TALLINN UNIVERSITY OF TECHNOLOGY
DOCTORAL THESIS
53/2024

On the Use of Defensive Schemes for Hardware Security

MOHAMMAD ESLAMI

TAL TECH PRESS

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

The dissertation was accepted for the defense of the degree of Doctor of Philosophy in Information and Communication Technologies on 7 July 2024

Supervisor: Prof. Dr. Samuel Pagliarini,
Department of Computer Systems, School of Information Technologies,
Tallinn University of Technology
Tallinn, Estonia

Co-supervisor: Dr. Tara Ghasempouri,
Department of Computer Systems, School of Information Technologies,
Tallinn University of Technology
Tallinn, Estonia

Opponents: Prof. Dr. Kaveh Razavi,
Department of Information Technology and Electrical Engineering,
Federal Institute of Technology (ETH) Zurich
Zurich, Switzerland

Prof. Dr. Philippe Maurine,
Laboratory of Computer Science, Robotics and Microelectronics of Montpellier (LIRMM),
University of Montpellier
Montpellier, France

Defence of the thesis: 4 October 2024, 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.

Mohammad Eslami _____
signature

European Union
European Social Fund
Investing
in your future

ISSN 2585-6898 (publication)
ISBN 978-9916-80-196-3 (publication)
ISSN 2585-6901 (PDF)
ISBN 978-9916-80-197-0 (PDF)
DOI https://doi.org/10.23658/taltech.53/2024

Eslami, M. (2024). On the Use of Defensive Schemes for Hardware Security [TalTech Press]. https://doi.org/10.23658/taltech.53/2024

TALLINNA TEHNIKAÜLIKOOL
DOKTORITÖÖ
53/2024

Kaitseskeemid riistvara turvalisuse tagamiseks

MOHAMMAD ESLAMI

TAL TECH
KIRJASTUS

Contents

List of Publications

The present Ph.D. thesis is based on the following publications.

I M. Eslami, T. Ghasempouri and S. Pagliarini, "Reusing Verification Assertions as Security Checkers for Hardware Trojan Detection," *In Proceedings of the 2022 23rd International Symposium on Quality Electronic Design (ISQED)*, Santa Clara, CA, USA, 2022, pp. 1-6, DOI: https://doi.org/10.1109/ISQED54688.2022.9806292

II M. Eslami, J. Knechtel, O. Sinanoglu, R. Karri, and S. Pagliarini, "Benchmarking Advanced Security Closure of Physical Layouts: ISPD 2023 Contest," *In Proceedings of the 2023 International Symposium on Physical Design (ISPD '23)*, New York, NY, USA, 2023, pp. 256–264. DOI: https://doi.org/10.1145/3569052.3578924

III M. Eslami, T. Ghasempouri and S. Pagliarini, "SCARF: Securing Chips with a Robust Framework against Fabrication-time Hardware Trojans," in IEEE Transactions on Computers, pp. 1-15, 2024. DOI: https://doi:10.1109/TC.2024.3449082

IV M. Eslami, T. Perez, and S. Pagliarini, "SALSy: Security-Aware Layout Synthesis," *In arXiv*, under review for IEEE Transactions on Dependable and Secure Computing, 2024. DOI: https://doi.org/10.48550/arXiv.2308.06201

Other related publications

V J. Knechtel, M. Eslami, et al. "Trojan Insertion versus Layout Defenses for Modern ICs: Red-versus-Blue Teaming in a Competitive Community Effort," accepted for publication in IACR Transactions on Cryptographic Hardware and Embedded Systems, pp. 1-41, 2025.

Author's Contributions to the Publications

I In Publication I, I was the main author and proved the idea of reusing the verification assertions as security checkers by investigating the security properties of verification assertions among 40 different IPs of the OpenTitan SoC using Cadence formal tools. I also prepared the results and wrote the manuscript.

II In Publication II, I was the main author, and I co-organized a global security contest. My role in the project involved designing various types of hardware Trojans and creating a fully automated flow for Trojan insertion into the submitted layouts by participants. I also prepared the figures and wrote the manuscript.

III In Publication III, I was the main author and developed a methodology to add online checkers to digital designs at the physical synthesis stage to protect against fabrication-time attacks. I also obtained the results, prepared the figures and wrote the manuscript.

IV In Publication IV, I was the main author and utilized various techniques during the physical synthesis step to protect the layout, resulting in a secure 65nm CMOS chip. I conducted experiments, tested the chip outputs, and analyzed results. Moreover, I prepared the figures and wrote the manuscript.

Abbreviations

3PIP	Third-Party Intellectual Property
ABV	Assertion Based Verification
AI	Artificial Intelligence
AO	Always On
ASIC	Application-Specific Integrated Circuit
BEOL	Back-End-Of-the-Line
BMC	Bounded Model Checking
CAD	Computer-Aided Design
CED	Concurrent Error Detection
CTS	Clock Tree Synthesis
DF	Degrading Factor
DfHT	Design for Hardware Trust
DFT	Design For Test
DMR	Dual Modular Redundancy
DRC	Design Rule Check
ECO	Engineering Change Order
FEOL	Front-End-Of-the-Line
FI	Fault Injection
FIB	Focused Ion Beam
FPGA	Field-Programmable Gate Array
FSP	Front-Side Probing
GDS	Graphic Data System
HDL	Hardware Description Language
HT	Hardware Trojan
IC	Integrated Circuit
IoT	Internet of Things
I/O	Input/Output
IP	Intellectual Property
IR	Instruction Register
ISPD	International Symposium on Physical Design
OEM	Original Equipment Manufacturer
PDK	Process Design Kit
PO	Primary Output
POS	Product Of Sum
PPA	Power-Performance-Area
PSL	Property Specification Language
RE	Reverse Engineering
RIE	Reactive Ion Etching
RTL	Register-Transfer Level
SALSy	Security-Aware Layout Synthesis
SC	Security Coverage
SoC	System on Chip
SOP	Sum Of Product
SPV	Security Path Verification
STA	Static Timing Analysis
SVA	SystemVerilog Assertion

TSMC	Taiwan Semiconductor Manufacturing Company
UV	UltraViolet

1 Introduction

The impact of digitalization has brought about significant changes in our daily lives, affecting the way we communicate, work, and interact as a society [1]. Integrated Circuits (ICs), or simply, computer chips, are a crucial component in modern electronics and have played a fundamental role in this technological revolution [2].

ICs have been essential in driving the technological advancements we experience today. These complex microelectronic components, which consist of an array of transistors on a small silicon substrate, form the foundation for the development and operation of a wide range of devices, from smartphones to sophisticated computing systems that power our daily lives [2, 3]. Their ability to condense complex functionalities into compact dimensions has been crucial in the exponential growth of computational power, leading to innovations that permeate every aspect of our interconnected world.

The fabrication of ICs involves a thorough and elaborate manufacturing process, **compulsorily** conducted in controlled cleanroom environments to ensure precision and reliability [4]. Scaling emerges as a major trend in IC fabrication, with the shrinking of transistors and the concentration of functionalities into a single chip. This not only enhances the overall performance of electronic devices but also contributes to a reduction in physical footprint, paving the way for the development of smarter and more powerful gadgets.

Within the domain of the IC supply chain, design houses occupy a key role. These entities are responsible for designing and developing the complex architecture of these microelectronic marvels, utilizing their engineering expertise and creativity to create the blueprint for future technologies. However, the complexity of IC fabrication often leads design houses to collaborate with specialized foundries for the actual manufacturing process. The most prominent players in this landscape are the Taiwan Semiconductor Manufacturing Company (TSMC), Intel, and Samsung [5]. These companies are well known for their cutting-edge facilities and extensive knowledge of semiconductor manufacturing.

Many design houses opt to outsource the manufacturing of their ICs to specialized foundries such as TSMC. This is due to the high cost of establishing and maintaining an in-house fabrication facility, which can easily exceed billions of dollars [6]. Even large companies like Apple prefer to outsource their chip fabrication to take advantage of the latest technologies and facilities offered by these foundries [7]. Outsourcing enables these companies to concentrate on their core strengths, such as design, while leveraging the expertise of external partners, like TSMC, in fabrication. This approach enables businesses to optimize efficiency and reduce costs.

The collaborative model between design houses and foundries, which is referred to as the fabless model, allows for flexibility, efficiency, and cost-effectiveness, all of which are crucial elements in navigating the dynamic landscape of semiconductor technology. By partnering with foundries, design houses can avoid the substantial capital investment required for establishing and maintaining in-house fabrication facilities, while still benefiting from the latest technologies and facilities.

1.1 The Significance of Integrated Circuits in Our Lives

ICs are a key part of technology advancements. They have changed the way we live and helped make progress in many areas. Their small size, excellent performance, and flexibility have allowed them to be used in many different ways. This has led to new developments in communication, computing, healthcare, transportation, and entertainment. This section discusses the essential role ICs play in our modern lives and how they help drive progress.

ICs are founded on the principles of integration, which involve the interconnection of

millions to billions of electronic components on a single silicon substrate. The miniaturization of transistors, guided by the famous Moore's Law [8], has led to rapid increases in computational power. At the same time, improvements in fabrication techniques have allowed for complex functionalities to be integrated within smaller and smaller devices [9]. From the simple beginnings of the transistor to the complex designs of modern microprocessors and system-on-chips (SoCs), ICs have evolved significantly. This evolution has made ICs a powerful force that drives the digital age.

ICs are found everywhere and play a crucial role in many industries due to their unique capabilities and functions. In communication, ICs form the backbone of wireless networks, enabling smooth connectivity and allowing data exchange on a global scale [10]. In computing, ICs power various devices that drive productivity and innovation, from smartphones and laptops to supercomputers and data centers [11]. In healthcare, ICs contribute to significant advancements in medical imaging, diagnostics, and treatment, transforming patient care and improving examination results [12]. Furthermore, ICs are vital in transportation systems, enhancing safety, efficiency, and sustainability through innovations in automotive electronics, avionics, and navigation systems [13]. Lastly, ICs enrich our leisure and entertainment experiences by powering gaming consoles, audiovisual equipment, and digital media platforms that captivate audiences around the world.

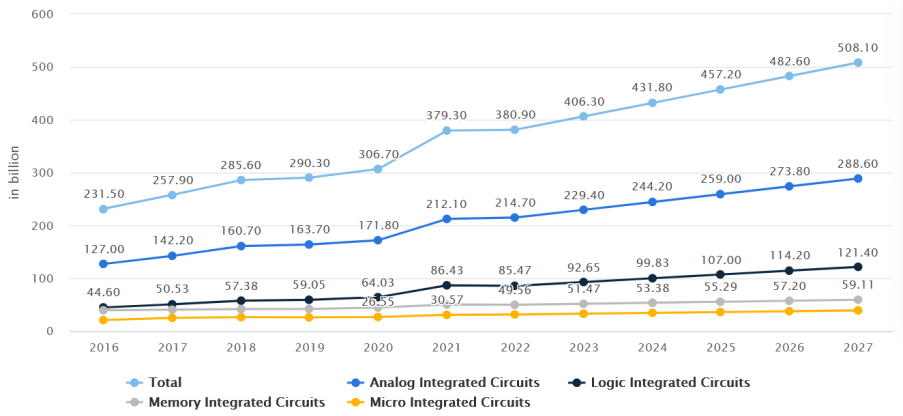

Figure 1: The estimated sales volume of the ICs based on the estimated end-market product volume (from [14])

The impact of ICs goes beyond just technological advancements, as they significantly influence societal dynamics, economic landscapes, and cultural norms. The widespread use of ICs has made information and resources more accessible, bridging geographical gaps and promoting global interconnectedness. Additionally, ICs have driven economic growth and innovation, creating new industries, job opportunities, and paths for entrepreneurship. As shown in Figure 1, the rapid increase in IC production is still expected to grow in the coming years [14]. This growth is driven by the increasing demand for electronic devices in various sectors, including consumer electronics, automotive, healthcare, and industrial applications. Moreover, the ongoing advancements in IC technology, such as the development of smaller, more powerful, and energy-efficient chips, are also contributing to the growth of the IC market. According to recent market research [14], shown in Figure 1, the sales volume of ICs is estimated to reach a value of over 508 billions by 2027.

Looking to the future, ICs hold enormous potential, with emerging technologies such as Artificial Intelligence (AI), the Internet of Things (IoT), and quantum computing set to

further expand the frontiers of possibility [15, 16]. Numerous companies are striving to capture a larger portion of the market by increasing their influence in the semiconductor industry.

The semiconductor industry is a significant player in the global economy, with IC fabrication and the sale of fabricated chips generating substantial revenue. Figure 2 presents the market shares of major semiconductor companies in the year 2022. However, this distribution is likely to change in the upcoming years, as only a few of these companies possess the capability to fabricate advanced ICs with feature sizes below 10nm [17]. This future also presents challenges that must be addressed, including ethical considerations, environmental sustainability, and ensuring equitable access to technological benefits. As those responsible for guiding technological progress, it is our duty to address these challenges thoughtfully and work towards creating a future where ICs contribute to a more sustainable, ethical, and equitable world.

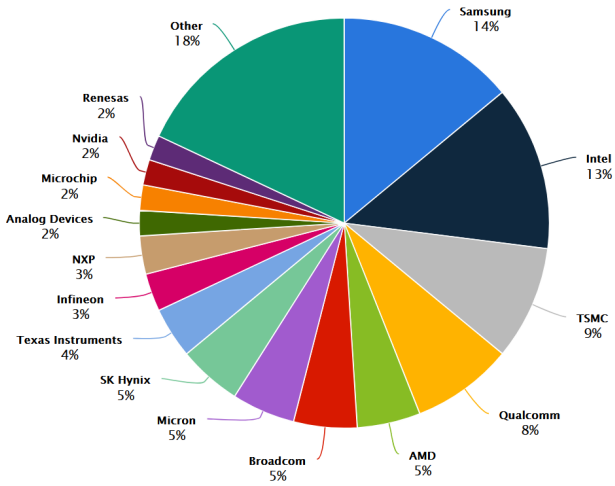

Figure 2: The market share of the selected leading brands of the total market size in the year 2022 (from [14])

It should be noted that in the context of this thesis, the term IC will almost always refer to Application-Specific Integrated Circuit (ASIC) unless otherwise specified. ASICs are a crucial class of ICs designed for specific applications, providing optimized performance and efficiency due to their custom-tailored design. To provide clarity and context, brief descriptions of various types of chips are presented in the following paragraphs.

Custom ASICs are specifically designed for a particular application or function. They offer unparalleled performance and efficiency because their design is optimized for specific tasks. Despite their high initial development costs, ASICs can achieve lower per-unit costs in high-volume production due to their fixed functionality once fabricated.

Field-Programmable Gate Arrays (FPGAs) provide flexibility and reprogrammability, allowing for post-manufacturing hardware configuration. This makes FPGAs ideal for rapid prototyping and for applications that require frequent updates. However, their general-purpose nature typically results in lower performance and higher power consumption compared to custom ASICs [18].

Microcontrollers combine a processor, memory, and peripherals on a single chip, optimized for control-oriented tasks. These chips are widely used in embedded systems for

applications such as sensor interfacing, motor control, and other low-power functions. Microcontrollers offer a balanced trade-off between flexibility and performance, capable of running software programs tailored to specific tasks.

1.2 Threats in the Integrated Circuit Supply Chain

While outsourcing the fabrication of ICs to specialized foundries offers numerous advantages, it also introduces a spectrum of threats and challenges that design houses must contend with [19]. In Figure 3, the overall life cycle of an IC, from conception to market distribution, is illustrated within a fabless model. The process begins at the design house, where the design phase is performed. During this phase, engineers meticulously define all specifications and intricacies of the IC, either by performing the digital design in its entirety or by incorporating Intellectual Properties (IPs) from Third-Party Intellectual Property (3PIP) vendors. The outcome of the design phase is a Graphic Data Stream (GDS) file, which encapsulates all components and interconnections of the design. This file is subsequently sent to the foundry for fabrication, where the chip is manufactured according to the design house's selected technology. Thereafter, the fabricated chip is forwarded to another facility for packaging. During the test and packaging phase, bare dies are packaged accordingly, and preliminary tests are conducted to ensure that the chips are free from defects. Finally, the chip is made available to the end-user through market distribution.

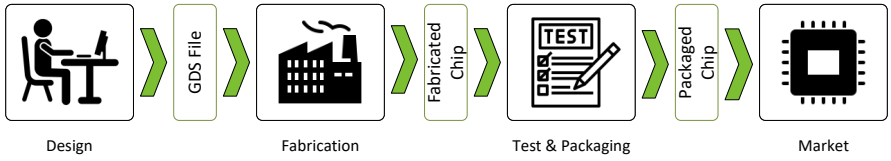

Figure 3: Different phases of IC's life cycle from design to market.

Once a chip is sent for fabrication, it moves beyond the direct oversight of the design house, which makes it difficult to monitor and ensure the integrity of the manufacturing process. This lack of oversight creates potential risks such as the insertion of Hardware Trojans (HTs) [20], Reverse Engineering (RE) [21], IP piracy [22], overproduction [23], counterfeiting [24], and broader supply chain security concerns. Even after verification and distribution in the market, chips remain susceptible to end-user threats such as Fault Injection (FI) [25], probing [26], and microarchitectural side-channel attacks [27]. Moreover, geopolitical factors can make these challenges even more complicated, adding uncertainties to the supply chain dynamics.

HTs refer to malicious modifications done to the IC during its life cycle before distribution in the market without the knowledge of the original designer [28]. The intention behind incorporating HTs comes from diverse entities, including dissatisfied employees, malicious third-party contractors, state-sponsored actors, competitors, and cybercriminals. Each group operates with distinct motivations that span from data theft and sabotage to spying, financial gain, competitive advantage, and the pursuit of political or ideological goals [20, 29].

RE is another challenging threat to the security of ICs, and is referred to as the process of analyzing and understanding the design, functionality, and composition of ICs without permission from the owner. A malicious user or a rogue element in the foundry can extract the gate-level netlist of a design, analyze it, and comprehend the functionality of the IC

[30, 31, 32]. This extracted information can be later used for IP piracy, IC counterfeiting, or as a guide for HT insertion by the adversary.

Another risk to the security of the ICs is IP piracy [22]. This method allows for the illegal and unauthorized use, reproduction, distribution, or exploitation of the IP without permission or proper licensing from the rightful owner. Such actions undermine the originality and exclusivity of the design house's IPs and can potentially harm the financial investments and competitive advantage of the design house.

Overproduction is the act of manufacturing more IC units than what is contractually required [23]. This surplus production often occurs without the knowledge or authorization of the owner. Overproduction can lead to a variety of risks and challenges, including the infiltration of counterfeit or unauthorized components into the market, which may lack the rigorous quality controls of legitimate products.

IC counterfeiting refers to the illegal manufacturing and distribution of fake ICs which are designed to look and function like genuine electronic components [24]. These counterfeit ICs are produced using substandard or inferior materials to mimic the design and performance of legitimate ICs made by established semiconductor manufacturers. This unlawful practice poses significant risks to the electronics industry since fake ICs can be used in various electronic devices, potentially compromising their security, reliability, and performance.

Concerning end-user threats, FI and probing represent two prominent security risks capable of compromising the integrity and confidentiality of ICs. These attacks aim to exploit the physical or logical vulnerabilities of the IC, with the objective of extracting sensitive information, altering the circuit's behavior, or causing malfunctions [25, 26, 33].

FI involves intentionally introducing errors or malfunctions into the IC's operation by manipulating its environment, such as voltage, temperature, or clock signals [33]. Attackers can use various FI techniques, including laser, electromagnetic, and voltage glitching, to induce transient or permanent faults in the circuit. These faults can lead to unintended behavior, such as bypassing security mechanisms, corrupting data, or revealing cryptographic keys.

Probing, on the other hand, involves physically accessing the IC's internal signals or data using specialized equipment, such as microprobes or Focused Ion Beams (FIBs) [26]. Attackers can use probing to extract sensitive information, such as cryptographic keys or proprietary data, or to modify the circuit's behavior by tampering with its internal connections. Probing attacks can be particularly effective against ICs with weak physical security measures, such as inadequate shielding or encapsulation.

Moreover, microarchitectural attacks, such as Spectre [34] and Meltdown [35], are a class of security vulnerabilities that exploit the microarchitectural features of modern processors, such as speculative execution, out-of-order execution, and cache hierarchies. These attacks can enable unauthorized access to sensitive data, including passwords, encryption keys, and personal information, stored in the memory of a computer system.

Spectre is a vulnerability that exploits speculative execution, a technique used by modern processors to improve performance by executing instructions before they are known to be needed. Spectre can trick a processor into executing malicious code that indirectly accesses sensitive data from the memory of other applications or the operating system.

Meltdown is a vulnerability that exploits the out-of-order execution and cache hierarchies of modern processors. Meltdown can enable an attacker to bypass the memory isolation mechanisms of the operating system and access sensitive data from the memory of other applications or the kernel.

In addition to the mentioned threats, geopolitical factors and international tensions add layers of complexity to the risks associated with outsourcing IC fabrication. Changes in trade policies, geopolitical disputes, or disruptions in diplomatic relations between countries can impact the supply chain and manufacturing processes [36]. Such uncertainties may lead to delays in the delivery of fabricated ICs, disrupting production schedules and affecting the timely release of products to the market.

Supply chain security risks can become more severe due to the absence of direct control over the manufacturing process [37]. Semiconductor supply chains have a global reach and involve numerous suppliers and subcontractors. If any of these entities encounter a security breach, it can result in the insertion of malicious components or unauthorized alterations during the fabrication process. This can have significant implications, ranging from compromised functionality to potential security breaches.

Due to these persistent threats, the semiconductor industry faces staggering financial losses, amounting to several billions of dollars annually [38]. Once the chip is sent for fabrication, the loss of oversight underscores the importance of addressing and mitigating the inherent risks associated with outsourcing IC production. Hence, design houses must implement robust security measures to safeguard against the mentioned threats.

1.3 Countermeasures

Countermeasures against the introduced threats are crucial for maintaining the integrity and security of the chips. The most common classes of techniques against these threats include HT detection [39, 40], fingerprinting [41, 42], and Design for Hardware Trust (DfHT) techniques [43] including hardware obfuscation [44], cryptography [45], watermarking [46, 47, 48], and split manufacturing [49, 50, 51, 52].

In addressing the challenges HT poses, both HT detection and DfHT techniques can be used to protect ICs. HT detection methods aim to identify potential HTs after the chip has been delivered. They employ various approaches, categorized into destructive and non-destructive methods. Conversely, DfHT methods aim to prevent HT insertion during IC fabrication or to ease their detection. The main difference is that HT detection methods are applied after IC fabrication, while DfHT methods require considering security measures in the IC design before fabrication.

Regarding RE, IP piracy, and IC counterfeiting, many techniques have been introduced in prior art. It is worth noting that methods developed to tackle one of these issues often provide solutions to the others as well. One of these methods is IC fingerprinting [41, 42], which is a process used to uniquely identify and authenticate individual ICs or chips based on their inherent physical characteristics and minor variations that occur during the fabrication process. These variations, which are impossible to clone or replicate, derive from factors such as process imperfections, environmental conditions, and material properties.

Watermarking [46, 47, 48] is another measure against IP piracy and IC counterfeiting, and it involves the incorporation of a distinct signature into an IP core, constructing the watermarked IP in a manner that preserves its original functionality. The key characteristic is that this process does not alter the core's intended operation. Upon the completion of the chip fabrication process, the IP owner can preserve it and extract its signature using pre-defined activation parameters. This extraction serves as a means to validate the lawful utilization of their IP core within the SoC by comparing it with the initially embedded signature. It is crucial for watermarking to be easily embeddable and verifiable without imposing significant overhead or succumbing to potential attacks.

Hardware obfuscation, [44, 53, 54, 55, 56, 57], is another security technique that pro-

tects ICs mainly by preventing RE in electronic systems. It involves intentionally making the design, functionality, or inner workings of a hardware component difficult to understand or interpret, thereby increasing the complexity and time required to reverse engineer a design. This technique can be implemented through various methods, such as logic obfuscation, layout obfuscation, gate-level obfuscation, functional obfuscation, and key-based obfuscation. These methods aim to modify the circuit's logic gates, alter the physical layout, add redundant or dummy gates, hide the actual functionality, or implement a secret key within the hardware design. By employing these techniques, design houses can prevent potential attackers and safeguard their IPs. However, it is essential to recognize that obfuscation is not a foolproof method and may still be bypassed by skilled attackers with sufficient resources and time. Examples of obfuscations schemes that have been 'broken' are SARLock [58], Anti-SAT [59].

While countermeasures like logic locking aim to prevent IP piracy and IC overproduction, watermarking primarily concentrates on verifying the legality of IP use. In light of numerous copyright violation incidents over the past two decades [60], having a unique identifier for each IP is considered essential for claiming ownership. The distinctive signature provided by watermarking facilitates a robust mechanism for IP owners to assert their rights and authenticate the rightful use of their IP within complex SoCs.

Cryptography provides a means to protect sensitive data, such as encryption keys, passwords, and personal information, from unauthorized access, modification, or disclosure [45]. In the context of hardware security, cryptography is used to implement various security mechanisms, such as secure boot, secure storage, and secure communication, to ensure the integrity, confidentiality, and authenticity of the hardware and the data it processes.

While cryptography is fundamental for securing data and communications, it is insufficient as a standalone solution for hardware security. Physical access to hardware can enable attackers to bypass cryptographic protections through side-channel attacks, fault injection, or direct probing. Additionally, vulnerabilities in the implementation of cryptographic algorithms and poor key management practices can undermine the effectiveness of cryptographic measures.

Split manufacturing [49] is a technique used to protect IP and prevent RE during the fabrication phase. This method involves dividing the IC manufacturing process into multiple stages, each performed by a different foundry or fabrication facility. By doing so, no single foundry has access to the complete IC design, making it significantly more difficult for an unauthorized party to reverse engineer or counterfeit the IC.

In split manufacturing, the goal is to divide the design into front-end-of-line (FEOL) and back-end-of-line (BEOL) processes. The FEOL process involves the fabrication of transistors and other active devices, while the BEOL process focuses on the creation of interconnects and metal layers. By separating these processes, the foundry responsible for the FEOL process has access only to the transistor-level design, while the foundry handling the BEOL process receives a partially fabricated IC with no information about the underlying transistor structure. This division of information helps maintain the confidentiality of the IP and reduces the risk of unauthorized replication or reverse engineering.

However, IC split manufacturing also presents challenges, such as increased coordination and communication between foundries, potential yield loss due to process mismatches, and the need for compatible sizes of metal layers and vias between the FEOL and BEOL processes. Despite these challenges, IC split manufacturing is a promising approach for enhancing the security and protection of IPs in the semiconductor industry.

In recent years, there has been a growing focus on security closure within the hardware

security community [61, 62, 63, 64, 65, 66, 67, 68, 69, 70]. Security closure refers to the procedure of verifying that the security measures and countermeasures integrated into a hardware system align with the intended security objectives, while simultaneously managing the trade-offs between PPA limitations [67]. This process is similar to timing closure, which aims to verify that the system meets the desired timing specifications under PPA constraints.

To promote research and development in the area of security closure, several academic conferences and workshops have been organizing security-focused design contests. For example, the International Symposium on Physical Design (ISPD) has been organizing the ISPD Hardware Security Contest in recent years [70, 71]. The contest aims to provide a platform for researchers and practitioners to showcase their innovative solutions for hardware security and to evaluate the effectiveness of their solutions against a set of benchmark circuits and attack scenarios.

The ISPD Hardware Security Contest 2022 and 2023 have attracted a large number of submissions from around the world, highlighting the growing interest and importance of security closure in the hardware security community. The contest results and the research papers presented at the conferences provide valuable insights and guidance for the design and implementation of secure hardware systems [61, 62, 63, 64, 65, 66, 68, 69]. This thesis has been shaped by these contests: in 2022, as a contest participant, I have secured a third place award after a 4-way tie between the top teams. In 2023, I have organized the contest in collaboration with my supervisor and external collaborators from New York University.

In addition to the aforementioned techniques, there are alternative methods such as Assertion Based Verification (ABV) primarily employed for verification and dependability objectives [72]. For instance, ABV is utilized to ascertain whether the design is free of bugs or resilient against faults. Although not primarily intended for security purposes, the similarity in effects between faults and security threats, such as when an HT is activated, suggests that these approaches could be repurposed for security applications [73, 74].

