Security and Trust for Wireless Integrated Circuits

Alán Rodrigo Díaz Rizo

THÈSE DE DOCTORAT
DE SORBONNE UNIVERSITÉ

SECURITY AND TRUST FOR WIRELESS INTEGRATED CIRCUITS

présentée par

ALÁN RODRIGO DÍAZ RIZO

École Doctorale Informatique, Télécommunications et Électronique (EDITE)

réalisée au laboratoire

LIP6

Presentée et soutenue publiquement le 10 janvier 2023

Devant un jury composé de :

Johanna Sepúlveda	Rapporteuse Airbus Defense and Space GmbH, Germany
Lilian Bossuet	Rapporteur Professeur, Université Jean Monnet, Lab. Hubert Curien, France
Jacques Fournier	Examinateur CEA-LETI, France
Giorgio Di Natale	Examinateur Directeur de Recherche au CNRS, Université Grenoble Alpes, TIMA, France
Hassan Aboushady	Co-direteur de thèse Maître de conférences, Sorbonne Université, LIP6, France
Haralampos Stratigopoulos	Directeur de thèse Directeur de Recherche au CNRS, Sorbonne Université, LIP6, France

Alán Rodrigo Díaz Rizo: *Security and Trust for Wireless Integrated Circuits,* 2023.

ABSTRACT

The origin of the hardware security threats is the massively globalized and outsourcing-based Integrated Circuit (IC) supply chain that we see today. The prohibitively cost of owning a first-rate semiconductor foundry forces IC design houses to go fabless and outsource their IC fabrication, assembly, and testing. Outsourcing these tasks intensifies the risk of IC piracy attacks and Hardware Trojan (HT) insertion, and both threats translate into know-how and financial losses for the IC owner. Moreover, complex Systems-on-Chip (SoCs) are built by integrating third-party Intellectual Property (IP) cores from multiple IP providers. However, SoC integrators and IP providers have an imbalanced trust relationship. While IP providers are vulnerable to IP overuse, IP cloning, and IC overproduction, SoC integrators fear integrating HT-infected IPs into their systems.

IC supply chain attackers target wireless ICs because wireless ICs have a critical role in society, government, and industry, and they exchange sensitive and valuable information through a publicly accessible medium. Therefore, this thesis focuses on wireless ICs and addresses two IC supply chain attacks: IP/IC piracy and HT insertion.

To overcome IP/IC piracy of wireless ICs, we propose a locking-based design-for-security methodology. Locking transforms the original design into a functionally equivalent one, but the functionality depends on a correct secret key, otherwise any incorrect key corrupts the functionality. Locking cannot be blindly applied to RF transceivers, to that end we show how to fine-tune state-of-the-art techniques to achieve piracy protection. In addition, we develop an RF transceiver-specific locking methodology that consists in two spatially separated mechanisms. The solution is called *SyncLock* since it locks the synchronization of the transmitter with the receiver. If a key other than the secret key is applied, synchronization and, thereby, communication fails.

To assess the security of wireless ICs, we propose an HT attack that leaks sensitive information from the transmitter within a legitimate transmission. A rogue receiver can recover the sensitive data using signal processing, while the legitimate receiver is inconspicuous and does not realize the information leaking. We demonstrate the stealthiness and robustness of the attack with measurement on a hardware platform.

RESUMÉ

L'origine des menaces de sécurité matérielle est la globalisation et l'externalisation massives de la chaîne d'approvisionnement des circuits intégrés (CI) que nous connaissons aujourd'hui. Le coût prohibitif de la possession d'une fonderie de semi-conducteurs de haut niveau oblige les sociétés de conception de circuits intégrés à opter pour une production sans usine et à externaliser la fabrication, l'assemblage et le test de leurs circuits. L'externalisation de ces tâches intensifie le risque d'attaques de piratage de CI et d'insertion de chevaux de Troie matériels (HT), et ces deux menaces se traduisent par des pertes de savoir-faire et des pertes financières pour le propriétaire du CI. En outre, les systèmes sur puce (SoC) complexes sont construits en intégrant des noyaux de propriété intellectuelle (IP cores) de tiers provenant de plusieurs fournisseurs des IP cores. Cependant, les intégrateurs de SoC et les fournisseurs des IP ont une relation de confiance déséquilibrée. Alors que les fournisseurs des IP sont vulnérables à la surutilisation de la IP, au clonage de la IP et à la surproduction des CI, les intégrateurs de SoC craignent d'intégrer des IP infectées par des HT dans leurs systèmes.

Les attaquants de la chaîne d'approvisionnement des circuits intégrés ciblent les circuits intégrés sans fil parce que ces derniers jouent un rôle essentiel dans la société, le gouvernement et l'industrie et qu'ils échangent des informations sensibles et précieuses par le biais d'un support accessible au public. Par conséquent, cette thèse se concentre sur les CI sans fil et aborde deux attaques de la chaîne d'approvisionnement des CI : le piratage IP/IC et l'insertion HT.

Pour surmonter le piratage des IP/CI des circuits intégrés sans fil, nous proposons une méthodologie de conception pour la sécurité basée sur le verrouillage. Le verrouillage transforme la conception originale en une conception fonctionnellement équivalente, mais la fonctionnalité dépend d'une clé secrète correcte, sinon toute clé incorrecte corrompt la fonctionnalité. Le verrouillage ne peut pas être appliqué aveuglément aux émetteurs-récepteurs RF. À cette fin, nous montrons comment affiner les techniques de l'état de l'art pour obtenir une protection contre le piratage. En outre, nous développons une méthodologie de verrouillage spécifique aux émetteurs-récepteurs RF qui consiste en deux mécanismes séparés spatialement. La solution est appelée *SyncLock* car elle verrouille la synchronisation de l'émetteur avec le récepteur. Si une clé autre que la clé secrète est appliquée, la synchronisation et, par conséquent, la communication échouent.

Pour évaluer la sécurité des circuits intégrés sans fil, nous proposons une attaque HT qui fait fuir des informations sensibles

de l'émetteur au cours d'une transmission légitime. Un récepteur malveillant peut récupérer les données sensibles en utilisant le traitement du signal, tandis que le récepteur légitime est discret et ne se rend pas compte de la fuite d'informations. Nous démontrons la furtivité et la robustesse de l'attaque par des mesures sur une plateforme matérielle.

PUBLICATIONS

During the doctorate program, the following scientific articles were published:

[1] A. R. Díaz Rizo, H. Aboushady, and H.-G. Stratigopoulos. "SyncLock: RF transceiver security using synchronization locking." In: *2022 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. IEEE. 2022, pp. 1153–1156.

[2] A. R. Díaz Rizo, J. Leonhard, H. Aboushady, and H.-G. Stratigopoulos. "RF Transceiver Security Against Piracy Attacks." In: *IEEE Transactions on Circuits and Systems II: Express Briefs* (2022), pp. 1–1. DOI: 10.1109/TCSII.2022.3165709.

[3] A. R. Díaz Rizo, H. Aboushady, and H.-G. Stratigopoulos. "Anti-Piracy Design of RF Transceivers." In: *IEEE Transactions on Circuits and Systems I: Regular Papers* (2022), pp. 1–14. DOI: 10.1109/TCSI.2022.3214111.

[4] A. R. Díaz Rizo, H. Aboushady, and H.-G. Stratigopoulos. "Leaking Wireless ICs via Hardware Trojan-Infected Synchronization." In: *IEEE Transactions on Dependable and Secure Computing* (2022), pp. 1–16. DOI: 10.1109/TDSC.2022.3218507.

The following scientific articles were published in collaboration:

[1] J. Leonhard, N. Limaye, S. Turk, A. Sayed, A. R. Díaz Rizo, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos. "Digitally Assisted Mixed-Signal Circuit Security." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41.8 (2022), pp. 2449–2462. DOI: 10.1109/TCAD.2021.3111550.

[2] A. A. Ibrahim, H. A. Mohamed, A. R. Díaz Rizo, R. Parra-Michel, and H. Aboushady. "Tunable Filtenna With DGS Loaded Resonators for a Cognitive Radio System Based on an SDR Transceiver." In: *IEEE Access* 10 (2022), pp. 32123–32131. DOI: 10.1109/ACCESS.2022.3160467.

PATENTS

During the doctorate program, the following patent application was completed:

[1] "Method for securing telecommunication transceiver integrated circuit designs against piracy, counterfeiting and unauthorized use." PCT/FR2022/050437. A. R. Díaz Rizo, H. Aboushady, and H.-G. Stratigopoulos. Mar. 12, 2022.

AWARDS

The *SyncLock* project, presented in Chapter 5, is one of the laureates of the 2022 **i-PhD concours d'innovation**.

[1] A. R. Díaz Rizo. "SyncLock: Synchronization Locking to secure RF transceivers against piracy and unauthorized use." In: *Concours d'innovation i-PhD, BPI France*. 2022. URL: https://bit.ly/3x4NYvz.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

3PIP Third-Party Intellectual Property Block
A/M-S Analog and Mixed-Signal
ACE Adaptive Channel Estimation
ADC Analog-to-Digital Converter
AES Advanced Encryption Standard
AFE Analog Front-End
ALUT Adaptive Look-Up Table
AM Amplitude Modulation
ASIC Application-Specific Integrated Circuit
AWGN Additive White Gaussian Noise
BER Bit Error Rate
BPSK Binary Phase-Shift Keying
CAC Corrupt-And-Correct
CFO Carrier Frequency Offset
CP Cyclic Prefix
CSMA/CA Carrier Sense Multiple Access with Collision Avoidance
DAC Digital-to-Analog Converter
DC Direct Current
DCO DC Offset
DCO-IQI DC Offset and I/Q Imbalance
DIP Distinguishing Input Pattern
DNN Deep Neural Network
DSP Digital Signal Processing
DRAM Dynamic Random-Access Memory
EDA Electronic Design Automation
EDF Electronic Disposal Facility
EVM Error Vector Magnitude
EMS Electronic Manufacturing Services
FEC Forward Error Correction
FPGA Field-Programmable Gate Array
GDSII Graphic Database System II
HD Hamming Distance

HDL Hardware Description Language
HT Hardware Trojan
IC Integrated Circuit
IDM Integrated Device Manufacturer
IF Intermediate Frequency
IFFT Inverse Fast Fourier Transform
IFT Information Flow Tracking
IoT Internet-of-Things
IP Intellectual Property
IQI I/Q Imbalance
ISM Industrial, Scientific and Medical
KPA Known-Plaintext Attack
LNA Low Noise Amplifier
Low-IF Low Intermediate Frequency
LR-WPAN Low-Rate Wireless Personal Area Network
LSB Least Significant Bit
LTS Long Training Sequence
MAC Medium Access Control
ODM Original Device Manufacturer
OEM Original Equipment Manufacturer
OFDM Orthogonal Frequency Division Multiplexing
OSAT Outsourced Semiconductor Assembly and Testing
OSI Open Systems Interconnection
PA Power Amplifier
PCA Principal Component Analysis
PCB Printed Circuit Board
PHY Physical Layer
PIP Protected Input Pattern
PLL Phase Locked Loop
PPA Power, Performance, and Area
PPDU Physical layer Protocol Data Unit
PSD Power Spectral Density
PUF Physical Unclonable Function
QAM Quadrature Amplitude Modulation
QPSK Quadrature Phase-Shift Keying
RD Research and Development

RF Radio Frequency
RSA Rivest-Shamir-Adleman
RTL Register Transfer Level
SDR Software Defined Radio
SFLL Stripped Functionality Logic Locking
SMT Satisfiability Modulo Theory
SNR Signal-to-Noise Ratio
SoA state-of-the-art
SoC System-on-Chip
SPS Signal Probability Skew
SSCF Statistical Side-Channel Fingerprinting
STS Short Training Sequence
SVM Support Vector Machine
TPM Tamper-Proof Memory
WLAN Wireless Local Area Network
WPAN Wireless Personal Area Network
Zero-IF Zero Intermediate Frequency

1 INTRODUCTION

The semiconductor supply chain is massively globalized. Electronic device makers, usually referred to as Original Equipment Manufacturers (OEMs), typically design their products and decide which components to use from which suppliers. OEMs often have their headquarters in one country, their team of design engineers in another country, and have their devices manufactured in a different country or in many different countries. The companies in charge of the various manufacturing tasks are referred to as Original Device Manufacturers (ODMs) or Electronic Manufacturing Services (EMS) [1]. This globalization of the supply chain, can be divided into five specializations: design, equipment, materials, manufacturing, and commercialization. To illustrate the interdependence of these specialization tasks, Fig. 1 shows the global journey of a smartphone application processor [1] having an Intellectual Property (IP) core, or IP block[1], of a European design house as a building block. The journey is as follows:

1. A European semiconductor company designs and licenses an IP block to a system integrator, also known as a System-on-Chip (SoC)[2] integrator, in the United States of America (U.S.).

2. The SoC integrator, which is a *fabless*[3] U.S. company, uses advanced Electronic Design Automation (EDA) tools for chip design to integrate the European IP block, along with other Third-Party Intellectual Property Blocks (3PIPs), and designs the SoC.

3. A U.S. smartphone OEM selects the SoC to be embedded in its new smartphone.

4. Advanced manufacturing equipment is developed by companies in the U.S., Europe, and Japan, relying on decades of Research and Development (RD) efforts.

5. Materials for the semiconductor industry are mined, refined, polished and shipped from the U.S., Japan, and South Korea.

1 A semiconductor Intellectual Property core, or IP block, is a reusable circuit, cell, or Integrated Circuit (IC) layout design that is the intellectual property of one party.

2 A system on a chip is a complete system integrated on a single IC consisting of several IP blocks, such as processors, memories, power management units, and peripheral interfaces, to perform the intended function.

3 A fabless company designs and commercializes ICs while outsourcing their fabrication to a specialized manufacturer called a semiconductor foundry.

6. A semiconductor foundry in Taiwan receives the SoC design and materials, then uses the manufacturing equipment to imprint polysilicon wafers with an array of Integrated Circuits (ICs).

7. The individual chips are separated, packaged, and tested by an Outsourced Semiconductor Assembly and Testing (OSAT) in Malaysia.

8. The fabricated chip is shipped to the smartphone OEM's assembly partner in China for mounting on a Printed Circuit Board (PCB) inside the smartphone.

9. The smartphone is sold to an end-user in the U.S.

10. After some time of use, the smartphone is discarded.

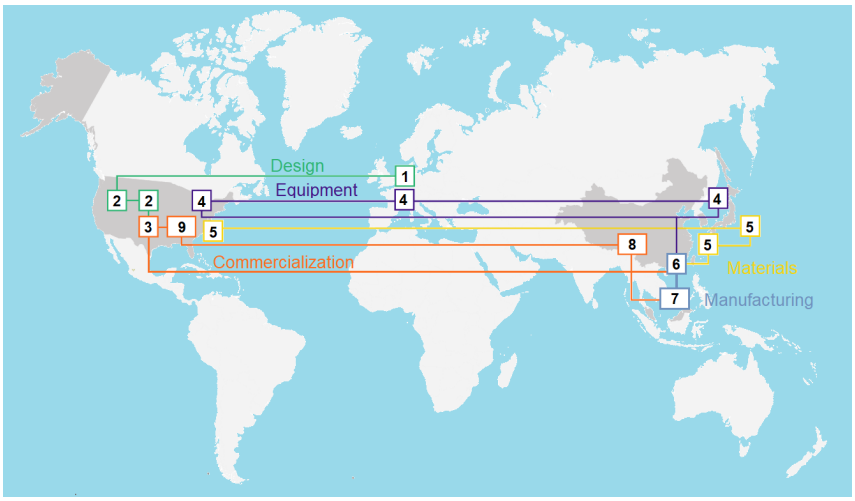


Figure 1: Simplified life cycle trajectory of an IC. Illustration based on "*The global journey of a smartphone application processor*" [1].

In contrast to the *fabless model* presented in the previous example, during the early decades of the semiconductor industry, the Integrated Device Manufacturer (IDM) model was prevalent. An IDM is a semiconductor company that designs, manufactures, and commercializes its own ICs. However, the rapid growth in capital expenditure and investment to produce state-of-the-art (SoA) ICs created the need for scale and specialization, which resulted in the advent of the fabless model. In 2020, 33% of the global IC sales where from fabless companies [1], and even IDMs rely on semiconductor foundries and OSATs for a portion of the manufacturing, assembly, packaging, and testing needs [1]. Europe has fallen behind in semiconductor manufacturing, declining from 24% of global production capacity in 2000 to 8% in 2022 [2].

The main stakeholders in the IC supply chain can be identified from the example presented previously and illustrated in Fig. 1. Their

role in the supply chain is summarized in Fig. 2. For instance, a fabless design firm may have four roles: (1) establish the requirements and specifications of an IC, (2) develop IP blocks, (3) integrate their IP blocks and likely 3PIP blocks into an SoC, and (4) design the layout of the IC. However, the rest of the tasks in the IC supply chain are outsourced to other stakeholders by the fabless design firm. An IP broker only intervenes in the storage and trading of IP blocks and SoCs. An example of an IP broker is Design & Reuse [3], a French company dedicated to IP and SoCs dissemination. A notable example of an OEM is Apple, a fabless outsourcing-based company.

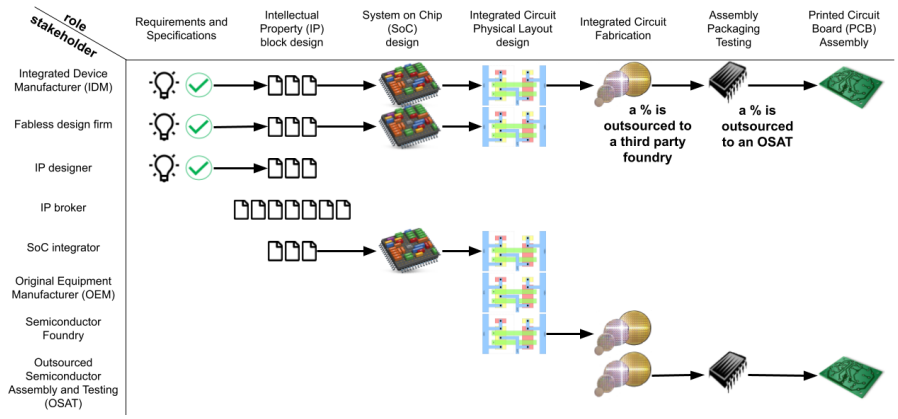


Figure 2: IC supply chain stakeholders and their role in the supply chain.

This globalized, fabless, and outsourcing-based model which characterizes the current IC supply chain and life cycle, raises vulnerabilities and boosts IC supply chain attacks, i.e., the possibility of cloning and tampering ICs [4].

1.1 IC SUPPLY CHAIN VULNERABILITIES AND THREATS

Threats to the IC supply chain consist of counterfeiting[4] and tampering.

To accord the terminology to the SoA, we defined the IC supply chain threats, a.k.a. IC supply chain attacks, as follows:

1. Cloning: unauthorized copying of an IP/IC.
2. Overproducing (a.k.a. overbuilding): out-of-contract production of ICs by the authorized contractor.
3. Remarking: indicating an out-of-specification or defective IC as working.

4 The U.S. department of Commerce defined a counterfeit component as the one that is an unauthorized copy; imitation; does not conform to original design, model, and/or performance standards; is produced by unauthorized contractors; is an off-specification, defective, or used IC sold as new or working; or has incorrect or false markings and/or documentation [5].

4. Recycling: reintroducing a used IC as new.

5. Tampering (a.k.a. Hardware Trojan (HT) insertion): malicious modification of the hardware of an IP block or IC performed by an adversary.

6. Unauthorized use: utilization of an IP/IC in an unauthorized application or sector.

Cloning, overproducing, remarking, and recycling are known as piracy attacks. Tampered (HT-infected) IP/ICs can potentially leak valuable information to an eavesdropper, cause performance degradation, or completely deny service. The tampering and piracy threats associated to each stage of the IC life cycle are illustrate in Fig. 3.

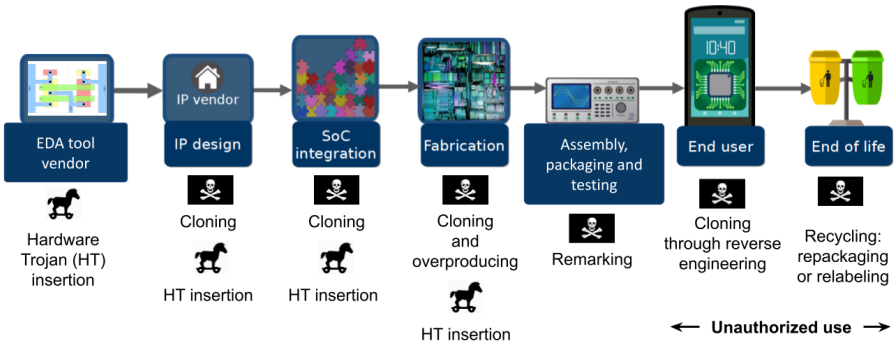


Figure 3: IC supply chain threats at each stage of an IC's life cycle.

HT insertion may originate from a malicious EDA tool. It may also be part of the design, implemented during SoC integration, or introduced into the IC design empty spaces by a rogue foundry.

Cloning may be carried out internally by a rogue employee, externally by an SoC integrator who acquires a license for using a 3PIP block, or by a malicious foundry. Specifically, the blueprint, i.e., the Graphic Database System II (GDSII) file of the IP/IC, is given away and, thereby, such a malicious entity can easily clone the IP/IC and illegitimately produce cloned counterfeits without the consent or knowledge of the IP/IC owner. Cloning may also be carried out by a malicious end-user who has capabilities for reverse-engineering the chip to extract its netlist [6, 7]. Nowadays there are increased capabilities for performing reverse-engineering of chips to extract the design netlist and other technology secrets [8]. In addition, a malicious foundry may overproduce chips beyond the number agreed on in the contract with the IP/IC owner and sell them illegitimately. Remarking may be performed by OSATs, and recycling may be done at an Electronic Disposal Facility (EDF). Repackaging (a.k.a. salvaging) is recycling an IC with a new package and a new label, and relabeling is recycling an IC with a used package but a new label. Although recycling ICs from discarded equipment may seem commendable, these

chips can have problems with electromigration of metals, reliability, and short life span.

1.2 CONSEQUENCES OF IP/IC PIRACY AND HT INSERTION

IP/IC piracy and HT insertion effects and consequences on government, industry, and society as a whole, are resumed in Table 1.

Piracy of ICs has been a major security threat since the past few decades. For instance, in 2011, VisionTech Components sold counterfeit semiconductor chips to more than 1,100 customers in every sector of the electronics industry, including the military [9]. Among the companies that claimed damages for chip fraud are major chip companies such as Analog Devices and Raytheon [9].

With respect to governments and consumers, in a report [10] out in march 2022, the European Union's law enforcement agency Europol highlighted the risks posed by counterfeit semiconductors to critical infrastructure as well as to people's private devices. The U.S. Department of Homeland Security tied counterfeit components to national security in a January 2022 report [5] about the semiconductor supply chain. Concerning the industry, cloning and overproducing result in know-how and financial losses for the IP/IC owner. IP piracy issues alone incur annual losses up to $4 billion for the semiconductor industry [11]. In addition to the loss of revenue suffered by counterfeit victims, estimated at $100 billion annually for the entire electronics industry, due to counterfeit semiconductors, end users and the systems that comprise them can suffer premature or critical failures [5]. The U.S. Semiconductor Industry Association testified in 2011 that it estimated that counterfeiting costs U.S. more than $7.5 billion per year, translating into nearly 11,000 jobs lost [5]. The commercialization of HT-infected ICs could permanently damage a company's reputation and trust. In addition, an IC leaking users' private information puts security and respect for privacy at risk. A HT could be a time bomb that can be triggered under specific conditions controlled by the attacker, leading in a complete denial-of-service while the chip is in the field.

Furthermore, considering that the IC supply chain is complex and not necessarily each actor in the chain knows and controls all the stakeholders involved, the infiltration of counterfeit or tampered ICs affects more than one actor. Therefore, its negative effect is difficult to fully assess.

Security methods deployed in the industry for identifying counterfeit and/or HT-infected ICs relying on human intervention are time-consuming, increasing the time to market of the ICs. Moreover, there are costly to carry out in volume, often cannot detect all counterfeits, and require a high degree of expertise [5]. In addition, because the IC supply chain involves several stages, it is not easy to verify that all of

Table 1: Effect of IP/IC piracy and HT insertion.

GOVERNMENT	INDUSTRY	CONSUMER/SOCIETY
- National security if counterfeit or tampered chips are used in critical infrastructure or defense - Lost tax revenue due to illegal sales of counterfeit parts - Costs of law enforcement	- Lost sales revenue - Lost of know-how - Lost brand value or damage to business image - Costs to mitigate the risk	- Costs to replace failed products due to lower quality and reliability and shorter lifespan of counterfeit parts - Safety concerns if counterfeit parts are used in critical applications, i.e., automotive - Privacy and safety concerns if a tampered IC leaks users' private information

these security methods are carried out from end-to-end. Thus, there is an urgent need for attack-resistant design methods which can protect an IP/IC against potential piracy attacks occurring at any point in the supply chain, as well as HT detection, prevention, and deactivation mechanisms.