In general, ABV is a widely used functional verification methodology that employs formal properties, known as assertions, to validate the correctness of a digital circuit design. As shown in Figure 4, ABV is an essential part of modern design and verification flows, as it helps to improve the quality and efficiency of the verification process by providing a systematic and automated way to check the design's functional behavior.

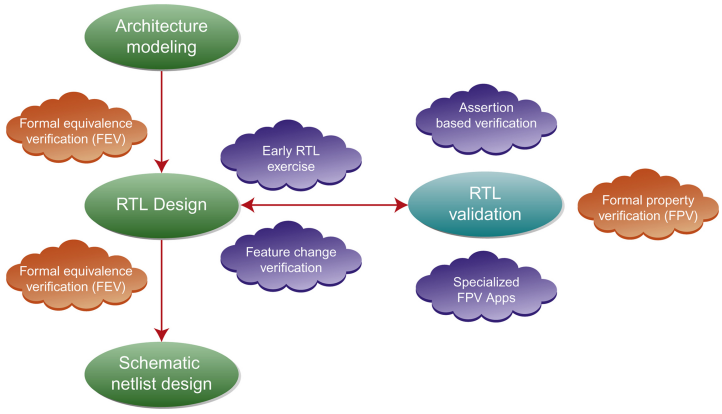

Figure 4: Different verification methods as part of modern design and verification flows (from [75]).

Assertions are formal statements written in a Hardware Description Language (HDL)

or a specialized assertion language, such as Property Specification Language (PSL) or SystemVerilog Assertion (SVA) [76]. They describe the expected behavior of a design under specific conditions, such as data dependencies, timing constraints, or protocol compliance. By embedding these assertions into the design or testbench, engineers can monitor the design's behavior during simulation and automatically detect any violations of the specified properties.

It is important to highlight that the mentioned countermeasures may exhibit efficacy against multiple threats. However, for the sake of brevity and clarity, the most common applications for each technique have been outlined.

1.4 Contributions and Outline of the Thesis

The core of this thesis is the investigation of methods and countermeasures against threats in the post-design stage, with particular emphasis on mitigating risks associated with HTs after the design phase. It involves several techniques in different phases of the IC design to enhance the security of ICs. Figure 5 presents the overall structure of the thesis.

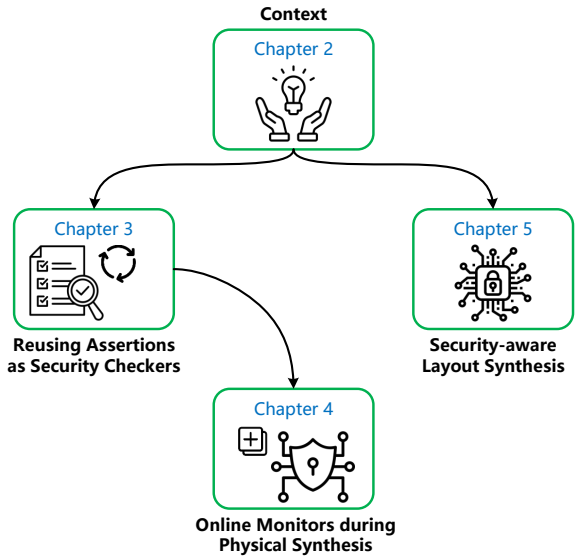

Figure 5: Structure of the thesis.

- **Chapter 2** This chapter offers a comprehensive exploration of advanced IC fabrication processes, along with an examination of various threats encountered throughout the IC life cycle and the latest countermeasures, with a specific emphasis on methods targeting HTs. It begins by providing an overview of the IC life cycle, detailing the steps involved from silicon to IC and encompassing aspects such as design and fabrication. Following this, the chapter explores the most critical security vulnerabilities of ICs, with a focus on HTs as a significant security concern. Finally, it surveys the existing countermeasures found in the current literature against HTs.

- **Chapter 3** This chapter introduces a novel approach to enhance the security of digital designs against HTs by repurposing verification assertions. The chapter explains how

assertions can be transformed into online monitors for efficient HT detection, and introduces a security metric and an assertion selection methodology. The chapter presents a comprehensive analysis of experimental results, demonstrating the adaptability and scalability of the method by applying over 100 assertions to a diverse set of IPs within the OpenTitan SoC. The chapter concludes by emphasizing the practicality and flexibility of the proposed detection solution, which is independent of the specific activation mechanisms of HTs, offering an adaptable security enhancement for digital designs.

- **Chapter 4** The main focus of this chapter is to present a comprehensive approach to enhancing IC security throughout the design process, particularly in the back-end stage. The chapter builds upon the previously introduced method for repurposing verification assertions as security checkers at the front-end phase of IC design. To further improve security, the chapter proposes a novel technique for incorporating online monitors during physical synthesis, providing an additional layer of protection at the back-end phase. The back-end flow can be considered as a complementary approach to the front-end method, but both techniques can also be employed independently, depending on user preferences and specific requirements, offering a more customizable and adaptable solution to enhance IC security.

- **Chapter 5** This chapter introduces Security-Aware Layout Synthesis (SALSy), a novel methodology for designing ICs with inherent security considerations, similar to the established practice of balancing PPA metrics and security, known as security closure. SALSy is a proactive strategy at the back-end phase that enhances IC security against fabrication-time and post-fabrication adversarial attacks, including HT insertion, FI, and probing. The methodology has been validated through a silicon-based demonstration, confirming its compatibility and effectiveness with a commercial Process Design Kit (PDK) and library. SALSy achieves this enhanced security with only a minimal impact on power consumption, thus maintaining a balanced trade-off between security and efficiency.

- **Chapter 6** As the last part of this thesis, this chapter serves as a summary of the main results and ideas presented in the study. It also suggests future research to improve IC security and the proposed methods, discussing possible ways for further development in the field.

2 Background

ICs have become a fundamental component of modern-day electronics, enabling the miniaturization and enhanced performance of various devices. The fabrication of ICs involves intricate processes that require advanced technology and precision engineering. This chapter provides a detailed overview of the complexities of IC fabrication and the efforts that have been made to enhance the security of these ICs against possible threats and attacks.

2.1 Life cycle of an Integrated Circuit

The life cycle of an IC involves several key stages. It begins with the design phase, where engineers create detailed schematics and layouts. This is followed by fabrication, where the IC is manufactured in a cleanroom environment using complex processes like photolithography. After fabrication, the IC undergoes testing and packaging to ensure it meets quality standards. Once packaged, the IC is distributed and integrated into electronic devices.

2.1.1 Design

The journey from idea to chips begins at the design stage, where the goal of fabricating the IC is established, and the relevant system specifications or requirements are set to meet user demands. Subsequently, engineers at the design house carefully define the implementation details of the chip. The design process is generally divided into two stages: front-end and back-end [77]. Each stage includes several steps that collectively contribute to the comprehensive design and fabrication of an IC.

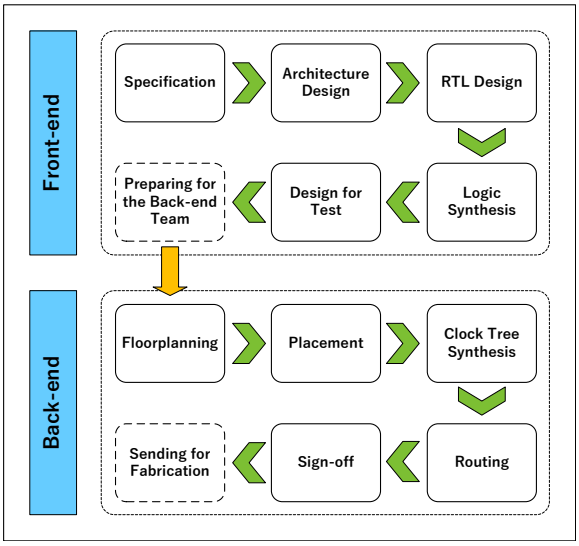

Figure 6: An overall view of different steps in IC design

As shown in Figure 6, each of the front-end and the back-end phases are divided into several main steps. In the following, more details are provided about each of the front-end and back-end procedures.

Front-end: The front-end phase focuses on the functional aspects of the IC. The primary goal of front-end design is to create a high-level description of the IC's behavior that meets the desired Power, Performance, and Area (PPA) requirements. The outcome of the front-

end phase is a gate-level netlist, which is passed to the back-end engineers for further implementation steps, and it involves the following steps:

- Specification: The first step in front-end design is to define the functional requirements, performance goals, and constraints of the IC. Moreover, the number of the pins and the packaging type is considered in this step. These specifications serve as a guideline for the subsequent design stages and ensure that the final product meets the intended purpose. At this stage, it is crucial to consider the interaction between hardware and software components to ensure that the IC is optimized for both hardware functionality and software compatibility, leading to a more efficient and effective design.

- Architecture Design: Based on the specifications, a high-level architecture is developed to meet the PPA requirements. This step involves defining the overall structure, organization, and functionality of the system or chip being developed. Additionally, designing interface and communication is performed in this step, which defines the interfaces between modules and creates the communication protocols and data transfer mechanisms, including the selection of interconnect protocols and the definition of buses. Domain separation between digital and analog components is also addressed, along with the establishment of clock domains and power domains to ensure proper operation and power management.

- Register-Transfer Level (RTL) Design: The IC's functionality is described using an HDL, such as Verilog or VHDL, at the RTL. This level of abstraction allows designers to focus on the data flow and control logic without worrying about the underlying gate-level implementation.

- Logic Synthesis: The RTL design is converted into a gate-level netlist, which consists of logic gates and their interconnections. During logic synthesis, the design is optimized for PPA metrics considering the target technology and fabrication process.

- Design for Test (DFT): Test structures and methodologies, such as scan chains, are implemented in this step to facilitate testing and debugging of the fabricated IC. This step involves a set of techniques to measure the reliability of the IC.

Back-end: Back-end design, also known as physical design, focuses on the physical implementation of the IC. This stage involves the use of Computer-Aided Design (CAD) tools to develop a digital representation of the IC [78]. It begins with transforming the gate-level netlist into a layout that meets the geometric and electrical constraints imposed by the fabrication process. The primary goal of back-end design is to create a manufacturable IC layout with optimal performance, power, and area. The outcome of the design phase is a GDS file, which contains all the necessary information for the subsequent fabrication stages. The back-end design stage includes the following main steps:

- Floorplanning: In this step, the overall outline of the IC is defined, including the placement of major blocks, Input/Output (I/O) pads, and power distribution networks. This step sets the foundation for the subsequent placement and routing stages.

- Placement: The exact location of standard cells (pre-designed logic gates) and macro blocks (larger functional units, such as memory or processors) within the IC layout is determined during placement. The primary objective of placement is to minimize the interconnect length and optimize the IC's performance, power, and area.

- Clock Tree Synthesis (CTS): A balanced clock distribution network is generated at this step to ensure that clock signals reach all parts of the IC with optimal skew and delay. This step is critical for synchronizing the IC's operation and optimizing its performance.

- Routing: Once the CTS is completed, the interconnections between standard cells and macro blocks are created using metal layers. Routing must adhere to design rules and optimize for signal integrity, power, and performance.

- Sign-off: Final checks and analyses, such as Static Timing Analysis (STA), power analysis, and reliability analysis, are performed during sign-off to ensure that the design meets all specifications and is ready for fabrication.

2.1.2 Fabrication

Silicon is the most commonly used material for IC fabrication due to its abundant availability, chemical stability, and semiconductor properties. The use of silicon as a base material can be traced back to the invention of the transistor in 1947 by John Bardeen, Walter Brattain, and William Shockley [79]. In the subsequent years, the development of silicon-based ICs revolutionized the electronics industry, leading to the rapid advancement of technology.

The fabrication of ICs is a complex process that involves several stages, including wafer preparation, photolithography, etching, doping, metallization, and dicing. Each stage is critical in ensuring the overall performance and reliability of the final product. An overview of IC fabrication flow is illustrated in Figure 7.

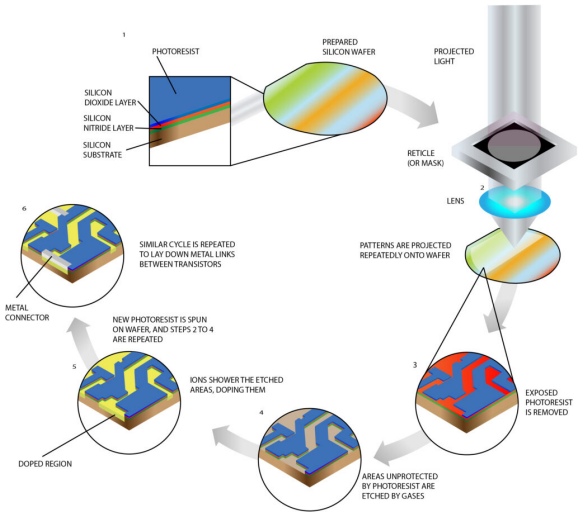

Figure 7: Overall view of IC fabrication flow (from [80])

Wafer Preparation: In order for silicon to be suitable for use in computer chips, it undergoes a rigorous purification process to achieve a purity level of less than one foreign atom per billion atoms. The purified silicon is first melted and then pulled to create a solid, resulting in a single, continuous, and unbroken crystal lattice structure in the form of a cylindrical ingot.

As depicted in Figure 8, this ingot is subsequently sliced into thin, circular surfaces called *wafers*. Engineers use these silicon wafers as the substrate for semiconductor devices.

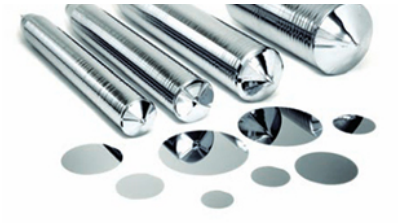

Figure 8: Silicon ingots (top) and wafers (bottom) of different diameters (from [81])

Photolithography: Photolithography is a process used to transfer the circuit pattern from the design phase onto the silicon wafer. The principle of photolithography hinges on replicating a structure delineated on a lithographic mask onto a light-sensitive resist, previously coated on a substrate. This process offers two options: utilizing either positive resist or negative resist. The procedure of using positive resist involves the following steps:

- Cleaning and Deposition of Metal Film: The silicon wafer is thoroughly cleaned to remove any impurities or contaminants that may affect the fabrication process. Afterward, a metal film is deposited on the substrate.
- Coating: The wafer is coated with a light-sensitive material called photoresist, which undergoes chemical changes when exposed to UltraViolet (UV) light. This coating is commonly heated for 30 minutes between 60 and 100 °C.
- Exposure: The circuit pattern is projected onto the photoresist-coated wafer using a mask aligner, which exposes the photoresist to UV light in the desired pattern. Thus, the structure of the mask is imaged on the resist and causes photochemical changes therein.
- Development: The exposed photoresist is chemically developed, resulting in a patterned layer that serves as a template for the subsequent etching and doping processes.

Etching: Etching is the process of selectively removing layers of material from the silicon wafer to create the desired circuit structure. This process can be performed using wet chemical etching or dry etching techniques, such as plasma etching or Reactive Ion Etching (RIE) [82]. The choice of etching technique depends on the specific material being etched and the desired level of precision.

Doping: Doping is the process of intentionally introducing impurities, known as dopants, into the silicon wafer to alter its electrical properties. During this process, high-energy ions are accelerated and implanted into the wafer, altering its conductivity and creating regions of n-type or p-type semiconductor material essential for device functionality. Common dopants include boron, phosphorus, and arsenic, which are used to create p-type and n-type semiconductors [83]. Doping can be achieved through various techniques, such as diffusion, ion implantation, and epitaxy.

Metallization: Metallization is a process that involves depositing metal layers onto the surface of a silicon wafer to create interconnects and contact pads, which are crucial for linking various components of an IC. Techniques such as sputtering or electroplating are employed to deposit metals like aluminum, copper, or tungsten onto the wafer.

Dicing: Once the fabrication of individual ICs on the wafer is complete, the wafer undergoes dicing to be cut into individual chips. Dicing is typically performed using specialized

cutting tools, such as diamond saws or laser systems, to precisely cut along predefined lines, known as scribe lines, that separate the chips. The separated chips are then picked up and forwarded to the packaging facilities for further testing and integration into electronic systems.

2.1.3 Testing and Packaging

After the fabrication process is completed, the chips are subjected to electrical testing to verify the correct functioning. Once the functional dies have been identified, they are separated from the wafer and packaged as individual semiconductor devices. The final stage in IC fabrication is packaging, which involves encapsulating the silicon chip in a protective casing and connecting it to external components. Packaging helps to ensure the mechanical stability, electrical performance, and thermal management of the IC [84].

2.1.4 Distribution

After successful packaging and testing, the chips are shipped to distributors, who act as intermediaries between the manufacturers and Original Equipment Manufacturers (OEMs). Distributors often maintain relationships with multiple manufacturers, which enables them to offer a diverse portfolio of chip products to answer the varying needs of OEMs. Finally, OEMs integrate these packaged chips into their products, such as computers, smartphones, and other electronic devices, which are ultimately sold to end-users through retail channels.

2.2 Security Vulnerabilities of Integrated Circuits

As previously mentioned, the process of bringing an IC to market involves numerous complex stages, often spanning multiple countries for various design, fabrication, packaging, and testing phases. At each stage, various threats can compromise the security of the ICs. Although a rogue engineer could manipulate the IC during different design stages, design houses typically have good control over their staff and can prevent such irregularities to a large extent unless they use infected 3PIP [85]. Attacks during the testing and packaging phases primarily involve false test report generation and the use of lower-quality materials in packaging and bonding. Furthermore, these attacks are more likely to be discovered by the design house upon IC delivery.

The threats persist even after the chip is distributed in the market, with attackers potentially attempting to reverse engineer the chip for various purposes. However, a significant threat lies in the foundry, where IC fabrication takes place, and where there is no control over what happens to the chip. This is particularly concerning because most recognized threats to IC security, such as HT insertion, RE, IP piracy, and IC counterfeiting, can occur individually or simultaneously during the fabrication process. Consequently, this thesis focuses on threats in the post-design stages, especially those within the foundry.

Among all the threats in the fabrication phase, HT insertion is more likely to remain undetected due to its stealthy nature and has received significant attention in recent years within academia. It is worth noting that implementing techniques to countermeasure HTs may also be effective against other security attacks. Further details about this will be provided in the following sections.

2.2.1 Hardware Trojan as a Major Security Risk

HTs are malicious modifications or additions that are intentionally made to the design, layout, or functionality of an IC in order to compromise the security, reliability, or performance of electronic systems. These modifications are typically inserted during the design and fabrication stages of the IC's lifecycle, often without the knowledge of the

original designers. An HT imposes a significant threat to any hardware design intended for deployment in critical operations.

HTs are typically composed of *trigger* and *payload* parts. The trigger is an optional part of the HT and is referred to as a particular input sequence, temperature, or voltage level at which the HT can be activated. If an HT does not include the trigger, it is called an "always-on" HT. However, the trigger part is typically designed to only activate in extremely rare conditions, in order to make the detection of HT difficult.

The payload is the part of an HT that is responsible for carrying out the malicious action once the trigger condition is satisfied. This action can result in various forms of harm, including data theft, denial of service, or unauthorized access to the system. As long as the trigger condition is not met, the circuit operates normally, making it difficult to detect the presence of the HT. When the payload becomes activated, the malicious behavior is executed. This stealthy nature of HTs, where they remain dormant until the payload is activated, makes their detection particularly challenging.

A malicious foundry can incorporate three categories of HTs into an IC layout: additive, substitution, and subtractive [86]. Additive HTs involve introducing additional circuit components and/or wiring into an existing design. Substitution HTs necessitate the removal of logic to accommodate extra HT circuit components and/or wiring within an existing circuit design. Lastly, subtractive HTs include the removal of circuit components and/or wiring to modify the behavior of an existing circuit design. This thesis focuses on evaluating the vulnerability of a circuit layout to additive HT attacks due to their significant impact on system behavior, their detectability through changes in different characteristics of the design, and the extensive body of research that provides a solid foundation for further study.

Figure 9 presents an HT taxonomy based on the trigger and payload types of additive HTs. The triggers can be created by adding, modifying, or removing hardware components within an IC and can be either digital or analog. An ideal trigger for an HT is identified by its key characteristics, which include a small size that requires minimal additional circuit components, stealth that demands rare events for activation, and controllability that enables easy activation by attackers but not by defenders or during normal operation. Previously demonstrated triggers exhibit a range of characteristics, from large and stealthy (requiring many additional gates) to small and easily triggered. Sophisticated HTs are typically small, stealthy, and controllable.

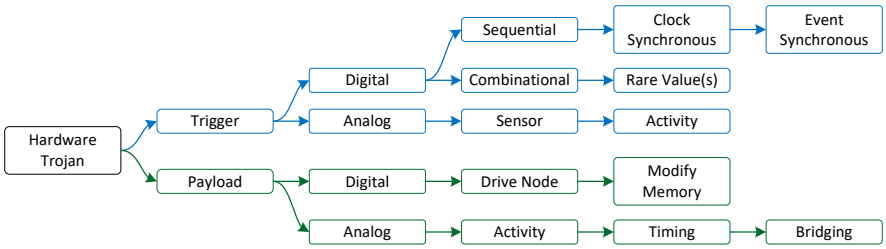

Figure 9: HT Taxonomy based on trigger and payload types (from [87])

On the other hand, both analog and digital payloads exist, with various effects such as information leakage, alteration of the IC's internal state, or rendering the system unusable

through a denial-of-service attack. Regardless of the effect, the payload mechanism must establish a connection to or near a target, which can be a *security-critical* component within the IC design.

HTs pose a substantial security threat due to their complex nature and the challenges they present in detection. Traditional testing and verification methods, such as functional verification and design rule checking, are often inadequate for detecting HTs. This is because HTs are designed to be stealthy, remaining dormant until activated by a specific, often complex, trigger. The relatively simple test vectors used in conventional testing are unlikely to activate these triggers.

Moreover, HTs are designed to make only subtle changes to the IC's behavior, which can be difficult to detect using traditional fault models. These models are intended to identify accidental faults, such as manufacturing defects, and are not equipped to detect intentional, malicious modifications like HTs.

The task of detecting HTs requires a unique set of test vectors that can activate the target fault. This can be a challenging and time-consuming process, particularly for cyclic sequential designs. Furthermore, the increasing complexity and size of modern ICs exacerbate the detection process, making it even more difficult to identify and isolate these malicious modifications. Therefore, while test vectors and fault models are essential tools in IC testing, they may not be sufficient for detecting HTs, and more advanced, specialized methods are needed.

Another issue is that HTs can be designed to be stealthy and adaptable, making them suitable for various malicious activities. For example, an HT can be programmed to leak sensitive information, disable critical system functions, or create a backdoor for unauthorized access, depending on the attacker's objectives.

Furthermore, since ICs are used in various electronic systems, a single compromised IC can have far-reaching consequences. For instance, an HT in a widely used microprocessor could affect millions of devices, leading to significant security and privacy breaches.

Lastly, the globalized nature of the semiconductor industry increases the risk of unauthorized access to the IC design or manufacturing process, making it easier for adversaries to insert HTs. This global supply chain presents numerous vulnerabilities that attackers can exploit and highlights the need for robust security measures throughout the IC life cycle. Due to these rising concerns, protecting measures should be added to ICs before sending them for fabrication in order to mitigate the risks posed by HTs.

2.3 Countermeasures against Hardware Trojans

As mentioned in the previous section, HTs pose a significant threat to ICs, as they are embedded at the hardware level, which makes software-level countermeasures inadequate for addressing the risks they present. The detection of HTs in hardware designs is a complex task, primarily due to the absence of a golden version or a known-good reference for comparison during the verification process.

In principle, an effective method for detecting an HT would be to activate it and observe its effects. However, this approach is challenging, as an HT's type, size, and location are generally unknown, and its activation is likely to be a rare event. Consequently, an HT can remain hidden during the normal operation of the chip and only becomes active when the specific triggering condition is met. This stealthy nature of HTs necessitates the development of advanced detection and mitigation strategies at the hardware level to ensure the security and reliability of ICs.

To minimize the risk of HTs, researchers have been exploring different methods in recent years. These methods are mainly classified into DfHT techniques and HT detection

techniques, which are approaches used to enhance the security and reliability of ICs. An overview of these techniques is shown in Figure 10.

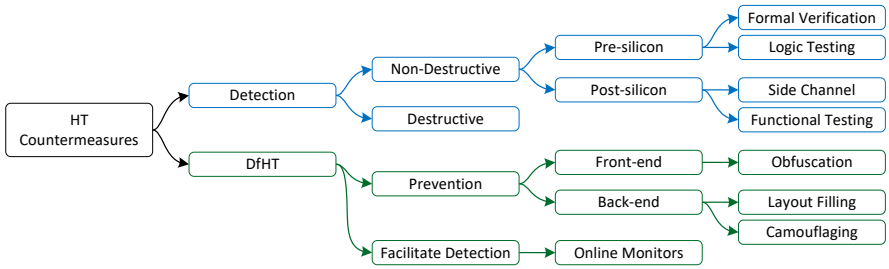

Figure 10: An overview of different protection methods against HT (adopted from [88])

2.3.1 Detection Techniques

HT detection is the most widely adopted approach by researchers to tackle HTs [89]. The primary goal of this method is to verify the integrity of existing designs and fabricated ICs without requiring additional circuitry. These techniques are categorized into destructive and non-destructive methods.

Destructive techniques typically involve reverse engineering the IC by depackaging and obtaining images of each layer to reconstruct the golden behavior of the chip. This approach has the potential to provide very high assurance that any malicious modification in the IC can be detected. However, it comes with significant drawbacks, such as high cost and time consumption, taking several weeks or months for an IC of reasonable complexity. Furthermore, at the end of this invasive process, the IC cannot be used, and the information obtained is limited to a single IC sample. An overview of the process of delayering an IC is presented in Figure 11.

It is important to note that reverse engineering modern complex chips is a labor-intensive and error-prone task. Obtaining the entire chip structure through RE may necessitate the use of tens of ICs, as depackaging and delayering procedures can introduce unintended errors in the RE process. Nonetheless, employing destructive RE on a limited number of samples may be appealing for acquiring the characteristics of a golden batch of SoCs. However, destructive method is proven to be the only effective approach among HT detection techniques in practice.

On the other hand, non-destructive methods, as their name indicates, aim to detect HTs without causing harm to the IC. Some of these methods are performed in the pre-silicon stage, where the design has not yet been sent for fabrication. These techniques are primarily used to validate 3PIPs purchased from third-party vendors [91, 92]. In this stage, the design house retains control over the circuit and has the ability to simulate and observe internal signals to detect potential malicious behavior. The main existing pre-silicon detection methods include formal verification and logic testing.

Formal verification methods [91, 92, 93, 94, 95, 96, 97, 98, 99] involve creating mathematical models of the circuit design and its specifications, and then using automated tools, such as theorem provers or model checkers, to exhaustively analyze the design for any discrepancies or violations of the specified properties. This process can help detect HTs, as well as other design errors or vulnerabilities in 3PIPs, that might be missed by traditional simulation-based testing methods.

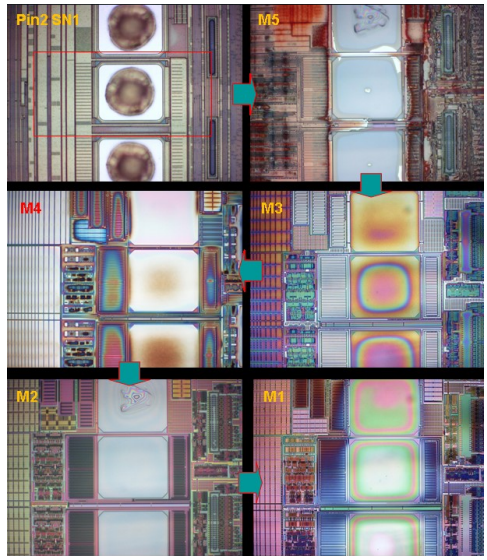

Figure 11: The process of delayering an IC by removing each layer of die (from [90])

Some common formal verification techniques used in HT detection include property checking [93], equivalence checking [94], model checking [95], and information flow [96]. While traditionally applied to software systems to uncover security bugs and enhance test coverage, these methods have also proven effective in verifying the trustworthiness of 3PIP [91, 92].

Property checking involves verifying whether a given circuit design satisfies specific safety or security properties, such as the absence of unauthorized information flow or the presence of proper access control mechanisms. Equivalence checking compares the original circuit design with a trusted version or a higher level of abstraction to ensure they exhibit the same functionality, helping identify any malicious modifications. Model checking explores all possible states of a circuit design to verify that it adheres to the specified properties and does not contain any unintended behavior. Information flow analysis is used to analyze and track the flow of sensitive data within a digital circuit design. The primary goal of information flow analysis is to ensure that confidential information does not leak to unauthorized parts of the circuit or external entities, which could be a result of malicious modifications or HTs.

In [97], a model-checking technique is introduced to formally verify the presence of malicious modifications in 3PIP caused by HTs. This method is based on the concept of Bounded Model Checking (BMC). BMC generates reports detailing the sequence of input patterns that violate specific defined properties. The main advantage of this approach is its feasibility to extract the triggering condition of the HT from these reported input patterns. Another approach, as presented in [98], focuses on formally verifying unauthorized information leakage in 3PIPs. However, due to the challenge of space explosion, these approaches are constrained by the limited processing capability of model checking. While these techniques offer a promising solution for HT detection, they each encounter specific challenges and limitations [99].

Another category of pre-silicon approaches is logic testing [100, 101, 102] , which focuses on analyzing the functional behavior of a digital circuit design to identify potential malicious modifications. In this approach, test patterns or vectors are applied to the circuit's

inputs, and the corresponding outputs are observed and compared with the expected results. The primary objective of logic testing is to activate any hidden or rare trigger conditions associated with HTs, thereby exposing their malicious payloads. While some works distinguish logic testing from functional analysis based on the type of input patterns used (specific patterns for logic testing and random patterns for functional analysis), both methods share the same fundamental concept of applying inputs and observing outputs. Consequently, this thesis considers them as part of the same category.

A technique is introduced in [100], which identifies suspicious nets with weak input-to-output dependency. This approach is based on the observation that an HT is triggered under extremely rare conditions, which assumes the logic implementing the trigger circuit nearly unused or inactive during normal operation. The authors proposed a metric to find those nearly unused logic by quantifying the degree of controllability of each input net on its output function. This metric is computed by applying random input patterns and measuring the number of output transitions. If the threshold for a net is lower than a predefined one, it is flagged as suspicious. However, this technique has significant limitations, such as producing a large number of false-positive results and not providing any method to verify if the suspicious signals perform malicious operations. Another approach presented in [101] demonstrates how to design HTs that can evade [100] by distributing the trigger vector over multiple clock cycles.

The authors of [102] presented a technique to identify potential triggering inputs of an HT. The proposed technique is based on the observation that input ports of the triggering circuit of an HT remain inactive during normal operation. It performs functional simulation of the IP with random input patterns and traces the activation history of the input ports in the form of Sum-Of-Product (SOP) and Product-Of-Sum (POS). It then identifies redundant inputs by analyzing the SOPs and POSs, which are unactivated during functional simulation. These redundant input signals are potential triggering inputs of a HT. However, this method also produces a large number of false-positive results due to incomplete functional simulation and unactivated entries belonging to normal operations.

Pre-silicon techniques can be used effectively in many applications. However, the main challenge in HT detection arises when HTs are inserted during the fabrication process. In this scenario, the design house receives only the fabricated chip, making it impossible to observe all internal signals.

Post-silicon HT detection schemes are employed after the chip fabrication process. As depicted in Figure 10, these techniques can be categorized into two main classes: side channel and functional testing.

Side-channel analysis approaches [103, 104] aim to detect HTs by measuring various circuit parameters, such as delay, power (static and dynamic), temperature, and electromagnetic radiation. These methods exploit the side effects caused by additional circuits or activities resulting from HT trigger/payload activation. However, most detection techniques rely on the availability of “golden ICs” (HT-free ICs) for comparison to identify HT-infected ICs.

The authors in [103] demonstrate the use of side-channel profiles such as power consumption and electromagnetic emanation for HT detection. They generate power signature profiles from a small set of ICs randomly selected from a batch of manufactured ICs, which serve as golden chips. After profiling, the golden chips undergo rigid destructive RE to compare them against the original design. If found to be HT-free, the ICs are accepted as genuine, and their profiles serve as power templates. The remaining ICs are then tested efficiently and non-destructively by applying the same stimuli and building their power profiles. These profiles are compared using statistical techniques, such as principal component

analysis, against the templates obtained from the golden chips.

While side-channel analysis methods may achieve some success in detecting HTs, they face challenges in providing high coverage for every gate or net and extracting the abnormal side-channel signals of HTs in the presence of process and environmental variations. As IC feature sizes shrink and the number of transistors increases, growing process variations can easily mask the small side-channel signals induced by low-overhead and rarely triggered HTs. The authors in [104] proposed a backside imaging method based on filler cell patterns in the IC layout, which does not require a golden chip. However, the comparison between simulated and measured optical images still suffers from manufacturing process variations. Additionally, capturing clear images at higher resolutions is time-consuming.

Functional testing techniques [105, 106] aim to activate HTs by applying test vectors and comparing the responses with the correct results. The effectiveness of these techniques relies on the availability of a golden response. Although this approach may seem similar to manufacturing tests used for detecting manufacturing defects, conventional manufacturing tests using functional, structural, or random patterns have limited success in detecting HTs [107]. Skilled adversaries can design HTs that are activated under extremely rare conditions, allowing them to evade detection during the manufacturing test process.

To address this challenge, researchers in [105] and [106] have developed test pattern generation methods to trigger rarely activated nets and improve the probability of observing HT effects from primary outputs. However, due to the vast number of logical conditions in a circuit, it is impractical to enumerate all conditions of a real design. Moreover, HTs that transmit information via nonfunctional means, such as through an antenna or by modifying the specification, can evade detection by functional tests [108]. These limitations highlight the need for more advanced and sophisticated functional test techniques to effectively detect HTs.

In summary, destructive methods involve physically dissecting or altering the chip to analyze its internal structure, while nondestructive methods rely on non-invasive techniques to inspect the chip's functionality and behavior without causing any physical damage. Both approaches have their advantages and limitations, and their selection depends on various factors, including the specific application, available resources, and desired level of detection accuracy.

2.3.2 Design for Hardware Trust

As mentioned in the previous section, detecting a smartly-designed HT with a small size remains a significant challenge using existing techniques. As depicted in Figure 10, DfHT approaches aim to address this issue by incorporating additional logic to either facilitate the detection of HTs [109, 110, 111] or to prevent an adversary from inserting an HT in the first place [91, 92, 95, 96, 104, 112, 113]. Although it is impossible to achieve complete prevention against HT insertion in practice, research efforts have concentrated on limiting available chip resources to make it extremely difficult for adversaries to exploit them for the insertion of malicious logic [91, 92, 95, 96].

To facilitate HT detection, prior works try to add different redundancies to the design before sending it for fabrication in order to verify if the IC is HT-free after it is delivered. One of the most common redundancies is adding **online monitors** to the design. Online monitoring [114, 115, 116, 117, 118] is an effective approach to increase trust in hardware systems concerning HT attacks, as triggering all types and sizes of HTs during pre-silicon and post-silicon tests is exceptionally challenging.

These techniques have been widely employed for enhancing reliability and dependability, with a focus on Concurrent Error Detection (CED) methods. CED techniques introduce

redundancy through parity codes or hardware duplication and incorporate a dedicated checker to identify any discrepancies or errors [114]. Moreover, these techniques can be repurposed to identify unwanted changes caused by HTs [115, 116]. The authors in [119] and [116] have proposed leveraging a diverse range of 3PIP vendors to mitigate the impact of HTs. The work presented in [119] suggests verifying the integrity of a design by comparing multiple 3PIPs with an alternative untrusted design that serves a similar function. Meanwhile, [116] utilizes some constraints to prevent collusion among 3PIPs sourced from the same vendor to enhance the security of the design.

Another group of approaches utilizes a distributed software scheduling protocol to establish a trustworthy system resilient to HT activation in a multicore processor [117, 118]. These methods seek to mitigate the impact of HTs by coordinating the allocation and execution of tasks across multiple cores.

Furthermore, online monitoring techniques can utilize existing or supplemental on-chip structures to monitor chip behaviors [120, 121] or operating conditions, such as transient power [122, 123] and temperature [110]. Upon detection of any irregularities, the chip can be disabled or bypassed to ensure reliable operation, even though with some performance overhead. The authors in [124] propose a design for an on-chip analog neural network that can be trained to distinguish trusted from untrusted circuit functionality based on measurements obtained via on-chip measurement acquisition sensors.

The second category of DfHT techniques comprises HT prevention methods. As the name suggests, these approaches aim to prevent HT insertion by employing various techniques at different stages of IC design. In order to insert targeted HT, attackers usually need to gain an understanding of the design's functionality first. Since HT attacks are mostly performed in a location other than the design house, the attacker typically accomplishes HT insertion by reverse engineering the circuit to identify its intended functionality.

The process of reverse engineering a circuit can be both time-consuming and complex, as it requires analyzing the circuit's structure and behavior to understand its purpose and potential vulnerabilities. However, since an attacker has access to advanced CAD tools and the same PDK used by the design house, the threat of reverse engineering the design and, consequently, HT insertion remains. Therefore, preventive techniques are used to make the insertion of HTs as difficult as possible and minimize the risk of successful attacks. Depending on the implementation phase where the defensive technique is applied, these approaches can be further classified into front-end and back-end techniques.

Front-end engineers may employ various obfuscation techniques, such as logic locking [125, 126], to safeguard the IP of the design. The primary goal of logic obfuscation is to hide the original functionality of a design by incorporating several locking schemes into the original design. These locking circuits reveal the correct function only when the correct key is applied. This can make it more difficult for attackers to insert HTs without knowing the right input vectors.

Combinational logic obfuscation can be achieved by using XOR/XNOR gates at specific locations in a design [127]. In sequential logic obfuscation, additional states are introduced to a finite state machine to hide its functional states [106]. Some techniques also suggest the insertion of reconfigurable logic for logic obfuscation [128, 129]. The design operates correctly only when the reconfigurable circuits are correctly configured by the design house or end-user.

On the other hand, the focus of the back-end phase is to improve the security of the layouts during the physical synthesis stage. The HT prevention techniques in the back-end phase include layout filling [130, 131, 132] and camouflaging [133, 134, 135, 136, 137].

Layout filling is a technique aimed at restricting available resources, such as gaps and

free routing tracks, to prevent adversaries from inserting malicious logic [130, 131, 132]. During the back-end phase, CAD tools are often unable to completely fill the area with regular standard cells. Consequently, unused spaces are typically occupied by filler cells or decap cells, which serve no functional purpose. The main purpose of filler cells is to occupy the empty spaces in the layout, which can help to improve the overall density and reduce the risk of manufacturing defects, while decaps are utilized to manage peak current in the chip, particularly in areas with significant instantaneous power.

As a result, one method for attackers to insert HTs into a circuit layout involves replacing filler cells and, to some extent, decaps. Removing these nonfunctional cells minimally impacts electrical parameters, causing the presence of maliciously inserted cells challenging to detect. By occupying these spaces with functional cells, dummy vias, or other design elements, the attacker's ability to exploit the layout for inserting HTs is significantly reduced. This approach makes it more challenging for an attacker to find suitable locations to insert malicious logic without being detected or causing functional issues in the design.

An approach called BISA is presented in [130], and it involves filling empty spaces with functional filler cells during layout design. Subsequently, these cells are interconnected to establish combinational circuitry that can be tested later. Any failure detected during subsequent testing indicates that a functional filler has been substituted by a potential HT. The general BISA insertion flow includes preprocessing (gathering detailed information about the standard cell library), identifying unused spaces, placing BISA cells, and routing BISA cells.

Camouflaging [133, 134, 135, 136, 137] is a layout-level obfuscation technique that aims to create indistinguishable layouts for different gates by adding dummy contacts and faking connections between the layers within a camouflaged logic gate. It involves replacing standard logic gates with functionally equivalent but visually different gates to make it challenging to identify the actual logic function implemented in the design. By camouflaging the gates, attackers are hindered from extracting a correct gate-level netlist of a circuit from the layout through imaging different layers. As a result, the original design is protected from the insertion of targeted HTs. The authors in [138] employed a similar dummy contact approach and developed a set of camouflaging cells. These camouflaging cells can further enhance the security of the design by making it more difficult for attackers to identify the actual functionality of the gates.

3 Reusing Verification Assertions for Security Purposes

This chapter presents a novel approach to improve the security of digital designs by reusing verification assertions, particularly in the context of HT detection. It demonstrates that by transforming existing verification assets, one can create efficient security mechanisms capable of detecting HTs. The process by which assertions are leveraged as online monitors is explained, and a security metric alongside an assertion selection methodology employing the advanced capabilities of the Cadence JasperGold Security Path Verification (SPV) tool [139] is introduced.

Moreover, this chapter presents a comprehensive analysis of experimental outcomes, by applying over 100 assertions to a diverse collection of IPs within the OpenTitan SoC [140]. This demonstrates the presented method's adaptability and scalability to circuits of industry-relevant sizes. The chapter concludes by proving the practicality of this detection solution, emphasizing its independence from the specific activation mechanisms of HTs, thereby offering an adaptable security enhancement to digital designs [73].

As discussed in Section 2.3.1, several approaches exist to facilitate HT detection, with online monitors being the most common technique. Online monitoring techniques rely on embedding checker circuits in different locations of the design to catch unwanted behavior. One approach for building these checker circuits is using assertions, which describe the expected behavior of the circuit and help detect deviations between the intent and actual behavior. However, although online monitoring techniques offer high detection coverage, they impose significant overheads on the circuit. Recent efforts have aimed to decrease these overheads while maintaining maximum detection coverage, but the trade-off remains unfavorable [141].

In this chapter, I propose a methodology for selecting and reusing assertions written by verification engineers for functional verification purposes to achieve security goals, such as HT detection. This approach leverages existing design knowledge, which is often underutilized after the verification process. A new metric called **Security Coverage (SC)** is presented to evaluate the efficiency of online checkers in detecting HTs while considering the imposed overhead on the circuit. This metric helps automate the removal of assertions that are not helpful for HT detection and eliminates the need for detailed knowledge about the design for the engineer who is responsible for ensuring the security of the circuit.

3.1 Assertions as Hardware Trojan Detectors

This section explores the possibility of reusing assertions to detect HTs. To investigate this, the `B19-T500` benchmark from Trust-Hub [142] is selected, which is a Trojan-inserted version of the `B19` circuit from the `ITC'99` benchmark suite [143]. The Trust-Hub benchmarks, with their small sizes and rare triggering conditions, provide a suitable starting point for validating the effectiveness of HT detection schemes [142]. Consequently, these HTs remain hidden during standard verification checks [144]. In my initial study, I utilize these benchmarks to explain and evaluate my proposed approach. However, the goal is to extend this concept and apply it to more complex and realistic circuits.

The ideal characteristics for an assertion to be considered an effective security checker are:

1. It has a minimal overhead on the circuit once synthesized, as many assertions may be required in complex designs for high detection coverage.

2. It has a broad scope that captures high-level behavior, rather than focusing on local signals.

The assertions that satisfy the above conditions are referred to as **top-level assertions**. To illustrate this concept, a set of assertions that satisfy these conditions have been manually written for detecting the HTs inside `B19_T500`. The `B19` benchmark consists of four copies of the `Viper` processor, with an HT circuit embedded within each processor. The HT is triggered by a counter that counts specific vectors and resets with other specific input vectors. If the counter value falls between `3'b100` and `3'b110`, the HT is activated. The payload of the HT manipulates the bits of the Instruction Register (IR) of the embedded `Viper` processor, thereby altering the functionality of the circuit [142].

Although the easiest way to detect this embedded HT would be to write assertions to check the IR bits directly, this approach is not practical for two reasons: First, in a realistic scenario, the HT locations are unknown. Second, this style of assertion writing does not describe any system-level behavior, violating the second condition of being a good security checker. Moreover, as the defender is unable to anticipate the specific trigger for an HT, writing an assertion such as "`(IR == 3'b110 => alert)`" is not a practical solution.

Table 1 presents the top-level assertions considered for detecting HTs in the `B19_T500` circuit, written in PSL. These assertions check the correctness of transactions between the memory and processor by comparing the contents of the IR with the Store Operation (OP_STORE) and the Read Operation (OP_READ) instructions. The first two assertions presented in Table 1 check for invalid write operations in the memory, while the subsequent ones perform similar checks for read operations. Considering the first assertion (ASR_1), if the IR does not contain the OP_STORE operation, then the write signal (wr) must not be asserted. Similar checks are performed using the other assertions presented in the table. For more information on the memory access mechanism in the `Viper` processor, the reader is referred to its documentation [145]. Simulation results demonstrate that these assertions can effectively detect the HT inserted in the `B19_T500` benchmark, with further discussion on their effectiveness provided in Section 3.6.

Table 1: Considered assertions for detecting HTs on B19-T500 *benchmark*

Name	Assertion definition
ASR_1	`assert always {(!(IR == OP_STORE)) -> (!wr)};`
ASR_2	`assert always {(IR == OP_STORE) -> (wr)};`
ASR_3	`assert always {(!(IR == OP_READ)) -> (!rd)};`
ASR_4	`assert always {(IR == OP_READ) -> (rd)};`

3.2 Binding the Assertions to the Main Design

Simulation provides valuable insights into the incorrect behavior of a circuit and its internal values. However, it does not offer sufficient information about the design's PPA characteristics. Consequently, it is impossible to assess the quality of the assertions using simulations alone. To obtain accurate PPA reports, the design must be synthesized.

As mentioned before, PSL and SystemVerilog are the most popular languages for describing assertions, but they are not directly synthesizable. To address this issue, the MBAC tool [146] is used to convert PSL and SystemVerilog assertions into a synthesizable Verilog format. This allows the PPA results to be obtained after synthesis.

Once the synthesizable code is generated from the assertions, it can be bound to the main circuit to evaluate the effectiveness of the assertions based on the overheads imposed on the circuit. For this purpose, first, the main circuit without the assertions is synthesized,

and reports on the maximum clock frequency, power, and area are obtained. Then, the original circuit with the bound assertions is synthesized, effectively turning the assertion into an embedded online checker. Finally, the results of the two syntheses are compared to evaluate the overheads of the assertions.

3.3 Security Coverage

Despite having information about the PPA results for each assertion, it is not possible to make a definitive decision regarding their effectiveness. While the exact cost of these assertions is known, there is a lack of knowledge about their success in HT detection. Therefore, a new evaluation scheme is needed to balance the costs and the benefits. In this context, I propose a new metric for assessing assertions based on security properties.

To achieve this, every node n in the design is categorized into either the set of covered nodes C or the set of vulnerable nodes V. Initially, all the nodes are considered to be vulnerable. As indicated in Equation 1, in order to consider a node as covered, there must be a functional path between the node and the output of any assertion. Therefore, merely having a connection between the node and the output of the assertion is not enough, and conventional methods, such as extracting the input cone(s), are not applicable in this case.

$$n \in \begin{cases} C & \text{if a functional path exists between } n \text{ and any } assertion_k \\ V & \text{if no functional path exists between } n \text{ and every } assertion_k \end{cases} \quad (1)$$

Where k denotes the identifier number of the assertion. Therefore, the SC metric for a design *Des* is defined as follows:

$$SC_{(Des)} = \frac{|C|}{|C| + |V|} \quad (2)$$

Where $|C|$ and $|V|$ are the number of the covered and vulnerable nodes, respectively. In other words, The SC is defined as the ratio of nodes covered by the assertion to the total nodes within the design.

Figure 12 shows an example of a design with two integrated assertions, namely Assr_1 and Assr_2. The assertion circuit, which includes multiple logic gates upon synthesis and integration with the original design, is depicted as a single rectangle in Figure 12 for the sake of simplicity. It is important to note that adding the assertion circuit introduces a new Primary Output (PO) to the design, which can be used to verify if the assertion rises. To calculate the SC for the entire design, Equation 2 is applied. This equation can also be used to determine the SC for each individual assertion, allowing for a comparative analysis of their security properties.

In this scenario, the SC for each assertion is computed by finding the number of covered nodes (C). As shown in Figure 12b, nodes 1, 2, 3, 6, 7, 9, 12, 13, and 16 (highlighted in green) have at least one functional path to Assr_1. Similarly, the covered elements for Assr_2 (highlighted in blue) include nodes 2, 3, 4, 5, 7, 8, 10, and 17, as depicted in Figure 12c. Therefore, the SC values for Assr_1 and Assr_2 are 50% and 44.44%, respectively. If both assertions are bound to the design at the same time, the overall SC for the design can be increased to 77.78%.

It is important to note that some nodes may appear in multiple subsets of covered nodes for different assertions, as exemplified by nodes 2, 3, and 7. Conversely, certain nodes may not be covered by any assertions, such as nodes 11, 14, 15, and 18. Furthermore, the proposed approach is not limited to combinational designs, as sequential elements

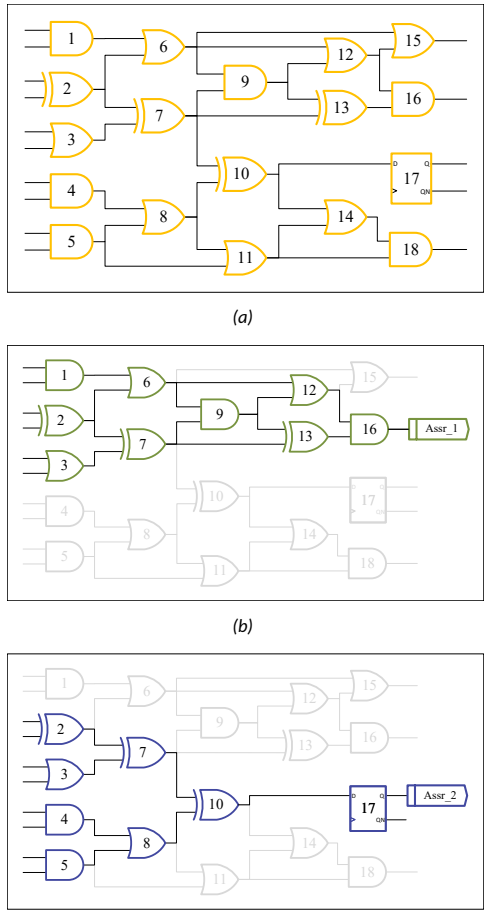

Figure 12: An example of a) original design, b) nodes covered by bound assertion Assr_1, and c) nodes covered by bound assertion Assr_2 (from [88])

such as flip flops (node 17 in Figure 12) are also taken into account as covered elements if there is a path between them and the assertion.

Hence, if any node in the design is the location of the payload of an HT, an assertion that can be reached by that node can detect the malicious logic. A higher number of nodes reachable from the original circuit to the assertion output indicates better coverage in HT detection for that assertion.

To obtain SC for each assertion, the circuit is first synthesized with the assertion bound to it. Then, all the nodes in the synthesized netlist are extracted using a tool that generates a list containing all nodes inside the netlist for further analysis. The generated list is then submitted to the SPV tool. This tool is primarily used for taint analysis [139], checking if design parts are securely isolated. With modifications, this tool can be used to calculate the SC by creating a list of node pairs (origin, destination), where all circuit nodes are possible origin nodes, and the destination node is the output of an assertion. To check the existence of functional paths between pairs, the SPV tool creates properties for each pair and attempts to prove the absence of a functional path or provide counterexamples. Once

the analysis is completed, SC is calculated using Equation 2.

With the SC information, the effectiveness of assertions can be evaluated in terms of security. This metric can be used to perform trade-off analysis to help users decide which assertions best suit their needs. It is important to note that not all circuits require 100% SC. For instance, if certain sensitive parts of the circuit have already been identified and only those need to be secured, covering those parts may be sufficient to meet user requirements.

3.4 OpenTitan - A Case Study

In Section 3.1, the use of custom assertions as security checkers for detecting HTs was studied. However, writing top-level assertions for security purposes is time-consuming and challenging to generalize. The main contribution of the approach presented in this chapter is to ***reuse existing verification assertions*** as security checkers.

For this study, I used OpenTitan, an open-source project featuring an embedded RISC-V-based processor and IPs from various vendors. The project comes with functional assertions for different IPs, making it an ideal candidate for evaluating the suitability of these assertions as security checkers. Obtaining realistic designs and verification assets from the industry is often challenging, so OpenTitan stands out as a valuable resource for this research.

I selected the Register Top modules of each IP, which control transactions between the IP and the bus, grant access to read/write requests for IP registers, and have a unique error generation mechanism for invalid addresses. Since the Register Top modules of different IPs share the same assertions, they provide a good comparison among experiments. A set of selected assertions is shown in Table 2, with a total of 108 different assertions studied on 35 individual IPs of the OpenTitan SoC. A brief description of each assertion is presented as follows:

- wePulse: This assertion ensures that once the `reg_we` signal goes high (indicating a write enable condition), it must go low in the subsequent clock cycle. This can be used to validate that `reg_we` is only briefly asserted for a single clock cycle, ensuring that the write enable signal is properly managed and does not stay high across multiple cycles.

- rePulse: Similar to the previous assertion, this assertion ensures that when the `reg_re` signal is asserted (indicating a read enable condition), it must be deasserted in the following clock cycle. This verifies that `reg_re` is only briefly activated for a single clock cycle, ensuring proper management of the read enable signal and preventing it from remaining high across multiple cycles

- reAfterRV: This assertion ensures that whenever there is a rising edge on either the read enable (`reg_re`) or write enable (`reg_we`) signals, the signal `tl_o` should be asserted in the subsequent clock cycle. This helps in verifying that the `tl_o` signal is correctly managed and activated following an enable condition (either read or write).

- en2addrHit: This assertion ensures that whenever either the write enable (`reg_we`) or read enable (`reg_re`) signals are active, the `addr_hit` signal must be one-hot encoded. This helps in verifying that the address hit signal is correctly managed and represents a valid single address hit when either read or write operations are enabled.

To obtain the SC of each assertion, the same flow as explained in the previous sections is used: MBAC translation followed by assertion binding and synthesis. The nodes from the synthesized netlist are then fed into the SPV tool to calculate the SC.

Table 2: Considered assertions for Register Top modules of different IPs in OpenTitan SoC

Name	Assertion definition
wePulse	`assert property (@(posedge clk_i) disable iff ((!rst_ni) !== 1'b0) $rose(reg_we) |=> !(reg_we));`
rePulse	`assert property (@(posedge clk_i) disable iff ((!rst_ni) !== 1'b0) $rose(reg_re) |=> !(reg_re));`
reAfterRv	`assert property (@(posedge clk_i) disable iff ((!rst_ni) !== 1'b0) $rose(reg_re || reg_we) |=> tl_o);`
en2addrHit	`assert property (@(posedge clk_i) disable iff ((!rst_ni) !== 1'b0) ((reg_we || reg_re) $onehot0(addr_hit)));`

3.5 Optimizing the Assertion List

Manually checking assertions to determine if they are top-level is a time-consuming process, questioning the efficiency of the proposed approach. To address this, a methodology is presented to help users select efficient security checkers from available assertions based on their needs. This step is necessary since not all functional assertions are useful for security purposes.

Figure 13 presents an automated flow for the assertion selection process. The first step in this process involves selecting an assertion from a candidate list, which consists of assertions that can be synthesized. Assertions with a simulation-based nature that cannot be synthesized will be filtered out at this stage. The selected assertion is then converted into synthesizable logic and integrated into the design. The prepared design is used for overhead evaluation, where synthesis is performed with the assertion, and PPA metrics are compared against the values obtained from the original design.

After evaluating the overheads and ensuring they meet user-defined requirements, the next step is to calculate the SC. This is accomplished by using Equation 2 and employing the SPV tool. If the assertion passes the overhead evaluation and achieves the desired SC, it is added to the final list of assertions. However, if the assertion fails to meet the criteria (e.g., unacceptable overheads or insufficient SC), an alternative assertion from the candidate list is considered (if available). The decision in this step can be based on either the individual SC of the assertion or the overall SC threshold specified by the user for the entire design. Once the flow is completed, the generated netlist with the embedded assertion(s) is considered finalized.