1.2.1 *A case study of a disruption scenario*

The Covid-19 pandemic disrupted the IC supply from China and spread to other parts of the world. That disruption showed that the semiconductor supply chain is not resilient, and a failure can cause multi-millions losses due to chip shortages [12, 13]. The manufacturing lead time for ICs has increased since the pandemic and even at the date of this work, OEMs and IDMS must wait longer to have their designs manufactured [14–16]. Moreover, the scarcity of semiconductors created a huge demand that could not be met and a growing desperation on the part of the companies as they saw multi-million losses due to the shutdown of their production lines. In such a disruptive scenario, many companies were and are willing to buy (fabricate) the necessary components from (with) unreliable parties. Thus flooding the market with counterfeit chips [17], or further risking the intellectual property of their designs.

ERAI is an online counterfeit electronics database. ERAI monitors, investigates, and reports issues affecting the global electronics supply chain [18]. Since 2019 more ERAI members from automotive and similar industries are requesting ERAI for a counterfeit mitigation solution. Through ERAI, fake electronics can be anonymously reported, which provides tools to its members to avoid counterfeit components. "In 2020, about 80 percent of those reported devices were reported for the very first time that year" [19].

The counterfeit chips resulting from the semiconductor crisis, since the beginning of the pandemic, added to the counterfeits that were already on the rise. Their infiltration into the supply chain causes damage that is difficult to fully quantify as counterfeits affect the

products in which they are embedded, the companies that sell them, and the millions of users who use them.

1.3 IC SUPPLY CHAIN THREATS FOR WIRELESS COMMUNICATIONS ICS

Wireless communication devices are ubiquitous. A Radio Frequency (RF) transceiver is a fundamental building block in an electronic system that allows it to communicate with another device wirelessly. RF transceivers are present in all devices with wireless connectivity. The wireless connectivity sector can be classified by technology type (Wi-Fi, Bluetooth, NFC, Zigbee, LTE, LoRa, SigFox, 2G, 3G, LTE, 4G, 5G, etc.), by end-user (Internet-of-Things (IoT), wearables, consumer electronics, healthcare), and by application (aerospace, satellite, automotive, mobile communication, etc.).

With the rise of IoT and 5G, hundreds of new markets are expected to emerge. For example, the number of IoT devices is forecasted to surpass 25.4 billion in 2030 [20]. Moreover, the wireless connectivity sector is one of Europe's strategic industries [2]. Therefore, its development and security is very important to achieve digital sovereignty.

In addition to their widespread use, IC supply chain attackers target wireless integrated circuits for two main reasons: they have a critical role in society, government, and industry; they exchange sensitive and valuable information through a publicly accessible medium. For these reasons, this thesis focuses on securing RF transceiver ICs.

1.4 THESIS OBJECTIVES

The thesis has two primary objectives:

1. To investigate and analyze the attack surface for HT insertion into RF transceiver ICs.
2. To propose novel solutions to protect the design of RF transceivers and ascertain their security against piracy attempts.

1.5 THESIS STRUCTURE

The overall structure of the thesis takes the form of six chapters, including this introduction. The rest of the thesis is structured as follows. Chapter 2 examines the prior art on security and trust for integrated circuits. The chapter is composed of two sections, the previous work on HTs and the evolution of thinking in anti-piracy design techniques, focusing on RF transceivers. Chapter 3 presents a new HT attack for RF transceivers aiming at leaking sensitive information through a covert channel. Chapter 4 provides a methodology to secure RF transceivers against IP/IC piracy by leveraging a SoA

logic locking technique. In Chapter 5, we present *SyncLock*, a domain-specific anti-piracy design technique to secure RF transceivers. Chapter 6 concludes the thesis and discusses prospective investigations and developments on RF transceiver security and trust.

2

PRIOR ART ON SECURITY AND TRUST FOR WIRELESS INTEGRATED CIRCUITS

This chapter gives an overview of the prior art on security and trust techniques to protect IP blocks and ICs against IC supply chain attacks. The chapter is composed of two sections. Section 2.1 deals with the prior art on HT insertion in the context of RF transceivers, and Section 2.2 focuses on the SoA anti-piracy design techniques for RF transceivers.

2.1 HARDWARE TROJANS IN RF TRANSCEIVERS

In this section, we present the prior art on HTs. We introduce different models of insertion and attacks, as well as the defense mechanisms to detect or prevent HTs. The scope of the thesis focuses on the context of RF transceivers, their main threats, and countermeasures.

2.1.1 *Hardware Trojans in ICs*

A HT is a malicious modification of the hardware performed by an adversary [21–26]. HTs are classified according to the insertion phase (i.e., design, fabrication, assembly, post-silicon, etc.), insertion level (i.e., Register Transfer Level (RTL), gate-level, transistor-level, layout, etc.), location on die (i.e., processor, memory, analog, etc.), triggering mechanism (i.e., always-on, activation after some operation time elapses, activation when some rare input condition is met), and payload or effect (i.e., performance degradation, denial-of-service, or leaking of sensitive data such as a cipher key).

From the attacker's perspective, the goal is to design a small footprint and stealthy HT that evades detection. From the defender's perspective, the goal is to prevent HT insertion or detect the presence of a HT, for example with reverse engineering, post-manufacturing testing, or during run-time.

Numerous HT designs have been proposed in the literature, the vast majority of which target digital ICs. The simplest HT is a combinational circuit that monitors a set of nodes to generate a trigger on the simultaneous occurrence of rare node conditions and, subsequently, once the trigger is activated, the payload is simply flipping the value of another node. Another common HT design are the sequential HTs which are triggered with a sequence of conditions and not with a specific state or condition like the combinational HTs. More complex HTs include silicon wearout mechanisms [27], hidden side-

channels [28], changing dopant polarity in active areas of transistors [29], siphoning charge from victim wires known as A2 attack [30, 31], activating a row in Dynamic Random-Access Memory (DRAM) to corrupt data in nearby rows known as rowhammer attack [32], exploiting capacitive crosstalk effects [33], leveraging characteristics of emerging Non-Volatile Memories (NVMs) [34], etc.

For analog ICs, HT insertion is more challenging because analog performance is sensitive to circuit alterations, thus a HT-infected analog IC is likely not to pass testing, and also because analog layouts have few components, thus a HT can be easily detected via reverse engineering and layout inspection. Proposed HT designs for analog ICs include bringing the circuit into an undesired state or operation mode [35–39] and digital-to-analog HTs that exploit the on-chip test infrastructure [40–42]. In the latter scenario, the HT resides inside a digital IP block where it is triggered. The generated payload is transferred to the victim analog IP via the common test access mechanism and is applied to it via its built-in self-test or programming interface to the test access mechanism. There exist also analog HT designs to infect digital ICs, such as the A2 attack [30, 31]. The A2 attack can be used to infect the digital section of a mixed-signal IC, but it has not been demonstrated inside the analog section.

2.1.2 *Covert channel attacks for wireless ICs*

Targeting RF transceivers, a covert channel is a HT attack in wireless ICs aiming at leaking sensitive information from the transmitter within a legitimate signal transmission. A rogue receiver can listen to the transmission to recover the sensitive information, while the legitimate receiver is inconspicuous and does not realize the information leaking. Several studies have demonstrated this type of HT attack. The HT can be embedded within the Medium Access Control (MAC) protocol [43], within the digital baseband Physical Layer (PHY) [44–48], or its payload mechanism can partially act upon the Analog Front-End (AFE) [49–53]. In parallel, these studies propose defenses for detecting the HT attack at test time or during run-time. Table 2 provides a concise summary of existing attack models and corresponding defenses. These are explained below in more detail.

2.1.3 *Attack models*

In [43], a spyware is demonstrated exploiting the timing channel resulting from inter-arrival times of the legitimate transmitted packets. It can be embedded within any MAC protocol that avoids collision in packet re-transmissions by using an exponential back-off rule, i.e., the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol. In [44], covert messages are hidden within a "dirty" pay-

Table 2: HT attack models and defenses.

REF.	ATTACK MODEL	DEFENSE MECHANISM
[43]	Modifies the MAC layer CSMA/CA protocol to leak data into the timings of the transmitted packet sequence.	Evades statistical tests that detect covert timing channels. No other defense is studied.
[44]	Encodes leaked data on the I/Q mapping and hides the encoding by introducing imperfections to the transmitted signal.	Certain tests, such as EVM, show a distinguishing behavior compared to HT-free operation.
[45]-1	Leaks data by introducing an additional phase shift into all STS symbols of the preamble.	Analysis of the preamble constellation.
[45]-2	Leaks data by introducing artificial CFO into each OFDM symbol.	Analysis of CFO changes over time.
[45]-3, [46]	Leaks data in extra camouflage subcarriers added to the OFDM signal.	Decode the signal field to determine if the number of subcarriers is correct.
[45]-4, [47]	Leaks data into parts of the OFDM CP.	Compare the last 16 samples of an OFDM symbol with its CP; spectrum analysis.
[48]	Leaks data by substituting some legitimate data in the FEC block.	Channel noise profiling.
[49–51]	Leaks data by modulating amplitude and/or frequency of transmitted signal.	SSCF; ACE; Use hardware dithering as a prevention mechanism [54] .
[52]	Leaks data using spread spectrum techniques.	Spectral analysis.
[53]	Leaks data into artificial controlled impairments.	No defenses are studied.
HT in Ch. 3 [55]	Leaks data through amplitude modulation of some subcarriers in the STS of the preamble.	Evades any known defense for α <15%.

load data constellation by taking advantage of the I/Q impairments and noisy channel conditions. In [45], four covert channel schemes are shown for the PHY of the IEEE 802.11. These include leaking data by: (a) introducing an additional phase shift in the Short Training Sequence (STS) symbols of the preamble; (b) introducing an artificial Carrier Frequency Offset (CFO) into each Orthogonal Frequency Division Multiplexing (OFDM) symbol; (c) introducing camouflaged subcarriers, i.e., extra subcarriers to those of the standard, into the OFDM signal; and (d) by replacement of the OFDM Cyclic Prefix (CP). In [48], the attack is staged in the Forward Error Correction (FEC) block, exploiting the fact that the FEC block offers more error correcting capabilities than the channel needs. In [49–51], the idea is to exploit the margins that exist between the operating point of the circuit and the boundaries defined by the circuit and communication standard specifications. In particular, the HT performs minute modifications in the parameters of the transmitted signal, such as amplitude and frequency, to leak sensitive information from the tampered device. Two HT payload mechanisms are shown in [51], one that uses a single pole double throw switch and a pair of resistors to alter the input termination impedance of the power amplifier, and another one that

reprograms the gain stages. In [52], it is proposed to use spread spectrum techniques to hide an unauthorized transmission signal within the legitimate signal below the noise level. In [53], first feasible transmitter impairments that do not affect appreciably Bit Error Rate (BER) are determined, then leaked data are mapped on such artificially introduced impairments. The adversary learns these impairments and extracts the leaked data using deep learning.

2.1.4 *Defense mechanisms*

Most of the aforementioned studies also study resilience to various defenses, oftentimes finding a working defense. Defenses range from standard measurements, i.e., measuring SNR, EVM or Bit Error Rate (BER); examining compliance with the spectral mask specifications; analyzing I/Q constellation diagrams, etc., to more elaborate techniques such as Statistical Side-Channel Fingerprinting (SSCF) [50], Adaptive Channel Estimation (ACE) [51], and channel noise profiling. SSCF consists in training a one-class classifier in a feature space composed of parametric measurements, e.g., transmitted power from golden HT-free devices. The HT-infected devices have a feature vector that lies outside the classification boundary and, thereby, can be distinguished from HT-free devices. The ACE defense leverages the slow-fading characteristics of indoor communication channels to distinguish between channel impairments and HT activity. In [51], it is claimed that the ACE defense is successful in detecting any HT regardless of the attack specifics. There exist also defenses that are specific to the attack model focusing on the particular encoding of the leaked data. Finally, it is possible to design a proactive defense mechanism that challenges the operation principle of the HT with the aim to neutralize it. An example is the hardware dithering technique proposed in [54].

2.2 ANTI-PIRACY DESIGN TECHNIQUES FOR RF TRANSCEIVERS

The IP of IC design is protected under patent law in various countries. IP theft is a recurring industrial problem being pursued in the court [56]. However, this deterrent measure does not prevent IP theft or unauthorized use and often involves lengthy litigation to penalize the infringer. Therefore a technological countermeasure against piracy and unauthorized use is needed. In this Section, we present the SoA design techniques to prevent IP/IC piracy.

2.2.1 *Countermeasures against piracy of IP blocks and ICs*

A classification of anti-piracy measures used in RF transceivers design and fabrication is illustrated in Fig. 4. The three major cate-

gories are: (a) split manufacturing, (b) camouflaging, and (c) locking. Unlike split manufacturing and camouflaging, locking offers end-to-end protection against all potential piracy threat scenarios. The DARPA[1] agency has chosen locking as a defense mechanism to prevent "four fundamental silicon security vulnerabilities: side channel attacks, hardware Trojans, reverse engineering, and supply chain attacks, such as counterfeiting, recycling, re-marking, cloning, and overproduction" [57]. Locking has also been integrated into the Mentor Graphics framework [58].

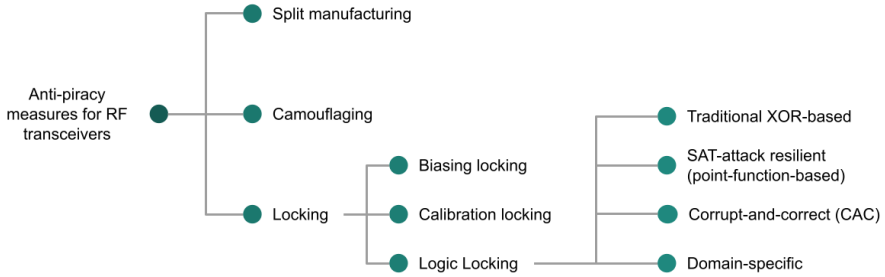


Figure 4: Anti-piracy measures used in RF transceivers design and fabrication.

Split manufacturing divides the fabrication process of ICs between two foundries [59]. An untrusted high-end foundry fabricates the front-end of line layers (transistors and lower metal layers), and a trusted low-end foundry manufactures the back-end of line layers (higher metal layers). This approach prevents overproduction and cloning from an untrusted foundry [60]. Split manufacturing has been demonstrated for RF designs in [61]. In [61], the authors claimed that split manufacturing may be more suitable for RF designs than for digital circuits, however, split manufacturing only protects against one piracy entity, i.e., an untrusted foundry.

The purpose of camouflaging is to obfuscate the hardware to protect the IC against reverse engineering [62]. The gates of the camouflaged designs are similar but implement different Boolean functions [63]. Camouflaging ideas for Analog and Mixed-Signal (A/M-S) ICs include multi-threshold transistor design [64] and obfuscating the geometry of layout components [65]. However, camouflaging lacks protection against other piracy attacks, like cloning by an SoC integrator or an end-user, overproducing, remarking, and recycling.

Locking, illustrated in Fig. 5, is performed by the designer and consists in embedding a lock mechanism inside the IP/IC. The lock mechanism is a circuit that is mingled with the original circuit and controlled by a key. The key is typically in the form of a digital bit-string. The lock mechanism is transparent to the IP/IC such that

1 The Defense Advanced Research Projects Agency (DARPA) is a research and development agency of the United States Department of Defense responsible for the development of emerging technologies for use by the military.

upon application of the correct key the nominal functionality is restored. However, applying an incorrect key corrupts the functionality. IP/IC locking protects an IP/IC against potential attackers located anywhere in the supply chain, as well as against malicious end-users. It can also protect against recycling facilities as long as the key can be reloaded every time the IC is powered on.

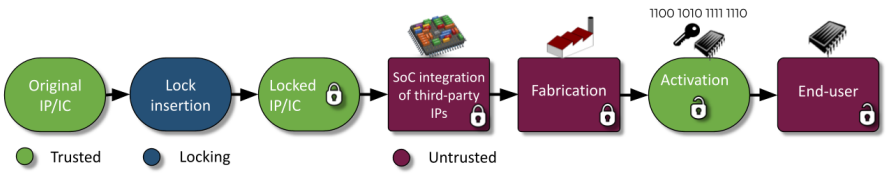


Figure 5: IC design flow and lifetime using locking as a countermeasure against IP/IC piracy.

IP/IC locking is considered as the strongest countermeasure against IP/IC piracy [66]. Therefore, we thoroughly studied this design technique and present, in the following subsections, the three main defenses used for the protection of A/M-S and RF transceiver ICs. The three locking defenses are (1) biasing locking, (2) calibration locking, and (3) logic locking.

2.2.2 *Biasing locking*

For locking analog blocks the existing technique is biasing locking. Biasing locking aims at controlling the bias generation with the key. Unless the correct key is provided, the analog block is incorrectly biased, meaning that the quiescent point of transistors is not the desired one, resulting in performance degradation or malfunction. For RF transceivers, one can perform biasing locking in blocks of the Analog Front-End (AFE), i.e., Low Noise Amplifier (LNA), Power Amplifier (PA), Phase Locked Loop (PLL), data converters, etc. Several embodiments of biasing locking exist, including obfuscating the geometry of a bias transistor [67, 68], designing key-controlled current mirrors [69], and replacing the biasing circuit with an alternative key-controlled bias generator, e.g., based on an on-chip neural network [70] or a programmable memristor crossbar [71]. Biasing locking may result in imprecise or unstable biasing. Besides, recently counter-attacks were proposed based on Satisfiability Modulo Theory (SMT) [72] and optimization [73, 74] that break this type of defense.

2.2.3 *Calibration locking*

Calibration locking achieves system-level locking making the compensation of process variations or adaptation to different operation modes key-dependent. Techniques in this category include locking

the digital section of the calibration loop [75], treating digital programmability as a natural secret key [76–78], and making the calibration range key-dependent [79]. Secure calibration locking requires that the calibration algorithm be complex enough to be devised or redesigned in hardware by the attacker. Such an assumption does not always hold.

2.2.4 *Logic locking*

Logic locking adds supplementary logic to a circuit design to protect the original logic. Logic Locking incurs a justifiable yet non-negligible area and power overhead. The first logic locking, a.k.a. logic encryption, technique was originally proposed for digital circuits [80]. Since then, several logic locking defenses were proposed aiming at reducing Power, Performance, and Area (PPA) penalties, increasing corruption for invalid keys, and circumventing counterattacks that were developed in the meantime aiming at exposing security vulnerabilities of logic locking, i.e., finding the secret key with reasonable effort or identifying and subsequently removing the lock [66]. There is an ongoing "ping-pong" game between logic locking defenses and counterattacks. Every newly introduced logic locking technique is considered secure until shortly after a counterattack heuristic breaks it.

Logic locking had also be used in other applications fields, e.g., for upgrading processors [81], or for locking the bitstream to configure and program a Field-Programmable Gate Array (FPGA) [82].

2.2.4.1 *Evolution of thinking in logic locking*

A timeline illustrating the evolution of thinking about logic locking protection techniques and counterattacks in the SoA is presented in Fig. 6. We have grouped **Logic Locking Defenses** into three main sets, traditional XOR-based, SAT-attack resilient, and Corrupt-And-Correct (CAC) defenses. The **Logic Locking Attacks** are also grouped into three major sets, brute-force and optimization, input-output query, and structural attacks. This section presents a walk-through of the multiple attacks and defenses developed using logical locking.

1. **Traditional XOR-based defenses:** The first logic locking defense, called "EPIC", inserts XOR/XNOR key-gates into the design in a random fashion. A key-gate interrupts a digital line controlling its value with a key-bit. The objective is to guarantee that incorrect keys will produce incorrect output [80]. In a **brute-force or optimization attack**, the attacker searches in the key space either randomly in a brute-force manner or more efficiently by employing an optimization algorithm hoping to find a key that produces correct outputs. Traditional XOR-based defenses protect against these attacks by setting a larger number of

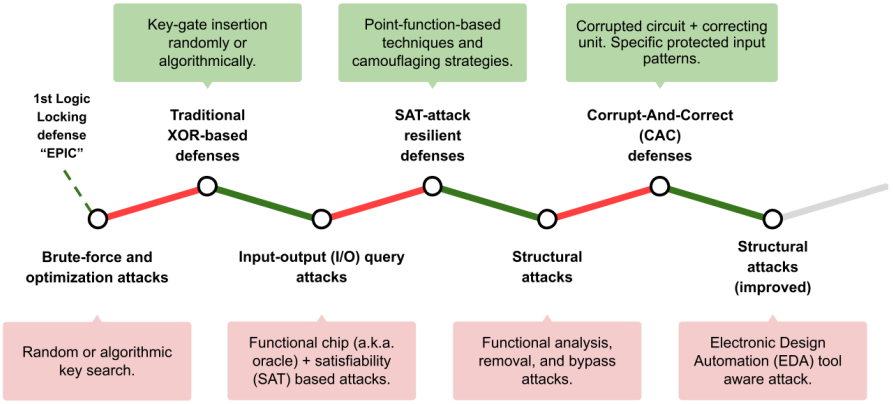


Figure 6: Timeline of the evolution of thinking about logic locking attacks and defenses.

bits for the key size, thus increasing the search space. Then, the first logic locking specific attack, the sensitization attack, broke the EPIC defense by employing a functional IC (a.k.a an oracle) and querying the locked IC with specific input patterns that propagate key values to the outputs [83].

Since EPIC failed, other traditional techniques, consisting of key-gates insertion [80, 83–88], followed-up targeting high output corruption for incorrect keys [84], resilience to sensitizing the key-bits to the output [86], optimal positioning of the key-gates [85], reducing PPA overheads [87], or thwarting the ability of the attacker to learn the key-bit value from the key-gate type [88]. In all these XOR-based techniques, key-gates are inserted randomly or algorithmically.

2. **Input-output (I/O) query attacks:** XOR-based defenses are efficient against brute-force and optimization attacks, but vulnerable to I/O query attacks, like the sensitization attack [83]. Attacks based on Boolean satisfiability (SAT) [89] belong to the I/O query attacks category and were shown to be very powerful, breaking traditional logic locking approaches [80, 83, 84, 86, 87] by recovering the key with little effort.

   The SAT attack computes Distinguishing Input Patterns (DIPs), defined as inputs which produce different output for at least two different keys, and prunes down multiple incorrect keys iteratively using DIPs and querying the oracle. In [90], another SAT-based attack is presented using an Satisfiability Modulo Theory (SMT) solver. Another group of I/O query attacks is the approximate attack, e.g., AppSAT [91] and 2-DIP [92]. Approximate attacks seek to reduce the security level and find the best set of keys that establish an incorrect but approximate functionality.

3. **SAT-attack resilient defenses:** To thwart I/O query attacks, researchers used **point-functions** as the protection logic [93, 94]. A point-function is a Boolean function that produces the output value '1' at exactly one point. For instance, when a point-function, such as a comparator whose first input is the key and the second input is the functional input, is XORed with the original circuit, the output is flipped/inverted/corrupted only for one input pattern for any incorrect key. In addition, the implementation of point-functions can be provably obfuscated [95].

   The point-function-based technique SARLock [93] maximizes the required number of DIPs to recover the secret key. SARLock thwarts the SAT attack by rendering the attack effort exponential in the number of bits in the secret key, while its overhead grows only linearly. The Anti-SAT defense [94] thwarts any I/O attack because the probability to find the input patterns (i.e., the key) that corrupt the outputs is an exponential function of the key-size, thereby making the SAT attack computationally infeasible. As the SAT attack makes use of the scan chain, another proposed solution to thwart the SAT attack is to withdraw the secret key upon detection of access to the scan chain [88].

   However, the security of point-function-based techniques has a drawback. The point-function can be identified and removed or bypassed by performing a structural attack.

4. **Structural attacks:** Attacks in this category aim to detect the point-function circuit and remove it [96, 97], bypass it [98], deobfuscate it [99], or recover the key from their structure properties [100–102]. For instance, the Signal Probability Skew (SPS) attack identifies large AND gates and/or comparators, using signal probabilities, and removes the locking mechanism of Anti-SAT and SARLock defenses [96]. Another structural attack performs a netlist analysis to locate and remove the locking mechanism [97].

   To thwart structural attacks, researchers developed a subset of point-function-based techniques which rely on their structure to be merged or hidden inside the vast Boolean area after logic synthesis. These defenses are discussed in the following section.

5. **Corrupt-And-Correct (CAC) defenses:** A CAC circuit contains a corrupted circuit and a correcting unit that restores the corrupted circuit only for the correct key. The CAC techniques differ in the construction of the corrupted circuit. The corrupted circuit could be generated using a corrupting unit [103] which can be a point-function that transforms the original circuit into a corrupted version, or it could be corrupted by another method, e.g., a fault-injection campaign [104, 105].

SoA CAC logic locking techniques are SFLL-hd/flex [103], SFLL-fault [104], SFLL-rem [105], and CAS-Lock [106]. SFLL-hd, illustrated in Fig. 7, inserts a corrupt unit which compares the input to a hard-coded secret key. If the Hamming Distance (HD) between the key and the input is h, then the output of the corrupt unit is '1', thus flipping the output of the circuit using an XOR gate. The inputs that satisfy this condition are called Protected Input Patterns (PIPs). The correct unit is identical to the corrupt unit, but in this case the key is sourced from the Tamper-Proof Memory (TPM). The correct unit flips the output a second time for the PIPs to restore correct functionality only when the correct secret key is loaded into the TPM. If an incorrect key is used, the functionality will be corrupted for all PIPs. A larger set of PIPs will result in a higher error rate at the output of the circuit. By choosing appropriate h, designers change the number of PIPs and thus trade-off the resilience against input-output query and structural attacks.