As a case study, the `alert_handler` IP from the OpenTitan SoC is selected, which contains several assertions, and I demonstrate how to create a list of security checkers from these assertions using the proposed methodology. In the first step, a candidate list of 13 different assertions is created, all predefined by OpenTitan developers to ensure the design's functionality remains as intended.

Since the final security checker list is based on user needs, two different strategies are defined for selecting appropriate candidates: **fixed-threshold** and **dynamic-threshold**. It is important to note that defining strategies is completely flexible and depends on the desired level of security and acceptable PPA overheads. In the following, the proposed optimization flow for the defined strategies is explained.

Fixed-Threshold Strategy: In this strategy, a fixed threshold margin in terms of SC is set

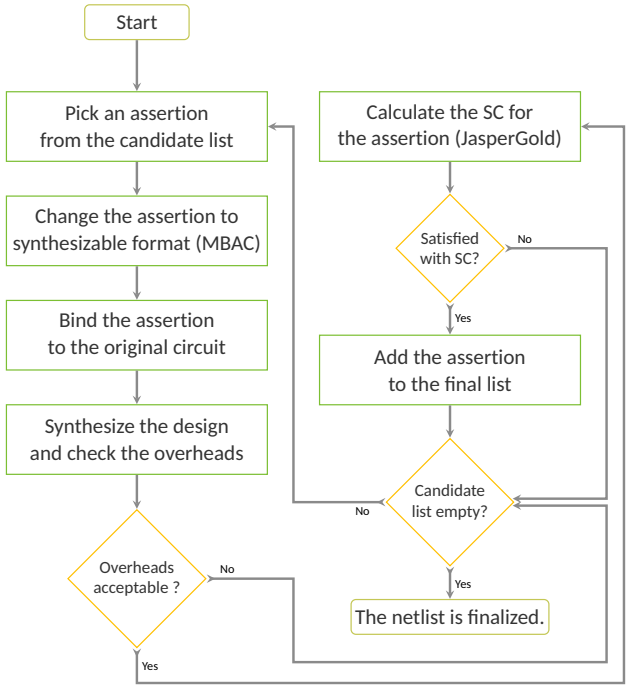

Figure 13: Optimization flow for selecting the assertions to be used as security checkers (adopted from [73])

for the overheads. If the overheads of a selected assertion exceed the defined threshold, it is immediately discarded, and another assertion is picked from the candidate list. This process continues until all assertions in the candidate list are evaluated. For example, if the performance overhead for a given assertion is X (percent), the SC should be at least 10X (percent). It should be noted that the number 10 is a configurable parameter that can be set by the user based on their specific requirements and the characteristics of the design under analysis. In this study, it has been empirically observed that this value works well for achieving a balance between the SC and the introduced overheads.

Dynamic-Threshold Strategy: In this strategy, the threshold margin for overheads is first adjusted based on the average overheads. A similar threshold is then set for the SC, taking into account the impact of each assertion on the overall SC. Consequently, each assertion is evaluated dynamically by considering both its overheads and its impact on the overall overheads and SC. For instance, the maximum PPA overhead for a given assertion should not exceed twice the average PPA overhead. For SC, only assertions that have a positive impact on the overall SC of the circuit compared to other assertions are chosen.

Unlike the previous strategy, where assertions are assessed individually, the dynamic strategy performs comparisons between competing assertions. Since looking at SC results for individual assertions does not provide information about their positive impact, this strategy selects only assertions that perform better than average.

The first strategy is simple and easy to implement but requires the user to define a constant (i.e., 10) for the threshold. The second strategy does not require such a constant but needs a sufficient number of assertions to define average overhead and coverage. The next section shows how the dynamic strategy can be more effective than its fixed-threshold counterpart. However, more complex strategies can be defined, which can be explored in

future work.

3.6 Experimental Results

This section presents the experimental results, encompassing PPA overheads and SC values achieved for various designs, as detailed in earlier sections. Cadence Genus, with the target cell library being a commercial 65nm CMOS library, is employed for all the experiments conducted and reported here.

Figure 14 illustrates the normalized PPA overheads for the considered assertions in the `B19-T500` benchmark. As depicted in this figure, three assertions (*ASR_1*, *ASR_2*, and *ASR_4*) have zero area overhead. The highest overheads are observed in *ASR_2* and *ASR_1*, which cause the circuit to consume 9% more power when bound. Furthermore, the timing overhead for all assertions is below 6%. It is important to note that, according to the heuristic, normalized numbers lower than one fall within the noise margin and do not impact performance

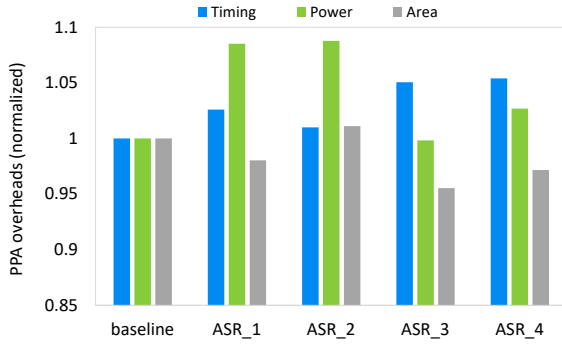

Figure 14: PPA overheads imposed by different assertions on the B19-T500 benchmark from Trust-Hub (from [73])

Table 3 presents the SC calculated for the same assertions in the `B19-T500` benchmark. The second and third columns present the total number of nodes and the number of covered nodes for a given assertion bound to the design, respectively. The fourth column indicates the percentage of the SC. Since the total nodes encompass both the nodes in the original design and those associated with the assertion, the total node count varies for each assertion.

As shown, proposed assertions cover an average of 6.8% of the total nodes in the circuit. This means that they can detect HTs within their covered areas, regardless of how rarely the HTs are triggered and what impacts they might have on the circuit. This is one of the primary advantages of the presented method, as the user no longer needs to worry about activating rare HTS.

Figure 15 illustrates the SC obtained for different IP Register Top modules of the OpenTitan SoC. The highest SC is 4.77% for the `nmi_gen_reg_top` module. However, the majority of IPs have a SC value below 1%, which does not make them good candidates for being security checkers. This is primarily because these assertions only perform small interface checks and do not describe the top-level behavior of the circuit. Instead, they cover only some local nodes, resulting in low SC for the entire circuit. This highlights the importance of the optimization step in the proposed methodology to avoid selecting unnecessary assertions that have minimal impact on HT detection.

Table 3: SC for the synthesized assertions bound to B19−T500 benchmark

Assertion name	**Total nodes**	**Covered nodes**	**Security Coverage** (%)
ASR_1	5014	315	6.28 (%)
ASR_2	5062	304	6.01 (%)
ASR_3	4916	367	7.47 (%)
ASR_4	4944	369	7.46 (%)
Average	**4984**	**339**	**6.80** (%)

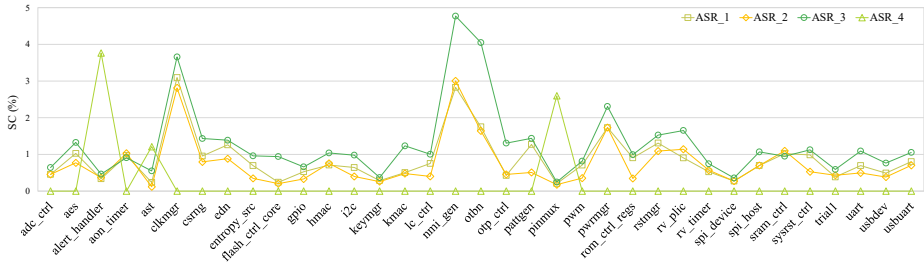

Figure 15: SC percentage for the Register Top modules of OpenTitan IPs (from [88])

3.6.1 Optimizing the Assertions

This section presents a practical experiment using the optimization flow shown in Figure 13. Previously, two strategies for selecting appropriate assertions were defined in Section 3.5, and now more details are provided about the assertion selection procedure.

Fixed-threshold strategy: The PPA overhead results for the assertion candidate list of `alert_handler` IP are shown in Figure 16a. As shown in this figure, the maximum overhead is due to the timing degradation of the `ah_asr_8` assertion (2.99%), while the minimum overhead is from the `ah_asr_3` and `ah_asr_4` assertions with a value of 0.75%. The next step is to check the SC, which should be at least ten times higher than the maximum overhead for each assertion. Figure 16b shows the SC results obtained from each assertion using the SPV tool. For clarity, numbers are associated with the assertion names. Based on these results, `ah_asr_12` and `ah_asr_13` can be disregarded since they do not meet the required SC condition, and the remaining candidates are considered as the final security checkers. Although 15% of the assertions were removed based on this strategy, defining smarter strategies can improve the effectiveness of the final list. Therefore, a second strategy (dynamic threshold) on the same candidate list is defined to achieve better efficiency.

Dynamic-threshold strategy: For the first condition of this strategy, the average overhead for all assertions is calculated to be 1.79%. Consequently, all candidates pass this condition since they have less than twice the average overhead in all cases (Figure 16a). However, for the second condition, referring to the SC results alone is insufficient, as they do not provide a basis for comparison. Therefore, an additional step is required to select the security checkers.

For this purpose, the assertions are arranged in descending order of SC, starting from the highest (`ah_asr_11`) to the lowest (`ah_asr_12`). Beginning with `ah_asr_11`, the assertion with the next highest number (`ah_asr_10` in the first round) is added, and the SC for the newly formed set of assertions is calculated. This process is repeated until the

lowest number is added to the list.

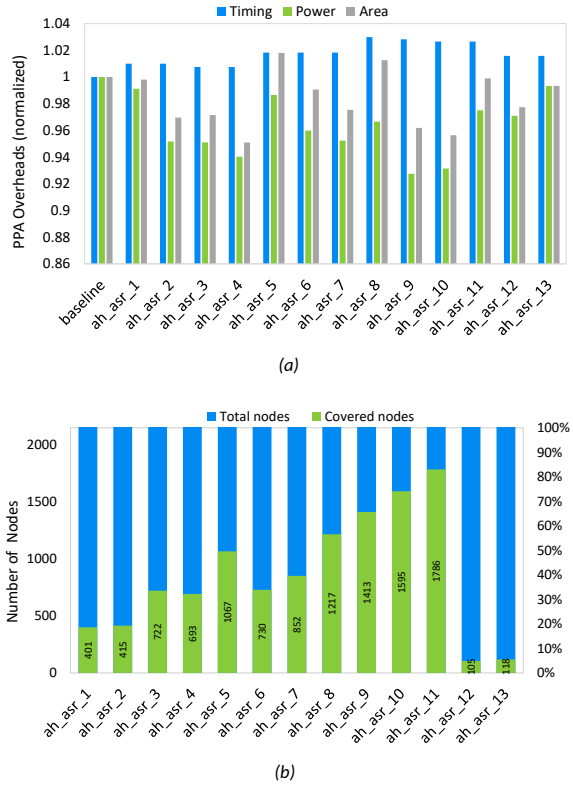

Figure 16: The figures for a) PPA overheads imposed by different assertions, b) Number of covered nodes for the individual assertion of `alert_handler` IP (from [73])

Figure 17 shows the SC numbers for each set of assertions. The assertion numbers are included in the set names to identify the effect of the added assertion in each step. For example, `ah_asr_11_10` represents a set starting from `ah_asr_11` (the highest coverage) and ending with `ah_asr_10` (the last assertion added), while `ah_asr_11_9` includes `ah_asr_11`, `ah_asr_10`, and `ah_asr_9`. A moving average trend-line is added to the figure to aid in selecting the best assertions. Since the moving average trend-line has a period of 2, it enables a good comparison between the SC of the newly added assertion in each stage and the two previous assertions. If the SC obtained after adding an assertion crosses the moving average trend, it indicates a noticeable difference.

Referring to the second condition of Dynamic-threshold Strategy and Figure 17, only three assertions have SC numbers that cross the moving average trend-line (`ah_asr_5`, `ah_asr_7`, and `ah_asr_13`), and they can be added to the final list. Additionally, the `ah_asr_11` assertion is added to the final list due to having the highest SC.

Compared to the SC numbers of different Register Top modules of various IPs (Figure 15), the SC numbers of different assertions in the `Alert Handler` IP are relatively higher (Figure 16b). This is primarily because the assertions written for this specific IP describe a top-level behavior of the design, rather than checking only local signals and interfaces.

These two examples demonstrate that different strategies can be defined based on user needs, making the presented approach flexible. Furthermore, one of the advantages of

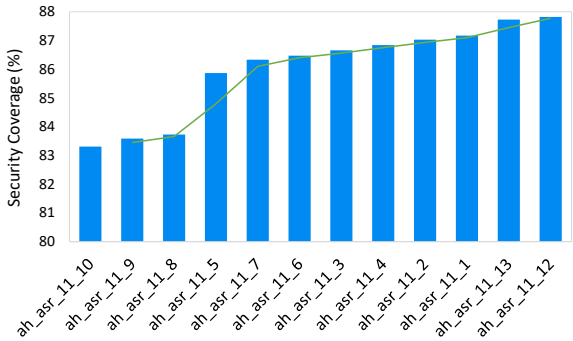

Figure 17: SC percentage for different assertion sets of `Alert Handler` IP (from [73])

this work compared to current approaches is its simplicity and lack of complex procedures. For instance, the work presented in [141] supports HT detection with flexible overheads, but it requires substantial effort and complicated steps. In contrast, my approach uses commercial tools that are widely available, increasing the portability and scalability of the presented work.

4 Enhancing IC Security by Embedding Online Checkers during Physical Synthesis

This chapter presents a comprehensive approach to enhance IC security throughout the design process in the back-end stage. In the previous chapter, a systematic method for converting existing verification assets into effective security checkers was introduced by repurposing verification assertions at the front-end phase of IC design [73]. To further enhance security, I propose a novel technique for incorporating online monitors during physical synthesis, offering an additional layer of protection at the back-end phase [88].

While integrating online monitors is not a new concept, most previous works have focused on introducing these checkers during the front-end phase of design. In contrast, this work takes a different approach by directly incorporating the checkers into the layout during the back-end phase, while still taking into account the front-end inserted assertions. This strategy allows for a more comprehensive security solution that spans both front-end and back-end design phases [88]. Integrating security checkers in the back-end design phase provides unique benefits, as the design is close to its final form at this stage. One such advantage is the ability to achieve more efficient area utilization, as the precise location of each design element is determined during the physical synthesis process.

Although this back-end methodology can be considered as a complementary approach to the front-end method, both techniques can be employed independently, depending on user preferences and specific requirements. This flexibility allows for a more customizable and adaptable solution to enhance IC security.

As deeply discussed in Section 2, the IC production process consists of multiple stages, as depicted in Figure 18. Front-end engineers convert the high-level design description into a gate-level netlist through logic synthesis. This netlist is then passed to the back-end team, where engineers modify it according to specific constraints such as area, power, and timing. The resulting layout is sent to a foundry for fabrication. Upon receiving the chip, specific tests are performed to verify its functionality.

As shown in Figure 18, the assumption of this work is that the foundry is considered an untrusted facility where an adversary (e.g., rogue engineer) may be present. The design and test stages, including the design house and test house, are assumed to be trusted. Defensive techniques are implemented in the design by front-end and back-end teams before sending it for fabrication, aiming to counter fabrication-time attacks.

It is also assumed that the attacker within the foundry has the capability to insert sophisticated, small, and rarely activated HTs that can evade side-channel analysis, logic testing, and simple forms of chip inspection such as electrical testing. The attacker has access to the target technology's PDK and advanced commercial CAD tools. This work focuses on functional HTs that alter the chip's functionality, allowing their effects to be observed by comparing internal signals with the expected ones.

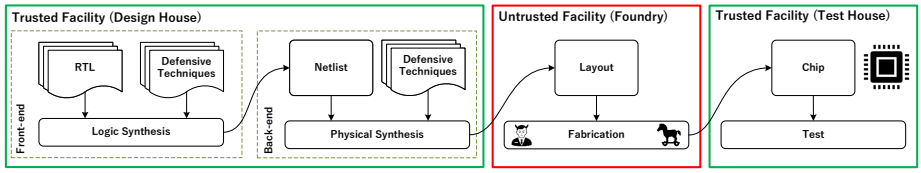

Figure 18: Different stages of IC design: The design house and the test house are considered trusted, while the foundry is assumed to be untrusted.

4.1 Limitations of the Concept of Reusing Verification Assertions as Security Checkers

The primary focus of the work presented in Chapter 3 is introducing of a new metric (i.e., SC) for assessing the security properties of the generated security checkers. However, it is crucial to acknowledge that while achieving higher SC numbers for various assertions might suggest improved design security, ensuring design security involves more than just relying on SC metrics, even when they are in high ranges (e.g., exceeding 80%). There are several challenges associated with SC, and some significant ones include:

1. **Ineffectiveness of verification assertions at runtime:** Functional assertions can cover various security properties by ensuring the expected behavior of the circuit. However, they may not be precise enough to thoroughly cover the negative or unexpected behavior of a circuit under attack. Although the SPV tool is used to calculate taint propagation coverage, the effectiveness of detecting HT behavior by the synthesized assertions at runtime is not guaranteed.

2. **Scalability concerns:** The requirement to "bind the assertion" and synthesize the entire design for characterizing overheads may present scalability challenges for larger designs. The resource-intensive nature of this process could restrict its practicality for more extensive and complex circuits, making it difficult to efficiently apply this approach to large-scale projects.

3. **Assertion availability:** The approach focuses on reusing existing assertions instead of generating new ones. However, a potential challenge arises when no suitable assertion with acceptable SC is found for a given design. This situation was illustrated by the varying SC numbers obtained for different assertions in the design in Section 3.6. In such cases, alternative security measures may need to be considered to ensure adequate protection.

To address the aforementioned challenges and limitations, it is crucial to integrate an additional layer of security into the design. However, it is important to note that this additional layer does not necessitate significant changes to the existing circuit design and fabrication processes. Instead, the goal is to develop a security solution that can be seamlessly incorporated within the current design flows, ensuring both practicality and effectiveness in enhancing the overall security of the system. This supplementary measure can help enhance the overall security of the system and provide more comprehensive protection against potential threats.

4.2 Adding Online Monitors during Physical Synthesis

Figure 19 presents a simplified view of the layout of a block within an IC. The green polygons represent standard cells, which are later interconnected through various metal layers to establish the logical function of the design. Due to fabrication complexities, especially in modern process nodes, achieving 100% density with a layout entirely filled with standard cells is impractical. Consequently, gaps are present in the layout, highlighted in red in Figure 19. These gaps can potentially be exploited by an adversary for inserting malicious logic (i.e., HTs) [20, 87, 130, 147].

Although these gaps are typically filled with filler cells or decap cells before being sent to fabrication, these cells lack functionality and are not connected to the design's logic. In the case of decap cells, they can be removed, but this may have a slight impact on the overall design. The proposed approach takes advantage of these gaps and available resources to

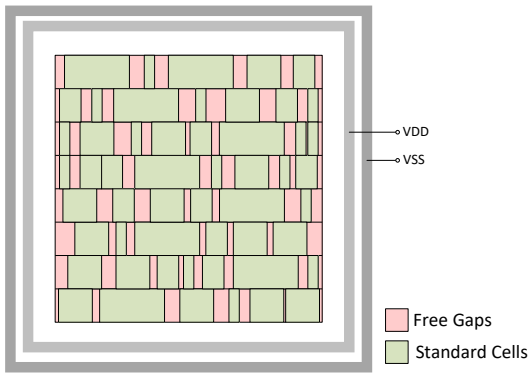

Figure 19: Illustrative example of a block layout within a chip (from [88])

insert online checkers into the design. Doing this not only adds an extra security layer to the design but also limits the adversary's ability to insert malicious logic by increasing the density and congestion in the layout.

While the presented method can be employed independently, it is used as a complementary approach alongside reusing assertions for detecting HTs to address its limitations. It is important to note that this technique leverages the Engineering Change Order (ECO) capabilities of the CAD tools for inserting online monitors into the layout. ECO features enable engineers to make last-minute modifications to the existing layout, such as adding or removing components and changing connections. By utilizing ECO features, alterations to the original layout with each added online monitor are minimized, which provides optimal overheads compared to front-end approaches. This makes the presented approach enhance security while maintaining design efficiency.

4.2.1 Generation of Online Monitors for the Back-end Phase

The online monitors provide an additional layer of protection for nodes that are not covered by assertions. To implement this protection strategy, a Dual Modular Redundancy (DMR) scheme is used, as depicted in Figure 20. In the left image (Figure 20a), a segment of a design is shown, where the covered nodes (10 and 17) are highlighted in green, and the vulnerable or uncovered nodes (11, 14, and 18) are highlighted in red. To create an online monitor for this design, an exact duplicate of the uncovered gates, with the same equivalent gates from the library (i.e., the same gate type and drive strength), is first generated. Then, the output of the duplicated part is compared with the output of the original part by XORing these two signals, as illustrated in Figure 20b. Consequently, implementing an online monitor for this segment results in seven new covered nodes (11, D11, 14, D14, 18, D18, and V18) being added to the previously covered nodes (10 and 17). In this figure, the duplicated nodes are labeled with the prefix "D", while the voter nodes are labeled with the prefix "V". For instance, node D11 is the duplicated version of node 11, and V18 is the voter which compares the output of node 18 with D18.

It is important to note that the output of node V18 is utilized only for security purposes and has a minimal impact on the performance of the design. This is because it only adds some wire capacitance to node 18, which is a negligible effect in most cases.

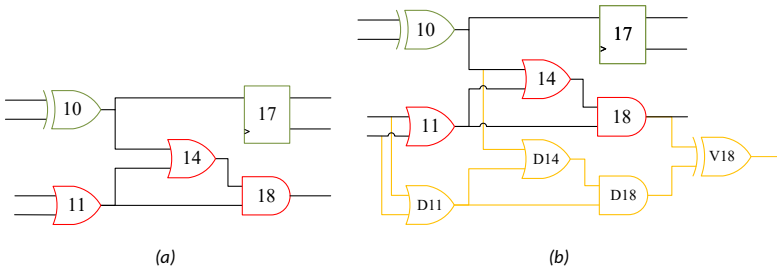

Figure 20: An example of a) design before adding online monitors, and b) design with the protection logic (D11, D14, and D18 as the duplicates and V18 as the voter) to protect the uncovered gates (11, 14, and 18)

4.2.2 Embedding Online Monitors into the Layout

The complete flow for incorporating online monitors during physical synthesis is illustrated in Figure 21. It consists of four primary steps, which are explained below:

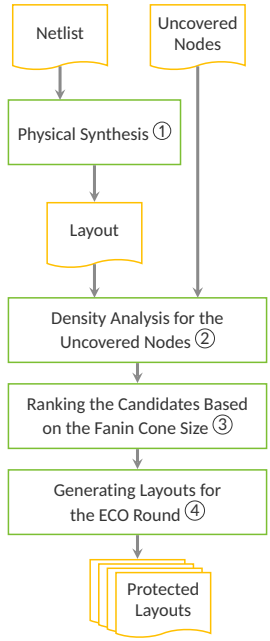

Figure 21: An overall flow of integrating online monitors during physical synthesis (from [88])

1) Physical Synthesis: The initial step in the comprehensive flow involves using a physical synthesis tool to convert the netlist into a layout. This netlist includes the original design and all specified assertions associated with it. The process consists of several stages such as placement, clock tree synthesis, and routing. Physical synthesis provides important details such as the precise placement of standard cells, the physical arrangement of the clock, and the structure of interconnections concerning wire length and the utilization of each available metal layer. In essence, physical synthesis provides insights into the spatial configuration of the design. It is important to note that this time-consuming step is only performed once for each design, making the proposed approach compatible with

industry-standard flows without requiring any modifications to them.

2) Density Analysis for the Uncovered Nodes: The resulting layout, combined with the report of uncovered nodes from the SPV tool, serves as the input to a developed analytical tool. The goal here is to identify uncovered nodes with available gaps around them, which can accommodate the online checker responsible for protecting the respective node. If there is no space available around the uncovered node, the online checker might be placed at a distance, leading to increased resource utilization and degradation of the PPA parameters of the design. This step is critical in the proposed flow to prevent such situations, minimizing layout modifications, ensuring compatibility with the ECO flow, and reducing overheads. In contrast to front-end protection schemes that only provide estimated overheads, this approach offers a distinct advantage by determining the actual overheads accurately. It is important to note that the radius of searching for an area around each node is adjustable and can be changed based on the design size and density.

3) Ranking the Candidates Based on the Fanin Cone Size: The candidates generated by the density analysis tool are subjected to a ranking process. As previously mentioned, the preference is to place the online checker for a group of interconnected uncovered nodes rather than individual gates. This choice provides benefits in terms of area, power, and routing resources. Consequently, this ranking system prioritizes subsets of candidate gates with larger input cones, optimizing the overall efficiency of the protection scheme. It is important to note that during this cone analysis, only candidates with a cone size[1] of 2 or greater are considered to enhance the efficiency of the proposed approach.

4) Generating Layouts for the ECO Round: In the final step, protected layouts are generated. To accomplish this, a tool is developed that takes the ranked list of candidates (generated in the previous step) and incrementally integrates the online checkers into the layout. Specifically, one online monitor is added to the design at a time, and a new layout containing the added monitor is generated. This process is repeated from the top of the ranked list to the end. Consequently, if there are *n* candidates (the nodes suitable for protection by online checkers) in the ranked list, *n* different layout files are created in an iterative manner. Each protected layout file, such as *Layout1*, contains the online monitor from *Candidate1*; *Layout2* contains the online monitors for *Candidate1* and *Candidate2*, and so on. It is important to note that these protected layouts are generated to be used along with the ECO flow, and the finalized layout, which includes all potential online checkers and is intended for fabrication, is obtained after the completion of the entire ECO rounds.

Since each new protected layout only adds one online monitor compared to the previous one, the user has the option to keep or discard the added online monitor. This incremental method helps to carefully integrate online monitors into the design while efficiently managing the resources needed for each addition. Additionally, it allows for a detailed evaluation of the impact on the PPA parameters of the design.

4.2.3 ECO Flow

As described, the proposed approach efficiently inserts online checkers and generates protected layouts, prioritizing area and resource utilization. This approach can be further enhanced by introducing a timing-aware element to the ECO flow. Hence, a new metric is introduced called the Degrading Factor (DF), which is set at 25% of the total positive setup slack of the design before the addition of online checkers. This DF serves as a threshold parameter for deciding whether to keep or discard protected layouts based on their impact on timing.

[1]Cone size refers to the number of gates or logic elements that have a direct or indirect influence on a particular node in the circuit.

To accomplish this, the PPA numbers of the initial layout are first stored before online monitors are integrated. The ECO flow starts by choosing the first protected layout, which includes the highest-ranked online monitor from the cone analysis, and then calculates the PPA numbers for this modified layout. Next, the total setup slack number is compared with the previous layout (the one without the newly added monitor). If the slack number is negative or if the difference between the new slack and the previous one exceeds the DF, the ECO flow rejects the newly added layout as it worsens the timing beyond user constraints and moves on to the next one. This process continues until all online monitors are integrated or there is no more slack available for the new logic. Furthermore, various checks are carried out at each ECO round to ensure compliance with design rules and prevent issues that may arise from implementing the new layout.

4.3 Experimental Results

This section presents the experimental results of integrating online monitors for different IPs of OpenTitan.

Figure 22 presents the calculated SC percentages for individual assertions within three selected IPs. These IPs have been chosen since the SC for each assertion is significantly higher compared to the figures obtained from the initial experiment in Section 3.6. The average SC is 85.83% for the selected assertions in the `alert_handler` (Figure 22a), 46.03% for the selected assertions in the `alert_handler_esc` (Figure 22b), and 38.35% for the selected assertions in the `flash_phy_rd` module (Figure 22c).

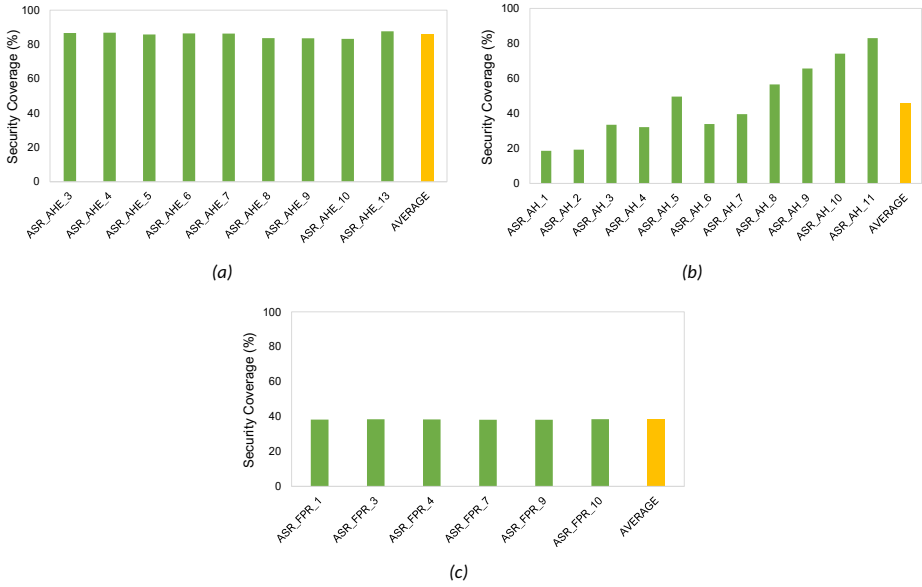

Figure 22: Calculated SC percentage for the different assertions in selected IPs of OpenTitan: a) alert_handler_esc, b) alert_handler, and c) flash_phy_rd (from [88])

To assess the efficiency of integrating online monitors during the back-end phase, five different IPs were chosen from all IPs studied earlier. Two of these IPs were selected for security reasons: `alert_handler_esc_timer` has the highest SC, and `keymgr_reg_top` has the lowest SC. The other three IPs were chosen based on their sizes: `ast_reg_top` is the largest design, `flash_ctrl_core_reg_top` has an average size, while the smallest

one is `nmi_gen_reg_top`. It is important to note that all these IPs are selected among 40 different IPs that already have some assertions to maintain the concept of repurposing existing assertions. This work does not introduce new assertions or modify existing ones across different IPs. The calculated SC for all these IPs is previously shown in Figure 22 and Section 3.6.

For all the results reported in this section, the Cadence suite is used: logical synthesis is performed by Genus, while physical synthesis is carried out by Innovus. The formal tool for performing taint analysis is JasperGold SPV, as mentioned earlier. The target technology node is a commercial 65nm CMOS one.

4.3.1 Impact of Adding Online Monitors on SC

To evaluate the impact of integrating online monitors on the security properties of each design, the same evaluation scheme given by the SC equation presented in Section 3.3 is used. In this equation, the total covered nodes (C) in the numerator now include both the nodes previously covered by the assertion and the newly covered nodes introduced by the online monitor.

Table 4 presents the results regarding the impact of added online monitors on the security of the considered designs. In this table, the first column denotes the IP name, while the second column enumerates the instances in each IP. The third column indicates the SC before the integration of online monitors. These values are obtained by binding all available assertions to each IP and analyzing the coverage using the SPV tool. The fourth and fifth columns represent the increase in SC specifically due to the online monitors and SC after adding the online monitors, respectively. As indicated, the lowest increase in SC is 0.43% for the `alert_handler_esc_timer` IP.

Table 4: The impact of adding online monitors on the security of selected IPs

IP Name	Instances	SC Before	SC Added	SC Total	# NCbM	# AM	# IM	# TM	Preventing Factor
`alert_handler_esc_timer`	1404	87.82%	0.43%	88.25%	6	1	0	1	Density
`ast_reg_top`	7048	2.49%	17.45%	19.94%	1382	183	7	190	Density
`flash_ctrl_core_reg_top`	7048	1.38%	16.6%	17.98%	954	105	264	369	Timing
`keymgr_reg_top`	4611	0.91%	9.98%	10.89%	490	55	86	141	Timing
`nmi_gen_reg_top`	214	3.53%	33.58%	37.11%	92	14	0	14	Density

NCbM: Nodes Covered by Monitors, **AM:** Added Monitors, **IM:** Ignored Monitors, **TM:** Total Monitors

This is mainly because of the IP's existing extensive coverage of nodes, making it difficult to identify suitable candidates that pass all stages of the online monitor insertion flow (as shown in Figure 21). This means finding a group of uncovered connected nodes with adequate space around them is challenging, given the limited total number of uncovered nodes. However, the increased SC does not exclusively signify the added security to each design. The introduced logic for inserting online monitors also occupies the gaps in the layout and utilizes the routing resources that could potentially be exploited by an attacker.

In column 6, the total number of covered nodes is displayed after the inclusion of online monitors. These covered nodes encompass those not covered by the assertions, as well as the new redundant logic added to form the online checker. Columns 7 and 8 show the number of applied and ignored online monitors in the IP, respectively. Column 9 indicates the number of online monitors that can be generated for each design after conducting density and cone analysis (Figure 21). The total number of online monitors equals the number of individual protected layouts produced for the ECO flow. It is important to note that not all generated online monitors can be integrated into the design due to timing constraints.

The final column identifies the factor that limits the addition of more online monitors. If all available online monitors are successfully integrated into the design, it indicates that no additional monitors can be generated, primarily due to the high density around the uncovered nodes. On the other hand, if some online monitors remain unembedded in the design (excluding those exceeding the DF), it is mainly because the design's timing resources have been exhausted, necessitating their exclusion. More details about the PPA restrictions are discussed in the next section.

4.3.2 Impact of Adding Online Monitors on PPA

The baseline layout for each IP is set up such that the design density ranges between 60% and 65%. This configuration provides a positive setup slack of approximately 10% of the clock period[2] for each design. Since online monitors introduce new logic that can affect the design's timing, the 10% margin enables the use of positive slack for integrating these monitors.

In Table 5, a comparison of various PPA metrics before and after the implementation of online monitors for the selected IPs is presented. The first column lists the names of the IPs, while the subsequent two columns provide details regarding the area and placement characteristics of the layouts. The second column displays the total area for each design, and the third column represents the placement density. It is noteworthy that the smallest design, `nmi_gen_reg_top`, exhibited a significant increase in area parameters following the addition of online monitors. This can be attributed to the fact that the size of the added logic became comparable to the overall design, consequently impacting the cell area, which denotes the space on a chip occupied by logic cells.

The fourth column represents the total power consumption for each IP in the study. The IP denoted as `nmi_gen_reg_top` exhibited the largest increase in power consumption due to its small size. The fifth and sixth columns present the timing characteristics of the design. While the hold slack remains relatively constant across all designs, the setup slack undergoes notable changes for most designs, attributed to the impact of redundant logic in different timing paths. The two designs with the preventing factor of timing, `flash_ctrl_core_reg_top` and `keymgr_reg_top`, display the most significant decrease in setup slack.

The last column represents the total wire length for each design. These metal wires are utilized to connect different parts of the design. The increase in the total wire length suggests that the design has become more congested, limiting the free routing resources available to be utilized by an adversary.

To obtain a more comprehensive understanding of the effects of incorporating each online monitor on the design characteristics, I have analyzed various PPA results at the end of each successful ECO round, where a new online monitor is integrated into the design. This thorough examination enables monitoring of the individual impacts on different attributes throughout the iterative ECO process. Figure 23 demonstrates the deterioration of setup slack after each successful ECO round, with the vertical axis representing the total setup slack time in nanoseconds (ranging from the worst to the best slack time) and the horizontal axis depicting the progression of ECO rounds.

As mentioned earlier, the ECO flow begins with the layout containing the assertions, which is labeled as the "Baseline" in Figure 23. As shown in the figure, some rounds have minimal impact on the timing, while others significantly degrade the total setup slack. In certain cases, the slack may even improve due to the heuristics of the physical synthesis

[2]A clock period is the time it takes for a clock signal to complete one full cycle. This is the time between two consecutive rising edges or falling edges of the clock signal.

Table 5: The impact of adding online monitors on the PPA metrics of selected IPs

IP Name	Total Area (μm^2)	Placement Density	Total Power (mW)	Setup Slack (ns)	Hold Slack (ns)	Total Wire Length (μm)
`alert_handler_esc_timer` (before)	3692.88	63.94%	1.46	0.328	0.133	22305.40
`alert_handler_esc_timer` (after)	3700.80	64.07%	1.47	0.328	0.133	22351.75
Difference	+0.21%	+0.20%	+0.68%	0.00%	0.00%	+0.21%
`ast_reg_top` (before)	28644.48	61.46%	9.93	0.287	0.084	353799.80
`ast_reg_top` (after)	30848.76	66.18%	10.26	0.091	0.07	397612.20
Difference	+7.69%	+7.68%	+3.32%	-68.29%	-16.67%	+12.38%
`flash_ctrl_core_reg_top` (before)	14243.40	64.25%	4.94	0.210	0.181	148407.90
`flash_ctrl_core_reg_top` (after)	15542.28	70.11%	5.13	0.007	0.186	170461.90
Difference	+9.12%	+9.12%	+3.85%	-96.67%	+2.76%	+14.86%
`keymgr_reg_top` (before)	18325.80	62.68%	8.11	0.278	0.152	186978.10
`keymgr_reg_top` (after)	19062.72	65.20%	8.29	0.000	0.151	199832.90
Difference	+4.02%	+4.02%	+2.22%	-100%	-0.66%	+6.87%
`nmi_gen_reg_top` (before)	769.68	61.16%	0.23	0.118	0.178	4841.51
`nmi_gen_reg_top` (after)	918.00	72.94%	0.27	0.043	0.179	6077.74
Difference	+19.27%	+19.26%	+17.39%	-63.56%	+0.56%	+25.53%

tool. However, the degradation does not exceed the DF, which is set at 25% of the total setup slack. It is important to note that this parameter can be adjusted based on the user's preferences. For example, if set to lower values, online monitors causing a sudden decrease in the total setup slack (e.g., the online monitor added in round 81 in Figure 23b) will be discarded, resulting in a more smooth overall trend for setup slack decrease.

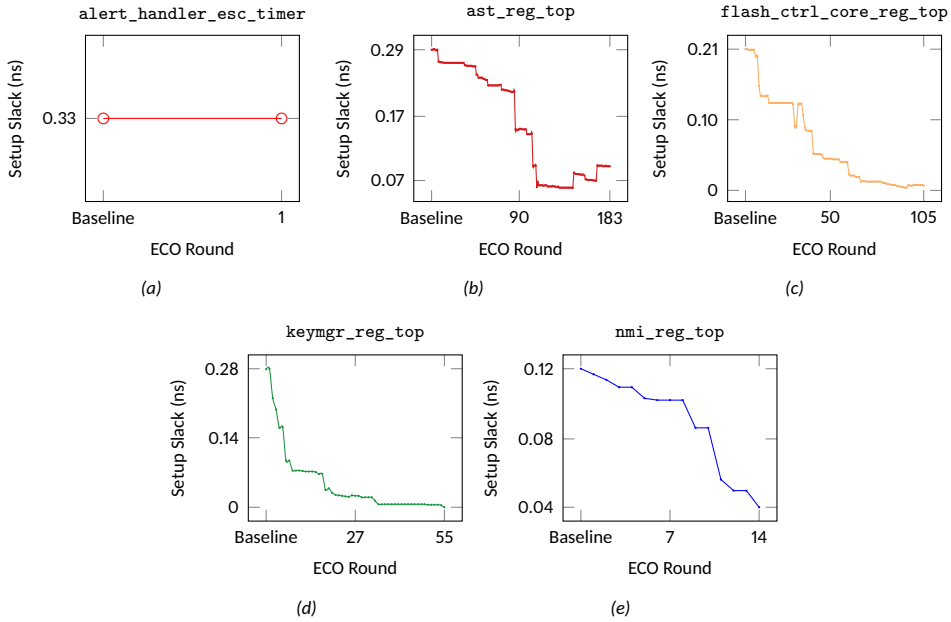

Figure 23: Changes in setup slack after each round of adding the online monitors for different IPs (from[88])

Figure 24 shows the progressive increase in wire length for each metal layer in the protected layouts. The wire length, measured in μm, is depicted on the vertical axis, and the horizontal axis identifies the protected layout for each round. Similar to Figure23, the term *Baseline* denotes the layout that includes assertions but lacks online monitors. Despite the varying number of metal stacks in different target technologies, newer technologies generally offer ten or more metal layers, and a higher utilization of upper metal layers suggests heightened congestion in the design. As a defender, the emphasis is on the greater use of upper metal layers, signifying an overall rise in congestion in the protected layouts. With the exception of `alert_handler_esc_timer`, where only a single online checker was available to be added, a steady trend of increased wire length in metal layers M3-M6 is noticeable for all designs in Figure 24. This trend reduces the available free routing resources for potential adversaries.

Figure 25 illustrates the layout views, showcasing the placement configuration before and after the integration of online monitors. Each row presents a pair of images, where the left image represents the cell placement of the layout before integrating online monitors, and the right image corresponds to the final protected layout after successfully completing all ECO rounds.

Similarly, Figure 26 showcases the layout views, including the routed view before and after the integration of online monitors. Each row also presents a pair of images, with the left image in each pair representing the routed view of the layout before integrating online

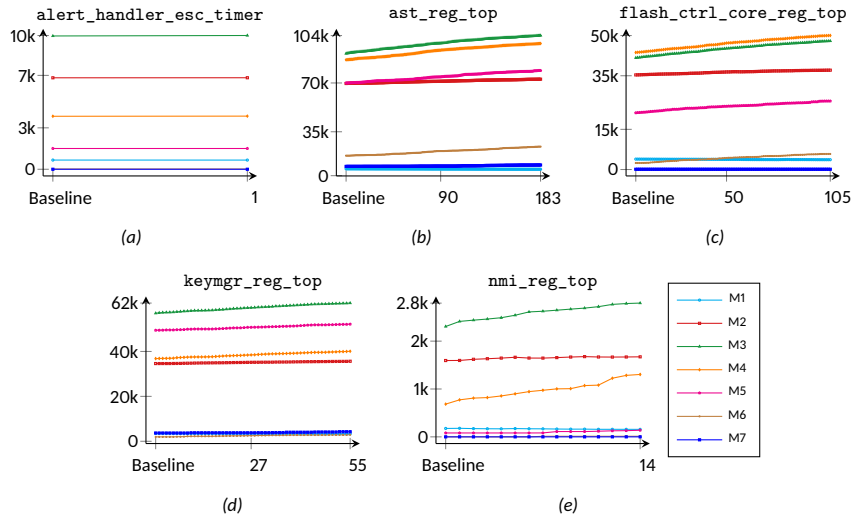

Figure 24: The impact of adding online monitors on the length of different metal layers in the protected layouts for different IPs (from [88])

monitors, and the right image corresponding to the final protected layout after successfully completing all ECO rounds.

Comparing the layout images on the right with those on the left in Figure 25 and Figure 26, reveals that the overall placement and routing configuration of the layouts remained unchanged, even for larger designs. This underscores the more efficient utilization of resources, a key advantage of the presented approach in adding online monitors during physical synthesis compared to similar works conducted in the front-end phase of IC design.

4.3.3 Comparison of the Presented Work with Other Techniques

Table 6 presents a comparison of the proposed method with various detection and DfHT techniques. The first column outlines the specific technique or category, while the second column references relevant works in that category. The third column categorizes the technique as detection, DfHT, or a combination of both. The subsequent column describes the chip design stage where the method is applied for protection. Column 5 identifies the location of the potential attacker, and column 6 indicates whether the technique can also be used to prevent HTs. The final two columns offer a concise summary of the advantages and drawbacks of the techniques, respectively.

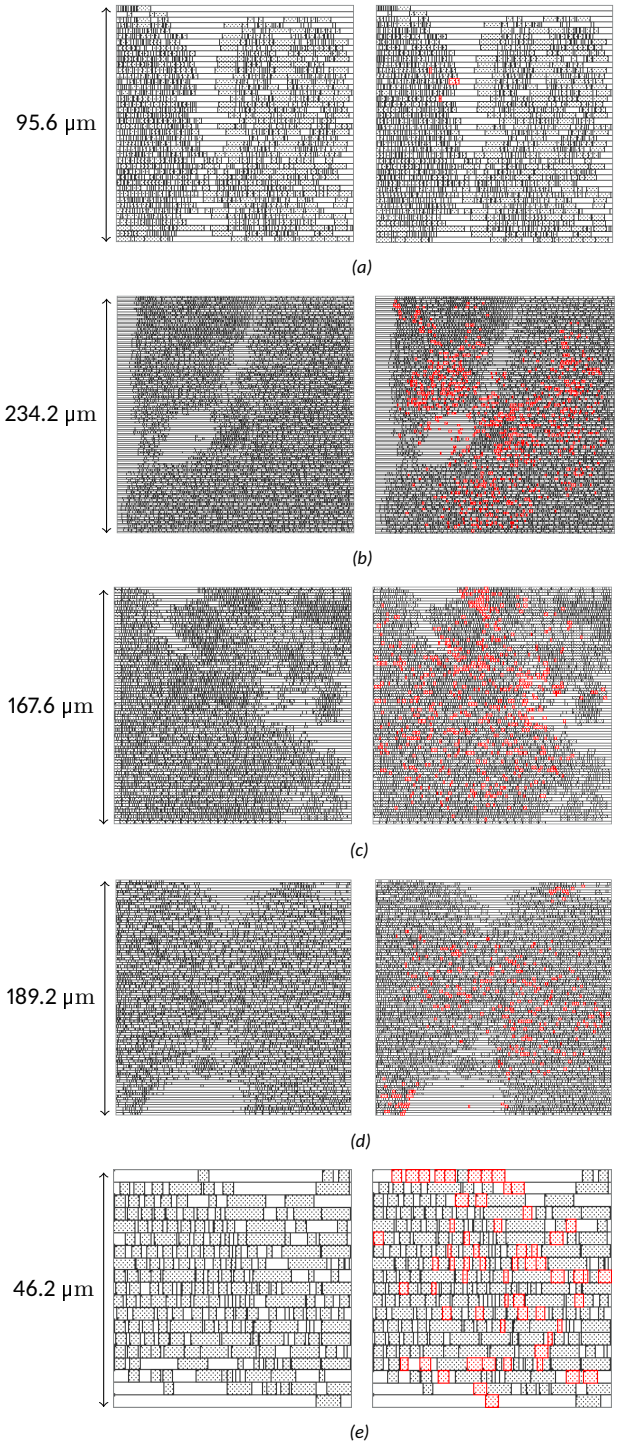

Figure 25: The layout view of selected IPs, whereas the left images in each row represent the placement configuration before adding online monitors, while the right one represents the view of each design after integrating the online monitors: a) `alert_handler_esc_timer`, b) `ast_reg_top`, c) `flash_ctrl_core_reg_top`, d) `keymgr_reg_top`, and e) `nmi_reg_top` (from [88])

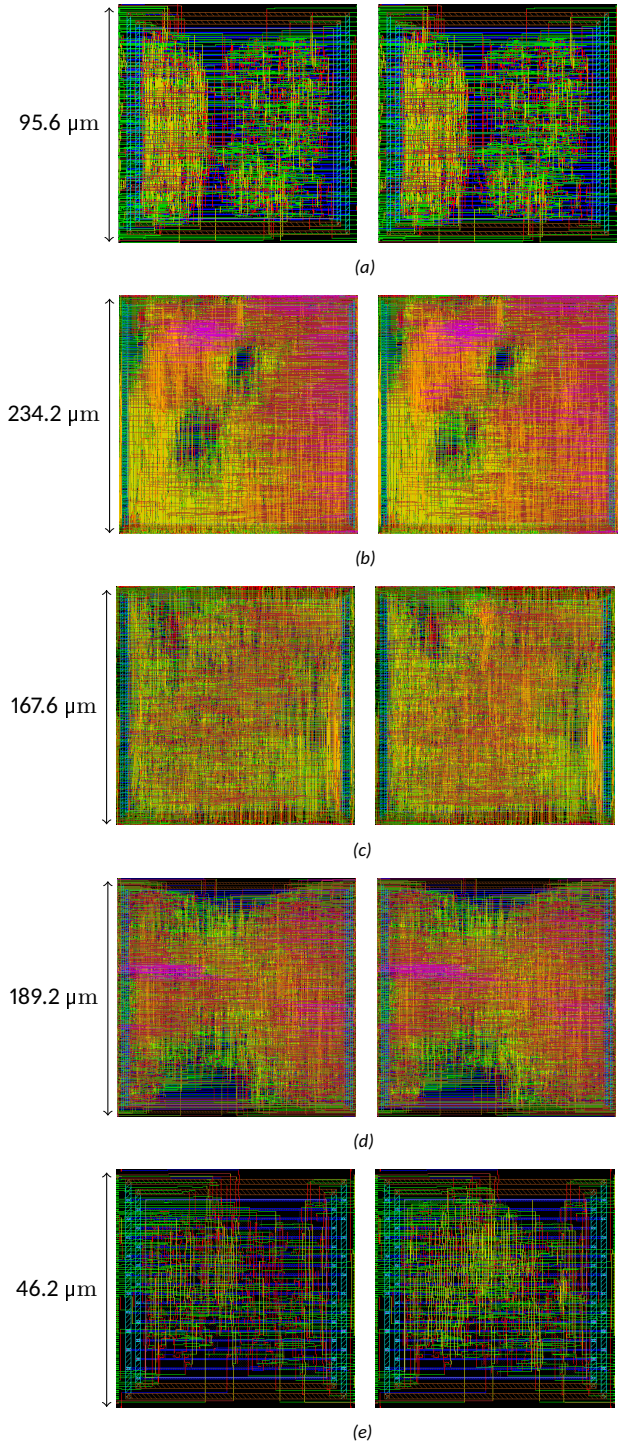

Figure 26: The layout view of selected IPs, whereas the left images in each row represent the routed view before adding online monitors, while the right one represents the view of each design after integrating the online monitors: a) `alert_handler_esc_timer`, b) `ast_reg_top`, c) `flash_ctrl_core_reg_top`, d) `keymgr_reg_top`, and e) `nmi_reg_top` (from [88])

Table 6: Comparison of Different Detection and DfHT Techniques

Technique	Refs	Detection or DfHT	Design Stage	Attacker Location	Prevents HT?	Pros.	Cons.
IC Fingerprinting, Delay Measurement	[103, 148, 149, 150]	Detection	Test	Des. house, Foundry	No	Nearly zero overheads, Non-destructive method	Needs reverse engineering for attributes of the Golden chip, Confusion with environmental and process variation effects
Logic Testing	[151, 152]	Detection	Test	Des. house, Foundry	No	Very fast technique, Can be fully automatic	All input combinations are not covered to activate HTs, Ineffective when dealing with HTs with sequential triggers
BISA, Layout Filling	[130, 131, 132]	DfHT	Back-end	Foundry	Yes	Replacement of filler cells with the functional ones, Independent testing circuit from the original one	Design becomes unroutable in high occupation ratios, Considerable overheads
Selective Placement	[65, 67, 68]	DfHT	Back-end	Foundry	Yes	Lowest overheads among HT prevention techniques	Risk of violating the critical paths in the routing stage, Timing degradation
TPAD	[141]	Both	Front-end, Back-end*	Des. house, Foundry	Yes	Zero false positives, Applicable to FPGA	Considerable overheads, Risk of degradation
This work		Both	Front-end, Back-end	Foundry	Yes	Reusing verification assets, Low PPA overheads	Achieving high density for some designs is challenging

* This technique is mainly applied in the front-end stage. To prevent CAD tool attacks, the design is split into two parts, each processed by different CAD tools independently. The final layouts from these tools are later merged in the back-end stage for being set to the foundry for fabrication.

5 SALSy: Security-Aware Layout Synthesis

This chapter presents a new methodology called Security-Aware Layout Synthesis (SALSy), which enables the design of ICs with inherent security considerations [69]. This approach is similar to the well-established practice of balancing PPA metrics and security, a concept that is referred to as security closure.

However, unlike PPA metrics, commercial layout synthesis tools do not offer any direct settings or options for security. Therefore, the task of SALSy is to work within the constraints and capabilities of these tools to indirectly achieve security properties in the final layouts. This involves modifying and adapting the existing algorithms and heuristics for placement, routing, CTS, and other physical synthesis tasks to make them more security-aware and resilient to various attacks and threats. Therefore, SALSy is a proactive strategy at the back-end phase that enhances the security of ICs against both fabrication-time and post-fabrication adversarial acts, including HT insertion, FI, and probing.

This methodology has been validated through a silicon demonstration, confirming its compatibility and effectiveness with a commercial PDK and library. SALSy achieves this enhanced security enhancement with only a minimal impact on power consumption, thus maintaining a balanced trade-off between security and PPA.

As discussed in detail in Section 2, in the fabless model, foundries are regarded as untrusted entities as design houses lack ownership or oversight over them. Consequently, IC design houses must prioritize safeguarding their designs (layouts) against potential threats originating from these untrusted foundries [153, 154].

Moreover, beyond fabrication-time attacks, numerous other threats exist. Once a finalized IC becomes available to malicious end-users on the open market, it becomes susceptible to attacks such as fault injection [21, 25]. In fault injection attacks, adversaries attempt to compromise the chip's security by introducing various faults into its operation.

Another post-fabrication attack is probing, where attackers seek unauthorized access to a chip's internal data through physical probing techniques [26, 155]. Typically, this attack aims to extract sensitive information like cryptographic keys or proprietary data, posing significant risks, particularly in critical or dependable applications [25].

To address these concerns, hardware security researchers have pursued the concept of security closure [61, 62, 63, 64, 65, 67, 68, 70]. This approach involves accepting certain overheads in terms of PPA to implement heightened security measures. The goal is to minimize vulnerabilities and potential attack surfaces, aiming to create trustworthy and resilient ICs capable of withstanding potential security breaches while ensuring reliable performance.

The methodology presented in this chapter outlines a comprehensive approach to achieving security closure, encompassing various techniques. The proposed flow (SALSy) is designed to be adaptable to layouts of any size, type, or technology. SALSy presents a generic and comprehensive approach to enhancing the security of IC designs during the physical synthesis stage. The proposed methodology addresses multiple security threats, including HTs, FI, and probing attacks. SALSy has been validated through the prototyping of a chip using a commercial 65nm CMOS technology, demonstrating its effectiveness and compatibility with current industry practices.

In addition to the development and validation of SALSy, this work also provides a comparative analysis of the utilization of commercial libraries and PDKs with open-source alternatives for security research. The analysis highlights the limitations and constraints associated with using open-source PDKs. To further facilitate and promote security research in the IC design community, this work also provides publicly accessible scripts that can be used to thoroughly verify and validate the techniques outlined in the study. These

scripts are designed to operate within a commercial physical synthesis tool, ensuring their compatibility and relevance to the current industry standards and practices.

5.1 Security Assessment Scheme

Researchers have introduced various metrics to evaluate the complexity of inserting HTs into a specific layout [71, 87]. This work uses the scoring framework presented in [71] to assess security. This framework is chosen because it considers a variety of threats (HT insertion, FI, and probing) instead of focusing on just one specific threat. Furthermore, to reflect the real challenges faced by an engineer during physical synthesis, design quality (i.e., PPA) is also taken into account in the final scores. Consequently, the overall score is a function of both *design quality* and *security*, as shown in Equation 3.

$$Score = DesignQuality \times Security \tag{3}$$

Where $DesignQuality$ consists of a weighted distribution of power, performance (in terms of clock frequency), area, and routing quality, and $Security$ consists of equally weighted metrics for HT insertion, FI, and probing. It should be noted that in this evaluation scheme, Front-Side Probing (FSP) is considered a proxy for FSP and FI.

The security scores for FSP/FI are determined by identifying a set of sensitive (security-critical) cells and their related interconnecting wires. These cells, known as ***cell assets***, and the related wires, called ***net assets***, are used to calculate a metric known as the ***exposed area***. This metric is calculated for each set of cell and net assets in each design and represents any spatial area that can be accessed from the top through the metal stack. An example of an exposed area is depicted in Figure 27. In this illustration, the red-marked cell areas indicate the exposed regions, making them susceptible to FI or probing attacks due to the lack of protection from other elements (i.e., metal wires) on the front side.

Figure 27: Example of exposed area (highlighted in red) for cell assets (from [71]).