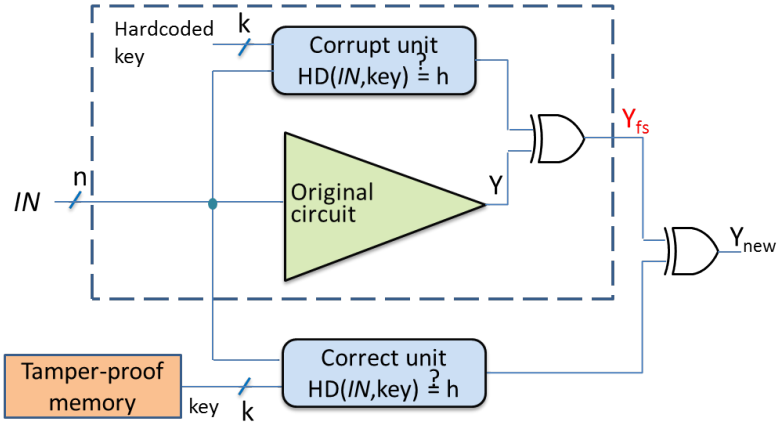


Figure 7: SFLL-hd principle of operation.

Similar to other CAC techniques, the hypothesis of SFLL-hd is that the corrupting unit is immersed into the circuit after logic synthesis becoming indistinguishable to the attacker. However, this hypothesis was not proven strong and structural attacks, a.k.a functional analysis attacks, were developed that identify and remove the corrupting unit [100], or recover the key using the HD properties [101].

The structure of CAC techniques compensates for the target security level against different attacks by defining the number of PIPs and the PPA overhead. SFLL-flex specifies the set of PIPs by means of a lookup table. The locked circuit produces the correct output if all PIPs are loaded. The structural attack in [102] aims at recovering the PIPs. It demonstrates how the optimization performed to minimize the PPA overhead executed by the

Electronic Design Automation (EDA) tools may expose the PIPs. The attack presented in [102] broke SFLL-hd/flex and .

SFLL-rem does not have a corrupting unit [105]. To create the corrupting circuit, it implements a fault-injection campaign to generate potential PIPs. SFLL is detailed and discussed further in Section 4.2 and it is employed to secure an RF transceiver. SFLL-rem has provable-resilience against all existing attacks. However, since it relies on EDA tools to hide the traces of the corrupted circuit, considerations to safely implement the defense are presented in [102].

2.2.4.2 *Generic logic locking techniques for RF transceivers*

Leveraging logic locking techniques to lock an Analog and Mixed-Signal (A/M-S) design via locking its digital section was proposed in [75, 107–109]. In [75], locking targeted the digital processor in the feedback calibration loop, while in [107–109] locking targeted digital blocks within the signal processing chain, a locking approach called *MixLock*.

2.2.5 *Passive and active countermeasures*

The countermeasures mentioned in Sections 2.2.1, 2.2.2, 2.2.3, and 2.2.4 are active countermeasures because they prevent or deter piracy attacks. In contrast, passive countermeasures are security mechanisms based on unique chip identification. Passive hardware techniques developed to protect ICs are watermarking and passive IC metering. Watermarking refers to a mechanism containing a unique IP/IC signature embedded in the IP/IC design. The signature can be detected on a post-fabrication IC to assess whether the IC holding the signature is and an authorized instance by the IP/IC owner [110]. Passive IC metering is a set of security protocols that allow tracking of the ICs after fabrication to control and count the ICs produced. Passive IC metering aims at embedding a unique per-device ID in each manufactured instance. In this way, the owner can analyze a manufactured IC, extract its ID and check if that IC corresponds to an IC requested by the owner or, on the contrary, is a cloned or overproduced instance [111]. In [112], the authors present a method to detect counterfeits by monitoring the supply voltage distribution over the IC surface through a network of sensors. IC metering was further improved with active mechanisms, like locking [80, 113, 114].

A secure countermeasure should achieve all security attributes, i.e., confidentiality by actively protecting intellectual property from third parties; integrity and availability by ensuring that the design has not been tampered with and can operate continuously; and authenticity by providing a unique per-device identity [115].

2.2.6 *Key management*

Common to any locking approach, this section briefly summarizes different key management schemes. The correct key is a designer's secret, and it is not shared with any potentially untrusted party, i.e., System-on-Chip (SoC) integration house, foundry, or end-user. The chip is securely activated after fabrication by storing the secret key in a Tamper-Proof Memory (TPM) such that it is erased on detecting a probing attempt. In this case, the secret key is common to all chips, thus if it is leaked any chip instance can be unlocked. Alternatively, a key provisioning on-die unit can be used to ensure that each chip is unlocked only by a user key, which is unique to that chip [116]. A standard scheme [69] uses a Physical Unclonable Function (PUF) [117, 118] to generate on-die a chip identification key, then a chip-unique user key is generated by XORing the identification key with the common key. The common key is generated internally by XORing the user and identification keys. Read access to the PUF output is disabled after recording to prevent probing attacks by end-users. Another key management scheme uses a PUF and Rivest-Shamir-Adleman (RSA) encryption to securely activate the chip remotely [80]. In a remote activation scheme, which requires an applied cryptographic function, the cryptographic hardware must be very lightweight and selected according to area and power consumption constraints [119].

2.3 CONTRIBUTIONS TO THE STATE-OF-THE-ART

2.3.1 *Hardware Trojans attacks in RF transceivers*

Chapter 3 proposes a novel and practical HT attack in the context of RF transceivers where the HT operates exclusively in the digital baseband PHY. The underlying idea is to modulate leaked data with the preamble part of the transmitted frames which serves for the synchronization of the receiver with the transmitter [55]. By doing so the synchronization still succeeds and the HT is non-intrusive and transparent to the communication link as the transmitted data is not affected.

2.3.2 *Anti-piracy RF transceiver design*

MixLock is thoroughly studied and demonstrated in Chapter 4 to protect RF transceivers at the system level. Generic logic locking techniques, like the ones presented in this section, cannot be blindly applied to RF transceivers. In this regard, Chapter 4 details how to fine-tune the SoA SFLL-rem [105] generic logic locking defense in the context of RF transceivers. The methodology and results presented in Chapter 4 were published in [120].

Chapter 5 is dedicated to *SyncLock* [121, 122], a domain-specific logic locking technique to secure RF transceivers. *SyncLock* takes advantage of a specific part of the signal processing chain found in any RF transceiver. The solution is called *SyncLock* since it locks the synchronization of the transmitter with the receiver. If a key other than the secret key is applied, synchronization and, thereby, communication fails. *SyncLock* is implemented using a novel locking concept consisting of two spatially separated mechanisms. A hard-coded error is hidden in the design to break synchronization while error correction, i.e., unlocking, takes place in another part of the design by applying the secret key.

3

LEAKING WIRELESS ICS VIA HARDWARE TROJAN-INFECTED SYNCHRONIZATION

Section 2.1 has analyzed the prior art on Hardware Trojans (HTs) in the context of RF transceivers. This chapter presents an HT attack in wireless ICs that aims at leaking sensitive information within a legitimate transmission, a.k.a covert channel. The HT is hidden inside the transmitter modulating the sensitive information into the preamble of each transmitted frame which is used for the synchronization of the transmitter with the receiver. The data leakage does not affect synchronization and is imperceptible by the inconspicuous nominal receiver as it does not incur any performance penalty in the communication. A knowledgeable rogue receiver, however, can recover the data using signal processing that is too expensive and impractical to be used during run-time in nominal receivers. The HT mechanism is designed at circuit-level and is embedded entirely into the digital section of the RF transceiver having a tiny footprint. The proposed HT attack is demonstrated with measurements on a hardware platform. We demonstrate the stealthiness of the attack, i.e., its ability to evade defenses based on testing and run-time monitoring, and the robustness of the attack, i.e., the ability of the rogue receiver to recover the leaked information even under unfavorable channel conditions.

The rest of the chapter is structured as follows. Section 3.1 describes the theory of the proposed HT attack, including the threat model, working principle, and applicability. In Section 3.2, we show the HT attack implementation, including the circuit-level design, overhead, rogue receiver design, and throughput of the covert channel. Section 3.3 presents the hardware platform and we physically demonstrate the attack with measurements, including its transparency to the legitimate communication, its resilience to test-based and run-time defenses, a demonstrator from the attacker's perspective, and the reliability analysis of the covert channel. In Section 3.4, we discuss other generic HT countermeasures potentially applicable in the context of RF transceivers. Section 3.5 concludes the chapter.

3.1 PROPOSED ATTACK: THEORY

3.1.1 *Threat model*

We consider three wireless ICs as depicted in Fig. 8, namely Alice that has been tampered with by an attacker, Bob that establishes a communication link with Alice and is a legitimate receiver, and Eve

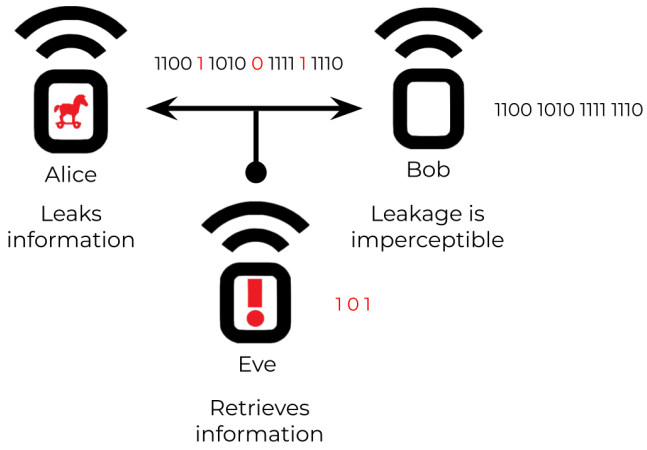


Figure 8: Threat model.

that is a rogue receiver having knowledge of the existence of the HT into Alice's transmitter hardware. The HT has been implanted into the baseband of the transmitter of Alice during the design or fabrication phases. Unbeknownst to Alice, while performing an authorized communication with Bob, she is disclosing valuable information to Eve, e.g., the secret cipher key that is thereafter used to decrypt the transmitted data, sensitive data from body sensors or other Internet-of-Things (IoT) devices or the weights of a proprietary Deep Neural Network (DNN) model. The information is encoded into bits that are well hidden within the transmission signal of Alice.

The attacker can be: (a) a Third-Party Intellectual Property Block (3PIP) provider that delivers the HT-infected digital section of the RF transceiver to the inconspicuous design house which then integrates it together with the AFE; (b) the design house itself that has easy access to perform the malicious modifications; or (c) the foundry which receives the GDSII file and can insert the HT either by directly modifying the layout or by first reverse engineering the file to extract the circuit netlist and then inserting the HT at the transistor-level or gate-level [123].

3.1.2 *Working principle*

In any wireless communication protocol the payload is transmitted along with the PHY specifications. The baseband Digital Signal Processing (DSP) prepares the payload in a frame format for transmission. The Physical layer Protocol Data Unit (PPDU) frame format of an OFDM IEEE 802.11 transmission consists of several OFDM symbols. These symbols are divided into 3 parts, namely preamble (a.k.a. SYNC), header (a.k.a. SIGNAL), and payload (a.k.a. DATA). The preamble section is composed of two different training symbol sequences, namely a Short Training Sequence (STS) and a Long Training

Sequence (LTS). Fig. 9 shows the PPDU of an IEEE 802.11 transmission with the 3 mentioned parts as defined in the IEEE 802.11 standard [124]. The STS field consists of ten identical short symbol repetitions and it is used for timing acquisition based on the Schmidl and Cox algorithm [125], i.e., to detect the start of the frame, and for coarse CFO estimation[124]. The LTS field consists of two long symbol repetitions and is used for channel estimation and fine CFO estimation[124].

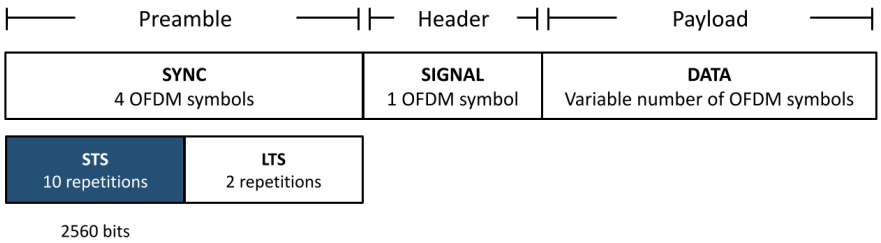


Figure 9: PPDU frame format of an OFDM IEEE 802.11 transmission.

The STS is described by the IEEE 802.11 standard[124] using its frequency domain representation, denoted by STS_F. STS_F is composed of 64 complex values, i.e., having real (I) and imaginary (Q) components, also called subcarriers or frequency bins, each of 16-bit length, i.e., STS_F has $N = (64 + 64) * 16$ bits = 2048 bits. The 64 subcarriers are indexed from -32 to 31, and there are 12 non-zero subcarriers whose signed decimal floating-point value and hexadecimal fixed-point value using a 16-bit word length are shown in the second and third columns of Table 3, respectively.

Table 3: STS_F as defined in the IEEE 802.11 standard [124].

Index (k)	Floating-point (I,Q)	Fixed-point (I,Q)
$-24, -16, -4, 12, 16, 20, 24$	$1.4720, 1.4720$	$16'h5E35$, $16'h5E35$
$-20, -12, -8, 4, 8$	$-1.4720, -1.4720$	$16'hA1CA$, $16'hA1CA$
others	$0.0, 0.0$	$16'h0000, 16'h0000$

The STS that is prepended to the frame is derived by performing an Inverse Fast Fourier Transform (IFFT) on the 64 subcarriers composing STS_F, as depicted in Fig. 10. The IFFT generates a time-domain representation of the STS, denoted by STS_t, of the same size as STS_F, composed of 64 I/Q samples, each of 16-bit length. STS_t has a periodicity of 16 samples, i.e., it contains 4 repetitions of 16 samples. The final STS is formed by concatenating two and a half STS_t to obtain the 10 repetitions required by the standard. The transmitter performs those operations at the OFDM Modulation block, which contains an IFFT unit, and at the Framer block, performing the concatenation operation, as depicted in Fig. 11. The same operation is performed to the frequency-domain representation of the LTS, denoted by LTS_F, to obtain the time-domain version, denoted by LTS_t. The preamble is

formed by concatenating the final STS and two and a half LTS_t. The resulting preamble is prepended to any transmitted frame as it is depicted in Fig. 9.

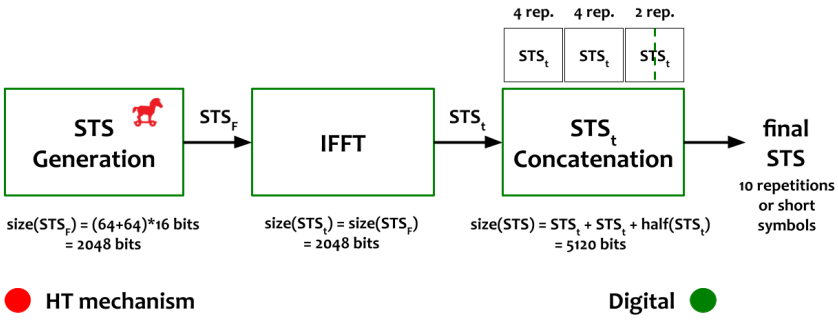


Figure 10: The detailed STS generation.

In this chapter, we propose an HT attack which leaks secret information through the baseband STS_F. Fig. 11 shows a block diagram where sensitive information from outside the baseband processor is driven to the STS generation block, as it is depicted by a red dotted line, where the HT will be implemented. For simplicity, Fig. 11 shows only parts of the transmitter of the wireless IC. This leakage scheme where secret information in one part of the design is driven to another part of the design, i.e., the DSP or AFE, is used in all of the previous works [44, 45, 48–52]. Indeed, in an Application-Specific Integrated Circuit (ASIC) implementation of the RF transceiver where all parts are integrated on the same substrate, sniffing the valuable information from the memory where it is stored and establishing the connection to the STS generation block is totally feasible if the attacker is the design house or foundry. More specifically, the proposed HT attack, called **Amplitude Modulation (AM) STS HT attack**, consists in modulating the amplitude of the STS_F subcarriers with the information bits being stolen.

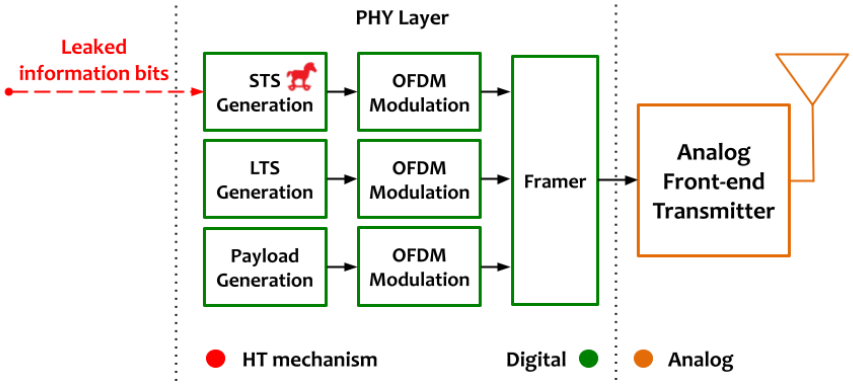


Figure 11: AM STS HT location within the architecture of a wireless IC, showing only part of the transmitter's sub-blocks.

In detail, from the 12 non-zero subcarriers of the STS_F, we choose to leak one byte of information using 8 subcarriers and we use the other 4 subcarriers to set a non-modulated amplitude threshold to reduce the error rate for the rogue receiver. Therefore, in each frame, a byte of the disclosed information is leaked into 8 subcarriers, called *corrupted subcarriers*, with the indexes of these subcarriers being the same for all frames. The amplitude of the corrupted subcarriers is multiplied by $\alpha < 1$, i.e., it is slightly lowered, when the leaked bit is $'1'$, otherwise the amplitude is preserved for leaked bits $'0'$.

Let us consider for example that the 8 corrupted subcarriers have indexes $k = \{-24, -20, -16, -8, 4, 8, 16, 24\}$. The STS_F infected by the AM STS HT is then given in floating-point values by

$$\begin{aligned}STS^{HT}_{F_{-32,31}} =\{&0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ \alpha'_{-24}(1.472+1.472j)\ 0\ 0\ 0\ \alpha'_{-20}(-1.472-\\ &1.472j)\ 0\ 0\ 0\ \alpha'_{-16}(1.472+1.472j)\ 0\ 0\ 0\ -1.472-1.472j\ 0\ 0\ 0\\ &\alpha'_{-8}(-1.472-1.472j)\ 0\ 0\ 0\ 1.472+1.472j\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ \alpha'_{4}(-1.472-\\ &1.472j)\ 0\ 0\ 0\ \alpha'_{8}(-1.472-1.472j)\ 0\ 0\ 0\ 1.472+1.472j\ 0\ 0\ 0\ \alpha'_{16}(1.472+\\ &1.472j)\ 0\ 0\ 0\ 1.472+1.472j\ 0\ 0\ 0\ \alpha'_{24}(1.472+1.472j)\ 0\ 0\ 0\ 0\ 0\ 0\ 0\}\end{aligned}$$

where $\alpha'_k = \alpha$ if the leaked bit in subcarrier k is 1 and $\alpha'_k = 1$ if the leaked bit in subcarrier k is 0.

At the receiver side, the synchronization process to find the start of the frame consists of performing a cross-correlation operation between the received samples and the ideal OFDM modulated samples of the standard-defined STS_F. Since the synchronization process is done with the received STS_t, the STS_F generation at the transmitter side can be freely modulated by the attacker to leak data under the condition that the correlation properties at the receiver side are kept. Therefore, the amplitude modulation operation in the proposed HT attack, dictated by the choice of α, is performed while ensuring that it will have no effect on the correlation properties. In this way, the synchronization process is not affected and the covert channel is unnoticed by the inconspicuous receiver.

As we will see in Section 3.3, the magnitude of α represents a tradeoff between stealthiness of the AM STS HT attack and effective recovery of leaked data by the rogue receiver. Larger α increases the probability of detection while reducing the probability of error in the recovered data. The attacker can choose a small α to circumvent detection, and exploit several consecutive transmissions of the sensitive bits to perform an error correction scheme.

3.1.3 *Applicability*

A synchronization process is present and necessary in any wireless communication protocol. For instance, Wireless Local Area Network (WLAN) IEEE 802.11 (i.e., Wi-Fi), Wireless Personal Area Network (WPAN) IEEE 802.15.1 (i.e., Bluetooth), and Low-Rate Wireless

Personal Area Network (LR-WPAN) IEEE 802.15.4 (i.e., Zigbee), use correlation-based synchronization algorithms. Moreover, all of the above standards use a preamble for synchronization, thus the proposed AM STS HT attack is virtually applicable to all of them.

3.2 PROPOSED ATTACK: IMPLEMENTATION

3.2.1 *Circuit-level design*

Fig. 12 shows the hardware implementation of the STS block in Fig. 11. The STS block creates the 16-bit fixed-point values for each element of the sequence $STS_{F_{-32,31}}$, starting with the element with index $k = 0$ up to $k = 31$, then from $k = -32$ up to $k = -1$. These fixed-point values are reported in Table 3. Note that the real (I) and imaginary (Q) components have the same value. Therefore, for simplicity and without loss of generality, in the following examples we refer only to the real value. Taking as an example the element with index $k = 0$ that has floating-point value 0.0, the STS block has to create the fixed-point value $16'h0000$. The SEL input is a 6-bit word and selects the index k of one of the 64 subcarriers to be created. The three multiplexers (MUXes) in Fig. 12 receive a fixed 64-bit value as shown in the upper part of Table 4, where DATA_CX and MuxX denote the input and output of MUX X, respectively. The position of the selected bit transferred at the output of each MUX equals the decimal representation of the SEL input. The output values of the MUXes are then concatenated according to the scheme in the first row in the upper part of Table 5 to create the 16-bit fixed-point value of the selected element of the sequence $STS_{F_{-32,31}}$.

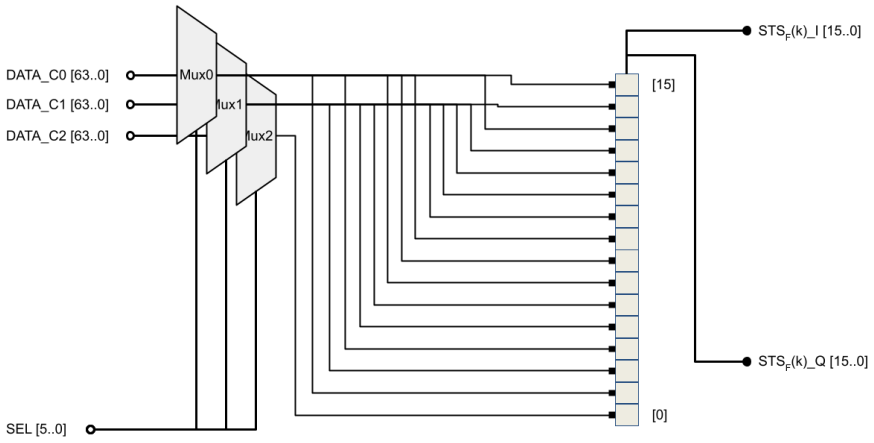


Figure 12: Circuit schematic of the nominal STS block.

For example, for element with index $k = 4$ that has fixed-point value $16'hA1CA$ in hexadecimal representation, SEL = $6'b000100$, bit position $k + 1 = 5$ is selected at the inputs of the MUXes as shown in

Table 4: Input values of MUXes.

MUX	Name	Input (binary value)
HT-free STS block implementation (Fig. 12)		
Mux0	DATA_C0	64'b0000 0001 0001 0000 0001 0000 0000 0000 0000 0000 0000 0000 0000 0001 0001 0000
Mux1	DATA_C1	64'b0001 0000 0000 0001 0000 0001 0000 0000 0000 0001 0001 0001 0001 0000 0000 0000
Mux2	DATA_C2	64'b0001 0000 0000 0001 0000 0001 0000 0000 0000 0001 0001 0001 0001 0000 0000 0000
HT-Infected STS block implementation (Fig. 13)		
MUX	Name	Input (binary value)
Mux0	DATA_C0	64'b0000 0001 0001 0000 0001 0000 0000 0000 0000 0000 0000 0000 0000 0001 0001 0000
Mux1	DATA_C1	64'b0001 0000 0000 0001 0000 0001 0000 0000 0000 0001 0001 0001 0001 0000 0000 0000
Mux2	DATA_C2	64'b0001 000σ[7] 0000 000¬σ[6] 000σ[5] 000¬σ[4] 0000 0000 0000 000¬σ[3] 0001 000¬σ[2] 0001 000σ[1] 000σ[0] 0000
Mux3	DATA_C3	64'b0000 000¬σ[7] 0001 000σ[6] 000¬σ[5] 000σ[4] 0000 0000 0000 000σ[3] 0000 000σ[2] 0000 000¬σ[1] 000¬σ[0] 0000
Mux4	DATA_C4	64'b0000 0001 0001 000σ[6] 0001 000σ[4] 0000 0000 0000 000σ[3] 0000 000σ[2] 0000 0001 0001 0000

Note: ¬σ is the inverse of σ.

blue in the upper part of Table 4, and the MUXes output concatenation is as shown in blue in the upper part of Table 5 resulting in the desired value of 16'hA1CA. The same hardware and concatenation operations as shown in the upper part of Table 5 are used to generate any element k by setting the input SEL equal to k in decimal.

Table 5: Concatenation operation of the outputs of the MUXes.

	Concatenation of MUXes (M#)	M0,M1,M0,M1	M1,M1,M1,M0	M0,M0,M1,M1	M0,M1,M0,M2
HT-free STS block implementation (Fig. 12)					
Subcarriers (k): $-24,-16,-4,12,16,20,24$	Fixed-point value	0101	1110	0011	0101
	Hexadecimal value	5	E	3	5
Subcarriers (k): $-20,-12,-8,4,8$	Fixed-point value	1010	0001	1100	1010
	Hexadecimal value	A	1	C	A
Subcarriers (k): others	Fixed-point value	000	0000	0000	0000
	Hexadecimal value	0	0	0	0
HT-infected STS block implementation (Fig. 13)					
	Concatenation of MUXes (M#)	M0,M1,M0,M1	M2,M1,M2,M0	M3,M3,M2,M2	M3,M2,M4,M2
Concatenated output of subcarrier $k=4$	Fixed-point value	1010	σ[0]0σ[0]1	¬σ[0]¬σ[0]σ[0]σ[0]	¬σ[0]σ[0]1σ[0]
Leaked bit $\sigma[0]=0$	Fixed-point value	1010	0001	1100	1010
	Floating point value			-1.4720	
Leaked bit $\sigma[0]=1$	Fixed-point value	1010	1011	0011	0111
	Floating point value			-1.3248	

Note: ¬σ is the inverse of σ.

Fig. 13 shows the hardware modifications in the STS block to implement the AM STS HT attack. The stolen bits are streamed into the STS block whose output is the 16-bit fixed-point values of the elements of the sequence $\mathrm{STS}^{\mathrm{HT}}_{\mathrm{F}_{-32,31}}$. The attacker needs to define which subcarriers will be corrupted, as well as the amplitude modulation α. Without loss of generality, similarly to Section 3.1.2, let us assume that 8 subcarriers are corrupted with indexes $k=\{-24,-20,-16,-8,4,8,16,24\}$. Let us also assume $\alpha = 10\%$. As shown in Fig. 13, the design is

modified to add two extra MUXes. The inputs of the five MUXes are shown in the lower part of Table 4. The first two MUXes have constant input values, while MUXes $2-4$ have some constant bits and some bits coming from the leaked secret information. The 8 stolen bits per frame are denoted by $\sigma[j]$, $j = 0, \cdots, 7$ in Table 4, where $\sigma[0]$ is the least significant bit and $\neg\sigma[j]$ denotes the inverse of $\sigma[j]$. In this implementation, stolen bits $\{0, \cdots, 7\}$ are mapped to subcarriers with indexes $\{4, 8, 16, 24, -24, -20, -16, -8\}$ in this exact order. When the value of the leaked bit is $'0'$ the corresponding subcarrier according to this mapping has the same amplitude as in the HT-free case. On the other hand, when the value of the leaked bit is $'1'$ the amplitude of the corresponding subcarrier according to this mapping is multiplied by α. The concatenation of the outputs of the MUXes is shown in the first row of the lower part of Table 5.

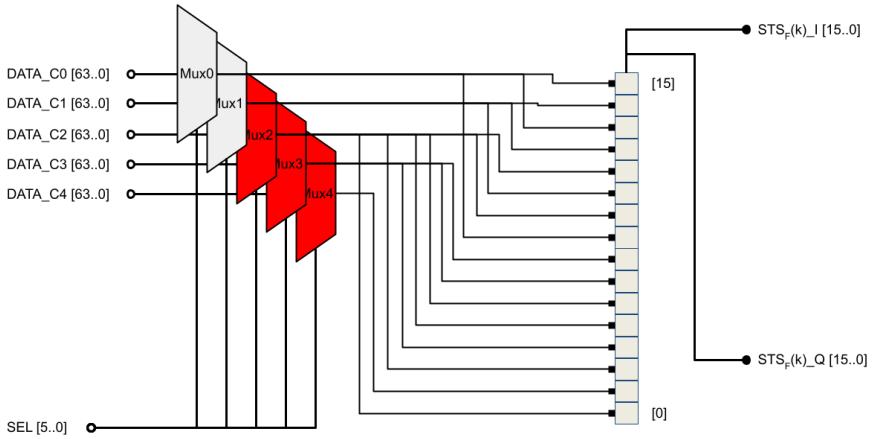


Figure 13: Circuit schematic of the HT-infected STS block.

As an example, let us assume that the leaked byte with the secret information is $\sigma = 8'b00010001$. In this scenario, subcarrier with index $k = 4$ will have a modulated amplitude of $0.9 \times (-1.4720) = -1.3248$, subcarrier with index $k = -24$ will have a modulated amplitude of $0.9 \times (1.4720) = 1.3248$, while the rest of the subcarriers will remain at the non-modulated amplitude, i.e., -1.4720 or $+1.4720$. Let us further consider subcarrier with index $k = 4$. The outputs of the MUXes for SEL input $k + 1 = 5$ are shown in red in the lower part of Table 4. The second row of the lower part of Table 5 shows in red the generation of the element of STS_F^{HT} with index $k = 4$ after the concatenation operation. Certain bits depend on the leaked bit $\sigma[0]$. The third and fourth rows of the lower part of Table 5 show the 16-bit fixed-point and floating-point values in the case where $\sigma[0] = 0$ and $\sigma[0] = 1$, respectively. For $\sigma[0] = 0$ the concatenated output value remains at -1.4720, whereas for $\sigma[0] = 1$ the concatenated output value is reduced to -1.3248, i.e., 10% below the standard.

The HT-infected STS block implementation shown in Fig. 13 resulted from a synthesis using the software Quartus II 16.0 from Intel™. It should be noted that the implementation is not unique and different implementations can result using different optimization levels of synthesis.

Moreover, the implementation of the PHY of the IEEE 802.11 standard can vary depending on available resources or developer preferences, as long as it still complies with the standard. Herein, we showed the HT design in the case where STS_t is re-computed for every frame. Alternatively, the STS_t may be computed once, then stored and reused for each frame. In this scenario, the HT mechanism would modulate the stored STS_t samples to implement the information leakage.

3.2.2 *Overhead*

To calculate the HT overhead and prove the low footprint of the proposed HT attack, we utilized as base HT-free implementation an open-source IEEE 802.11 compatible Software Defined Radio (SDR) Hardware Description Language (HDL) modem [126]. The project is called **bladeRF-wiphy** as it implements the PHY of the IEEE 802.11 in the bladeRF board [127]. More details about the bladeRF board will be given in Section 3.3.1. Starting from the HT-free implementation, we made the modifications of Section 3.2.1 to incorporate the HT into the PHY of the modem and we re-synthesized the project using Quartus II 16.0 from Intel™ to find the resultant overhead. The HT design requires the addition of 58 Adaptive Look-Up Tables (ALUTs) which translates into a negligible increase of 0.109% in the total number of required ALUTs to implement the entire PHY of the modem. According to Intel, the physical resources represented by an ALUT differ depending on the device family. For example, in Stratix V, Arria V and Cyclone V devices, there is one combinational look-up table (LUT) and two registers per ALUT.

3.2.3 *Rogue receiver*

To extract the leaked data from the received signal, after synchronization the rogue receiver needs to demodulate the OFDM symbols containing the STS_t, extract the values stored in the corrupted subcarriers, and demodulate the AM signal. The hardware needed to retrieve the leaked data is depicted in Fig. 14.

The above process is not performed in the nominal receiver since after finding the start of the frame the STS_t is no longer processed. In Section 3.3, we investigate different defenses aiming to spot the HT activity. One of them, namely the STS constellation test, looks for anomalies in the constellation of the demodulated STS_t, i.e., it emu-

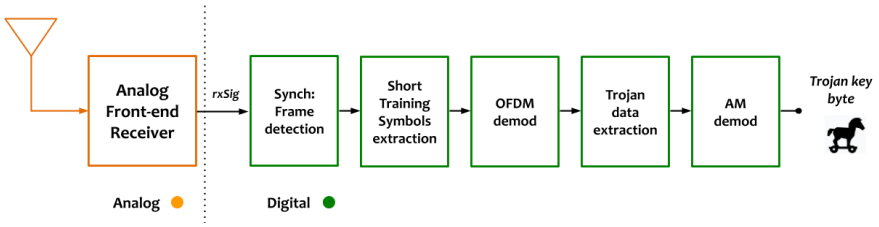


Figure 14: Rogue receiver block architecture for the AM STS HT attack.

lates some of the stages performed by the rogue receiver. Although this defense may be able to detect HT attacks that act on the STS, it is not cost-effective to be implemented into every nominal receiver because the purpose-specific hardware needed to achieve this would incur a large area, power, and delay overhead. In addition, the nominal receiver does not know which subcarriers are corrupted and which are affected only by noise, therefore a very high SNR would be required to detect any anomalies. We will return to this point in our experimental results in Section 3.3.

3.2.4 *Throughput of the covert channel*

In the proposed HT attack, the throughput of the covert channel can reach 12 bits per frame (bpf), whereas the number of transmitted frames per second (fps), denoted by n_{fps}, depends on the length of the transmitted frames. There are three types of IEEE 802.11 frames, namely management, control, and data. The second layer of the Open Systems Interconnection (OSI) model, namely the Data Link layer, defines the number of bytes contained in each of the different IEEE 802.11 frame types. All types include a PHY preamble, meaning that every transmitted frame can carry up to 12 bits of the covert channel payload. Therefore, the throughput of the covert channel, denoted by Th_{HT}, expressed in bits per second (bps) is given by $Th_{HT} = 12 \cdot n_{fps}$. For instance, short control frames, such as the Acknowledge (ACK) and Clear to Send (CTS), have a frame length of 16 µs, which implies a frame rate of 62,500 fps, and thus a throughput of 750,000 bps. Longer control frames, such as the Request to Send (RTS) or variable-length data frames, reduce the number of fps according to their frame duration, thus the throughput of the covert channel is also reduced.

3.3 MEASUREMENT RESULTS

3.3.1 *Hardware platform*

To demonstrate the proposed AM STS HT attack we use the SDR bladeRF board from Nuand™ [127] shown in Fig. 15. This board contains three main chips: an RF transceiver LMS6002 from Lime

Microsystems™, a Field-Programmable Gate Array (FPGA) Cyclone IV from Intel™ (formerly ALTERA™), and a USB 3.0 peripheral controller FX3 from Cypress™.

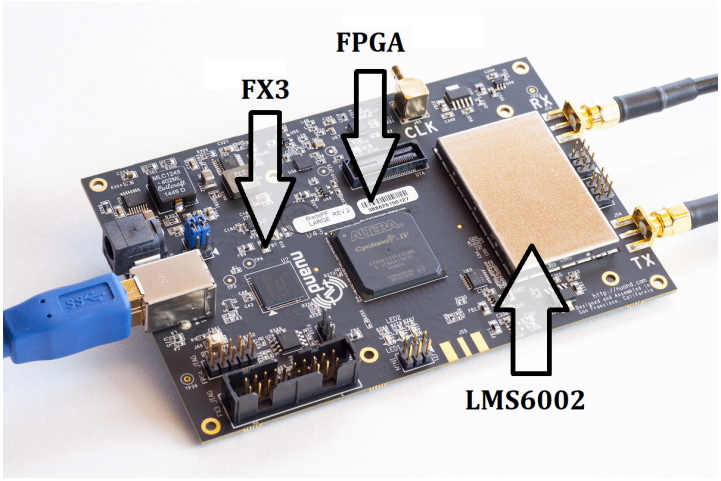


Figure 15: SDR bladeRF board from Nuand.

The RF transceiver has on-chip baseband and RF loopback modes allowing us to perform measurements using the same board. For our measurements we employ the baseband loopback mode and we model the communication channel with Additive White Gaussian Noise (AWGN). Fig. 16 shows a block diagram architecture of the test setup used for the experiments. It shows the interaction between the baseband DSP, the FPGA, and the AFE, as well as the loopback modes. The HT attack mechanism at the transmitter side is placed in the preamble generation depicted by a Trojan horse. The defense mechanisms at the receiver side were implemented in the baseband at the synchronization process during run-time or at test time and are depicted by a crossed-out Trojan horse.

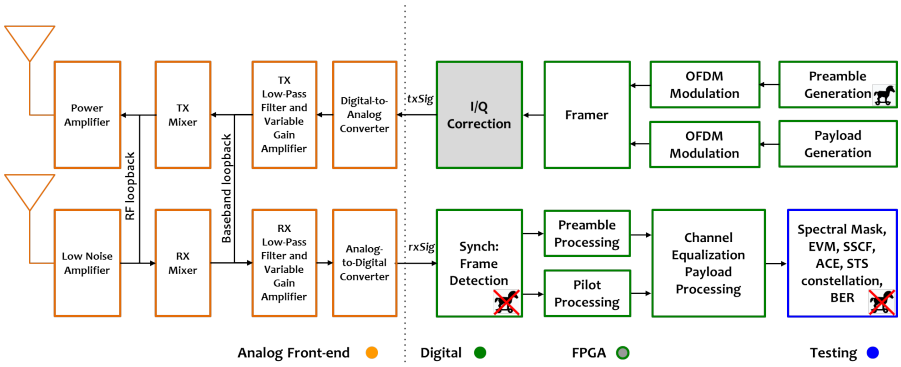


Figure 16: RF transceiver test setup using the loopback mode. The Trojan horse shows the stage of the attack and the crossed-out Trojan horse show the stages of the defense mechanisms.

3.3.2 *Transparency to legitimate communication: choice of α*

The AM STS HT attack leaks data through the STS, thus it affects the coarse CFO estimation. However, the standard uses the LTS, which is left intact by the AM STS HT attack, for fine CFO estimation and correction. The hypothesis is that there exists a value of α below which the introduced CFO due to the AM STS HT attack can still be compensated, thus making the AM STS HT attack transparent to legitimate communication. In essence, while the inconspicuous legitimate receiver successfully synchronizes and thereafter discards the preamble, the rogue receiver who knows the leaking mechanism processes further the preamble to retrieve the leaked data.

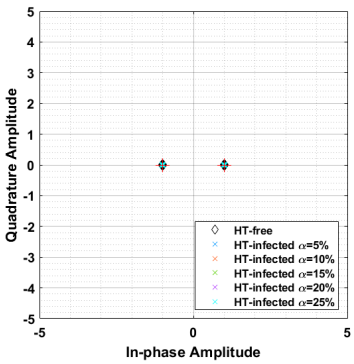
To demonstrate that our hypothesis is valid, we studied the impact of different α values on the constellation diagram of the received decoded payload signal and on the BER performance. We performed these measurements using an OFDM transmission using BPSK modulation for different values of SNR for an HT-free device and an AM STS HT-infected device using different values of α.

Fig. 17 shows the constellation diagrams of the received payload. The HT-free device and the AM STS HT-infected device are visually indistinguishable, regardless of the α value. Fig. 18, however, shows that for α greater than 15% the AM STS HT-infected device presents a degraded BER for SNR greater than 2 dB that distinguishes it from the HT-free device. In conclusion, the proposed AM STS HT attack does not have any impact on either the synchronization or the performance of legitimate communication when the value of α is chosen to be less than 15%.

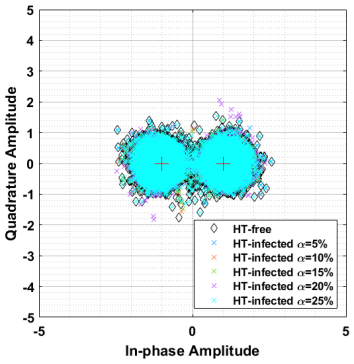
3.3.3 *Resilience to test-based and run-time defenses*

The proposed AM STS HT attack was tested against known defences, previously introduced in Section 2.1.4, aiming at detecting HT-infected chips at post-manufacturing test or HT activity during run-time. For all the following experiments, unless explicitly mentioned, we use $\alpha = 10\%$.

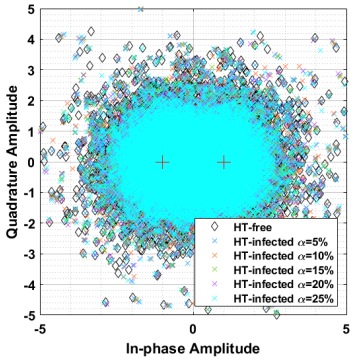
Spectral mask test: We performed spectral measurements to analyze whether transmissions with an AM STS HT-infected device and transmissions with an HT-free device were distinguishable. The Power Spectral Density (PSD) of the transmitted signals are shown in Fig. 19. The signals are centered at the carrier frequency 2.41 GHz and they occupy a 20 MHz bandwidth. Along with the signals, the spectral mask margins specified by the IEEE 802.11 standard are shown [124]. As it can be seen, the PSD of the HT-infected device is indistinguishable from that of the HT-free device and is standard-compliant. Note that there is no pre-processing that can be performed on the transmitted data to make the two PSD curves distinguishable. Any



(a) SNR = 40 dB.



(b) SNR = 10 dB.



(c) SNR = 2 dB.

Figure 17: Constellation diagrams of the decoded payload of the received OFDM-BPSK signal for different SNR values. The result is shown for an HT-free device and an AM STS HT-infected device using different values of α.

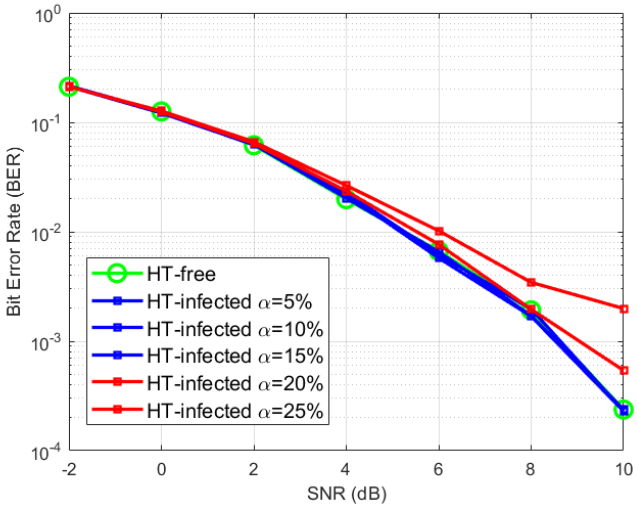


Figure 18: Measured BER of the received OFDM-BPSK signal using an HT-free device and an AM STS HT-infected device for different values of α.

subtle difference is due to the channel noise and RF impairments and not due to the HT activity.

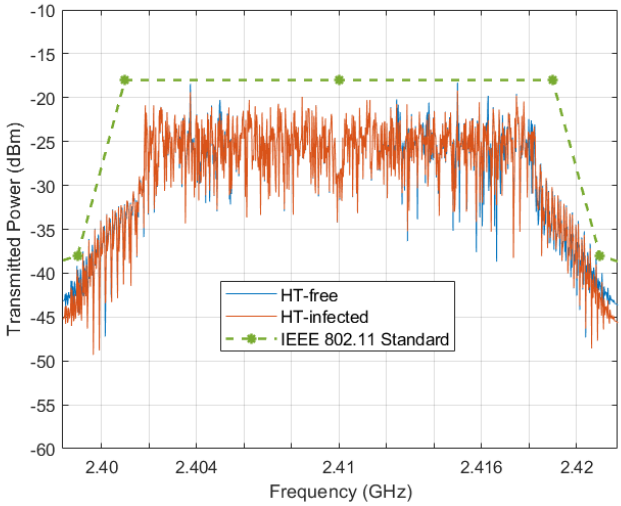


Figure 19: Measured Power Spectral Density (PSD) of signals transmitted with an AM STS HT-infected device and an HT-free device, along with the IEEE 802.11 standard spectral mask specification[124].

EVM test: Fig. 20 shows the Error Vector Magnitude (EVM) measurements at different transmission power levels for an OFDM-BPSK transmission with an AM STS HT-infected device and an HT-free device. As it can be seen, the EVM measurements are compliant with the IEEE 802.11 standard [124] and there is no impact on EVM due to HT insertion. It should be noted that the variability between transmissions from the HT-infected device and the HT-free device is due to electronic noise and hardware impairments in the SDR circuitry and not due to the HT activity. This is because the HT activity takes place in the preamble and the EVM test detects payload divergence from the ideal symbols of a target constellation.

SSCF test: To detect the HT presence using Statistical Side-Channel Fingerprinting (SSCF), we generated two populations with 1000 HT-free devices and 1000 HT-infected devices using 2000 different random combinations of I/Q gain and phase imbalance. For the gain imbalance we used 5% variability and for the phase imbalance 10% variability. To vary I/Q imbalance we used the I/Q correction module in the FPGA of the board. Then, for each device we transmitted the same information at six different power levels in the range from -27 dBm to -14 dBm and considered as feature the total received power at the receiver through the baseband loopback of the transceiver. We used half of the devices to train a one-class Support Vector Machine (SVM) classifier and the other half was used as an independent validation set.

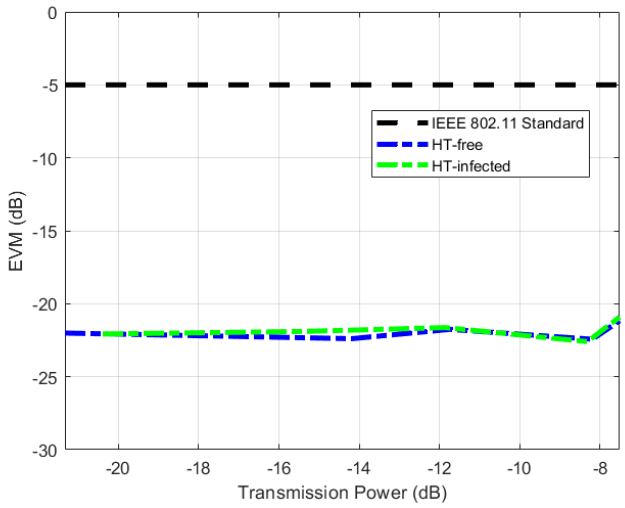
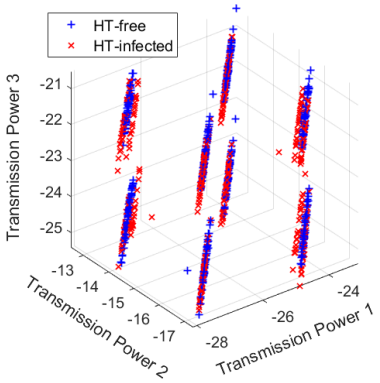
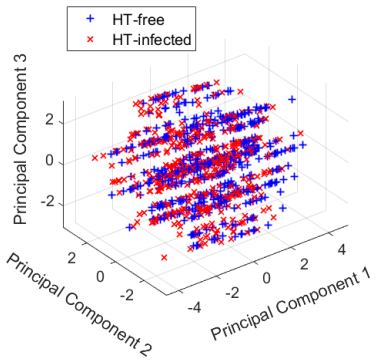


Figure 20: Measured Error Vector Magnitude (EVM) of the transmitted BPSK signal from an AM STS HT-infected device and an HT-free device, along with the IEEE 802.11 standard EVM specification [124].

The SVM shows a very poor accuracy to correctly classify the HT-infected devices, with a misclassification rate of 44.5%, essentially pointing to a random decision. Fig. 21a shows a 3D plot of the first 3 transmission power levels and Fig. 21b shows a 3D plot of the first 3 principal components resulting from a Principal Component Analysis (PCA). In both cases, the 2 populations are overlapped and, thereby, the HT-infected devices cannot be screened out. Due to the total overlapping, there is no classifier that can separate the two classes, i.e., using other popular classifiers such as a decision tree or a deep feed-forward neural network produces the same result.



(a) Projection of HT-free and HT-infected devices onto a 3D space composed of the first three transmission power levels.

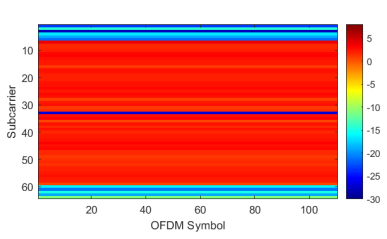


(b) Projection of HT-free and HT-infected devices onto a 3D space composed of the first three principal components.

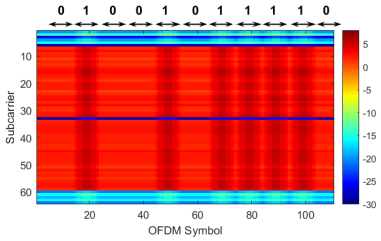
Figure 21: Measurement results from the SSCF test.