To determine the HT-related portion of the Security score, an ***exploitable region*** metric is established. This metric defines a set of continuous placement sites[3] that are either i) free, ii) occupied by filler cells or non-functional cells, or iii) unconnected cells. When the number of these continuous placement sites reaches a minimum threshold of 20, they are identified as an exploitable region. Furthermore, free routing tracks around the exploitable region(s) are also considered. The motivation behind this is that an adversary needs both placement and routing resources to successfully insert an HT. Consequently, there should

[3]A placement site refers to a predetermined valid position within a layout where a cell can be legally positioned. These placement sites are typically determined by factors such as the standard cell height and the contacted poly pitch.

be enough gaps in the layout or some logic that can be easily removed to accommodate the HT.

The baseline layouts, before applying any security closure techniques, have a default score of 1. Layout modifications that improve design quality and/or security would be scored within the range of $[0, 1)$, while poor modifications would be scored within the range of $(1, \infty]$.

To derive each component of the scoring formula (i.e., power) outlined in Equation 3, first the relative metric for the baseline layout should be calculated. Subsequently, the corresponding metric for the modified (secured) layout can be obtained. The score for each specific component is then computed by dividing the values of the secured version by those of the baseline version. The final score is obtained by assigning relative weights to each element and summing them. Hence, Equation 3 can be expanded as follows:

$$Score = \overbrace{\left(0.1 \times \frac{(des\_p\_total)_{\text{s}}}{(des\_p\_total)_{\text{bl}}} + 0.3 \times \frac{(des\_perf)_{\text{s}}}{(des\_perf)_{\text{bl}}} + 0.3 \times \frac{(des\_area)_{\text{s}}}{(des\_area)_{\text{bl}}} + 0.3 \times \frac{(des\_issues)_{\text{s}}}{(des\_issues)_{\text{bl}}}\right)}^{\textit{DesignQuality}} \times \left(\overbrace{\frac{1}{2} \times \left(0.5 \times \frac{(fsp\_fi\_ea\_c)_{\text{s}}}{(fsp\_fi\_ea\_c)_{\text{bl}}} + 0.5 \times \frac{(fsp\_fi\_ea\_n)_{\text{s}}}{(fsp\_fi\_ea\_n)_{\text{bl}}}\right)}^{\textit{Security (fsp/fi)}} + \overbrace{\frac{1}{2} \times \left(0.6 \times \frac{(ti\_sts)_{\text{s}}}{(ti\_sts)_{\text{bl}}} + 0.4 \times \frac{(ti\_fts)_{\text{s}}}{(ti\_fts)_{\text{bl}}}\right)}^{\textit{Security (ti)}}\right) \tag{4}$$

In this equation, des_p_total, des_perf, des_area, and des_issues represent the power, performance concerning timing violations (if any), area, and Design Rule Checks (DRCs) respectively. The terms $fsp\_fi\_ea\_c$ and $fsp\_fi\_ea\_n$ indicate the exposed area of the cell assets and the exposed area of the net assets, respectively, and ti_sts and ti_fts terms denote the exploitable regions and available routing resources (free tracks) of exploitable regions.

5.2 SALSy Techniques

This section introduces different techniques used in SALSy, the primary contribution of this study. In this section, two perspectives of SALSy will be presented: one is the pre-silicon view, compatible with open-source PDKs, and the other is the post-silicon view, more aligned with commercial PDKs. Consequently, the results obtained from a real fabricated chip implementing SALSy concepts will be provided in the following section. Comparing to open-source PDKs is crucial as it enables this work to be assessed relative to others within the framework outlined in [71].

An outline of the employed techniques and their respective sequence is depicted in Figure 28. Notably, not all techniques suitable for an open-source PDK can be applied in an actual tapeout. The color scheme adopted in the figure designates green-colored rectangles to indicate techniques fully compatible with commercial PDKs.

The techniques outlined in steps 1 through 6 are primarily used to increase the design's security against FSP/FI attacks. The last two steps, on the other hand, are mainly focused on eliminating exploitable regions to protect the design against HT insertion. To help the user determine whether the design has achieved the desired level of security, two checkpoints are included. The first checkpoint is placed after the completion of Edge Cell Placement in step 4. If the user is satisfied with the enhanced security against FSP/FI at this point, they can choose to skip steps 5 and 6. Otherwise, the techniques in steps 5 and 6 can be applied to further improve the design's security against these types of attacks. The second

checkpoint is placed after step 8, and is used to evaluate the enhanced security against HT insertion. If the security scores for the TI analysis meet the user's requirements, the layout can be considered final. If not, steps 7 and 8 can be repeated until the TI scores reach the threshold defined by the user.

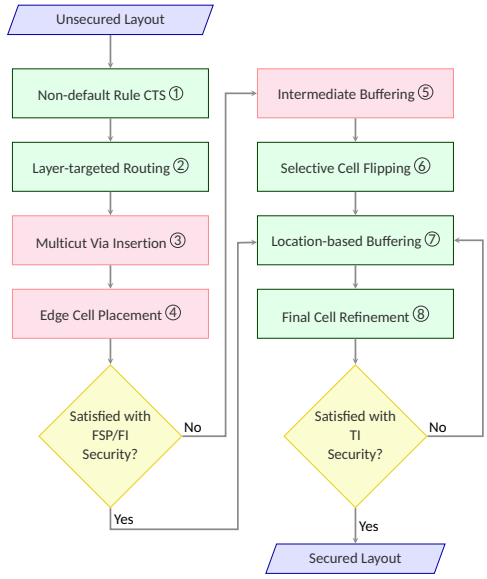

Figure 28: SALSy framework. Red boxes highlight techniques that are not feasible for the tapeout. Green boxes highlight techniques that can be used in both open-source PDKs and in the tapeout. (from [69])

5.2.1 Benchmarks

The selected benchmarks for the open-source experiment predominantly comprise crypto cores, including CAST, Camellia, MISTY, PRESENT, OpenMSP430_1, three versions of AES, SEED, TDEA, OpenMSP430_2, and SPARX [156, 157, 158].

5.2.2 Open-source PDK

Consistent with comparable academic efforts, the selected PDK/standard cell library in [71] is the Nangate 45nm Open Cell Library [159], given its unrestricted availability. The metal stacks taken into account are 6M and 10M, contingent upon the benchmark's complexity.

It is worth emphasizing that the scoring formula prioritizes a delicate balance between security and PPA, as outlined in Equation 4. Regarding the design aspect, while customized implementation scripts were utilized for each benchmark, it is important to note that these scripts primarily focused on traditional parameter exploration in physical synthesis and will not be extensively elaborated upon. The subsequent discussion will primarily center around the security aspects. Moreover, given that the scoring formula accounts for distinct metrics concerning front-side probing/fault injection (*fsp_fi*) and HT insertion (*ti*), the relevant SALSy techniques will be explained separately.

5.2.3 Countermeasures against FSP/FI

1) Non-default Rule Clock Tree Synthesis: The core idea of this strategy is to modify the default rules for CTS[4] in order to protect a broader range of assets by widening the clock distribution wires. It is worth noting that CTS routing consumes fewer resources compared to signal routing. Therefore, CTS wires can be significantly widened, often several times more than signal wires. As illustrated in Figure 30a, the enlarged clock tree can significantly cover more exposed areas under it. Often, the quality of the CTS is improved by using non-default rules.

2) Layer-targeted Routing: Recall that the exposed area metric, pertaining to both cells and nets, denotes the area of assets directly accessible from the front side. In the initial step, the objective is to shield the net assets under other non-asset nets to safeguard them against FSP/FI, as outlined in Algorithm 1. To achieve this, the lowest available metal layers[5] are exclusively assigned to the net assets (line 3). It is important to note that the minimum width for routing these asset-related wires is utilized to hide them under other nets (line 5).

Algorithm 1 Layer-targeted Routing Algorithm

1: $net\_assets \leftarrow List\_of\_net\_assets$
2: $other\_nets \leftarrow List\_of\_other\_nets$
3: $prf\_lays\_assets \leftarrow [M2, M3]$
4: $prf\_lays\_others \leftarrow [M4, M5, M6]$
5: $width\_for\_assets \leftarrow width(M2)$ ▷ This value is the minimum width according to the library
6: $width\_for\_others \leftarrow width(M2) \times 2$
7: **foreach** net **in** net_assets **do**
8: **assign** *prf_lays_assets* **to** *route_layer*
9: **assign** *width_for_assets* **to** *width_rule*
10: **end for**
11: **route** *net_assets* **with** *width_rule* **in** *route_layer*
12: **if** $(route\_err)$ **then**
13: **route** *net_assets* **with** *default_rules*
14: **end if**
15: **foreach** net **in** $other\_nets$ **do**
16: **assign** *prf_lays_others* **to** *route_layer*
17: **assign** *width_for_others* **to** *width_rule*
18: **end for**
19: **route** *other_nets* **with** *width_ruler* **in** *route_layer*
20: **if** $(route\_err)$ **then**
21: **route** *other_nets* **with** *default_rules*
22: **end if**

Subsequently, all remaining non-asset nets are designated to be routed using higher metal layers (line 4). Additionally, it is tried to opt for a wider width, different from the

[4]CTS is a crucial phase in the design workflow of digital ICs, involving the creation of a network of clock branches to efficiently distribute the clock signal across the entire circuit. By carefully constructing the network, observing delay balancing, and managing skew, timing can be improved, and power consumption can be reduced. CTS routing typically takes priority over signal routing, which is leveraged for security purposes.

[5]Metal layers are organized into a metal stack, with the lower layers typically being thinner.

default width, to enhance the chance of covering net and cell assets (line 6). If the routing tool encounters difficulties in routing the nets with the adjusted width or in the preferred metal layer, it will attempt to route them using the default width and default metal layers (lines 12-14, 20-22). It is important to note that for the physical synthesis tool employed, routing constraints are considered soft constraints, meaning that the tool will make every effort to adhere to the constraints. However, if it faces challenges, the constraints may be relaxed.

As an example, in Algorithm 1, M2 and M3 layers are dedicated exclusively to routing the net assets (line 3), while the higher metal layers M4-M6 are allocated for routing non-asset nets (line 4). In this scenario, the width of the non-asset nets is set to be twice as wide as that of the net assets (line 6). However, this value can be adjusted if additional resources become available[6]. Upon implementing this technique, congestion significantly increases, which enables more cell assets and net assets to be protected against FSP/FI, as depicted in Figure 30b.

3) Multi-cut Via Insertion: In an IC's metal stack, a vertical connection known as a *via* enables the connection between different metal layers. Typically, the physical synthesis tool optimizes routing resource utilization and minimizes congestion by employing the minimum number and smallest size of vias available for connections. However, the proposed strategy aims to deliberately increase congestion on the top of the cell assets to improve their coverage. To achieve this, the insertion of multi-cut vias between the M1 and M2 layers allows for a larger metal piece to be routed over the cell assets, leading to improving coverage, as illustrated in Figure 29. The decision to use multi-cut vias exclusively between the M1 and M2 layers is deliberate, as it avoids affecting the resources in higher metal layers, which are reserved for signal routing.

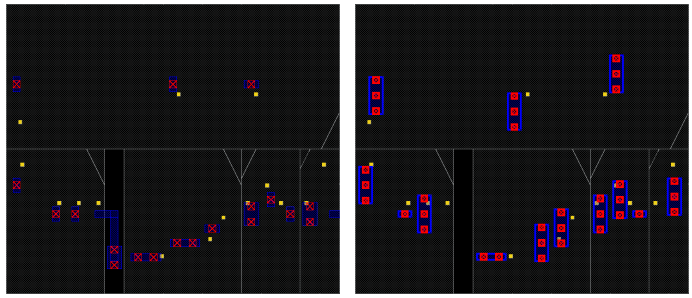

Figure 29: Using the default rules for via insertion (left) and multi-cut via insertion (right) to increase the coverage of cell assets (from [69])

4) Edge Cell Placement: In certain benchmarks, it has been observed that net assets encompass lengthy wires that extend from IO pins to their respective sinks (highlighted in green in Figure 30c). To address this, a technique is employed where the sink cell linked to the IO pins through net assets is moved to the nearest feasible position to their driver. This replacement substantially reduces the length of the net assets, increasing the chance of being covered by other nets on the upper layers since shorter nets typically exhibit fewer turns and jogs.

5) Intermediate Buffering: The aforementioned technique for shortening net assets is effective only for wires connected to IO pins, leaving other net assets vulnerable due

[6] In real ICs, the number of metal layers depends on the technology and metal stack agreed upon with the foundry. Current technologies often provide 10 or more layers.

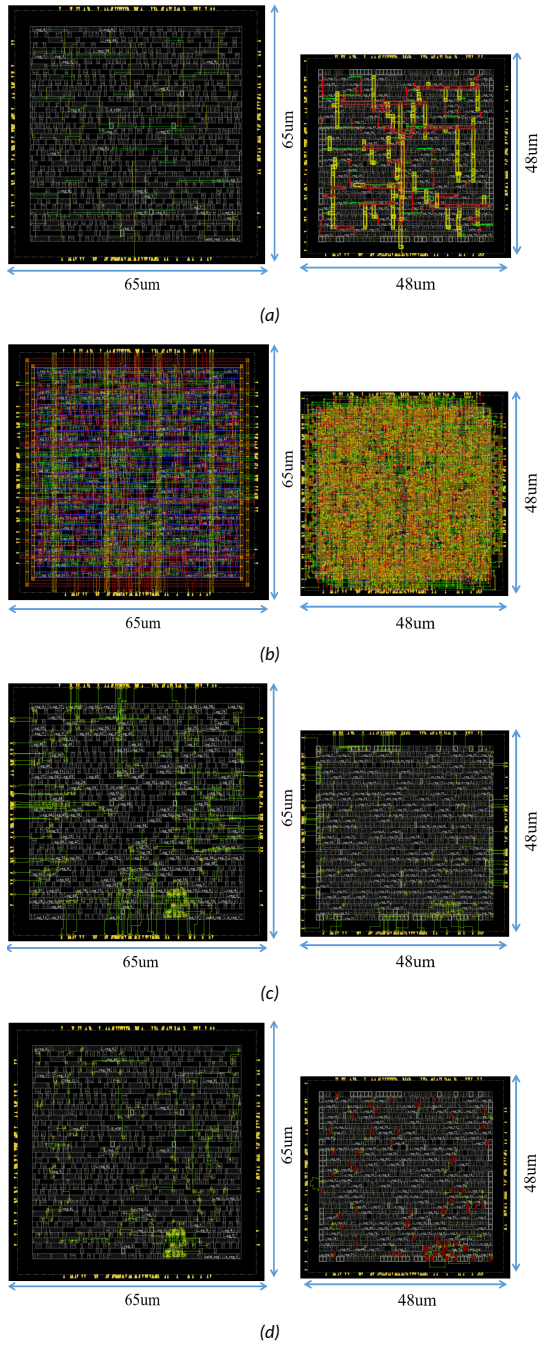

Figure 30: Different techniques used in SALSy (from [69]). The design on the left is always the BL variant, and the design on the right is always the SEC variant. a) Non-default Rule CTS, b) Increased congestion by applying Layer-targeted Routing, c) Edge Cell Placement for shortening the long net assets (highlighted in green), and d) Reducing the length of the net assets (highlighted in green) by applying Intermediate Buffering technique (added buffers appear in red).

to their extended length. When both the driver and sink are placed within the core area (the region containing all cells), addressing this challenge becomes essential. In such cases, buffers are inserted between the driver and sink to shorten the length of these lengthy net assets, as depicted in Figure 30d. It is important to note that buffer insertion can significantly impact the circuit's timing and power consumption. Therefore, this technique is implemented iteratively with multiple checkpoints. If the insertion of a buffer leads to a timing violation, the buffer can be removed, reverting the circuit to its previous state without the violation.

6) Selective Cell Flipping: In certain scenarios, the exposed area of net assets can be notably reduced by changing the orientation of the cell (i.e., flipping it over the Y axis). This makes the physical synthesis tool automatically re-route the nets connected to the flipped cell, thereby enhancing the chance of covering the net asset beneath other nets, as illustrated in Figure 31. It is important to emphasize that this technique is executed during the final stages of the proposed methodology, and only net assets with the most substantial exposed area are targeted for this adjustment.

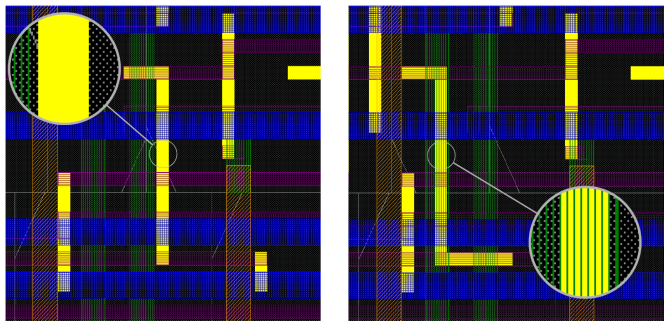

Figure 31: An example of covering a net asset by flipping the cell (from [69]): The exposed area (solid yellow regions in the left image) is totally covered by the nets in the upper metal layer(s) after the net is re-routed (right image)

5.2.4 Countermeasures against HT Insertion

This section outlines the techniques employed against HT insertion. The concept of an *exploitable region* is revisited here, which is defined as a set of continuous gaps, filler cells, disconnected cells, or non-functional cells, and could be exploited by an adversary for inserting malicious logic. Since HT components must connect to the existing design, the availability of routing resources is also a consideration in determining such regions. However, the primary focus here is on eliminating free placement sites. This is mainly because if all the gaps are eliminated, there would be no space for HT logic to be placed in the design. Consequently, available routing resources become irrelevant, as there would be no HT cells to connect to the original design.

7) Location-based Buffering: Despite reducing the design area to maximize the density of the core area, some gaps may still exist, creating large exploitable areas. Given that a continuous gap exceeding 20 placement sites qualifies as an exploitable region, a script has been developed to identify such regions and introduce buffers to either fill these gaps or reduce them to fewer than 20 sites. It is important to note that the insertion of buffers can incur additional power consumption and possibly affect timing. Nonetheless, the balance between enhanced security and the impact on PPA is considered advantageous for this specific technique.

8) Final Cell Refinement: In particular cases, the insertion of buffers may be unsuccessful due to insufficient routing resources within the congested regions. Additionally, even if successful, it could lead to the creation of timing violations. To address this, efforts are made to mitigate any remaining vulnerable areas by incrementally adjusting the adjacent cells. This straightforward method can be implemented through algorithmic strategies as outlined in the [65] framework or, in scenarios with a limited number of instances, manually by a physical design engineer.

Figure 32 illustrates the successful elimination of all exploitable regions within a design through the adoption of the mentioned techniques. It is important to clarify that eliminating all exploitable regions in the layout does not necessarily mean that there are no gaps or empty spaces left. Rather, it implies that any remaining gaps are smaller than the predefined threshold for an exploitable region.

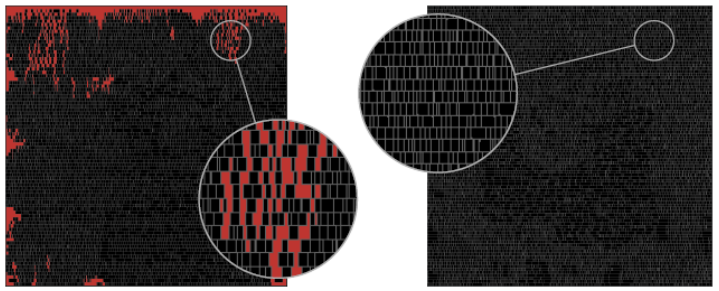

Figure 32: An example of a) design with exploitable regions (highlighted in red), and b) design with zero exploitable regions using the proposed techniques (from [69])

5.3 Scores for Open-source PDK and Comparisons

This section presents the benchmark scores achieved by employing the specified methods on the considered open-source PDK. Additionally, the results are compared with those from other studies aimed at improving the security of the same benchmark layouts. To facilitate this comparison, data from teams that participated in the security closure contest associated with the ISPD'22 conference have been compiled. A detailed version of the contest's logistics and framework is provided in [71].

The scoring equation presented in Equation 3 is designed to normalize results relative to baseline measures, attributing equal significance to both design quality and security considerations. Nevertheless, due to the presence of a multiplicative relationship between these two parts of the score, a hypothetical zero score in security—though impossible in reality—would result in a total score of zero.

Participants in the competition eventually sensed that implementing a singular metallic shield atop the entire layout effectively neutralized all security vulnerabilities. This approach is equivalent to employing a complete metal layer as a sacrificial layer. While this method does result in violations of DRC, the scoring formula regrettably does not impose penalties for such infractions: given that the security metric is nullified, the design aspect of Equation 3 becomes irrelevant. In conclusion, the ISPD'22 contest ended with several teams achieving a perfect score of zero. The final scores are outlined in Table 7. SALSy techniques are denoted as team 'K.'

It is worth mentioning that the proposed solution involving a sacrificial metal layer lacks any substantial value in practice. Its efficacy is limited to fulfilling the criteria for contest scoring and does not offer real protection against the threats under consideration. A clearer

Table 7: Overall scores of the participating teams

Benchmarks / Teams	J	N	O	E	L	A	Q	**K**
AES_1	0.764	0.025	0.000	0.000	0.271	0.000	0.000	**0.000**
AES_2	1.687	0.054	0.000	0.000	0.324	0.000	0.000	**0.000**
AES_3	1.332	0.000	0.000	0.000	0.295	0.000	0.000	**0.000**
Camellia	0.676	0.000	0.000	0.000	0.281	0.000	0.000	**0.000**
CAST	1.687	0.000	0.000	0.000	0.300	0.000	0.000	**0.000**
MISTY	3.178	0.000	0.000	0.000	0.254	0.000	0.000	**0.000**
OpenMSP430_1	0.841	0.000	0.000	0.000	0.344	0.000	0.000	**0.000**
PRESENT	0.629	0.000	0.000	0.000	0.319	0.000	0.000	**0.000**
SEED	2.203	0.000	0.000	0.000	0.207	0.000	0.000	**0.000**
TDEA	0.596	0.003	0.000	0.000	0.246	0.000	0.002	**0.000**
OpenMSP430_2	1.031	0.000	0.000	0.000	0.822	0.000	0.000	**0.000**
SPARX	0.476	0.000	0.000	0.000	0.262	0.000	0.000	**0.000**

understanding of the overheads imposed by the presented techniques can be obtained from Table 8. When considering only the design component of Equation 3, it is evident that SALSy scores are quite competitive. These findings, coupled with an assessment of the feasibility of adapting the proposed techniques for use with a commercial PDK, have prompted me to proceed with a tapeout, which is elaborated upon in the subsequent section.

Table 8: Design quality scores of the participating teams

Benchmarks / Teams	J	N	O	E	L	A	Q	**K**
AES_1	0.995	0.713	0.447	0.475	0.527	0.519	1.347	**0.481**
AES_2	3.737	0.702	0.425	0.458	0.539	0.509	0.817	**0.461**
AES_3	2.689	1.059	0.473	0.498	0.566	0.541	1.171	**0.523**
Camellia	0.753	0.746	0.398	0.420	0.470	0.418	0.960	**0.530**
CAST	1.663	0.851	0.412	0.409	0.463	0.439	0.908	**0.495**
MISTY	5.009	0.753	0.418	0.396	0.457	0.417	1.559	**0.458**
OpenMSP430_1	0.756	0.656	0.406	0.440	0.490	0.469	1.025	**0.632**
PRESENT	0.752	0.693	0.359	0.427	0.465	0.446	1.009	**0.306**
SEED	1.917	0.892	0.416	0.442	0.418	0.442	0.924	**0.522**
TDEA	0.750	0.846	0.459	0.526	0.534	0.524	0.808	**0.584**
OpenMSP430_2	0.995	0.777	0.464	0.543	0.524	0.570	0.848	**0.608**
SPARX	0.753	0.663	0.397	0.420	0.422	0.404	1.047	**0.509**

5.4 Silicon Validation of SALSy

In the prior section, various methodologies to enhance IC security were explained, anchored by specific evaluation of benchmark circuits. However, these techniques were designed for open-source PDKs. In contrast, industry-utilized commercial PDKs possess a greater level of complexity than their academic counterparts. Therefore, using commercial PDKs can increase design complexity and introduce certain practical limitations. These limitations may arise from factors such as compatibility issues and the need for specific design rules and guidelines to be followed. Consequently, in order to demonstrate the gaps and limitations of open-source PDKs for rigorous security closure assessment and offer solutions to address these issues, I decided to fabricate a chip incorporating the mentioned security features. This hands-on approach allows the community to gain valuable insights into the challenges and potential solutions when working with open-source PDKs in the context of secure chip design.

In designing the chip, the first step is to adapt the scoring system to the commercial library, which enables the evaluation of the security features of the chip using the same metrics from [71]. Next, it is needed to decide which designs to be included in the tapeout.

A small chip size (1 mm^2) has been chosen that can accommodate four designs arranged as eight blocks: four secured versions (SEC) and four baseline versions (BL). Having a pair of each benchmark on the same chip makes it possible to fairly evaluate and compare each block's security and design quality.

The selection of designs aims to contain a range of complexities and sizes, with Camellia, CAST, PRESENT, and SEED as the final candidates. The chip's floorplan, depicted in Figure 33, uses color differentiation for various blocks, while the core area is reserved for the comparison and control unit. To ensure a fair comparison, all BL version benchmarks maintain the density from the open-source experiment. In contrast, the density of the SEC versions varies as different SALSy techniques are applied. Additionally, separate power domains were created for each block, facilitating the activation of a single block at a time. This ensures the remaining blocks are powered down, allowing for accurate power consumption measurements for each block individually.

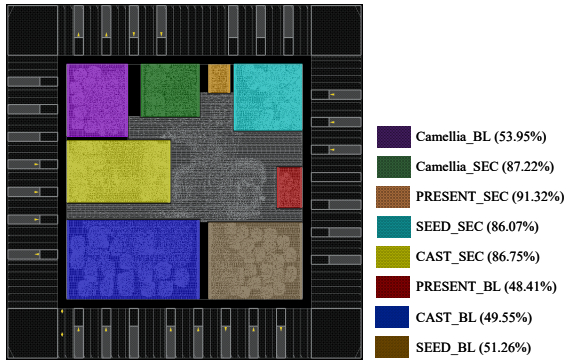

Figure 33: Floorplan view of the chip including eight blocks and density of each block (from [69])

Figure 34 presents a microscope view of the fabricated chip. The effectiveness of the proposed techniques has been validated across all four benchmarks that were demonstrated on this chip. It is important to emphasize that the presented methodology is generally applicable to any design with different functionality or scale. A notable advantage of this approach is its independence from prior knowledge of the design, as it operates at the layout stage. It is proposed that SALSy enables the possibility of assigning security closure to a separate design team, as no specific design details or characteristics are required for enhancing security. The interface between this team and the traditional physical synthesis team would be a straightforward list of assets. This separation allows for a more focused and efficient approach to security improvement during the design process.

5.4.1 Implementation for Commercial Process Design Kits

A crucial factor in enhancing the scores for HT and FSP/FI is to increase the design's density. By doing so, the possibility of HT insertion decreases due to the reduction in gaps, and more cell and net assets can be protected against FI/FSP as a result of increased wire congestion. Therefore, all designs are shrunk as much as possible before applying any specific technique (notice the smaller size of the SEC variants compared to the BL variants in Figure 33 and the remarkably high-density levels achieved for the SEC variants). In the following text, more details about the chip implementation are provided. It is worth mentioning that several scripts for the implementation flow have been released in a GitHub repository [160]. This sets the presented work apart from previous security closure attempts, which are nearly impossible to reproduce.

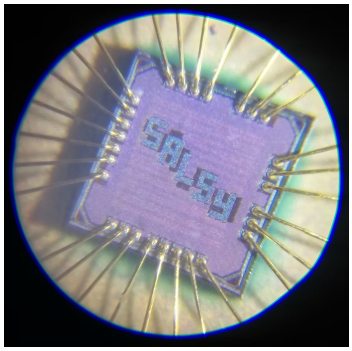

Figure 34: Microscope view of the fabricated 1mm^2 chip (from [69])

1) Non-default Rule CTS: This method can also be applied to commercial libraries but with certain limitations on the maximum wire width. The foundries set these limitations to ensure the designs remain compatible with their processing capabilities. Consequently, the enlargement of clock wires cannot be as extensive as in the open-source experiment; however, a moderate increase from the standard width is feasible. Despite its reduced efficacy relative to the open-source experiment, this technique is still employed due to its minimal impact on power consumption and other performance metrics.