ACE test: The Adaptive Channel Estimation (ACE) test is capable of discriminating between signal variations due to channel impairments and HT activity only in the case where the HT activity is generated in the AFE. If the HT is infecting the baseband of the transmitter, i.e., the preamble of the signal, as is the case for the proposed AM STS HT attack, the ACE test will not detect the HT activity. To demonstrate this, Fig. 22 shows three ACE tests with the resulting post-channel equalization of the received payload. The signal amplitude is indicated by color variations, and the color bar on the right-hand side of the plots shows the color used to represent a given amplitude value. The x-axis shows the received payload divided in OFDM symbols and the y-axis shows the subcarrier indexes k. Fig. 22a shows an HT-free transmission where the received amplitude of the payload remains constant along the OFDM symbols. Fig. 22b presents an example of HT activity hidden in the amplitude modulation of the payload, like the approach presented in [51]. As it can be seen, the HT increases the amplitude of the received signal for a certain amount of time. This amplitude modulation corresponds to a leaked bit. In particular, the dark columns correspond to a leaked bit 1, whereas the lack of darkness corresponds to leaked bit 0. In this case, the ACE defense spots the leaked data, e.g. in Fig. 22b the leaked message is '01001011110'. In contrast, the ACE test is unable to detect the HT activity in the case of the proposed AM STS HT attack, as shown in Fig. 22c. When the HT is hidden in the STS of the transmitted signal, Fig. 22c does not point to any HT activity despite the fact that the covert channel is enabled.

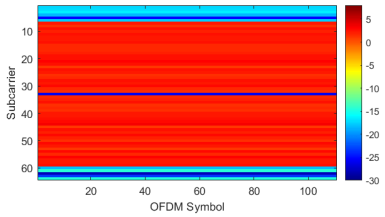
STS constellation test: Another possible defense for detecting an STS anomaly is to demodulate the preamble, extract the STS_F, and observe its constellation. In a HT-free transmission, the constellation is composed of only three symbols, namely $\{-1-j\ ,\ 0,\ 1+j\}$. If the observed constellation points can be distinguished from these expected three symbols, then the attack is deemed detected. For this defense, we compare the proposed AM STS HT attack with the attack proposed in [45], which also acts upon the STS in the preamble but implements the information leakage differently. In particular, the STS (alternatively called STF in [45]) contains BPSK symbols that are shifted by 45°. In [45], the data are leaked by introducing an additional phase shift $\Delta\phi$ into all STS symbols. This attack proposed in [45] is called PSK STF HT attack. This defense detects the PSK STF HT attack, whereas the proposed AM STS HT attack bypasses it successfully. This is demonstrated in Fig. 32. More specifically, Fig. 23a shows the resulting constellation for a 20 dB SNR and for 5 different phase shifts using the PSK STF HT attack in [45], whereas Fig. 23b shows the resulting constellation for the proposed AM STS HT attack for a 20 dB SNR and for 5 different amplitude modulations from $\alpha = 5\%$ to $\alpha = 25\%$. The black circles around the coordinates $-1-j$



(a) ACE test for an HT-free transmission.



(b) ACE test for an HT-infected transmission based on amplitude modulation at the AFE, like the approach presented in [51]. The HT activity on the payload is identified by the ACE test.



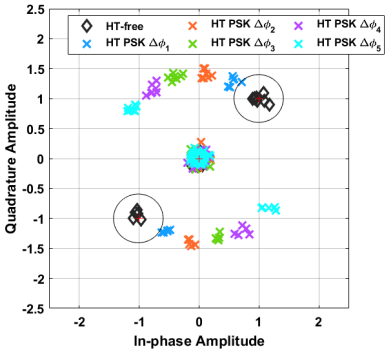
(c) ACE test for an HT-infected transmission with the AM STS HT. The HT activity on the STS cannot be identified by the ACE test.

Figure 22: Measurement results from the post-channel equalization of the received payload.

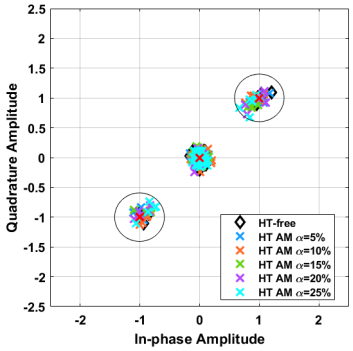
and $1 + j$ show where the STS symbols should be found ideally in the absence of a HT, along with the points at zero which are the subcarriers with zero amplitude value. In Fig. 23a most of the points are outside the circles making the PSK STF HT attack [45] easily noticeable, while in Fig. 23b all the points are inside the circles even for a large amplitude modulation of $\alpha = 25\%$ making the AM STS HT attack practically undetectable. To detect the AM STS HT attack a very high SNR would be required and a close examination of the constellation. Fig. 23c shows the constellation diagram of the STS_F preamble in such a scenario having 40 dB of SNR. To avoid being detected, an attacker must choose lower α values.

It should be noted that this STS constellation analysis is not cost-effective to be implemented as a run-time defense in every wireless receiver since specialised equipment is needed to perform it.

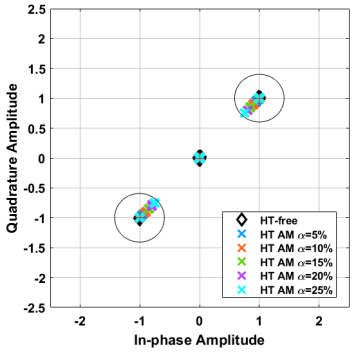
Single-branch STS_t correlation test: We propose a new less complex and low-cost run-time defense compared to the STS constellation test, which relies on comparing each individual I/Q branch of the stored nominal STS_t against the corresponding I/Q branch of the received STS_t. This defense mechanism is placed in the synchroniza-



(a) Constellation of the STS_F preamble using the PSK STF HT attack [45].



(b) Constellation of the STS_F preamble using the proposed AM STS HT attack.

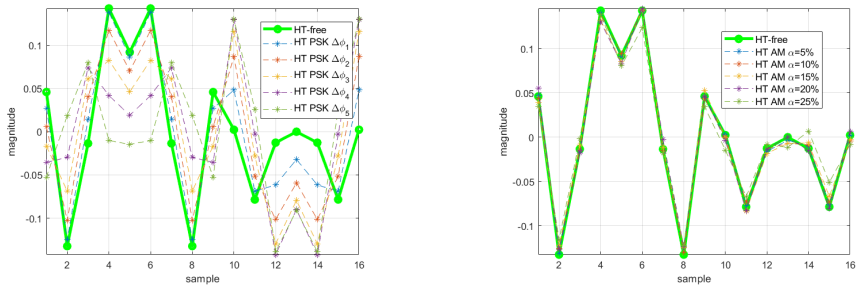


(c) Constellation of the STS_F preamble using the proposed AM STS HT attack at a high SNR scenario.

Figure 23: Measurement results from the STS constellation test.

tion process as depicted in Fig. 16 by a crossed-out Trojan horse in the Synch block.

The wireless receiver has access to the STS_t digital I/Q samples with which it performs the correlation operation to search for the start of the frame. STS_t corruption as a result of phase shifting due to the PSK STF HT attack [45] causes the I/Q branches to be unbalanced. To visualize the effect of a phase shifting in the STS_t, Fig. 24a shows a time domain comparison of the I branch of the HT-free STS_t against 5 different phase shifts. As it can be seen, the samples differ from the HT-free STS_t. Therefore, this proposed defense mechanism consists of performing a correlation operation between the first 16 samples of the I branch of the received STS_t and the first 16 stored samples of the I branch of the ideal STS_t. If the maximum value of the correlation is not found in the index 16, then the frame is infected by an HT. To strengthen the defense, the same correlation analysis can be repeated for the Q branch, and if any of the two correlation operations fails then an HT is detected.



(a) Real (I) part of the HT-free STS_t and the received STS_t for different phase shifts in the PSK STF HT attack [45].

(b) Real (I) part of the HT-free STS_t and the received STS_t for different amplitude values in the proposed AM STS HT attack.

Figure 24: Measurement results from the single-branch STS_t correlation test.

Although this low-cost and practical defense is capable of detecting the PSK STF HT attack in [45], as shown in Fig. 24a, the proposed AM STS HT attack still evades this defense, as shown in Fig. 24b for 5 different amplitude modulations.

3.3.4 *Demonstrator*

To demonstrate the HT functionality from the attacker's perspective, we performed an encrypted transmission using the HT-infected transmitter of Fig. 11, where the leaked information is the encryption key from an Advanced Encryption Standard (AES) core, and we attempted to decrypt the payload using the recovered leaked secret key. As payload, we transmitted the well-known standard 512 x 512 pixel color version test image of a mandrill (a.k.a. baboon[1]). The payload was processed, converted to plaintext, and finally encrypted using the AES algorithm with a 128-bit secret key. According to the threat model of Section 3.1.1, the demonstrator consists of Alice sending the encrypted image to Bob, while without her knowledge the encryption key is leaked via the corrupted subcarriers of the STS_F according to the proposed AM STS HT attack in Section 3.1.2. Eve who is the eavesdropper retrieves the stolen key from the HT-infected STS as described in Section 3.2.3 and tries to decrypt the cyphertext to reveal the message, i.e., the mandrill image in this case. As shown in Fig. 25, the received encrypted payload in Fig. 25a is decrypted correctly in Fig. 25b when applying the recovered leaked key. Section 3.3.5 details how the leaked key is recovered even at low SNR scenarios.

Key refreshment: Considering a wireless IC, where the encryption key is stored in a TPM, there will be no key updates unless more keys are stored in the TPM, which increases the size and complexity of the

1 Source of image: Signal and Image Processing Institute, University of Southern California

(a) Received encrypted payload.

(b) Decrypted payload using the recovered key leaked through the corrupted subcarriers of STS_F.

Figure 25: Demonstration of the AM STS HT attack: stealing the cypher key and recovering the transmitted encrypted mandrill (a.k.a baboon) image.

TPM. In addition, AES is a symmetric cipher, that is, if the transmitter refreshes the key then the key at the receiver must also be updated, leading to an increase in the complexity of the communication system. Therefore, we consider that an update of the encryption key is very unlikely in our scenario. However, even in the scenario of a key update, the number of frames used to transmit a payload while the key is constant is greater than the number of frames required to extract the key from the preamble of those frames. Thus, the rogue receiver could store the frames encrypted by some secret key, de-embed the key from the preambles of those frames, and decrypt the payload. Since the HT mechanism continuously leaks the key, changes in the register used to store the key would be reflected in the preamble of the new frames to be transmitted, and the rogue receiver could repeat the procedure described above for a new secret key.

3.3.5 *Reliability of the covert channel*

Assuming a 128-bit secret key and choosing to leak one byte per frame, i.e., only 8 out of the 12 non-zero subcarriers of the STS_F are corrupted, then after 16 frames the secret key would be completely transmitted. Starting from the 17th frame, the secret key will be repeated for the duration of the transmission creating redundancy of the leaked data. This redundancy allows the attacker to apply an error correction algorithm to recover the secret key even under low SNR conditions. We conducted an experiment to evaluate the resilience of the proposed technique in various SNR scenarios and for various HT amplitude modulations, i.e., α values. We used a simple voting sys-

tem in which, after having some redundancy, the value of each bit is chosen as the one that is repeated the most. For example, after 48 frames, the secret key has been repeated 3 times and we will have a redundancy of 3 for each of the 128 bits of the secret key. According to the voting system, the value of each bit is determined as the one that has been received at least twice. Then, after each key iteration we aimed at decrypting the ciphertext using the received key. If the ciphertext is decrypted correctly, then the recovered key is the correct one. Fig. 26 shows the results obtained from our experimentation. Each point is an average of 6 measurements to account for the channel noise. It can be observed that for larger α the number of iterations needed to recover the secret key without any error is lower. For example, for $\alpha = 10\%$, which is sufficient to thwart the single-branch STS_t correlation defense as shown from Fig. 24b, and for SNR values greater than 24 dB only one series of 16 frames is needed to obtain the secret key without errors, while for an SNR of 15 dB it is required to wait for 6 repetitions of the secret key, i.e., 96 frames in total. As it can be seen from Fig. 26, for the lowest unfavorable SNR of 15 dB and the lowest α of 5% that results in minute and imperceptible preamble deviations, 20 repetitions are needed. Finally, since only 8 out 12 subcarriers are corrupted, the attacker can use the other 4 not corrupted subcarriers to tune a threshold amplitude value to demodulate the leaked data and further reduce the error rate.

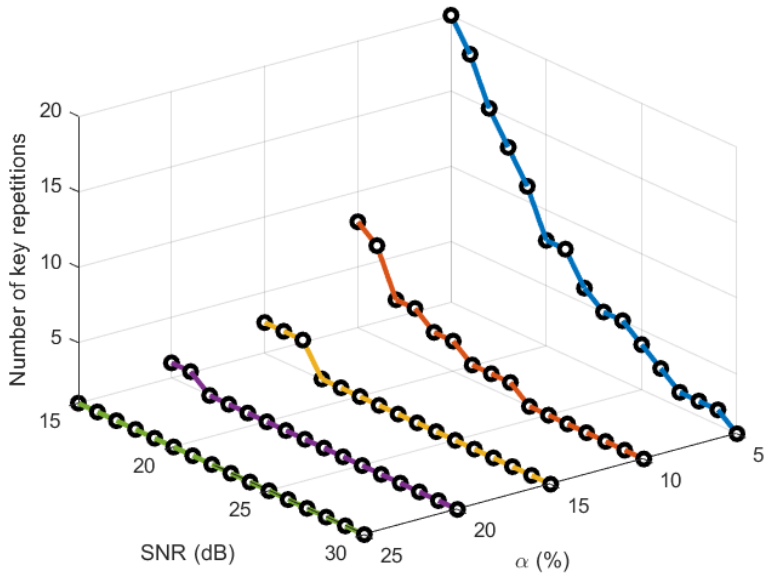


Figure 26: Reliability test for five different modulation amplitudes α under different SNR scenarios.

3.4 RELATED PREVENTION AND DETECTION DEFENSE MECHANISMS

In Section 3.3.3 we demonstrated experimentally that the proposed HT evades all known detection defenses at test time or during run-time. The covert channel is undetected when the nominal receiver is post-processing the HT-infected signal. There exist, however, several other generic HT countermeasures based on insertion prevention and pre-silicon or post-silicon detection that are in principle applicable under different threat model scenarios. The proposed HT relies on performing small malicious modifications in the digital section of the RF transceiver, i.e., baseband DSP and IP core from which the information is being leaked.

If the attack is staged by an EDA tool provider or if the digital section of the RF transceiver is a 3PIP core, then the owner of the RF transceiver can check for the presence of the HT prior to fabrication using: (a) functional verification of the 3PIP cores [128]; (b) structural analysis of HDL codes [128]; (c) logic testing tools [123, 129–133]; (d) specific simulation benches, i.e., performing aging simulations along with over-clocking [134] or short-term aging [135] to magnify the effect of the HT without triggering it; (e) search methods for unused components during design-time verification, which thereafter can be removed as potentially suspicious [136]; and (f) Information Flow Tracking (IFT) methods that track the propagation of sensitive data and verify that they do not reach unauthorized sites in the design [31, 137–141]. In our case, IFT could be used to spot the connection between the register in the IP core where the information is stored and the preamble generation block in the DSP.

If the attack is staged by the foundry, pre-silicon prevention methods include: (a) filling in all unused spaces on the layout, which are most likely insertion areas for the HT, with functional filler cells and checking if those have changed [142]; and (b) design obfuscation, for example using locking [66, 80, 109, 143, 144], camouflaging [62, 65, 145], or split manufacturing [60, 146], aiming at obscuring the circuit functionality so as to make it difficult for the attacker to insert the HT.

The test-based and run-time defenses discussed in Section 2.1.4 and tested in Section 3.3.3 are post-silicon HT detection methods. Other post-silicon HT detection methods include: (a) destructive reverse-engineering, which involves de-packaging and de-layering the chip, imaging the chip's layers, and using software to stitch together the prepared images, thereby recovering the layout and netlist, which thereafter can be carefully examined to detect the presence of HTs [7, 147, 148]; (b) non-destructive side-channel analysis to expose the HT location, for example using optical circuit analysis [149], electromagnetic emanation measurements [150–152], dynamic power measure-

ments [153], thermal map analysis [154], backscattering [155], and laser probing [156]; and (c) using on-chip monitors, i.e., current sensors [157], thermal sensors [158], voltage sensors [112], and invariance checkers [159], for run-time HT detection.

All the aforementioned defenses should be evaluated in the context of the proposed HT and can be the subject of future work.

3.5 CONCLUSION

This chapter has described a novel AM STS HT attack for leaking sensitive data out of wireless ICs. The AM STS HT attack acts on the preamble of a transmission frame, hiding the data into the transmission part that is used only for system synchronization, thereby not affecting the communication. The HT mechanism itself is hidden inside the dense digital baseband of the transmitter having a tiny footprint with 0.109% overheard. The proposed attack is stealthy being completely transparent to the normal RF transceiver operation. The leaked data can be recovered only by the intended rogue receiver using an inverse operation that is prohibitively high-cost and impractical to perform during run-time on every regular receiver. An indicative throughput of the established covert channel is 750 kbps. We demonstrated with hardware measurements using the SDR bladeRF board from Nuand™ that the proposed AM STS HT attack is capable of evading any previously reported defense either at run-time or based on performance testing. We also proposed a novel low-cost run-time correlation-based defense to detect HT activity hidden in the synchronization data. This defense also falls short in revealing the proposed AM STS HT attack.

Our experiments demonstrate the strength of the proposed AM STS HT attack calling for the development of a specific practical defense so as to ensure the security of wireless communications. To this end, we discussed several known generic HT countermeasures that could be potentially applicable and should be further evaluated. Finally, we demonstrated the AM STS HT attack from the attacker's perspective where an encrypted message with a 128-bit secret key is being leaked. We analyzed the reliability of the covert channel and we demonstrated that the key can be successfully recovered after less than 10 key transmissions even in the most unfavorable SNR scenario. The research results presented in this chapter were published in [55].

4

RF TRANSCEIVERS SECURITY AGAINST PIRACY ATTACKS THROUGH LOGIC LOCKING

Section 2.2 has analyzed the state-of-the-art (SoA) in anti-piracy design for ICs in the context of RF transceivers. This chapter demonstrates system-level locking for RF transceivers serving as an anti-piracy security technique. While all previous works show locking of individual Analog and Mixed-Signal (A/M-S) blocks (i.e. amplifier [69–71, 79], filter [69, 75], oscillator [69], phase-locked loop [67], data converter [107, 108]), herein we show system-level locking of a complete RF transceiver. The security of RF transceivers has been addressed only in [76], however, this solution applies only to highly-digitized programmable RF transceivers and consists in keeping the programming settings secret. Thus, there is no actual locking mechanism. Moreover, the assumption is that the calibration algorithm is unknown to the attacker. The locking strategy is to make RF performance key-dependent by leveraging a state-of-the-art logic locking technique, namely Stripped Functionality Logic Locking (SFLL)-rem [105] to obfuscate digital blocks in the signal path. The technique presents several advantages, including general applicability, effective locking for incorrect keys, attack resilience, transparency when the correct key is used, minimum overheads, and ease of implementation. Logic locking cannot be blindly applied in the context of RF transceivers, in this regard, this chapter show how it can be adapted towards effective RF transceiver locking. A proof-of-concept is demonstrated with hardware measurements.

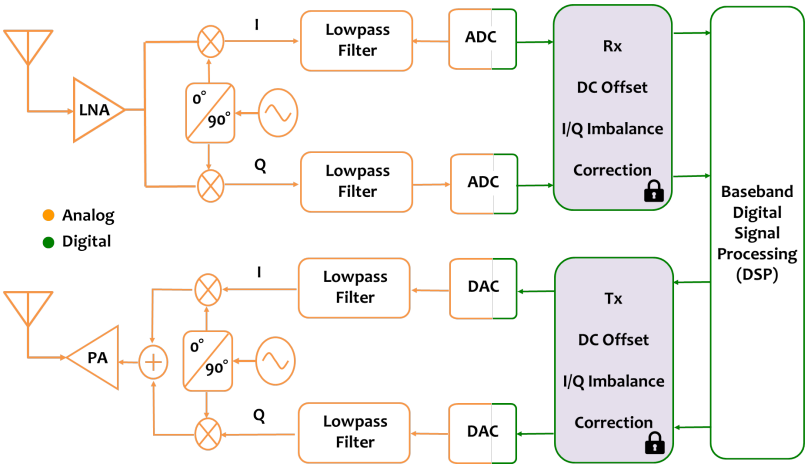
The rest of the chapter is structured as follows. Section 4.1 presents an overview of RF transceiver architectures and the locking methodology. Sections 4.2 and 4.3 describe the generic logic locking technique SFLL-rem and explain how to fine-tune it efficiently secure RF transceivers. The security analysis of the design methodology is evaluated in Section 4.4 Regarding the implementation results on the hardware platform, Section 4.5 shows the measured locking efficiency, resiliency to attacks, and overheads. Section 4.6 concludes the chapter.

4.1 RF TRANSCEIVER LOCKING

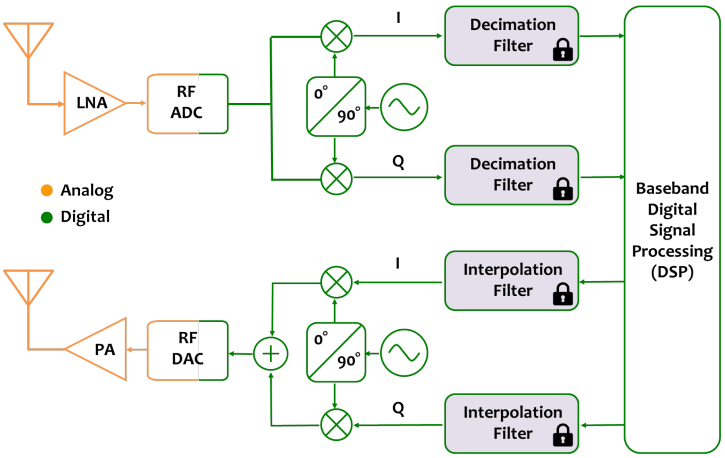
4.1.1 *Overview of RF transceiver architectures*

The most common architecture for integrated RF receivers is shown in Fig. 27a. It is used for both Zero Intermediate Frequency (Zero-IF) receivers and Low Intermediate Frequency (Low-IF) receivers [160]. In

Zero-IF receivers, the RF signal is directly down-converted to Direct Current (DC) using an analog down-conversion mixer. Zero-IF receivers are sensitive to DC Offset (DCO) and I/Q Imbalance (IQI). In Low-IF receivers, the analog mixer down-converts the RF signal to a low Intermediate Frequency (IF). In this case, the IF-signal is first converted to the digital domain by the Analog-to-Digital Converter (ADC), then it is down-converted to DC using a digital mixer. Since down-conversion to DC is performed in the digital domain, Low-IF receivers are not sensitive to DCO but only to IQI. Regarding the architectures for integrated RF transmitters, the most common is the Zero-IF transmitter shown in Fig. 27a, which is also sensitive to DCO-IQI. As illustrated in Fig. 27a, both the receiver and the transmitter require a digital block to correct for DCO and/or IQI.



(a) Conventional RF transceiver architecture. Locking can be performed in the DC Offset and I/Q Imbalance (DCO-IQI) correction circuit.



(b) Highly-digitized RF transceiver architecture. Locking can be performed in the decimation and interpolation filters.

Figure 27: Main RF transceiver architectures showing the most suitable digital block to lock: (a) conventional; (b) highly-digitized.

Recently, highly-digitized transceiver architectures have been proposed [161, 162], as shown in Fig. 27b. An RF ADC is used to directly convert the RF signal to the digital domain. The signal is then down-converted to DC by a digital mixer and filtered using a digital decimation filter. In the transmitter, the baseband signal passes through a digital interpolation filter before the digital up-conversion. An RF Digital-to-Analog Converter (DAC) is then used before the analog band-pass filter and the Power Amplifier (PA). The main advantage of this architecture is that down-conversion in the receiver and up-conversion in the transmitter are performed using digital mixers. The I and Q branches are in the perfectly matched digital domain, thus this architecture does not require any DCO or IQI correction. The main challenge is the design of high-speed ADC and DAC.

4.1.2 *Locking methodology*

The proposed locking strategy targets the interaction between analog and digital blocks in the RF transceiver. The underlying idea is to corrupt analog information propagation by performing logic locking in digital blocks in the signal path. In this way, system-level RF performances, e.g., Bit Error Rate (BER), are corrupted in a complex and unpredictable way.

Considering the conventional Zero-IF and Low-IF RF transceiver architectures in Fig. 27a, we can target locking the DCO-IQI correction blocks of both the transmitter and the receiver. Considering the highly-digitized RF transceiver architectures in Fig. 27b [162], we can target locking the digital decimation filter in the receiver and the digital interpolation filter in the transmitter. In this chapter, we demonstrate with hardware measurements the locked RF transceiver architecture in Fig. 27a.

The locking strategy presents the following advantages:

1. *General applicability*: It is applicable to any RF transceiver architecture and independent of the modulation scheme and constellation size.

2. *Locking effectiveness*: Only one key unlocks functionality while any incorrect key results in drastic performance degradation.

3. *Attack resilience*: It generates a large-size digital key, which is a prerequisite for achieving resiliency against counterattacks. It also borrows and capitalizes on the security properties of state-of-the-art logic locking mechanisms to provide strong security against counterattacks.

4. *Transparency*: There is no performance penalty since (a) analog blocks are left intact and (b) advanced logic locking techniques intentionally do not modify critical paths in the digital section,

thus the delay penalty if any is practically negligible having no effect on RF performance.

5. *Minimum overheads:* The small and justifiable area and power overheads for the digital blocks resulting from the locking operation become negligible when projected to the entire RF transceiver.

6. *Ease of implementation:* The A/M-S design flow does not change and no A/M-S block needs to be re-designed. The locking step can be seamlessly integrated into the digital design flow since logic locking is automated [66].

4.2 LOGIC LOCKING WITH SFLL-REM

We employ the state-of-the-art SFLL-rem logic locking technique which is fully supported by commercial EDA tools [105]. SFLL-rem will be fine-tuned in the context of RF transceiver locking to result in high BER corruption. Let us assume a circuit F with n inputs I and m outputs O implementing the Boolean function $O = F(I)$. Without loss of generality, we assume a single logic cone, i.e., $m = 1$. The objective is to insert a locking mechanism into the circuit that is controlled with a key K composed of k key-bits, $k \leqslant n$, thus transforming the circuit F to a circuit F_l with Boolean function $O = F_l(I, K)$. There is a single secret key k_{corr} that unlocks the circuit, i.e., $F(I) = F_l(I, K)|_{K=k_{corr}}, \forall I$. Any other key is incorrect and corrupts the output for some inputs, i.e., $\exists I\, F(I) \neq F_l(I, K)|_{K \neq k_{corr}}$. The steps of the SFLL-rem procedure are described below with the help of Fig. 28, which shows a toy application on the c17 circuit from the ISCAS benchmark suite [163] reproduced from [105]. The targeted logic cone from the c17 circuit is shown in Fig. 28a having $n = 5$ inputs $\{I1, I2, I3, I6, I7\}$ and $m = 1$ output O23.

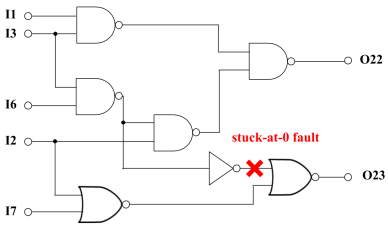
1. The first step is to perform a stuck-at fault injection campaign. A stuck-at fault means tying a net to a constant logical '0' (e.g., ground) or '1' (e.g., V_{DD}). For each injected fault we record the input test patterns that detect the fault, i.e., propagate the fault effect to the output resulting in a flipped output bit. These input test patterns are called *failing input test patterns* and their set is denoted by T_f. The fault injection campaign does not have to be exhaustive; it suffices to find a fault f that has a failing input test pattern with k care bits and $n - k$ don't care bits. This failing input test pattern is denoted by t_{secure} and, in essence, represents a total of 2^{n-k} failing input test patterns. These 2^{n-k} failing input test patterns are called Protected Input Patterns (PIPs). We select the k care bits of t_{secure} to be the secret key k_{corr}. In our example circuit, Fig. 28a shows the location of

such a fault and Fig. 28b lists the corresponding failing input test patterns that cause the circuit to fail on the output O23. All failing input test patterns have $k = 4$ care bits and $n - k = 1$ don't care bits. We arbitrarily select t_{secure} =x0111, which results in the secret key $k_{corr} = 0111$ whose bit positions map to the inputs {I2, I7, I3, I6}.

2. Due to the injection of fault f, the original Boolean function $F(I)$ is transformed to $F_f(I)$. Some internal nets now being tied high or low, allows us to remove logic and simplify the circuit F_f with regard to F. Compared to the original circuit F, F_f produces an erroneous output for the complete set T_f of failing input test patterns. For our example circuit, F_f is shown in Fig. 28c and the corresponding output is denoted by $O23^{I}$.

3. The next step consists in redesigning F_f by adding logic to restore the functionality for all failing input test patterns in the set $\{T_f - t_{secure}\}$, resulting in a circuit $F_{f'}$. Compared to the original circuit F, $F_{f'}$ produces an erroneous output only for the 2^{n-k} protected input patterns represented by t_{secure}. For our example circuit, $F_{f'}$ is shown in Fig. 28d and the corresponding output is denoted by $O23^{II}$.

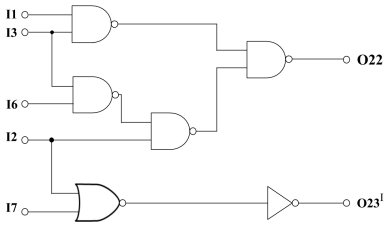
4. The final step consists in generating the target circuit F_l starting from $F_{f'}$. This is achieved by adding to $F_{f'}$ a restore unit and a 2-input XOR gate. Specifically, let $I^{'}$ be the concatenation of input bits whose positions map to the positions of the care bits of t_{secure} that compose the secret key k_{corr}. The restore unit implements a generic comparison function based on a look-up operation and compares $I^{'}$ to the key k_{corr}, which is stored in the TPM. The output of the restore unit is '1' when $I^{'} = k_{corr}$ and '0' when $I^{'} \neq k_{corr}$. The XOR gate is driven by the output O and the output of the restore unit, and the output of the XOR gate is the output of circuit F_l. In this way, we correct functionality for the remaining 2^{n-k} protected input patterns represented by t_{secure}. For our example circuit, F_l is shown in Fig. 28e and the corresponding output is denoted by $O23^{III}$.



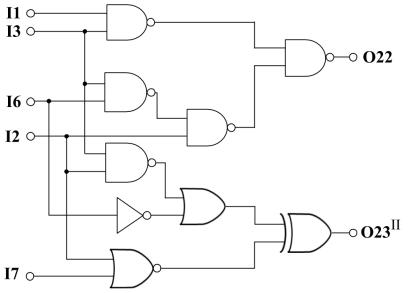
(a) Original circuit F.

Failing test patterns					(a)	(c)	(d)	(e)
I1	I2	I7	I3	I6	O23	$O23^{I}$	$O23^{II}$	$O23^{III}$
X	1	1	1	1	0	1	0	0
X	0	1	1	1	0	1	1	0
X	1	0	1	1	0	1	0	0

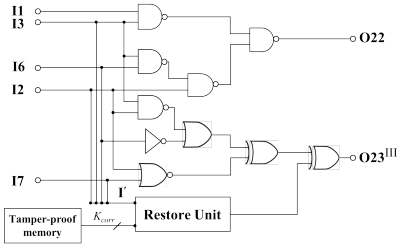
(b) Failing test patterns T_f for the stuck-at 0 fault of Fig. 28a highlighting the selected t_{secure}.



(c) Circuit F_f after stuck-at-0 fault injection.



(d) Circuit $F_{f'}$ after restoring logic in circuit F_f for all failing test patterns in Fig. 28b except those represented by t_{secure}.



(e) Circuit F_l after restoring logic in circuit $F_{f'}$ for the failing test patterns represented by t_{secure}.

Figure 28: The c17 circuit example from the ISCAS benchmark suite [105, 163].

4.3 SFLL-REM TUNED FOR RF TRANSCEIVER LOCKING

Herein, we show how SFLL-rem is tuned for RF transceiver locking considering the architecture in Fig. 27a. SFLL-rem is applied to the DCO-IQI correction blocks. The pseudo-algorithm is shown in Algorithm 1 where F is the DCO-IQI correction block and F_l is the DCO-IQI correction block with the lock embedded. The algorithm follows the steps of SFLL-rem detailed in Section 4.2 with the third step, i.e., the selection of t_{secure}, being the subject of the tuning. First, in the next paragraph, we explain why tuning is required.

Algorithm 1 SFLL-rem tuned for RF transceiver locking

Input: Original circuit netlist F
Output: Locked circuit netlist F_l

1: Perform stuck-at fault injection on F
2: Record set T_f of failing input test patterns
3: Select t_{secure} from T_f that is most frequently encountered during RF transceiver operation
4: Set key equal to the k care bits of t_{secure}
5: Generate F_f by removing redundant logic in F
6: Generate $F_{f'}$ by adding logic into F_f to restore functionality for all failing input test patterns in the set $\{T_f - t_{secure}\}$
7: Generate F_l by adding the restore unit and an XOR gate into $F_{f'}$

For digital ICs it suffices that logic locking corrupts one bit for a small set of PIPs. For example, to lock a microcontroller it suffices to lock one bit for one input in the program counter to safeguard against unauthorized execution [103]. However, for RF transceiver locking, to ensure an appreciable BER degradation, the transmitted/received data propagated to the input of the DCO-IQI correction block must frequently "hit" one of the PIPs. Thus, t_{secure} that represents the PIPs of the DCO-IQI correction block must be carefully selected specifically to the operation of the RF transceiver.

Let us consider first the DCO-IQI correction block of the receiver. Fig. 29 shows its final high-level block schematic modified by SFLL-rem. The circuit has a 96-bits input, i.e., $n = 96$, and a 32-bit output. We applied SFLL-rem aiming at locking two output bits $I_{out}[7]$ and $Q_{out}[7]$. Let us consider first $I_{out}[7]$. For the logic cone driving $I_{out}[7]$, in the first step of the SFLL-rem procedure, we injected a stuck-at-0 fault that resulted in a large number of failing input test patterns.

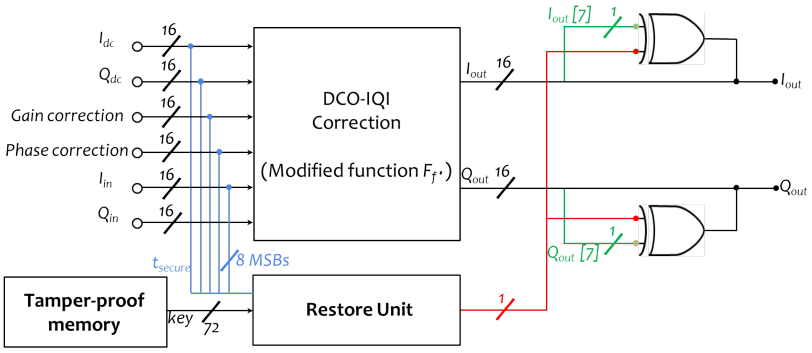


Figure 29: DC Offset and I/Q Imbalance (DCO-IQI) correction block with the locking mechanism.

To achieve a high security level against all foreseen counterattacks, as will be discussed in Section 4.4, we need to consider a key of large size k (typically k is chosen to be 64 or higher). At the same time, BER degradation requires first ensuring a high error rate expressed as the ratio of PIPs to all input patterns, i.e., $ER = 2^{n-k}/2^n = 2^{-k}$. Increasing k improves resiliency against attacks but reduces the error

rate. Furthermore, t_{secure} must be chosen to ensure that the 2^{n-k} PIPs frequently appear, thus resulting in BER degradation. A desired trade-off can be established by choosing appropriately k and t_{secure}.

In this regard, we consider further only those failing input test patterns having $k = 72$ care bits and $n - k = 24$ don't care bits with the don't care bits being the bits of Q_{in} and the 8 less significant bits (LSBs) of I_{in}. Table 6 lists a small subset of the failing input test patterns showing only the I_{in} segment. Among all failing input test patterns, we select t_{secure} to be the one that is most frequently encountered during the RF transceiver operation. To make this selection, we examine the histogram of I_{in} payload data, shown in Fig. 30. The histogram represents I_{in} in signed decimal values and shows their frequency of appearance. Similarly, the segments of I_{in} in Table 6 are represented with their signed decimal value range resulting from the don't care bits. Now, we can examine where the range of each failing input test pattern lies with respect to the peak of the histogram and select t_{secure} to be a failing input test pattern whose range is close to the peak. The selected t_{secure} is the failing input test pattern IV highlighted in red in Table 6, and its corresponding range is also depicted in Fig. 30. The key stored in the TPM is composed of the k=72 care bits of t_{secure}, as shown in Fig. 29.

Table 6: Subset of failing input test patterns for logic cone driving $I_{out}[7]$ showing only the I_{in} segment.

PATTERN	BINARY	SIGNED DECIMAL VALUE RANGE
I	1111 1101 xxxx xxxx	$[-640, -513]$
II	1111 1110 xxxx xxxx	$[-384, -257]$
III	11111111 xxxx xxxx	$[-128, -1]$
IV	0000 0000 xxxx xxxx	$[128, 255]$
V	0000 0001 xxxx xxxx	$[384, 511]$
VI	0000 0010 xxxx xxxx	$[640, 767]$

The next steps in SFLL-rem are to remove logic in the DCO-IQI correction block, resulting in version F_f of the circuit, and then add logic to restore the functionality at the $I_{out}[7]$ output for all failing input test patterns except t_{secure}, resulting in version $F_{f'}$. In the final step, the restore unit and a 2-input XOR gate are added to generate the target circuit F_l that restores functionality at the $I_{out}[7]$ output for t_{secure} when the correct key is applied, as shown in Fig. 29.

The procedure is repeated for the logic cone driving $Q_{out}[7]$ and we force the same t_{secure} to be part of the set of failing input test patterns and selected it. In this way, we have a single t_{secure} and, thereby, a single key locking both logic cones.

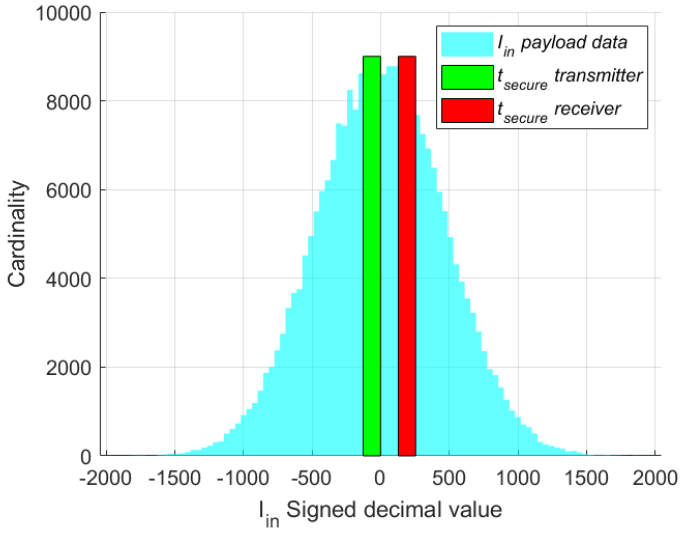


Figure 30: Histogram of I_{in} payload data during the RF transceiver operation.

The DCO-IQI correction block is the same for the receiver and the transmitter and the exact same procedure is followed for inserting the locking mechanism into the transmitter as well. However, among all failing input test patterns we selected a different t_{secure}, in particular the failing input test pattern III highlighted in green in Table 6 and in Fig. 30, such that the receiver and the transmitter have different secret keys.

4.4 SECURITY ANALYSIS

4.4.1 *Threat model*

We consider the threat model that is most favorable for an attacker. In particular, we assume that the attacker possesses the transistor-level netlist of the non-activated circuit and an unlocked functional chip which can be used as an oracle.

4.4.2 *RF transceiver performance corruption for incorrect keys*

The effectiveness of locking is assessed by the degradation of BER when an incorrect key k_{incorr} is used. This degradation is controlled by:

(a) The error rate (ER) of the locked digital block. We observe that any incorrect results in functionality corruption for the same PIPs. Thus, for a given transmission, the BER degradation is the same for all incorrect keys.

(b) The frequency with which the PIPs are encountered at the input of the locked digital block during normal operation of the RF transceiver. The higher the frequency is, the higher the BER degradation will be.

4.4.3 *Brute-force and optimization attacks*

The attacker may try to find the key by iterative simulation searching in the key space in a brute-force fashion or using optimization aiming at maximizing performance, i.e., minimizing BER. For a brute-force attack, the attack time is on average $2^k \cdot T/2$, where T is the run-time of a single simulation. The resiliency to this attack can be expressed as:

$$Res_{brute-force} = 2^k \cdot T, \tag{1}$$

This attack is impractical for a large key size k and considering also that T can be in the order of days.

For an optimization attack, the attack time is $m \cdot T$, where m is the number of iterations until convergence is achieved. As T is long for RF transceiver simulation, the attacker can afford running only a limited number of iterations. In this regard, the large key space, i.e., 2^{72} in our implementation, is a strong defense against these attacks. Moreover, since all incorrect keys result in the same degraded BER, the function BER$= g(K)$ relating BER with the key is a delta function and the search cannot be guided with optimization.

4.4.4 *I/O query and structural attacks*

The attacker may also attempt to break the defense by performing an attack on logic locking targeting solely the locked DCO-IQI correction blocks independently of the rest of the RF transceiver blocks. As mentioned in Section 2.2.4.1, main attacks are based on input-output query using the netlist and oracle to find the key or structural analysis that exploits the processing by logic synthesis tools to identify and remove the lock mechanism. In this case, the proposed RF transceiver locking strategy inherits the resiliency of the underlying logic locking technique. In [105], it is proven that with SFLL-rem the complexity of I/O query attacks, such as the SAT-attack [89], is equivalent to breaking a locked circuit with a k-bit key in brute-force, i.e., the resiliency is expressed in bits as:

$$Res_{SAT} = -\log_2 ER = k; \tag{2}$$

regarding structural attacks, SFLL-rem has recently shown vulnerability to a structural attack and a mitigation solution is proposed [102].

4.5 EXPERIMENTAL RESULTS

4.5.1 *Hardware platform*

We used the same hardware platform presented in Section 3.3.1. The RF transceiver of the hardware platform has a conventional Zero-IF architecture for both the receiver and the transmitter, shown in Fig. 27a. The DCO-IQI correction blocks are programmed inside the FPGA. The AFE has an on-chip loopback mode allowing us to perform BER measurements using the same board. This also greatly simplifies the channel model, allowing us to assume an Additive White Gaussian Noise (AWGN) channel model.

We implemented a wireless telecommunication protocol using for the payload an Orthogonal Frequency Division Multiplexing (OFDM) encoding with a 16-Quadrature Amplitude Modulation (QAM) scheme in each carrier, and the SDR is transmitting and receiving in the Industrial, Scientific and Medical (ISM) unlicensed band at 2.41 GHz.

As metric of performance, we consider the BER versus the energy per bit, E_b, to noise power spectral density ratio, N_0, i.e., E_b/N_0. We also present received constellation diagrams.

4.5.2 *Measured locking efficiency*

Fig. 31 shows the measured BER versus E_b/N_0 for five scenarios. The first scenario is the nominal design with no locking mechanism, while the other four scenarios correspond to the design with the locking mechanism embedded, where "unlocked" means that the correct key is applied and "locked" means that an incorrect key is used. As explained in Section 4.4, considering a given locking scenario with an incorrect key, the measured BER curve is exactly the same regardless which incorrect key is used. This theoretical implication is also verified experimentally by trying thousands of random incorrect keys. For this reason, in Fig. 31, we use the term "locked" to refer to applying an incorrect key in general.

The following observations can be made from Fig. 31:

1. Embedding the locking mechanisms into the DCO-IQI correction blocks has zero performance penalty since the BER curves of the RF transceiver without locking mechanism and the unlocked RF transceiver are identical.

2. Using an incorrect key for the receiver, transmitter or both, degrades BER and the degradation worsens with E_b/N_0. By locking both the receiver and the transmitter, BER is degraded by more than one order of magnitude for E_b/N_0 higher than 15 dB. Note that the goal is to achieve enough BER degradation to the

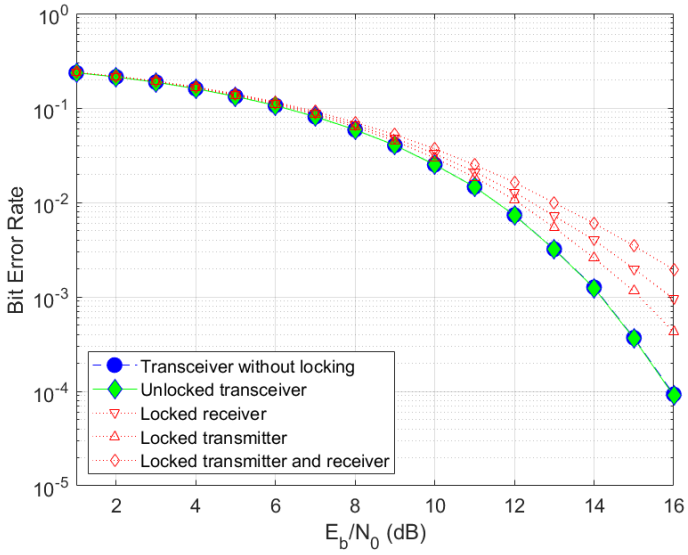


Figure 31: BER measurement results for different configurations.

point where the RF transceiver is deemed of unacceptable quality.

3. BER degradation is higher when locking only the receiver compared to locking only the transmitter. The reason is that the PIPs of the DCO-IQI correction block of the receiver are more frequently encountered in the communication compared to those of the DCO-IQI correction block of the transmitter. BER degradation is higher when both the receiver and transmitter are locked since now the set of PIPs becomes the union of the PIPs of the DCO-IQI correction blocks of the receiver and the transmitter.

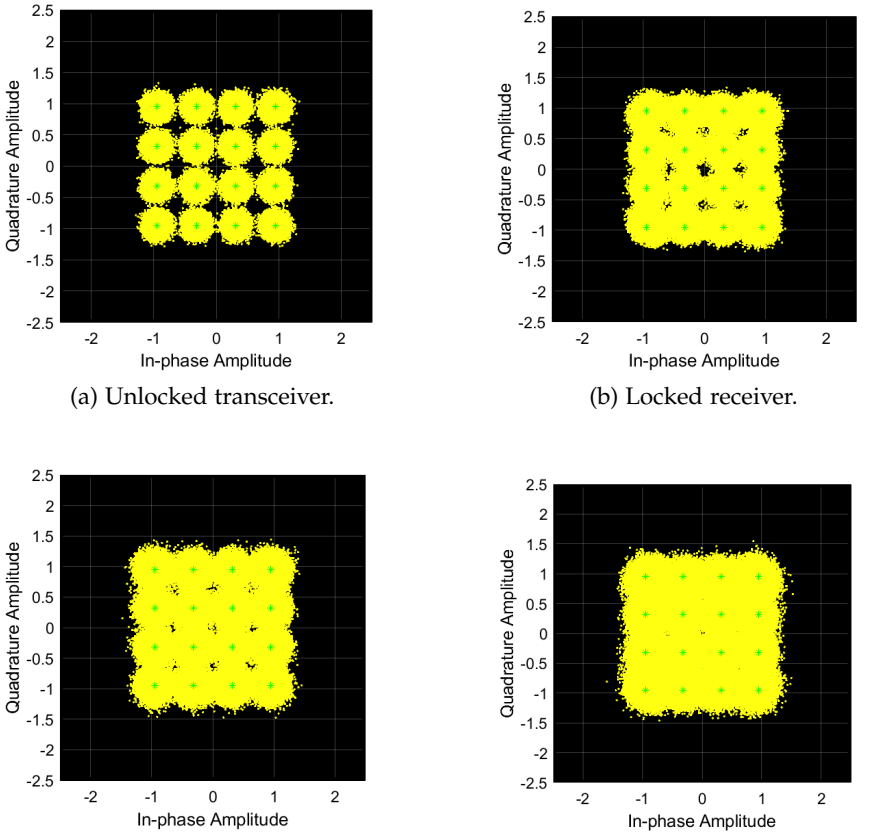
Fig. 32 shows the measured I/Q constellation diagram of the received signals for the last four scenarios. Locking causes constellation points to clearly deviate from their ideal locations compared to the unlocked transceiver.

4.5.3 *Resiliency to attacks*

With the selected t_{secure}, we have k=72 and $n-k$=24 and, thereby, we achieve strong security against all foreseen attacks described in Section 4.4. For example, we achieve a 72-bit resiliency against the most lethal SAT attack.

4.5.4 *Locking overheads*

Due to the locking operation, the area, power consumption, and delay of the DCO-IQI correction block are increased by 3.9%, 0.3%, and



(a) Unlocked transceiver.

(b) Locked receiver.

(c) Locked transmitter

(d) Locked receiver and transmitter.

Figure 32: I/Q constellation diagrams.

0.8%, respectively. As it can be seen from Fig. 31, there is no BER performance penalty implying that the small delay penalty is fully absorbed. Moreover, considering a fully integrated implementation of the RF transceiver, the DCO-IQI correction block is a small block, thus these small area and power overheads become negligible when projected to the entire RF transceiver. Therefore, we can claim a near zero area and power overhead due to locking.

4.6 CONCLUSION

We demonstrated for the first time RF transceiver locking against piracy. The methodology is based on logic locking of digital blocks in the signal processing chain. We employed the state-of-the-art SFLL-rem logic locking technique and we adapted it for effective RF transceiver locking. The methodology is virtually applicable to any RF transceiver architecture and inherits the security properties of logic locking. Hardware experiments demonstrated strong BER degradation for incorrect keys, while achieving zero performance penalty when applying the single correct secret key, and negligible

area and power overheads. The methodology can be seamlessly integrated into the RF transceiver design flow since its analog section is left intact and logic locking is fully automated. The research results presented in this chapter were published in [120].