2) Layer-targeted Routing: In a manner similar to the presented approach in the open-source experiment, the routing strategy delineated in Algorithm 1 is employed. The difference lies in the utilization of commercial libraries, which offer a more detailed and defined set of parameters to guarantee the accuracy of design rules and verification. Consequently, as the design's complexity increases, a more significant number of violations appear for various reasons, all in the service of ensuring the chip's quality and reliability throughout the manufacturing process. Therefore, achieving a high density (exceeding 90% for the specified 65nm technology) presents significant challenges. It becomes unfeasible to route every asset or non-asset net within their optimal metal layers as previously done in the open-source experiment. Nonetheless, despite the necessity to route some asset nets through the upper metal layers, this method continues to effectively cover a notable portion of the exposed areas of the assets.

3) Multicut Via Insertion: This method was the first one that had to be abandoned due to the strict constraints in the commercial PDK. While employing multi-cut vias for pin connections is a theoretical possibility, it leads to systemic DRC violations once the vias are connected to the wires. Given the significant challenge of addressing numerous DRC violations, adopting this method in the chip was not practical. However, this strategy may be applicable and worth reconsidering for an alternate commercial library/PDK. Typically, multi-cut vias are utilized in power routing rather than signal routing, which was the novel usage that was explored.

4) Edge Cell Placement: An important distinction between the open-source experiment and an actual tapeout is that each design was treated as a separate chip in the open-source experiment, while all the designs must be placed together on one chip in this case. This integration significantly restricts the flexibility in assigning IO pin locations for each design. These locations are typically determined during the top-level floorplanning stage, where decisions are made about which side of the block (left, right, bottom, or top) the IO pins should be placed. Taking the *PRESENT_SEC* block as an example, as depicted in Figure 33, putting the IO pins on the block's bottom edge is advantageous due to its closeness to the

centrally located control and comparison unit. This choice results in routing with fewer issues and avoids unnecessary resource utilization, leading to a more optimized floorplan.

The constraints on pin placement present a challenge for incorporating this technique into the chip's design. As illustrated in Figure 35, the closeness of the IO pins to the net assets, highlighted in white, offers only a limited space. This spatial limitation makes it infeasible to position all connected cells adjacent to their respective driver/sink pins, as doing so would lead to excessive congestion, making the design unroutable. Consequently, this technique is unsuitable for the current floorplan configuration. Nevertheless, it is important to note that this constraint does not prevent the application of this technique in different chip designs. For instance, in an open-source experiment, cells were successfully placed near their corresponding IO pins due to the square shape of the block and the availability of space around all four sides of the design (Figure 30).

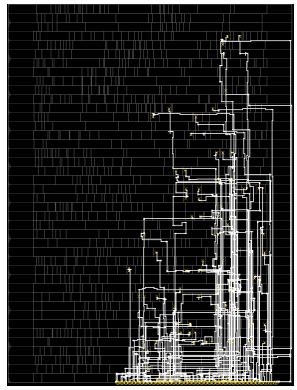

Figure 35: A design (PRESENT) with most of the IO pins on the bottom side and the net assets (highlighted in white) connected to their relative IO pin (from [69])

5) Selective Cell Flipping: This method is applicable both in the context of chip implementation and open-source experiment and presents no particular limitations. Nevertheless, its utilization is limited due to the inherent manual nature of this technique. The primary goal is to maintain a comprehensive and automated methodology, avoiding reliance on selective methods, thereby guaranteeing the comprehensive applicability of the presented work.

6) Intermediate Buffering: As mentioned in the previous section, buffer insertion can have undesirable effects on the timing and power of the design. In the open-source experiment, such issues were only considered as a negative factor in the final score to penalize the teams. However, in the actual chip, any single issue that violates the timing of the design (e.g., setup time, hold time) is considered unacceptable. Therefore, the timing closure of the design must be perfect, and the trade-off between timing issues and enhanced security is only possible as long as the timing slack remains positive. Due to this reason, this technique was replaced with a smarter buffer insertion algorithm, which is explained in the following text.

7) Location-based Buffering: As mentioned previously, the buffer insertion strategy is modified such that it can focus on filling the continuous gaps in the design, rather than shortening long net assets. This change transformed the buffer insertion technique into a location-based algorithm that targets exploitable regions instead of searching for long net assets. The sinks and drivers of the added buffers are selected from nearby cells to minimize the negative impact on the timing of the design, as illustrated in Figure 36.

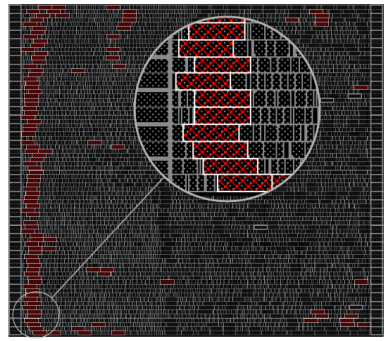

Figure 36: Added buffers (highlighted in red) using the smart algorithm to eliminate exploitable regions (from [69])

8) Final Cell Refinement: Similar to the open-source experiment, this technique is applied in the very late stages of chip implementation as a manual fix. If any exploitable region remains that can be eliminated with a few cell movements, this method can be used. However, it is decided to minimize the use of this technique in the chip implementation for two reasons: i) if an exploitable region is eliminated simply by moving cells, an adversary can potentially revert the changes to create enough space for their malicious logic, making this effort less effective in a realistic scenario; ii) it conflicts with the goal of creating an automated flow.

5.5 Results

This section presents the practical results of the chip design and measurement procedures. The Cadence suite was employed throughout the physical implementation phase, targeting a commercial 65nm CMOS technology. The results are categorized into pre-silicon and post-silicon parts. The former encompasses the data from the final layout sent for fabrication, including the block area and density. The latter part includes the actual chip metrics measured, such as power consumption.

5.5.1 Pre-silicon Results

As previously mentioned, the evaluation methods use the scoring system referenced in [71], and further details on each metric are explained below. The final scores of the presented methodology are depicted in Table 9, which clearly shows that the highest score belongs to HT insertion. This highlights the efficacy of SALSy as a preventive measure within a practical PDK environment. On the other hand, the expected lower scores in power are due to the security-enhancing buffer insertions. The power consumption consistently exceeded the baseline figures, as shown by the values exceeding 1.0 across the table. However, it is crucial to acknowledge the unavoidable trade-off between increased security and associated overheads. The power metrics in Table 9 are based on physical synthesis estimates, with exact figures detailed in Section 5.5.2.

This table demonstrates a significant decrease in the number of *exploitable regions* in the secured version for all benchmarks. This decline is especially notable, with the complete elimination of such regions in the Camellia and PRESENT benchmarks. Additionally, the CAST and SEED benchmarks show reductions of 95.3% and 90.3%, respectively. In terms of FSP/FI evaluation, the benchmarks show diverse results. The PRESENT benchmark stands out with a notable 43% reduction in the *exposed area* compared to the baseline, whereas

the CAST benchmark shows a more modest improvement of 18.5%.

Table 9: Final scores of SALSy for four different benchmarks

Metrics / Benchmarks			Camellia	CAST	PRESENT	SEED
Design Quality	DRC	des_issues	0.000	0.000	0.000	0.000
	PPA	des_perf	0.000	0.000	0.000	0.000
		des_p_total	1.184	1.072	1.161	1.041
		des_area	0.686	0.606	0.597	0.627
	Overall	des	0.467	0.419	0.439	0.417
Security	Trojan Insertion	ti_sts	0.000	0.015	0.000	0.026
		ti_fts	0.000	0.079	0.000	0.169
	Overall	ti	0.000	0.047	0.000	0.097
	FSP/FI	fsp_fi_ea_c	0.842	0.797	0.293	0.762
		fsp_fi_ea_n	0.624	0.833	0.568	0.835
	Overall	fsp_fi	0.733	0.815	0.430	0.799
Final score		OVERALL	0.171	0.181	0.094	0.187

To offer a thorough understanding of the relationship between each step of SALSy and the resulting scores, individual scores for the PRESENT benchmark are presented in Table 10 after applying each technique. This table shows that the Layer-targeted Routing technique has the most significant effect on the *fsp_fi* and overall scores, due to its considerable impact on increasing congestion. On the other hand, the Location-based Buffer Insertion technique has the most substantial impact on enhancing the *ti* score, as it drastically reduces the number of gaps in the layout. Notably, the overall trend of score improvement, as displayed in Table 10, remains consistent for all other three benchmarks.

5.5.2 Post-fabrication Results

This section details the measurement results obtained from the actual chip. The testing environment, as shown in Figure 37, consists of several components: a controller responsible for serial communication, input feeding, output reading, and data analysis; a power supply; a frequency generator providing a fast clock; and a precise measuring unit for assessing the chip's power consumption under various scenarios. The experiments are conducted on 20 packaged chips, selected from a total of 100 fabricated chips.

Verifying the Chip Functionality:

Before commencing power measurements, it is crucial to verify the proper functioning of the chips and their respective blocks. To achieve this, a Python script was developed to systematically activate each block at the target frequency while simultaneously ensuring the accuracy of the output data. All chips were found to be functional, allowing power measurements to be proceeded with. It is important to note that the target frequency for all blocks is 100 MHz, while the clock frequency for the comparison and control unit is set to 1 MHz. Additionally, a fast 100 MHz reference clock is generated by an external frequency generator, as depicted in Figure 37. The reader is reminded that total power consists of dynamic and static (leakage) power, which will be reported separately. The dynamic power results are reported at 100 MHz.

Leakage Power Measurement: Once the chip's functionality is verified, the next step involves measuring its power consumption. Initially, the Always On (AO) leakage power is evaluated. This type of power consumption refers to the power consumed by the circuit when it is in an idle or standby mode, but still powered on. This power consumption is caused by the leakage current that flows through the transistors and other components of the circuit or system, even when they are not actively switching or processing data.

Table 10: The changes in the scores of PRESENT benchmark after applying SALSy techniques in each step

Metrics / Steps			Non-default Rule CTS	Layer-targeted Routing	Location-based Buffer Insertion	Final Cell Refinement
Design Quality	DRC	des_issues	0.000	0.000	0.000	0.000
	PPA	des_perf	0.000	0.000	0.000	0.000
		des_p_total	1.018	1.138	1.159	1.161
		des_area	0.597	0.597	0.597	0.597
	Overall	des	0.404	0.434	0.436	0.439
Security	Trojan Insertion	ti_sts	0.010	0.011	0.005	0.000
		ti_fts	0.116	0.117	0.071	0.000
	Overall	ti	0.063	0.064	0.038	0.000
	FSP/FI	fsp_fi_ea_c	0.913	0.318	0.315	0.298
		fsp_fi_ea_n	0.985	0.586	0.583	0.568
	Overall	fsp_fi	0.949	0.452	0.449	0.430
Final score		OVERALL	0.204	0.112	0.106	0.094

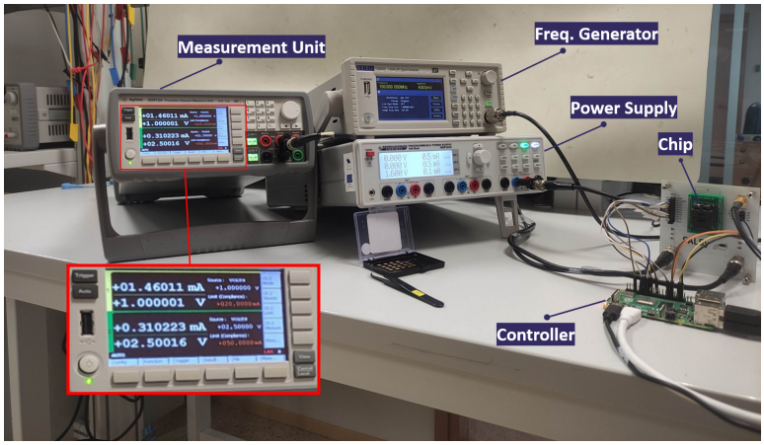

Figure 37: The testing environment for the fabricated chip

During this assessment, the inputs to the chip must be set and kept constant, without any changes or transitions. This allows the capturing of the baseline power consumption while the chip remains idle, without any specific operations underway.

Upon completing the AO leakage power assessment, the subsequent step involves evaluating leakage power for each specific block. This is accomplished by sequentially activating a single block, employing a customized configuration of input signals tailored for that block's voltage island. Each block is equipped with power switches, enabling selective activation or deactivation.

This precise power domain isolation significantly improves measurement accuracy by preventing power sharing with other blocks. Similar to the AO leakage measurement, no clock or other signals are toggled during this process. Consequently, this method accurately determines the power consumed by each individual block in isolation, providing insight into their specific power characteristics.

Figure 38 presents the results for leakage power. It is evident that different chips display unique power signatures, which can be attributed to process variation. These variations are inherent in the semiconductor fabrication process and can result in variations in power consumption among individual chips. The observed differences in leakage power emphasize the importance of process variation in chip manufacturing and underscore the necessity for comprehensive testing and analysis of power characteristics in real-world chip deployments.

Regarding static power, the overheads are on average 1.72%, 1.66%, 15.89%, and 7.24% for the PRESENT, SEED, Camellia, and CAST benchmarks, respectively.

Dynamic Power Measurement: The dynamic power measurement test is performed to evaluate the energy usage of each design block when operational. This is accomplished by sequentially activating each block and supplying them with the necessary inputs (i.e., plain text) at a clock frequency of 100MHz. These inputs are either derived from an internal register bank within the chip or provided by the host controller via the UART protocol. The results of this experiment are illustrated in Figure 39.

Across all benchmarks, the average overhead for dynamic power consumption remains below 3%. Specifically, the PRESENT, SEED, Camellia, and CAST benchmarks exhibit overheads of 0.79%, 0.86%, 2.02%, and 1.96%, respectively.

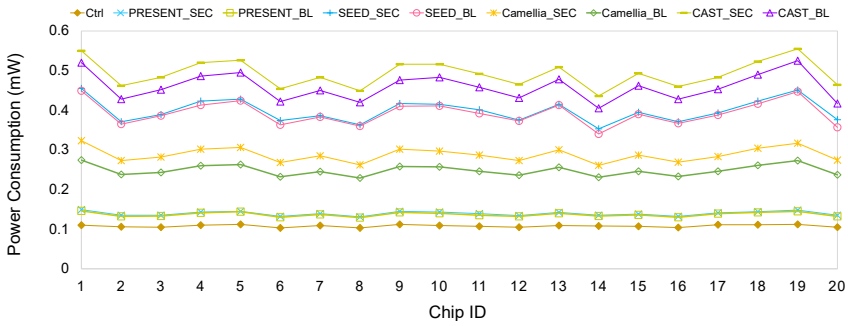

Figure 38: The measured leakage power (in mW) for 20 fabricated chips (from [69])

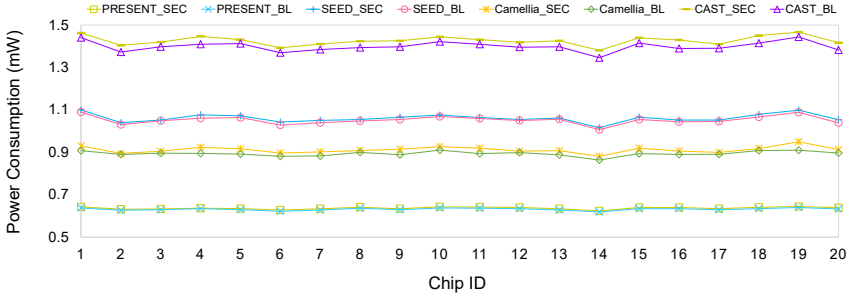

Figure 39: The measured dynamic power (in mW) for 20 fabricated chips (from [69])

5.6 SALSy Versus Other Techniques

SALSy is designed to counter post-design attacks, with its effectiveness measured using metrics from [71]. Nevertheless, it is suggested that redefining certain metrics may enhance the realism of the evaluation, thereby making the findings more applicable to industry practices. For instance, the threshold of 20 continuous gaps for the exploitable area might be overly optimistic, given that small HTs occupy only a few placement sites [161]. Additionally, in addressing FSP/FI threats, the goal is to protect the design. In an ideal scenario, attempts by attackers to compromise sensitive data, such as drilling holes (milling), should render the chip inoperable due to damage to protective nets above the sensitive ones. Consequently, simplistic defense strategies like covering the entire core area with a large metal plate should not be considered viable solutions, as such measures would not prevent attacks.

Furthermore, the scripted nature of SALSy reflects a deliberate decision that aligns with the scalability requirements of the industry, extending to even the most advanced technology nodes. While all proposed techniques can be readily adapted to sub-65nm technology, there are additional constraints for more advanced nodes. For instance, Non-default Rule CTS and Layer-targeted Routing rely on selecting wider wires, but there may be limited width options available. In older technologies, foundry recipes were more forgiving, allowing for virtually any width between 1X and 20X. However, in modern FinFET technology, available widths may be more discrete, such as 1X, 1.4X, 1.6X, and above. Thus, what was previously a continuum of valid widths has now become a discrete set.

While the concept of designing a security-aware place and route engine may hold academic appeal, implementing such a system could face scalability challenges when

applied to real-world, large-scale chip designs. Opting for a scripted approach with SALSy underscores its flexibility, enabling adaptation to diverse design sizes and complexities, including those featuring memory or analog macros. Notably, dealing with macros presents its own set of challenges, as security engineers have limited control over regions around macros due to high wire congestion. Consequently, employing techniques like Location-based Buffering in these areas poses challenges. However, it is worth noting that this issue also poses difficulties for potential attackers, as inserting malicious logic (i.e., HTs) in such congested layout regions is inherently challenging.

Additionally, standard deviation values for the leakage power consumption of both the baseline and secured versions of each block were calculated. For instance, the minimum values of standard deviation are 5.41×10^{-3} and 5.49×10^{-3} for the baseline and secured versions of the PRESENT benchmark, respectively. Conversely, the highest values of standard deviation are 3.46×10^{-2} for the baseline and 3.39×10^{-2} for the secured versions of the CAST benchmark. These results underscore that the SALSy approach demonstrates sensitivity to process variation comparable to that of the conventional security-unaware flow.

In addition, SALSy has been benchmarked against the most closely related existing works to provide readers with a clearer understanding of the significant differences in power, timing, area, and density promoted by SALSy. As illustrated in Table 11, this work stands out as the sole research that has verified the proposed techniques through practical implementation in silicon. In contrast, all other works aim for security closure, with some encountering various issues related to the utilization of limited PDKs/libraries. The ☺ symbol denotes improvement, the ☹ symbol signifies a decline in the metrics introduced, and the 😐 symbol indicates negligible changes after applying the individual technique. The term 'N/A' is used when the authors have not reported a metric. It is worth noting that an increase in density is considered beneficial as it enhances the security of the design in the face of the threats under consideration.

Table 11: Comparison of this work (SALSy) with the previous techniques

Ref.	Technique	Implications				Validated?
[155]	Internal Shielding	Power ☹	Timing ☹	Area ☹	Density ☺	✗
[66]	TroMUX	Power ☹	Timing ☹	Area 😐	Density ☺	✗
[130]	BISA	Power ☹	N/A	Area 😐	Density ☺	✗
[132]	Layout Filling	Power ☹	Timing ☹	Area 😐	Density ☺	✗
[67]	DEFense Framework	Power 😐	Timing ☹	Area 😐	Density ☺	✗
[65]	ASSURER	Power 😐	N/A	Area ☺	Density ☺	✗
[162]	T-TER	Power 😐	Timing 😐	Area 😐	Density 😐	✗
[68]	GDSII-Guard	Power ☹	Timing ☹	Area 😐	Density 😐	✗
	This thesis (SALSy)	Power ☹	Timing ☺	Area ☺	Density ☺	✓

6 Conclusions and Future Directions

In the modern era of technology, high-performance ICs have become essential across a wide range of applications, from AI and the IoT to automotive and aerospace sectors. However, the fabrication of these advanced ICs demands significant investment, leading to the globalization of the IC supply chain. This globalization, in turn, introduces various potential threats, including HTs, IP piracy, counterfeiting, and reverse engineering. This thesis addressed these challenges by developing novel methods and techniques to design ICs with inherent security considerations.

Chapter 3 introduced a novel approach to enhancing the security of digital designs through the reuse of verification assertions, specifically for HT detection. The chapter explained how assertions can be leveraged as online monitors and introduces a security metric and an assertion selection methodology utilizing the advanced capabilities of the Cadence JasperGold SPV tool. A comprehensive analysis of experimental outcomes demonstrated the adaptability and scalability of this method to industry-relevant circuits. The chapter concluded by highlighting the practicality of this detection solution, which is independent of the specific activation mechanisms of HTs, thus offering a versatile security enhancement for digital designs.

Chapter 4 presented a comprehensive method to enhance IC security throughout the back-end design process. It proposed a novel technique for incorporating online monitors during physical synthesis, adding an extra layer of protection at the back-end phase. While the integration of online monitors is not new, this work's unique contribution lies in directly incorporating checkers into the layout during the back-end phase, considering front-end inserted assertions. This strategy offers a more comprehensive security solution that spans both design phases. The flexibility of employing either technique independently allows for a customizable and adaptable solution to enhance IC security. Future work could focus on optimizing this methodology to reduce area and power overheads and incorporating path awareness for better utilization of positive setup slack time.

Chapter 5 introduced Security-Aware Layout Synthesis (SALSy), a new methodology enabling IC design with inherent security considerations. SALSy balances PPA metrics, and security. However, since commercial layout synthesis tools lack direct settings for security, SALSy adapts existing algorithms for placement, routing, CTS, and other tasks to make them security-aware. This strategy enhances IC security against both fabrication-time and post-fabrication adversarial acts, including HT insertion, FI, and probing. The methodology has been validated through a silicon-based demonstration, confirming its compatibility and effectiveness with a commercial PDK and library. SALSy achieves enhanced security with minimal impact on power consumption, maintaining a balance between security and efficiency. Although effective against HT insertion, future research will focus on improving defenses against fault injection and probing attacks, automating selective techniques, and introducing new evaluation metrics for a more comprehensive assessment.

This thesis has made significant contributions to the field of IC security, particularly in HT detection, enhancing security in the IC design process, and developing security-aware layout synthesis. The proposed methods and techniques, validated through extensive experimentation and simulation, have proven effective in mitigating various security threats. These findings provide a potent framework for developing more secure ICs and guiding future research in this critical area. By integrating security considerations into every phase of IC design, this work lays the groundwork for creating more secure ICs, essential for the technological advancements of the modern era.

In conclusion, the innovative approaches developed in this thesis offer a comprehensive and adaptable suite of solutions for enhancing IC security. They address the growing need

for secure protection mechanisms in the face of an increasingly globalized and vulnerable supply chain. As technology continues to grow, the insights and methodologies presented here will serve as a crucial foundation for ongoing advancements in hardware security, ensuring that the future of technology remains secure and resilient.

List of Figures

List of Tables

References

[1] I. Bojanova, "The digital revolution: What's on the horizon?," *IT Professional*, vol. 16, no. 1, pp. 8–12, 2014.

[2] IEEE Digital Reality, "The impacts that digital transformation has on society," last accessed on Apr. 03, 2024. Available at: `https://digitalreality.ieee.org/publications/impacts-of-digital-transformation`.

[3] Y.-Q. Lv, Q. Zhou, Y.-C. Cai, and G. Qu, "Trusted integrated circuits: The problem and challenges," *Journal of Computer Science and Technology*, vol. 29, pp. 918–928, Sep 2014.

[4] S. A. Campbell, "An introduction to microelectronic fabrication," in *Fabrication Engineering at the Micro and Nanoscale (3rd Edition)*, Oxford University Press, 2008.

[5] Yole Intelligence, "High-end performance packaging 2023," last accessed on Apr. 03, 2024. Available at: `https://www.yolegroup.com/product/report/high-end-performance-packaging-2023`.

[6] F. Alan, "TSMC's new 2nm chip production fab will cost it how much?," last accessed on Apr. 03, 2024. Available at: `https://www.phonearena.com/news/tsmc-to-spend-fortune-on-2nm-production-fab_id140626`.

[7] MacRumors, "Apple books nearly 90% of tsmc's 3nm production capacity for this year," last accessed on Apr. 03, 2024. Available at: `https://www.macrumors.com/2023/05/15/apple-tsmc-3nm-production-capacity/`.

[8] G. Moore, "Cramming more components onto integrated circuits," *Proceedings of the IEEE*, vol. 86, no. 1, pp. 82–85, 1998.

[9] S. Sze and K. K. Ng, *Physics of Semiconductor Devices*. John Wiley & Sons, Ltd, 3rd ed., 2006.

[10] A. S. Tanenbaum and D. J. Wetherall, *Computer Networks*. Prentice Hall Press, 5th ed., 2010.

[11] J. L. Hennessy and D. A. Patterson, *Computer Architecture, Sixth Edition: A Quantitative Approach*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 6th ed., 2017.

[12] N. Smith and A. Webb, *Introduction to Medical Imaging: Physics, Engineering and Clinical Applications*. Cambridge Texts in Biomedical Engineering, Cambridge University Press, 2010.

[13] N. Zaman, *Automotive Electronics Design Fundamentals*. Springer International Publishing, 2015.

[14] Statista, "Integrated circuits - worldwide," last accessed on Apr. 03, 2024. Available at: `https://www.statista.com/outlook/tmo/semiconductors/integrated-circuits/worldwide`.

[15] International Roadmap for Devices and Systems, "Semiconductors and artificial intelligence," last accessed on Apr. 03, 2024. Available at: `https://irds.ieee.org/topics/semiconductors-and-artificial-intelligence`.

[16] ElectronicsHub, "Integrated circuits-types , uses," last accessed on Apr. 03, 2024. Available at: https://www.electronicshub.org/integrated-circuits-types-uses/.

[17] C. A. Johan, G. Dieter, H. Guido, H. Denis, and H. Arndt, "How does the semiconductor industry landscape look today?," last accessed on Apr. 03, 2024. Available at: https://www.kearney.com/industry/technology/article/-/insights/how-does-the-semiconductor-industry-landscape-look-today.

[18] I. Kuon and J. Rose, "Measuring the gap between fpgas and asics," in *Proceedings of the 2006 ACM/SIGDA 14th International Symposium on Field Programmable Gate Arrays*, p. 21–30, 2006.

[19] A. Matthew, "Supply chain threats against integrated circuits," last accessed on Apr. 03, 2024. Available at: https://www.intel.com/content/dam/www/central-libraries/us/en/documents/2022-08/supply-chain-threats-against-integrated-circuits-whitepaper-july2020.pdf.

[20] T. D. Perez and S. Pagliarini, "Hardware trojan insertion in finalized layouts: From methodology to a silicon demonstration," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 7, pp. 2094–2107, 2023.

[21] S. E. Quadir, J. Chen, D. Forte, N. Asadizanjani, S. Shahbazmohamadi, L. Wang, J. Chandy, and M. Tehranipoor, "A survey on chip to system reverse engineering," *J. Emerg. Technol. Comput. Syst.*, vol. 13, apr 2016.

[22] S. M. Saeed, A. Zulehner, R. Wille, R. Drechsler, and R. Karri, "Reversible circuits: Ic/ip piracy attacks and countermeasures," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 11, pp. 2523–2535, 2019.

[23] F. Koushanfar, "Integrated circuits metering for piracy protection and digital rights management: An overview," in *Proceedings of the 21st Edition of the Great Lakes Symposium on Great Lakes Symposium on VLSI*, GLSVLSI '11, (New York, NY, USA), p. 449–454, Association for Computing Machinery, 2011.

[24] 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.

[25] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache, "Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures," *Proceedings of the IEEE*, vol. 100, no. 11, pp. 3056–3076, 2012.

[26] H. Wang, D. Forte, M. M. Tehranipoor, and Q. Shi, "Probing attacks on integrated circuits: Challenges and research opportunities," *IEEE Design & Test*, vol. 34, no. 5, pp. 63–71, 2017.