5

RF TRANSCEIVER SECURITY USING SYNCHRONIZATION LOCKING

For an RF transceiver, being an Analog and Mixed-Signal (A/M-S) design, one can leverage existing techniques for locking blocks in its analog section; for locking part of its digital section, e.g., in Chapter 4 a generic logic locking technique was fine-tuned for protecting RF transceivers; or for locking it at system-level, i.e., by exploiting its programmability features. These generic techniques, however, have shown to be vulnerable to attacks, as it is described in more detail in Section 2.2. In this chapter we propose a domain-specific logic locking technique that takes advantage of a specific digital signal processing path found in RF transceiver. The solution is called *SyncLock* since it locks the synchronization of the transmitter with the receiver. If a key other than the secret key is applied, synchronization and, thereby, communication fails. *SyncLock* is implemented using a novel locking concept consisting of two spatially separated mechanisms. A hard-coded error is hidden in the design to break synchronization while error correction, i.e., unlocking, takes place in another part of the design by applying the secret key. As it will be discussed in Section 5.5, this signal processing path has a fixed built-in input and part of the *SyncLock* mechanism is spatially separated from the input. These two properties render all known counter-attacks on logic locking, discussed in Section 2.2, non-applicable.

The rest of the chapter is structured as follows. In Section 5.1, we present *SyncLock* in detail, including the principle of operation, preamble generation, locking mechanism, specifics of the hardware implementation, key space analysis, incurred overheads, and practicality. Section 5.1 concludes by presenting the first *SyncLock* implementation in [121] as a sub-case of the new *SyncLock* implementation in [122] and comparing the two. For the purpose of completeness, Section 5.4 discusses related locking and obfuscation approaches and their differences with *SyncLock*. Section 5.5 provides the threat model and analyzes the resilience of *SyncLock* to all known counter-attacks. In Section 5.2, we present the hardware platform used for demonstrating *SyncLock*. In Section 5.3, we demonstrate the locking efficiency of *SyncLock* with hardware measurements. Section 5.6 concludes this chapter.

5.1 SYNCHRONIZATION LOCKING

5.1.1 *Principle of operation*

SyncLock is a security mechanism for preventing piracy of RF transceivers. The underlying idea is to lock the preamble that allows the synchronization process between the transmitter and the receiver. By blocking the synchronization, wireless receivers are unable to find the start of the received frame, thus the wireless communication fails.

The architecture of a wireless IC with *SyncLock* embedded is shown in Fig. 33. *SyncLock* acts on two different parts of the design. First, it modifies the frame generation block at the end of the baseband Digital Signal Processing (DSP) chain of the transmitter by corrupting the preamble of each transmitted frame. The introduced error is hard-coded such that after logic synthesis of the DSP it is impossible to be traced and recovered by netlist analysis. Then, it modifies the preamble generation block at the beginning of the baseband DSP chain such that the output preamble is key-controlled. To enable the synchronization process, the key must neutralize the, unknown to the attacker, later corruption in the frame generation block.

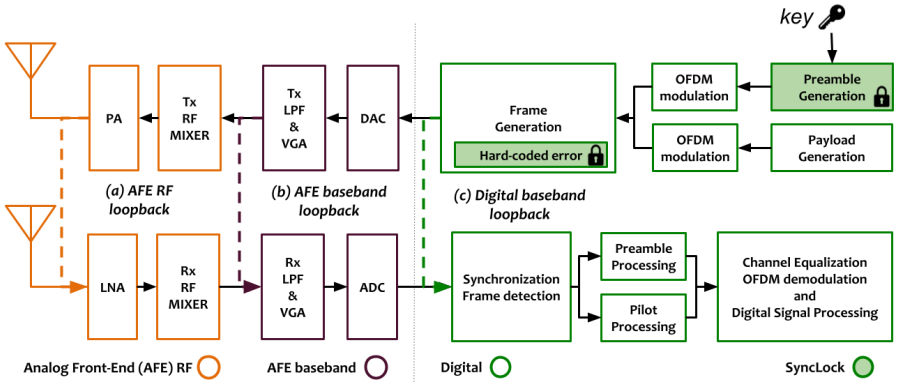


Figure 33: Simplified architecture of a wireless device IC with *SyncLock* embedded.

5.1.2 *Preamble generation*

In all wireless communication protocols, the payload is transmitted along with the Physical Layer (PHY) specifications. The baseband DSP prepares the payload in a frame format for transmission. The Physical layer Protocol Data Unit (PPDU) frame format of an Orthogonal Frequency Division Multiplexing (OFDM) IEEE 802.11 transmission consists of several OFDM symbols. These symbols are divided into three parts: preamble (a.k.a SYNC), header (a.k.a SIGNAL), and payload (a.k.a DATA). The preamble section is composed of two different training symbol sequences, namely a Short Training Sequence (STS)

and a Long Training Sequence (LTS). Fig. 9 shows the PPDU of an IEEE 802.11 transmission with the above three parts as defined in the IEEE 802.11 standard [124]. The STS field consists of 10 identical short symbol repetitions and is used for timing acquisition based on the Schmidl and Cox algorithm [125], i.e., for synchronization or start of frame detection and for rough Carrier Frequency Offset (CFO) estimation. The LTS field consists of 2 long symbol repetitions and is used for channel estimation and fine CFO estimation [124].

More specifically, as defined in the IEEE 802.11 standard [124], the nominal STS is divided into two parts, denoted here by $STS_{nom}I$ and $STS_{nom}Q$, corresponding to the real I and imaginary Q channels, respectively. $STS_{nom}I, Q$ is composed of 10 repetitions of the 16 samples of 16 bits each shown in Table 7 in floating-point and fixed-point hexadecimal representations. Thus, $STS_{nom}I, Q$ is composed of $10 * 16 * 16 = 2560$ bits in total.

Table 7: STS_{nom} as defined in the IEEE 802.11 standard [124].

Sample (k)	Floating-point (I,Q)	Fixed-point (I,Q)
0	0.04600 , 0.04600	16'h02F2 , 16'h02F2
1	−0.13245 , 0.00234	16'hF786 , 16'h0026
2	−0.01347 , −0.07853	16'hFF23 , 16'hFAF9
3	0.14276 , −0.01265	16'h0923 , 16'hFF31
4	0.09200 , 0.00000	16'h05E3 , 16'h0000
5	0.14276 , −0.01265	16'h0923 , 16'hFF31
6	−0.01347 , −0.07853	16'hFF23 , 16'hFAF9
7	−0.13245 , 0.00234	16'hF786 , 16'h0026
8	0.04600 , 0.04600	16'h02F2 , 16'h02F2
9	0.00234 , −0.13245	16'h0026 , 16'hF786
10	−0.07853 , −0.01347	16'hFAF9 , 16'hFF23
11	−0.01265 , 0.14276	16'hFF31 , 16'h0923
12	0.00000 , 0.09200	16'h0000 , 16'h05E3
13	−0.01265 , 0.14276	16'hFF31 , 16'h0923
14	−0.07853 , −0.01347	16'hFAF9 , 16'hFF23
15	0.00234 , −0.13245	16'h0026 , 16'hF786

Each sample of $STS_{nom}I, Q$ is generated in the baseband DSP by the preamble generation block shown in Fig. 34. There are in total 13 multiplexers (MUXes) per I/Q branch where the i-th MUX receives a constant 16-bit input DATA_i with values shown in Table 8. The SEL input of the MUXes is a 4-bit word and selects the creation of one of the 16 samples of the sequence. The position of the selected bit of DATA_i that is transferred at the output of each MUX equals the decimal representation of the SEL input. The 16-bit fixed-point I and Q values of the sample are then created by concatenating the outputs of the MUXes according to the schemes shown in the second and fifth rows of Table 9 for the I and Q branches, respectively. The

same hardware and concatenation operations are used to generate any sample k by setting the input SEL equal to k in decimal.

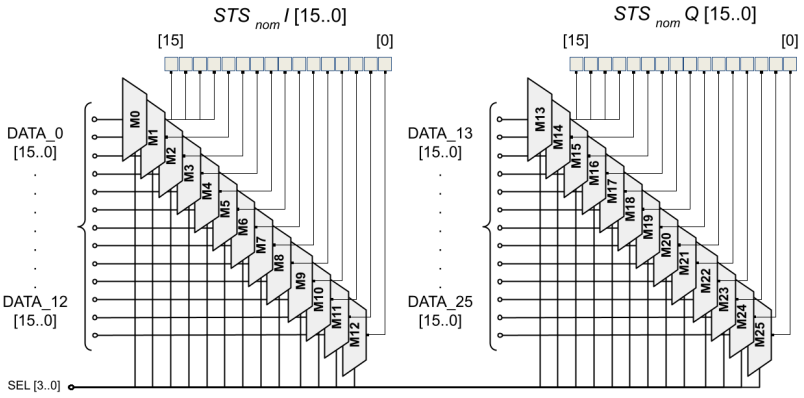


Figure 34: Original preamble generation block with no locking.

Table 8: Input values of MUXes in the preamble generation block.

MUX	Name	Input (16-bit)
	I branch	
M0	DATA_0	16'b0110 1100 1100 0110
M1	DATA_1	16'b0110 1100 0110 1100
M2	DATA_2	16'b0010 1000 1101 0110
M3	DATA_3	16'b0110 1101 1100 0111
M4	DATA_4	16'b0010 1000 1111 1110
M5	DATA_5	16'b0100 0101 1001 0011
M6	DATA_6	16'b0100 0101 0001 0001
M7	DATA_7	16'b1110 1111 0111 1101
M8	DATA_8	16'b0110 1101 0000 0001
M9	DATA_9	16'b0100 0100 0000 0000
M10	DATA_10	16'b1000 0010 1000 0010
M11	DATA_11	16'b1000 0011 1111 1111
M12	DATA_12	16'b0110 1100 0111 1100
	Q branch	
M13	DATA_13	16'b1100 0110 0110 1100
M14	DATA_14	16'b0110 1100 0110 1100
M15	DATA_15	16'b1101 0110 0010 1000
M16	DATA_16	16'b1100 0111 0110 1101
M17	DATA_17	16'b1111 1110 0010 1000
M18	DATA_18	16'b1001 0011 0100 0101
M19	DATA_19	16'b0001 0001 0100 0101
M20	DATA_20	16'b0111 1101 1110 1111
M21	DATA_21	16'b0000 0001 0110 1101
M22	DATA_22	16'b0000 0000 0100 0100
M23	DATA_23	16'b1000 0010 1000 0010
M24	DATA_24	16'b1111 1111 1000 0011
M25	DATA_25	16'b0111 1100 0110 1100

Table 9: Concatenation operation at the outputs of the MUXes.

		I branch			
	Concatenation of MUXes (M#)	M0,M0,M0,M0	M1,M2,M3,M4	M5,M6,M7,M8	M9,M10,M11,M12
SEL = 4'b0000	Fixed-point binary value	0000	0010	1111	0010
(first sample)	Fixed-point hexadecimal value	0	2	F	2
		Q branch			
	Concatenation of MUXes (M#)	M13,M13,M13,M13	M14,M15,M16,M17	M18,M19,M20,M21	M22,M23,M24,M25
SEL = 4'b0011	Fixed-point binary value	1111	1111	0011	0001
(fourth sample)	Fixed-point hexadecimal value	F	F	3	1

For example, let us consider the first sample, i.e., $k = 0$, in the I branch which has a fixed-point value of 16'h02F2 in hexadecimal representation. In this case, SEL = 4'b0000 selecting the first bit position of the DATA_i inputs of the MUXes, as shown in blue in the I branch part of Table 8. The concatenation of the MUXes output is shown in blue in the third row of Table 9 resulting in the desired value of 16'h02F2. As a second example, let us consider the fourth sample, i.e., $k = 3$, in the Q branch with a fixed-point value of 16'hFF31 in hexadecimal representation. SEL = 4'b0011 selecting the fourth bit position of the DATA_i inputs of the MUXes, as shown in red in the Q branch part of Table 8. The concatenation of the MUXes output is shown in red in the sixth row of Table 9 resulting in the desired value of 16'hFF31.

5.1.3 *Locking mechanism*

SyncLock acts specifically on the generation of the STS. The locking mechanism of *SyncLock* is divided into two parts embedded into the preamble and frame generation blocks, as shown in Fig. 35. In the frame generation block, the STS originally generated by the preamble generation block is embedded into the frame for transmission, with the final STS denoted by STS_{out}. The design owner deliberately corrupts the incoming STS to the frame generation block prior to frame creation by XORing it with the output of a nonlinear module $f(\cdot)$. This module implements a feedback loop involving STS_{out} and a hard-coded key, denoted by key_{h-c}.

In the preamble generation block, an XOR operation is performed between the key-bits stored in the Tamper-Proof Memory (TPM) and STS_{nom}, thus corrupting STS_{nom} to a faulty value, denoted by STS_{faulty}. The equations describing the operations are

$$STS_{faulty} = STS_{nom} \oplus key \tag{3}$$

$$STS_{out} = STS_{faulty} \oplus f(STS_{out}, key_{h-c}) \tag{4}$$

Combining Eqs. (3)-(4) and using the associative property $(A \oplus B) \oplus C = A \oplus (B \oplus C)$ of the XOR function, the system equation becomes

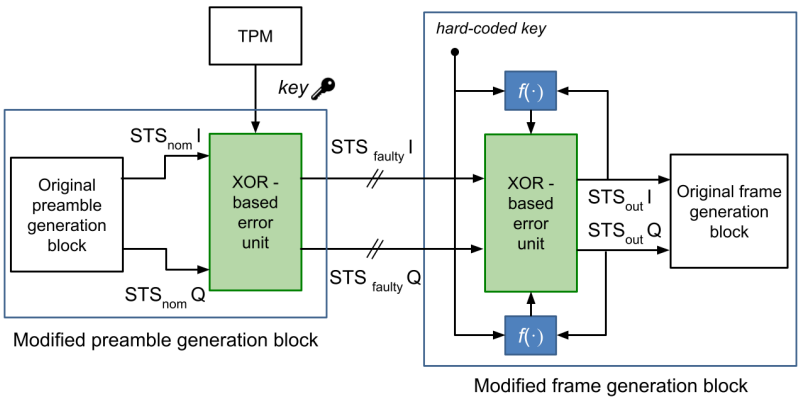


Figure 35: Preamble and frame generation blocks modified for *SyncLock*.

$$\mathrm{STS}_{\mathrm{out}} = \mathrm{STS}_{\mathrm{nom}} \oplus (\mathrm{key} \oplus \mathrm{f}(\mathrm{STS}_{\mathrm{out}}, \mathrm{key}_{\mathrm{h-c}})) \tag{5}$$

Thus, using the self-inverse property $A \oplus A = 0$ of the XOR function, $\mathrm{STS}_{\mathrm{out}} = \mathrm{STS}_{\mathrm{nom}}$ if and only if $\mathrm{key} = \mathrm{f}(\mathrm{STS}_{\mathrm{nom}}, \mathrm{key}_{\mathrm{h-c}})$

$$\mathrm{STS}_{\mathrm{out}} = \mathrm{STS}_{\mathrm{nom}} \iff \mathrm{key} = \mathrm{f}(\mathrm{STS}_{\mathrm{nom}}, \mathrm{key}_{\mathrm{h-c}}) \tag{6}$$

The *SyncLock* mechanism can be viewed as two spatially separated XOR-based stream ciphers controlled by two secret keys, one stored in the TPM and the other one being hard-coded. The generated $\mathrm{STS}_{\mathrm{nom}}$ by the original preamble generation block, i.e., the plaintext, is encrypted to $\mathrm{STS}_{\mathrm{faulty}}$, i.e., the ciphertext, so as to "match" the hidden hard-coded decryption that comes downstream in the DSP chain at the frame generation block. The secret key must be loaded in the TPM of the chip for correct deciphering that restores the synchronization. Applying incorrect keys introduces two uncorrelated STS corruptions at two distinct blocks of the DSP chain which breaks the synchronization.

As mentioned in Section 5.1.2, for each channel I or Q, at any point in the signal processing chain, STS is composed of 2560 bits and is processed in 160 blocks with each block corresponding to one sample of 16 bits shown in Table 7. Each key is composed of 512 bits divided into two parts of 256 bits for each channel. Thus, for each channel, a key is divided into 16 blocks of 16 bits each. This means that for each channel the key is repeatedly applied 10 times for every 16 blocks of STS.

The implementation specifics showing how $\mathrm{STS}_{\mathrm{out}}$ converges in Eq. (4) will be described in detail in Section 5.1.6.

5.1.4 *Choice of function* $f(\cdot)$

As will be explained in detail in Section 5.5.2-5, the function $f(\cdot)$ is introduced to circumvent the Known-Plaintext Attack (KPA), which is a vulnerability of the preliminary version of *SyncLock* in [121]. The choice of $f(\cdot)$ is free, leaving in theory unlimited freedom to the defender. It can also change from one design to another or across design iterations to update the key for increased security.

In our current implementation, $f(\cdot)$ is a two-step function. It first performs 16-bit parallel XORing of STS_{out} with the hard-coded key key_{h-c}, then it applies to the result a circular shift operation, a.k.a. bitwise rotation, i.e., $f(STS_{out}, key_{h-c}) = (STS_{out} \oplus key_{h-c}) >> b$, where $>>$ is the bitwise rotation operation and b is the number of bit rotations to the right. Other possibilities include bitwise logical operations between STS_{out} and key_{h-c}, bit scrambling or substitution after the XORing between STS_{out} and key_{h-c}, etc.

The function $f(\cdot)$ can be executed in a single clock cycle using any of the above bitwise operations with depth equal to one. For example, in our current implementation, the XOR operation $STS_{out} \oplus key_{h-c}$ needs one clock cycle, while the rotation can be simply implemented by rotating the wiring of the 16-bit output of $STS_{out} \oplus key_{h-c}$ when it is fed into the XOR function with STS_{faulty}. To accommodate this one clock cycle delay and guarantee convergence, as we will see in Section 5.1.6 that presents the implementation specifics, we let the first 16-bit block of STS pass without being processed by the XOR stream ciphers, whereas the XOR steam ciphers come into play starting from the second 16-bit block of STS. Essentially, from this point onward, the k-th block of STS_{faulty} is XORed with the output of $f(\cdot)$ that has been computed with the $(k-1)$-th blocks of STS_{out} and key_{h-c} to produce the k-th block of STS_{out}.

In general, if $f(\cdot)$ had a larger depth needing n clock cycles to be executed, then the XOR stream ciphers would come into play starting from the n-th 16-bit block of STS.

5.1.5 *Impact on performance*

The *SyncLock* mechanism has no impact on the performance of the RF transceiver. The delay of $f(\cdot)$ is accommodated by just enabling the XOR stream ciphers with the same delay. Thus, the STS generation is not delayed because of $f(\cdot)$. However, the two XOR stream ciphers introduce a delay of two clock cycles in the STS generation. This results in no timing violation because the preamble, composed of STS and LTS, and the payload are generated in parallel to compose the data frame, while the payload part is much longer than the preamble part. In other words, the preamble generation, despite being delayed, still finishes before the payload is generated.

5.1.6 *Implementation specifics*

Without loss of generality, let us consider the I channel. Let $s_I[k]$ denote the 16-bit k-th sample in Table 7, $k = 0, \cdots, 15$, e.g., $s_I[0] = 16'h02F2$, $s_I[1] = 16'hF786$, etc. Let also $STS_{faulty}I$ and $STS_{out}I$ denote the real parts of STS_{faulty} and STS_{out}, respectively, each composed of 2560 bits similarly to $STS_{nom}I$. As explained Section 5.1.2, by construction, the different STS values are divided into words of 16-bits corresponding to samples $s_I[k]$. Starting with $STS_{nom}I$, it is divided into 16-bit words $STS_{nom}I[j]$ corresponding to bit positions from $j * 16$ to $(j * 16) + 15$, $j = 0, \cdots, 159$. $STS_{nom}I[j]$ can be expressed in terms of $s_I[k]$ as

$$STS_{nom}I[j] = s_I[mod(j, 16)] \tag{7}$$

The key and hard-coded key are composed of 512-bits each and are reused in every repetition of the 16 samples. Each key can be divided into two equal 256-bit parts, with the first part corresponding to the I channel and the second part to the Q channel. For the I channel, the key and hard-coded key are denoted by $keyI$ and $key_{h-c}I$, respectively. Similar to STS values, each key is divided into 16-bit words corresponding to samples $s_I[k]$. For example, $keyI$ results from the concatenation $keyI = keyI[0]...keyI[15]$, where $keyI[n]$ is the part of $keyI$ in bit positions from $n * 16$ to $(n * 16) + 15$, $n = 0, \cdots, 15$.

Using the above definitions, we can now formally explain the *SyncLock* implementation. Since the nonlinear module evaluates STS_{out} in a feedback loop, the system essentially incorporates an internal memory and a valid STS_{out} would become available starting from the second sample of the first repetition. To remove this delay, we use the following technique. For the first sample of the first repetition, both XOR-based error units in the preamble and frame generation blocks are bypassed, i.e., $STS_{faulty}I[0] = STS_{nom}[0]$ and $STS_{out}I[0] = STS_{faulty}I[0]$, that is, we force the initial condition

$$STS_{out}I[0] = STS_{nom}I[0] \tag{8}$$

From the second sample of the first repetition onward, the key and the two XOR-based error units start intervening in the computation. Specifically, using the above definitions, for $j \geqslant 1$ we have

$$STS_{faulty}I[j] = STS_{nom}I[j] \oplus keyI[mod(j, 16)], \quad j = 1, \cdots, 159 \tag{9}$$

and

$$STS_{out}I[j] = STS_{faulty}I[j] \oplus f(STS_{out}I[j-1], key_{h-c}I[mod(j,16)]),$$
$$j = 1, \cdots, 159. \quad (10)$$

Substituting Eq. (9) into Eq. (10) we have

$$STS_{out}I[j] = STS_{nom}I[j] \oplus (keyI[mod(j,16)] \oplus$$
$$f(STS_{out}I[j-1], key_{h-c}I[mod(j,16)])),$$
$$j = 1, \cdots, 159. \quad (11)$$

The hard-coded key is set arbitrarily by the designer. The key is then selected such that $STS_{out}I = STS_{nom}I$ which from Eq. (11) is satisfied by the identity

$$keyI[mod(j,16)]) = f(STS_{nom}I[j-1], key_{h-c}I[mod(j,16)])$$
$$j = 1, \cdots, 159. \quad (12)$$

This results in

$$keyI[0] = f(STS_{nom}I[15], key_{h-c}I[0]) \quad (13)$$
$$keyI[n] = f(STS_{nom}I[n-1], key_{h-c}I[n]), \quad (14)$$
$$n = 1, \cdots, 15.$$

An excerpt of the computations for the first 19 real samples $j = 0, \cdots, 18$ of $STS_{nom}I$, i.e., comprising a complete first iteration and 3 samples in the second iteration, is shown in Table 10 for three different key cases, namely (a) incorrect zero key; (b) random incorrect key; and (c) correct key. Bitwise rotation with $b = 1$ is used as the nonlinear function. The key is repeated every 16 samples, but the XOR operations are bypassed for the first sample of the first iteration so as to force the initial condition for the feedback loop. The locking mechanism becomes active starting from the second sample of the first iteration and stays active until the end of the STS_{nom} transmission to the frame generation block. As it can be seen, STS_{out} is generated without errors only for the correct key.

5.1.7 *Key size*

The above *SyncLock* implementation has the advantageous property that there is a single correct key enabling synchronization, while any other key results in no synchronization, i.e., there are no approximate keys. This property stems from the nonlinear module inside the frame generation block. More specifically, considering for example a

Table 10: Values of the main signals of the *SyncLock* locking mechanism for three key cases: incorrect zero key, random incorrect key, and correct key. The example considers a bit rotation function with $b = 1$ and shows the computations during the transmission of the first 19 samples of STS_{nom} for the I channel.