[27] A. Javeed, C. Yilmaz, and E. Savas, "Microarchitectural side-channel threats, weaknesses and mitigations: A systematic mapping study," *IEEE Access*, pp. 48945–48976, 2023.

[28] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware trojans," *Computer*, vol. 43, no. 10, pp. 39–46, 2010.

[29] C. Helfmeier, D. Nedospasov, C. Tarnovsky, J. S. Krissler, C. Boit, and J.-P. Seifert, "Breaking and entering through the silicon," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, (New York, NY, USA), p. 733–744, Association for Computing Machinery, 2013.

[30] B. Lippmann, A.-C. Bette, M. Ludwig, J. Mutter, J. Baehr, A. Hepp, H. Gieser, N. Kovač, T. Zweifel, M. Rasche, and O. Kellermann, "Physical and functional reverse engineering challenges for advanced semiconductor solutions," in *Proceedings of the 2022 Conference & Exhibition on Design, Automation & Test in Europe*, DATE '22, (Leuven, BEL), p. 796–801, European Design and Automation Association, 2022.

[31] R. S. Rajarathnam, Y. Lin, Y. Jin, and D. Z. Pan, "ReGDS: A reverse engineering framework from GDSII to gate-level netlist," in *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 154–163, 2020.

[32] R. Torrance and D. James, "The state-of-the-art in ic reverse engineering," in *Cryptographic Hardware and Embedded Systems - CHES 2009* (C. Clavier and K. Gaj, eds.), (Berlin, Heidelberg), pp. 363–381, Springer Berlin Heidelberg, 2009.

[33] M. Eslami, B. Ghavami, M. Raji, and A. Mahani, "A survey on fault injection methods of digital integrated circuits," *Integration*, vol. 71, pp. 154–163, 2020.

[34] P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks: Exploiting speculative execution," in *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 1–19, 2019.

[35] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg, "Meltdown: Reading kernel memory from user space," in *27th USENIX Security Symposium (USENIX Security 18)*, pp. 973–990, 2018.

[36] J. Simpson, "Asml halts hi-tech chip-making exports to china reportedly after us request," last accessed on May 08, 2024. Available at: `https://www.theguardian.com/technology/2024/jan/02/asml-halts-hi-tech-chip-making-exports-to-china-reportedly-after-us-request`.

[37] European Union Agency for Cybersecurity, "ENISA threat landscape 2023," last accessed on Apr. 03, 2024. Available at: `https://www.enisa.europa.eu/publications/enisa-threat-landscape-2023`.

[38] Pricewaterhouse Coopers, "Global economic crime and fraud survey 2020," last accessed on May. 20, 2024. Available at: `https://www.pwc.com/gx/en/services/forensics/economic-crime-survey.html`.

[39] 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.

[40] F. Wolff, C. Papachristou, S. Bhunia, and R. S. Chakraborty, "Towards trojan-free trusted ics: Problem analysis and detection scheme," in *2008 Design, Automation and Test in Europe*, pp. 1362–1365, 2008.

[41] X. Chen, G. Qu, and A. Cui, "Practical IP watermarking and fingerprinting methods for ASIC designs," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–4, 2017.

[42] H. Huang, A. Boyer, and S. B. Dhia, "The detection of counterfeit integrated circuit by the use of electromagnetic fingerprint," in *2014 International Symposium on Electromagnetic Compatibility*, pp. 1118–1122, 2014.

[43] Y. Jin, E. Love, and Y. Makris, *Design for Hardware Trust*, pp. 365–384. New York, NY: Springer New York, 2012.

[44] M. Yasin, J. Rajendran, and O. Sinanoglu, "Trustworthy hardware design: Combinational logic locking techniques," Springer, Cham, 2019.

[45] O. Harrison and J. Waldron, "Practical symmetric key cryptography on modern graphics hardware," in *Proceedings of the 17th Conference on Security Symposium*, p. 195–209, 2008.

[46] 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 *Proceedings of the 35th Annual Design Automation Conference*, DAC '98, (New York, NY, USA), p. 776–781, Association for Computing Machinery, 1998.

[47] M. Khan and S. Tragoudas, "Rewiring for watermarking digital circuit netlists," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 7, pp. 1132–1137, 2005.

[48] M. Lewandowski, R. Meana, M. Morrison, and S. Katkoori, "A novel method for watermarking sequential circuits," in *2012 IEEE International Symposium on Hardware-Oriented Security and Trust*, pp. 21–24, 2012.

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

[50] J. Rajendran, O. Sinanoglu, and R. Karri, "Is split manufacturing secure?," in *2013 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 1259–1264, 2013.

[51] A. Sengupta, M. Nabeel, J. Knechtel, and O. Sinanoglu, "A new paradigm in split manufacturing: Lock the feol, unlock at the beol," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 414–419, 2019.

[52] Y. Wang, P. Chen, J. Hu, and J. J. Rajendran, "The cat and mouse in split manufacturing," in *Proceedings of the 53rd Annual Design Automation Conference*, DAC '16, (New York, NY, USA), Association for Computing Machinery, 2016.

[53] K. Zamiri Azar, H. Mardani Kamali, H. Homayoun, and A. Sasan, "Threats on logic locking: A decade later," in *GLSVLSI '19: Proceedings of the 2019 on Great Lakes Symposium on VLSI*, p. 471–476, 2019.

[54] Y. Xie and A. Srivastava, "Delay locking: Security enhancement of logic locking against IC counterfeiting and overproduction," in *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2017.

[55] M. Yasin and O. Sinanoglu, "Evolution of logic locking," in *2017 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, pp. 1–6, 2017.

[56] J. Sweeney, V. Mohammed Zackriya, S. Pagliarini, and L. Pileggi, "Latch-based logic locking," in *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 132–141, 2020.

[57] Y. Liu, M. Zuzak, Y. Xie, A. Chakraborty, and A. Srivastava, "Strong Anti-SAT: Secure and effective logic locking," in *2020 21st International Symposium on Quality Electronic Design (ISQED)*, pp. 199–205, 2020.

[58] M. Yasin, B. Mazumdar, J. J. V. Rajendran, and O. Sinanoglu, "Sarlock: Sat attack resistant logic locking," in *2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 236–241, 2016.

[59] Y. Xie and A. Srivastava, "Anti-sat: Mitigating sat attack on logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 2, pp. 199–207, 2019.

[60] S. D. Hampton and A. J. Bailey, "Intellectual property case filing trends over the last decade," last accessed on May. 20, 2024. Available at: `https://www.hamptonip.com/articles/post/intellectual-property-case-filing-trends-over-the-last-decade/`.

[61] V. Gohil, H. Guo, S. Patnaik, and J. Rajendran, "Attrition: Attacking static hardware trojan detection techniques using reinforcement learning," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, p. 1275–1289, 2022.

[62] X. Wei, J. Zhang, and G. Luo, "Rethinking ic layout vulnerability: Simulation-based hardware trojan threat assessment with high fidelity," in *2024 IEEE Symposium on Security and Privacy (SP)*, pp. 163–163, 2024.

[63] J. Bhandari, L. Mankali, M. Nabeel, O. Sinanoglu, R. Karri, and J. Knechtel, "Beware your standard cells! on their role in static power side-channel attacks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–14, 2024.

[64] F. Wang, Q. Wang, L. Alrahis, B. Fu, S. Jiang, X. Zhang, O. Sinanoglu, T.-Y. Ho, E. F. Y. Young, and J. Knechtel, "Trolloc: Logic locking and layout hardening for ic security closure against hardware trojans," 2024.

[65] G. Guo, H. You, Z. Tang, B. Li, C. Li, and X. Zhang, "Assurer: A ppa-friendly security closure framework for physical design," in *2023 28th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 504–509, 2023.

[66] F. Wang, Q. Wang, B. Fu, S. Jiang, X. Zhang, L. Alrahis, O. Sinanoglu, J. Knechtel, T.-Y. Ho, and E. F. Young, "Security closure of ic layouts against hardware trojans," in *Proceedings of the 2023 International Symposium on Physical Design*, ISPD '23, (New York, NY, USA), p. 229–237, Association for Computing Machinery, 2023.

[67] J. Knechtel, J. Gopinath, J. Bhandari, M. Ashraf, H. Amrouch, S. Borkar, S.-K. Lim, O. Sinanoglu, and R. Karri, "Security closure of physical layouts iccad special session paper," in *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pp. 1–9, 2021.

[68] X. Wei, J. Zhang, and G. Luo, "Gdsii-guard: Eco anti-trojan optimization with exploratory timing-security trade-offs," in *2023 60th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2023.

[69] M. Eslami, T. Perez, and S. Pagliarini, "Salsy: Security-aware layout synthesis," 2023. Available at: `https://arxiv.org/abs/2308.06201v2`.

[70] M. Eslami, J. Knechtel, O. Sinanoglu, R. Karri, and S. Pagliarini, "Benchmarking advanced security closure of physical layouts: Ispd 2023 contest," in *Proceedings of the 2023 International Symposium on Physical Design*, ISPD '23, (New York, NY, USA), p. 256–264, Association for Computing Machinery, 2023.

[71] J. Knechtel, J. Gopinath, M. Ashraf, J. Bhandari, O. Sinanoglu, and R. Karri, "Benchmarking security closure of physical layouts: Ispd 2022 contest," ISPD '22, (New York, NY, USA), p. 221–228, Association for Computing Machinery, 2022.

[72] Y. Li, W. Wu, L. Hou, and H. Cheng, "A study on the assertion-based verification of digital ic," in *2009 Second International Conference on Information and Computing Science*, vol. 2, pp. 25–28, 2009.

[73] M. Eslami, T. Ghasempouri, and S. Pagliarini, "Reusing verification assertions as security checkers for hardware trojan detection," in *2022 23rd International Symposium on Quality Electronic Design (ISQED)*, pp. 1–6, 2022.

[74] K. Alatoun, B. Shankaranarayanan, S. M. Achyutha, and R. Vemuri, "Soc trust validation using assertion-based security monitors," in *2021 22nd International Symposium on Quality Electronic Design (ISQED)*, pp. 496–503, 2021.

[75] E. Seligman, T. Schubert, and M. V. A. K. Kumar, "Chapter 1 - formal verification: from dreams to reality," in *Formal Verification (Second Edition)*, pp. 1–23, 2023.

[76] A. T. Sonny and S. Lakshmiprabha, "Ovl, psl, sva: Assertion based verification using checkers and standard assertion languages," in *2013 International Conference on Advanced Computing and Communication Systems*, pp. 1–4, 2013.

[77] F. Semiconductor, "A guide to ic design flow," last accessed on Jul. 11, 2024. Available at: `https://fpt-semiconductor.com/blogs/a-guide-to-ic-design-flow/`.

[78] C. Mead and L. Conway, *Introduction to VLSI Systems*. Addison-Wesley series in computer science and information processing, Addison-Wesley, 1980.

[79] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[80] Techovedas, "10 fabrication steps to build a semiconductor chip," last accessed on Apr. 03, 2024. Available at: `https://techovedas.com/10-fabrication-steps-to-build-a-semiconductor-chip/`.

[81] SUMCO, "About silicon wafers," last accessed on Apr. 03, 2024. Available at: `https://www.sumcosi.com/english/products/about.html`.

[82] M. Madou, *Fundamentals of Microfabrication: The Science of Miniaturization, Second Edition*. Taylor & Francis, 2002.

[83] R. Pierret, *Advanced Semiconductor Fundamentals*. Modular series on solid state devices, Prentice Hall, 2003.

[84] J. Lau, *Ball Grid Array Technology*. Electronic packaging and interconnection series, McGraw-Hill, 1995.

[85] D. Wang, "The golden age of ic design in taiwan," last accessed on May. 20, 2024. Available at: `https://nextrendsasia.org/the-golden-age-of-ic-design-in-taiwan/`.

[86] J. Vosatka, *Introduction to Hardware Trojans*, pp. 15–51. 2018.

[87] T. Trippel, K. G. Shin, K. B. Bush, and M. Hicks, "Icas: an extensible framework for estimating the susceptibility of ic layouts to additive trojans," in *2020 IEEE Symposium on Security and Privacy (SP)*, pp. 1742–1759, 2020.

[88] M. Eslami, T. Ghasempouri, and S. Pagliarini, "Scarf: Securing chips with a robust framework against fabrication-time hardware trojans," *IEEE Transactions on Computers*, pp. 1–14, 2024.

[89] S. Bhunia and M. Tehranipoor, *Hardware Security - A Hands-On Learning Approach*. Elsevier, 2019.

[90] IC Failure Analysis Lab, "Delayering / parallel lapping," last accessed on Apr. 03, 2024. Available at: `https://icfailureanalysis.com/delayering-parallel-lapping`.

[91] Y. Jin, B. Yang, and Y. Makris, "Cycle-accurate information assurance by proof-carrying based signal sensitivity tracing," in *2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pp. 99–106, 2013.

[92] X. Guo, R. G. Dutta, Y. Jin, F. Farahmandi, and P. Mishra, "Pre-silicon security verification and validation: A formal perspective," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2015.

[93] M. Qin, W. Hu, D. Mu, and Y. Tai, "Property based formal security verification for hardware trojan detection," in *2018 IEEE 3rd International Verification and Security Workshop (IVSW)*, pp. 62–67, 2018.

[94] G. Shrestha and M. S. Hsiao, "Ensuring trust of third-party hardware design with constrained sequential equivalence checking," in *2012 IEEE Conference on Technologies for Homeland Security (HST)*, pp. 7–12, 2012.

[95] A. Biere, A. Cimatti, E. Clarke, M. Fujita, and Y. Zhu, "Symbolic model checking using sat procedures instead of bdds," in *Proceedings 1999 Design Automation Conference (Cat. No. 99CH36361)*, pp. 317–320, 1999.

[96] A. C. Myers and B. Liskov, "A decentralized model for information flow control," in *Proceedings of the Sixteenth ACM Symposium on Operating Systems Principles*, SOSP '97, (New York, NY, USA), p. 129–142, Association for Computing Machinery, 1997.

[97] J. Rajendran, V. Vedula, and R. Karri, "Detecting malicious modifications of data in third-party intellectual property cores," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2015.

[98] J. Rajendran, A. M. Dhandayuthapany, V. Vedula, and R. Karri, "Formal security verification of third party intellectual property cores for information leakage," in *2016 29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems (VLSID)*, pp. 547–552, 2016.

[99] A. Nahiyan, M. Sadi, R. Vittal, G. Contreras, D. Forte, and M. Tehranipoor, "Hardware trojan detection through information flow security verification," in *2017 IEEE International Test Conference (ITC)*, pp. 1–10, 2017.

[100] A. Waksman, M. Suozzo, and S. Sethumadhavan, "Fanci: identification of stealthy malicious logic using boolean functional analysis," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, (New York, NY, USA), p. 697–708, Association for Computing Machinery, 2013.

[101] J. Zhang, F. Yuan, and Q. Xu, "Detrust: Defeating hardware trust verification with stealthy implicitly-triggered hardware trojans," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, (New York, NY, USA), p. 153–166, Association for Computing Machinery, 2014.

[102] J. Zhang, F. Yuan, L. Wei, Y. Liu, and Q. Xu, "Veritrust: Verification for hardware trust," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 7, pp. 1148–1161, 2015.

[103] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using ic fingerprinting," in *2007 IEEE Symposium on Security and Privacy (SP '07)*, pp. 296–310, 2007.

[104] B. Zhou, R. Adato, M. Zangeneh, T. Yang, A. Uyar, B. Goldberg, S. Unlu, and A. Joshi, "Detecting hardware trojans using backside optical imaging of embedded watermarks," in *Proceedings of the 52nd Annual Design Automation Conference*, DAC '15, (New York, NY, USA), Association for Computing Machinery, 2015.

[105] M. Banga and M. S. Hsiao, "A novel sustained vector technique for the detection of hardware trojans," in *2009 22nd International Conference on VLSI Design*, pp. 327–332, 2009.

[106] R. S. Chakraborty and S. Bhunia, "Security against hardware trojan through a novel application of design obfuscation," in *2009 IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*, pp. 113–116, 2009.

[107] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, "Hardware trojan attacks: Threat analysis and countermeasures," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1229–1247, 2014.

[108] X. Wang, M. Tehranipoor, and J. Plusquellic, "Detecting malicious inclusions in secure hardware: Challenges and solutions," in *2008 IEEE International Workshop on Hardware-Oriented Security and Trust*, pp. 15–19, 2008.

[109] J. Aarestad, D. Acharyya, R. Rad, and J. Plusquellic, "Detecting trojans through leakage current analysis using multiple supply pad I_{ddq}s," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 893–904, 2010.

[110] D. Forte, C. Bao, and A. Srivastava, "Temperature tracking: An innovative run-time approach for hardware trojan detection," in *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 532–539, 2013.

[111] F. Stellari, P. Song, A. J. Weger, J. Culp, A. Herbert, and D. Pfeiffer, "Verification of untrusted chips using trusted layout and emission measurements," in *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pp. 19–24, 2014.

[112] C. Sturton, M. Hicks, D. Wagner, and S. T. King, "Defeating uci: Building stealthy and malicious hardware," in *2011 IEEE Symposium on Security and Privacy*, pp. 64–77, 2011.

[113] C. Cadar, D. Dunbar, and D. Engler, "Klee: unassisted and automatic generation of high-coverage tests for complex systems programs," in *Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation*, OSDI'08, (USA), p. 209–224, USENIX Association, 2008.

[114] S. Mitra and E. McCluskey, "Which concurrent error detection scheme to choose ?," in *Proceedings International Test Conference 2000 (IEEE Cat. No.00CH37159)*, pp. 985–994, 2000.

[115] O. Keren, I. Levin, and M. Karpovsky, "Duplication based one-to-many coding for trojan hw detection," in *2010 IEEE 25th International Symposium on Defect and Fault Tolerance in VLSI Systems*, pp. 160–166, 2010.

[116] J. Rajendran, H. Zhang, O. Sinanoglu, and R. Karri, "High-level synthesis for security and trust," in *2013 IEEE 19th International On-Line Testing Symposium (IOLTS)*, pp. 232–233, 2013.

[117] D. McIntyre, F. Wolff, C. Papachristou, and S. Bhunia, "Trustworthy computing in a multi-core system using distributed scheduling," in *2010 IEEE 16th International On-Line Testing Symposium*, pp. 211–213, 2010.

[118] C. Liu, J. Rajendran, C. Yang, and R. Karri, "Shielding heterogeneous mpsocs from untrustworthy 3pips through security-driven task scheduling," in *2013 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*, pp. 101–106, 2013.

[119] T. Reece, D. B. Limbrick, and W. H. Robinson, "Design comparison to identify malicious hardware in external intellectual property," in *2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 639–646, 2011.

[120] G. Bloom, B. Narahari, and R. Simha, "Os support for detecting trojan circuit attacks," in *2009 IEEE International Workshop on Hardware-Oriented Security and Trust*, pp. 100–103, 2009.

[121] J. Dubeuf, D. Hély, and R. Karri, "Run-time detection of hardware trojans: The processor protection unit," in *2013 18th IEEE European Test Symposium (ETS)*, pp. 1–6, 2013.

[122] S. Narasimhan, W. Yueh, X. Wang, S. Mukhopadhyay, and S. Bhunia, "Improving ic security against trojan attacks through integration of security monitors," *IEEE Design & Test of Computers*, vol. 29, no. 5, pp. 37–46, 2012.

[123] Y. Jin and D. Sullivan, "Real-time trust evaluation in integrated circuits," in *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1–6, 2014.

[124] Y. Jin, D. Maliuk, and Y. Makris, "Post-deployment trust evaluation in wireless cryptographic ics," in *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 965–970, 2012.

[125] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *DAC Design Automation Conference 2012*, pp. 83–89, 2012.

[126] 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*, vol. 28, no. 10, pp. 1493–1502, 2009.

[127] J. A. Roy, F. Koushanfar, and I. L. Markov, "Epic: Ending piracy of integrated circuits," in *2008 Design, Automation and Test in Europe*, pp. 1069–1074, 2008.

[128] A. Baumgarten, A. Tyagi, and J. Zambreno, "Preventing ic piracy using reconfigurable logic barriers," *IEEE Design & Test of Computers*, vol. 27, no. 1, pp. 66–75, 2010.

[129] J. B. Wendt and M. Potkonjak, "Hardware obfuscation using puf-based logic," in *Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design*, ICCAD '14, p. 270–277, IEEE Press, 2014.

[130] K. Xiao and M. Tehranipoor, "Bisa: Built-in self-authentication for preventing hardware trojan insertion," in *2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pp. 45–50, 2013.

[131] K. Xiao, D. Forte, and M. Tehranipoor, "A novel built-in self-authentication technique to prevent inserting hardware trojans," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 12, pp. 1778–1791, 2014.

[132] P.-S. Ba, S. Dupuis, M. Palanichamy, M.-L. Flottes, G. Di Natale, and B. Rouzeyre, "Hardware trust through layout filling: A hardware trojan prevention technique," in *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 254–259, 2016.

[133] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security analysis of integrated circuit camouflaging," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, (New York, NY, USA), p. 709–720, Association for Computing Machinery, 2013.

[134] R. P. Cocchi, J. P. Baukus, L. W. Chow, and B. J. Wang, "Circuit camouflage integration for hardware ip protection," in *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–5, 2014.

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

[136] R. P. Cocchi, J. P. Baukus, L. W. Chow, and B. J. Wang, "Circuit camouflage integration for hardware IP protection," in *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–5, 2014.

[137] M. Li, K. Shamsi, T. Meade, Z. Zhao, B. Yu, Y. Jin, and D. Z. Pan, "Provably secure camouflaging strategy for IC protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 8, pp. 1399–1412, 2019.

[138] Y. Bi, P.-E. Gaillardon, X. S. Hu, M. Niemier, J.-S. Yuan, and Y. Jin, "Leveraging emerging technology for hardware security - case study on silicon nanowire fets and graphene symfets," in *2014 IEEE 23rd Asian Test Symposium*, pp. 342–347, 2014.

[139] Cadence Design Systems, Inc., "Jasper spv app," last accessed on May. 20, 2024. Available at: `https://www.cadence.com/en_US/home/tools/system-design-and-verification/formal-and-static-verification/jasper-gold-verification-platform/security-path-verification-app.html`.

[140] "Opentitan documentation index," last accessed on May. 24, 2024. Available at: `https://opentitan.org/book/doc/introduction.html`.

[141] T. F. Wu, K. Ganesan, Y. A. Hu, H.-S. P. Wong, S. Wong, and S. Mitra, "Tpad: Hardware trojan prevention and detection for trusted integrated circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 4, pp. 521–534, 2016.

[142] H. Salmani, M. Tehranipoor, and R. Karri, "On design vulnerability analysis and trust benchmarks development," in *IEEE 31st Intl. Conf. on Computer Design (ICCD)*, pp. 471–474, 2013.

[143] F. Corno, M. Reorda, and G. Squillero, "RT-level ITC'99 benchmarks and first ATPG results," *IEEE Des. Test Cmput.*, vol. 17, no. 3, pp. 44–53, 2000.

[144] B. Shakya, T. He, H. Salmani, D. Forte, S. Bhunia, and M. Tehranipoor, "Benchmarking of Hardware Trojans and Maliciously Affected Circuits," *J. of Hardw. and Sys. Security*, vol. 1, no. 1, pp. 85–102, 2017.

[145] W. Cullyer, "Implementing high integrity systems: the VIPER microprocessor," *IEEE Aerospace and Electronic Systems Magazine*, vol. 4, no. 6, pp. 5–13, 1989.

[146] M. Boulé and Z. Zilic, "Automata-based assertion-checker synthesis of PSL properties," *ACM TDAES*, vol. 13, no. 1, pp. 1–21, 2008.

[147] A. Hepp, T. Perez, S. Pagliarini, and G. Sigl, "A pragmatic methodology for blind hardware trojan insertion in finalized layouts," ICCAD '22, 2022.

[148] Y. Jin and Y. Makris, "Hardware trojan detection using path delay fingerprint," in *2008 IEEE International Workshop on Hardware-Oriented Security and Trust*, pp. 51–57, 2008.

[149] I. Exurville, L. Zussa, J.-B. Rigaud, and B. Robisson, "Resilient hardware trojans detection based on path delay measurements," in *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 151–156, 2015.

[150] X.-T. Ngo, I. Exurville, S. Bhasin, J.-L. Danger, S. Guilley, Z. Najm, J.-B. Rigaud, and B. Robisson, "Hardware trojan detection by delay and electromagnetic measurements," in *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 782–787, 2015.

[151] S. Dupuis, P.-S. Ba, M.-L. Flottes, G. Di Natale, and B. Rouzeyre, "New testing procedure for finding insertion sites of stealthy hardware trojans," in *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 776–781, 2015.

[152] N. Lesperance, S. Kulkarni, and K.-T. Cheng, "Hardware trojan detection using exhaustive testing of k-bit subspaces," in *The 20th Asia and South Pacific Design Automation Conference*, pp. 755–760, 2015.

[153] 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.

[154] W. Hu, C.-H. Chang, A. Sengupta, S. Bhunia, R. Kastner, and H. Li, "An overview of hardware security and trust: Threats, countermeasures, and design tools," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 6, pp. 1010–1038, 2021.

[155] H. Wang, Q. Shi, A. Nahiyan, D. Forte, and M. M. Tehranipoor, "A physical design flow against front-side probing attacks by internal shielding," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2152–2165, 2020.

[156] T. Sugawara, N. Homma, T. Aoki, and A. Satoh, "Asic performance comparison for the iso standard block ciphers," in *JWIS*, 2007.

[157] O. Girard et al., "openmsp430 at opencores.org," 2021. Available at: `https://opencores.org/projects/openmsp430`.

[158] M. Hicks et al., "Mit-ll common evaluation platform (cep) at github.com," 2021. Available at: `https://github.com/mit-ll/CEP`.

[159] SILVACO, "Nangate freepdk45 open cell library." Available at: `http://www.nangate.com/?page_id=2325`.

[160] "Salsy repository." Available at: `https://github.com/Centre-for-Hardware-Security/SALSy`.

[161] K. Yang, M. Hicks, Q. Dong, T. Austin, and D. Sylvester, "A2: Analog malicious hardware," in *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 18–37, 2016.

[162] T. Trippel, K. G. Shin, K. B. Bush, and M. Hicks, "T-ter: Defeating a2 trojans with targeted tamper-evident routing," in *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security*, ASIA CCS '23, (New York, NY, USA), p. 746–759, Association for Computing Machinery, 2023.

Abstract
On the Use of Defensive Schemes for Hardware Security

The digitalization era has profoundly transformed daily life, driven in part by the crucial role of Integrated Circuits (ICs) in modern electronics. These microelectronic components, central to devices ranging from smartphones to advanced computing systems, have been pivotal in technological advancements. However, the globalization of IC fabrication introduces significant security risks such as Hardware Trojans (HTs), intellectual property theft, counterfeiting, and reverse engineering.

This thesis addresses these security challenges by developing innovative methodologies for designing ICs with inherent security features. The first contribution is a novel approach that reuses verification assertions for HT detection. By leveraging the capabilities of the Cadence JasperGold Security Path Verification tool, assertions are transformed into online monitors, supported by a security metric and an assertion selection process. Experimental results, applied to diverse IPs within the OpenTitan System-on-Chip, validate the adaptability and scalability of this approach.

The second contribution enhances IC security during the back-end design phase by embedding online monitors directly into the layout, considering front-end inserted assertions. This dual-phase strategy, applied during physical synthesis, provides a comprehensive and flexible security solution that can be customized based on specific requirements.

The thesis further introduces Security-Aware Layout Synthesis (SALSy), a methodology that integrates security considerations into standard layout synthesis processes. SALSy adapts placement, routing, and clock tree synthesis algorithms to be security-aware, thereby protecting ICs against HT insertion, fault injection, and probing attacks. Validated through silicon-based implementations, SALSy demonstrates effectiveness with minimal impact on power consumption.

Overall, this thesis makes significant contributions to IC security by proposing methods for HT detection, improving security in the IC design process, and developing security-aware layout synthesis techniques. These solutions have been rigorously tested and proven to mitigate various security threats, providing a robust framework for future IC designs. By embedding security at every stage of the design process, this work lays the foundation for creating more secure and resilient technological advancements in an increasingly globalized supply chain.