(a) Incorrect zero key

	Preamble generation block			Frame generation block		
[j]	$STS_{nom}I[j]$	$keyI[mod(j,16)]$	$STS_{faulty}I[j]$	$key_{h-c}I[mod(j,16)]$	$f(STS_{out}I[j-1], key_{h-c}I[mod(j,16)])$	$STS_{out}I[j]$
0	16'h02F2	16'h0000 (bypassed)	16'h02F2	16'h0052 (bypassed)	(bypassed)	16'h02F2
1	16'hF786	16'h0000	16'hF786	16'hFAFA	(16'h02F2 ⊕ 16'hFAFA) >> b = 16'h7C04	16'h8B82
2	16'hFF23	16'h0000	16'hFF23	16'hFEA6	(16'h8B82 ⊕ 16'hFEA6) >> b = 16'h3A92	16'hC5B1
3	16'h0923	16'h0000	16'h0923	16'h5216	(16'hC5B1 ⊕ 16'h5216) >> b = 16'hCBD3	16'hC2F0
4	16'h05E3	16'h0000	16'h05E3	16'h5614	(16'hC2F0 ⊕ 16'h5614) >> b = 16'h4A72	16'h4F91
5	16'h0923	16'h0000	16'h0923	16'hCAFE	(16'h4F91 ⊕ 16'hCAFE) >> b = 16'hC2B7	16'hCB94
6	16'hFF23	16'h0000	16'hFF23	16'hFEF8	(16'hCB94 ⊕ 16'hFEF8) >> b = 16'h1AB6	16'hE595
7	16'hF786	16'h0000	16'hF786	16'h4516	(16'hE595 ⊕ 16'h4516) >> b = 16'hD041	16'h27C7
8	16'h02F2	16'h0000	16'h02F2	16'h0158	(16'h27C7 ⊕ 16'h0158) >> b = 16'h934F	16'h91BD
9	16'h0026	16'h0000	16'h0026	16'hCAFE	(16'h91BD ⊕ 16'hCAFE) >> b = 16'hADA1	16'hAD87
10	16'hFAF9	16'h0000	16'hFAF9	16'hFFAC	(16'hAD87 ⊕ 16'hFFAC) >> b = 16'hA915	16'h53EC
11	16'hFF31	16'h0000	16'hFF31	16'hAAAA	(16'h53EC ⊕ 16'hAAAA) >> b = 16'h7CA3	16'h8392
12	16'h0000	16'h0000	16'h0000	16'h003A	(16'h8392 ⊕ 16'h003A) >> b = 16'h41D4	16'h41D4
13	16'hFF31	16'h0000	16'hFF31	16'h0569	(16'h41D4 ⊕ 16'h0569) >> b = 16'hA25E	16'h5D6F
14	16'hFAF9	16'h0000	16'hFAF9	16'hFFC2	(16'h5D6F ⊕ 16'hFFC2) >> b = 16'hD156	16'h2BAF
15	16'h0026	16'h0000	16'h0026	16'h9623	(16'h2BAF ⊕ 16'h9623) >> b = 16'h5EC6	16'h5EE0
16	16'h02F2	16'h0000	16'h02F2	16'h0052	(16'h5EE0 ⊕ 16'h0052) >> b = 16'h2F59	16'h2DAB
17	16'hF786	16'h0000	16'hF786	16'hFAFA	(16'h2DAB ⊕ 16'hFAFA) >> b = 16'hEBA8	16'h1C2E
18	16'hFF23	16'h0000	16'hFF23	16'hFEA6	(16'h1C2E ⊕ 16'hFEA6) >> b = 16'h7144	16'h8E67

(b) Random incorrect key

	Preamble generation block			Frame generation block		
[j]	$STS_{nom}I[j]$	$keyI[mod(j,16)]$	$STS_{faulty}I[j]$	$key_{h-c}I[mod(j,16)]$	$f(STS_{out}I[j-1], key_{h-c}I[mod(j,16)])$	$STS_{out}I[j]$
0	16'h02F2	16'h2324 (bypassed)	16'h02F2	16'h0052 (bypassed)	(bypassed)	16'h02F2
1	16'hF786	16'hCAFE	16'h3D78	16'hFAFA	(16'h02F2 ⊕ 16'hFAFA) >> b = 16'h7C04	16'h417C
2	16'hFF23	16'h5249	16'hAD6A	16'hFEA6	(16'h417C ⊕ 16'hFEA6) >> b = 16'h5FED	16'hF287
3	16'h0923	16'h3216	16'h3B35	16'h5216	(16'hF287 ⊕ 16'h5216) >> b = 16'hD048	16'hEB7D
4	16'h05E3	16'hEFAC	16'hEA4F	16'h5614	(16'hEB7D ⊕ 16'h5614) >> b = 16'hDEB4	16'h34FB
5	16'h0923	16'h1234	16'h1B17	16'hCAFE	(16'h34FB ⊕ 16'hCAFE) >> b = 16'hFF02	16'hE415
6	16'hFF23	16'hAFC5	16'h50E6	16'hFEF8	(16'hE415 ⊕ 16'hFEF8) >> b = 16'h8D76	16'hDD90
7	16'hF786	16'hDE18	16'h299E	16'h4516	(16'hDD90 ⊕ 16'h4516) >> b = 16'h4C43	16'h65DD
8	16'h02F2	16'h0090	16'h0262	16'h0158	(16'h65DD ⊕ 16'h0158) >> b = 16'hB242	16'hB020
9	16'h0026	16'hFE10	16'hFE36	16'hCAFE	(16'hB020 ⊕ 16'hCAFE) >> b = 16'h3D6F	16'hC359
10	16'hFAF9	16'h3620	16'hCCD9	16'hFFAC	(16'hC359 ⊕ 16'hFFAC) >> b = 16'h9E7A	16'h52A3
11	16'hFF31	16'h5148	16'hAE79	16'hAAAA	(16'h52A3 ⊕ 16'hAAAA) >> b = 16'hFC04	16'h527D
12	16'h0000	16'h6696	16'h6696	16'h003A	(16'h527D ⊕ 16'h003A) >> b = 16'hA923	16'hCFB5
13	16'hFF31	16'hA5CD	16'h5AFC	16'h0569	(16'hCFB5 ⊕ 16'h0569) >> b = 16'h656E	16'h3F92
14	16'hFAF9	16'hB517	16'h4FEE	16'hFFC2	(16'h3F92 ⊕ 16'hFFC2) >> b = 16'h6028	16'h2FC6
15	16'h0026	16'h9ED1	16'h9EF7	16'h9623	(16'h2FC6 ⊕ 16'h9623) >> b = 16'hDCF2	16'h4205
16	16'h02F2	16'h2324	16'h21D6	16'h0052	(16'h4205 ⊕ 16'h0052) >> b = 16'hA12B	16'h80FD
17	16'hF786	16'hCAFE	16'h3D78	16'hFAFA	(16'h80FD ⊕ 16'hFAFA) >> b = 16'hBD03	16'h807B
18	16'hFF23	16'h5249	16'hAD6A	16'hFEA6	(16'h807B ⊕ 16'hFEA6) >> b = 16'hBF6E	16'h1204

(c) Correct key

	Preamble generation block			Frame generation block		
[j]	$STS_{nom}I[j]$	$keyI[mod(j,16)]$	$STS_{faulty}I[j]$	$key_{h-c}I[mod(j,16)]$	$f(STS_{out}I[j-1], key_{h-c}I[mod(j,16)])$	$STS_{out}I[j]$
0	16'h02F2	16'h003A (bypassed)	16'h02F2	16'h0052 (bypassed)	(bypassed)	16'h02F2
1	16'hF786	16'h7C04	16'h8B82	16'hFAFA	(16'h02F2 ⊕ 16'hFAFA) >> b = 16'h7C04	16'hF786
2	16'hFF23	16'h0490	16'hFBB3	16'hFEA6	(16'hF786 ⊕ 16'hFEA6) >> b = 16'h0490	16'hFF23
3	16'h0923	16'hD69A	16'hDFB9	16'h5216	(16'hFF23 ⊕ 16'h5216) >> b = 16'hD69A	16'h0923
4	16'h05E3	16'hAF9B	16'hAA78	16'h5614	(16'h0923 ⊕ 16'h5614) >> b = 16'hAF9B	16'h05E3
5	16'h0923	16'hE78E	16'hEEAD	16'hCAFE	(16'h05E3 ⊕ 16'hCAFE) >> b = 16'hE78E	16'h0923
6	16'hFF23	16'hFBED	16'h04CE	16'hFEF8	(16'h0923 ⊕ 16'hFEF8) >> b = 16'hFBED	16'hFF23
7	16'hF786	16'hDD1A	16'h2A9C	16'h4516	(16'hFF23 ⊕ 16'h4516) >> b = 16'hDD1A	16'hF786
8	16'h02F2	16'h7B6F	16'h799D	16'h0158	(16'hF786 ⊕ 16'h0158) >> b = 16'h7B6F	16'h02F2
9	16'h0026	16'h6406	16'h6420	16'hCAFE	(16'h02F2 ⊕ 16'hCAFE) >> b = 16'h6406	16'h0026
10	16'hFAF9	16'h7FC5	16'h853C	16'hFFAC	(16'h0026 ⊕ 16'hFFAC) >> b = 16'h7FC5	16'hFAF9
11	16'hFF31	16'hA829	16'h5718	16'hAAAA	(16'hFAF9 ⊕ 16'hAAAA) >> b = 16'hA829	16'hFF31
12	16'h0000	16'hFF85	16'hFF85	16'h003A	(16'hFF31 ⊕ 16'h003A) >> b = 16'hFF85	16'h0000
13	16'hFF31	16'h82B4	16'h7D85	16'h0569	(16'h0000 ⊕ 16'h0569) >> b = 16'h82B4	16'hFF31
14	16'hFAF9	16'h8079	16'h7A80	16'hFFC2	(16'hFF31 ⊕ 16'hFFC2) >> b = 16'h8079	16'hFAF9
15	16'h0026	16'h366D	16'h364B	16'h9623	(16'hFAF9 ⊕ 16'h9623) >> b = 16'h366D	16'h0026
16	16'h02F2	16'h003A	16'h02C8	16'h0052	(16'h0026 ⊕ 16'h0052) >> b = 16'h003A	16'h02F2
17	16'hF786	16'h7C04	16'h8B82	16'hFAFA	(16'h02F2 ⊕ 16'hFAFA) >> b = 16'h7C04	16'hF786
18	16'hFF23	16'h0490	16'hFBB3	16'hFEA6	(16'hF786 ⊕ 16'hFEA6) >> b = 16'h0490	16'hFF23

bit rotation function, for a single bit flip of the secret key, there is a large and arbitrary number of bit flips in STS_{out}. Thus, even for an incorrect key with HD of one from the correct key, STS_{out} contains a high number of errors. As a result, this *SyncLock* implementation has a full effective 512-bit key size.

So far, we have assumed a full key size of 512 bits. However, this is a rather unnecessarily large key size from a security point view. Typically, a key size of 64 bits suffices to guarantee high resilience against brute-force and optimization attacks. Therefore, we can consider a smaller key size that can be composed using any key-bits of the original 512-bit key since all of them are effective. Reducing the key size has the advantage of reducing the die area of the TPM and of the lock mechanism itself.

5.1.8 *Overheads*

1. **Area Overhead**: The hardware added by *SyncLock* are two XOR-based modules in the preamble and frame generation blocks, and one nonlinear module involving another XOR operation and a bitwise rotation in the frame generation block. To compute the area overhead of *SyncLock* we used as baseline unlocked implementation an open-source IEEE 802.11 compatible SDR HDL modem [126]. The project is called *bladeRF-wiphy* as it implements the IEEE 802.11 PHY on the Cyclone V FPGA integrated on the bladeRF board. More details about the bladeRF board are given in Sections 3.3.1 and 5.2. Starting from the unlocked implementation, we added the *SyncLock* locking mechanism into the PHY of the modem and we re-synthesized the project using Quartus II 16.0 from Intel to find the resultant overhead. Considering a full key size of 512 bits, *SyncLock* results in 1.22% area overhead for the baseband DSP section, which when projected to the entire RF transceiver is even smaller as the area is dominated by the AFE. The TPM overhead for the key management scheme is common to all locking schemes. Besides, the key can be shared across all blocks in a SoC. Thus, the overhead of the key management scheme is taken as fixed for any locking mechanism and is not considered.

2. **Power overhead**: Embedding *SyncLock* in the *bladeRF-wiphy* PHY implementation as above resulted in no noticeable power overhead.

3. **Performance penalty**: Since the *SyncLock* mechanism is entirely implemented into the baseband DSP of the RF transceiver it does not incur any performance penalty. This is confirmed with hardware measurements in Section 5.3.

4. **Design flow**: The AFE is left intact, thus there is no change in the analog IC design flow. This is an important attribute of *SyncLock* since analog IC designers are often reluctant to make any alternations in the circuit once it is finalized since this would typically add parasitics that would likely degrade performance. A lock mechanism inside the analog section inevitably would have to be co-designed with the circuit, possibly increasing design iterations and failing to meet the intent specifications. In contrast, *SyncLock* is a plug-in module added to the digital section of the RF transceiver once the design is completed without requiring any change in the design flow.

5.1.9 *Practicality*

Since a synchronization process is present and necessary in any wireless communication protocol, *SyncLock* is applicable to any of them. For instance, Wireless Local Area Network (WLAN) IEEE 802.11 (i.e., Wi-Fi), Wireless Personal Area Network (WPAN) IEEE 802.15.1 (i.e., Bluetooth), Low-Rate Wireless Personal Area Network (LR-WPAN) IEEE 802.15.4 (i.e., Zigbee), and any other standard using correlation-based synchronization algorithms, are natural candidates. Furthermore, since *SyncLock* only acts on the preamble generation, it is independent of the modulation scheme that is applied on the payload and, thereby, it is generally applicable for any modulation scheme. Finally, since *SyncLock* modifies only the DSP, it is independent of the AFE of the RF transceiver architecture; different RF transceiver architectures are presented in Section 4.1.1. Therefore, it can be applied to conventional RF transceiver architectures, such as Zero Intermediate Frequency (Zero-IF) and Low Intermediate Frequency (Low-IF), as well as to highly-digitized RF transceiver architectures.

5.1.10 *First SyncLock version*

The first *SyncLock* implementation in [121] is a special case of the *SyncLock* implementation proposed in this chapter. In particular, the first *SyncLock* implementation does not include the nonlinear module and the feedback in the part of the mechanism embedded inside the frame generation block. This can be expressed as

$$f(STS_{out}, key_{h-c}) = key_{h-c} \tag{15}$$

with the system equation being simplified from Eq. (5) to

$$STS_{out} = STS_{nom} \oplus (key \oplus key_{h-c}) \tag{16}$$

As will be explained in detail in Section 5.5.2-5, the motivation for the herein *SyncLock* implementation is that the first *SyncLock* implementation in [121] is vulnerable to the KPA. This new *SyncLock* implementation effectively thwarts the KPA.

Another difference is that the first *SyncLock* implementation in [121] does not provide a full effective key size. The reason is that a single bit flip in the input secret key results in a single bit flip in STS_{out}. To quantify the fraction of incorrect keys that are still capable of enabling synchronization we performed a Hamming Distance (HD) test. In particular, we generated incorrect keys with increasing HD from the correct key. For a full size key of 512 bits, there are 512 incorrect keys with HD=1 and $\binom{512}{k}$ keys with HD=k. Since $\binom{512}{k}$ is very high for $k > 1$, e.g., $\binom{512}{2}$=130816, for each $k > 1$ we tested a randomly generated set of 10^3 keys. We increased k until for all tested 10^3 keys synchronization failed. The results are shown in Table 11. For HD=1, only 192 incorrect keys, or 37% of the incorrect keys, did not allow the synchronization process. This percentage increases with k and for $k = 14$ all incorrect keys resulted in no synchronization. The number of incorrect keys that enabled synchronization can be estimated as:

$$\sum_{i=1}^{n=13} \left(1 - \frac{\omega_k}{100}\right) \cdot \binom{512}{i} \approx 10^{22}$$

where ω_k is the percentage of failing keys for HD=k and $n = 13$ is the highest HD showing keys that enable synchronization. Thus, a negligible percentage $(10^{22}/2^{512}) \cdot 100 = 10^{-131}\%$ of incorrect keys were capable of enabling synchronization.

The rest of 320 incorrect keys with HD=1 from the secret key that still enabled synchronization have the bit-flip in one of the 320 Least Significant Bit (LSB) positions of the secret key. This means that each of the 320 LSBs alone of the secret key is not effective, reducing the effective key size to 192.

Table 11: HD test for the first *SyncLock* implementation in [121].

HD	1	2	3	4	5	6	7
Percentage of failing keys	37.5%	63.5%	72.8%	84.6%	91.7%	94.4%	96.3%
HD	8	9	10	11	12	13	14
Percentage of failing keys	98.0%	98.3%	98.9%	99.3%	99.7%	99.9%	100%

Table 12 summarizes the comparison between the first *SyncLock* implementation and the new one presented in this chapter. The description of the different counter-attacks and the resilience to them will be described in detail in Section 5.5.

Table 12: Comparison between the new and first *SyncLock* implementations.

	First implementation [121]	New implementation [122]
Effective key size (bits)	192	512
Area overhead (projected to DSP)	1.12%	1.22%
Performance penalty	no	no
Attacks in the analog domain	✓	✓
Brute-force and optimization attacks	✓	✓
Input-output query attacks	✓	✓
Structural attacks	✓	✓
KPA through AFE baseband loopback	✓	✓
KPA through digital baseband loopback	✗	✓

✓: Safe , ✗: Not Safe

5.2 HARDWARE PLATFORM

SyncLock is demonstrated in hardware using a SDR bladeRF board (shown in Fig. 15) from Nuand [127]; the hardware platform is described in detail in Section 3.3.1. We implemented on the bladeRF board an IEEE 802.11 RF transceiver with a direct conversion AFE architecture for both the receiver and the transmitter. The bladeRF board has an AFE RF loopback mode as shown in Fig. 33, which allows us to perform BER measurements and symbol timing recovery, i.e., synchronization, using the same board. Note that this on-board loopback minimizes the impairments of the wireless communication channel, such as path loss, fading, and shadowing, and greatly simplifies the channel model. The measurements presented in Section 5.3 were obtained using this on-board loopback considering an Additive White Gaussian Noise (AWGN) channel model. The baseband DSP is designed in HDL [126] and implemented on the FPGA of the board. The HDL code of the preamble and frame generation blocks is modified to insert the *SyncLock* locking mechanism and is re-embedded into the same FPGA project. Detailed information on implementation overhead is presented in Section 5.1.8. As mentioned in Section 5.1.9,

SyncLock is independent of the modulation scheme. To show this, we repeated the demonstration by modulating the payload of the transmitted signal using Binary Phase-Shift Keying (BPSK), Quadrature Phase-Shift Keying (QPSK), and 16-Quadrature Amplitude Modulation (QAM), then encoding it into OFDM symbols. The frame generation block creates the PPDU frame format for an IEEE 802.11 transmission, as shown in Fig. 9. At the receiver side, the synchronization frame detection block searches for the start of the frame based on the Schmidl and Cox algorithm [125]. The received signal is processed and demodulated, and different performances of the RF transceiver are derived and visualized, such as BER and constellation diagram of the received payload.

The bladeRF board serves as a hardware platform for a fast prototyping of *SyncLock*, and it was preferred for a first full proof-of-concept compared to a fully integrated RF transceiver with *SyncLock* embedded, which would offer less experimentation freedom. The reason is that *SyncLock* is fully-digital, leaving the AFE intact, and the PHY can be quickly redesigned with the *SyncLock* mechanism embedded and then flashed into the FPGA of the bladeRF board. A chip fabrication would not add to the results presented herein. On the contrary, it would limit us experimenting with different functions $f(\cdot)$, hard-coded keys, modulation schemes for the payload, etc.

5.3 MEASURED *SyncLock* EFFICIENCY

5.3.1 *BER performance for correct key and incorrect keys*

The hardware platform is used for assessing the impact of *SyncLock* on the nominal performance when using the correct key and for demonstrating the locking efficiency when using an incorrect key. As discussed in Section 5.1.7, there is a single key enabling synchronization since any incorrect key, even those with HD=1 from the correct key, generate an arbitrary number of bit errors in the preamble of the outgoing data frame which impedes synchronization. For this reason, in the measurement results below we utilize a randomly selected incorrect key.

Fig. 36 shows the BER of an OFDM-BPSK transmission considering different Signal-to-Noise Ratio (SNR) values without *SyncLock*, with *SyncLock* when applying the correct key, and with *SyncLock* when applying a randomly selected incorrect key. A first observation is that when applying the correct key there is no BER penalty. The curves of BER without and with *SyncLock* are identical for all SNR values. This measurement proves that *SyncLock* is totally transparent when the correct key is used, thus there is zero performance penalty. This is expected since *SyncLock* leaves intact the sensitive AFE concentrating the lock mechanism inside the DSP. For each preamble bit line

SyncLock essentially introduces two spatially separated XOR gates in the path without causing any timing violation. A second observation is that with an incorrect key the system does not synchronize and erroneously demodulates the received signal. As a result, the BER is maximum and constant across all SNR values. It should be noted that for SNR values below -5 dB the synchronization was not possible even for the device with no locking.

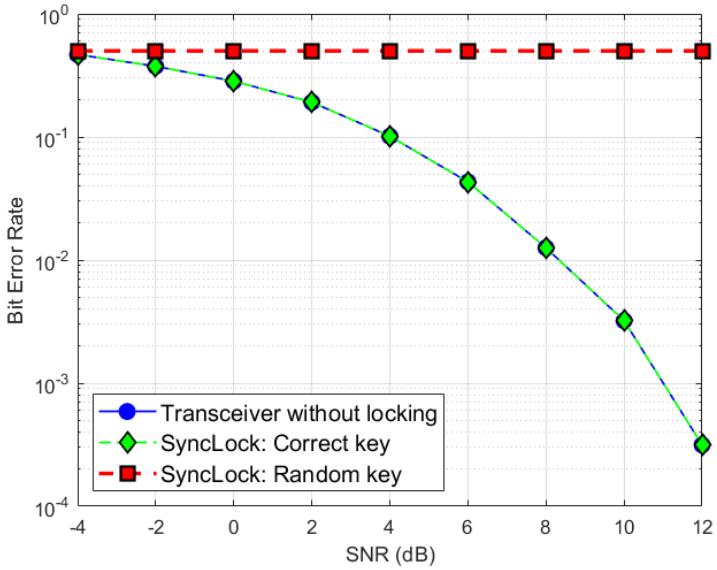


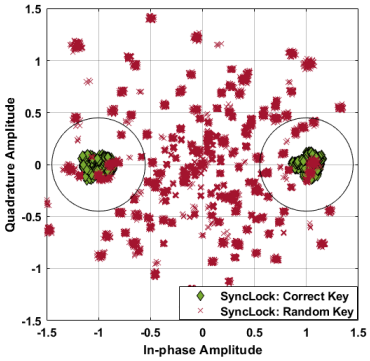
Figure 36: BER for RF transceiver with no locking and RF transceiver with *SyncLock* embedded using the correct and incorrect keys.

5.3.2 *Constellation diagrams for correct key and incorrect keys*

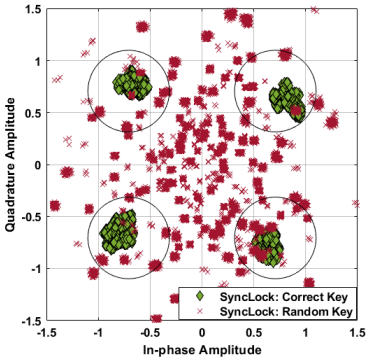
Fig. 37 shows the constellation diagrams of the received payload for three different modulation schemes, BPSK, QPSK, and 16-QAM, when applying the correct key and when applying a randomly selected incorrect key. The thin black circles show the reference constellation points for the modulation schemes. While the received signal lies inside the reference constellation for every modulation using the correct key, the non-synchronized signal is randomly distributed.

5.3.3 *Locking efficiency for approximate keys*

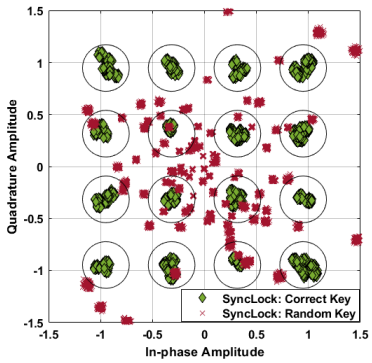
Finally, we tested the synchronization process for all the 512 incorrect keys with HD=1 from the correct key. All incorrect keys resulted in no synchronization with the smallest observed HD between $\mathrm{STS}_{\mathrm{out}}$ and $\mathrm{STS}_{\mathrm{nom}}$ being equal to 80. As discussed theoretically in Section 5.1.7, any incorrect key will show the same behavior observed in Figs. 36 and 37, with the only difference being the randomness of distribution



(a) BPSK.



(b) QPSK.



(c) 16-QAM.

Figure 37: Constellation diagram of the received payload with *SyncLock* embedded.

of payload data in Fig. 36 when different incorrect keys are loaded onto the chip.

5.4 RELATED LOCKING AND OBFUSCATION APPROACHES

Herein, we describe related locking approaches and explain their differences compared to *SyncLock*.

1. **Traditional XOR-based logic locking techniques**, discussed in Section 2.2.4.1-1, insert, randomly or algorithmically, key-gates into the design [80, 84, 86–88]. In contrast, *SyncLock* inserts key-gates on fixed binary sequences, i.e., $\mathrm{STS}_{\mathrm{nom}}$ inside the preamble generation block and $\mathrm{STS}_{\mathrm{faulty}}$ inside the frame generation block.

2. **Preamble obfuscation**: The XOR-based cipher of the *SyncLock* mechanism inside the preamble generator that encrypts the nominal preamble $\mathrm{STS}_{\mathrm{nom}}$ with a key was used in [164] as a Physical Layer (PHY) security to prevent man-in-the-middle

attacks such as eavesdropping. In this different context, the preamble obfuscation is performed through unique keys that are independently generated at both the transmitter and the receiver based on channel characteristics known only to the pair. Using only the XOR-based cipher inside the preamble generator is not sufficient for anti-piracy since the attacker can identify and straightforwardly remove this XOR-based cipher by tracing the key-bits from the TPM. *SyncLock* hides a second XOR-based cipher inside the frame generation block to achieve the anti-piracy objective.

3. **Corrupt-And-Correct (CAC) logic locking**: *SyncLock* belongs to the family of CAC logic locking techniques; these techniques are described in detail in Section 2.2.4.1-5. A state-of-the-art (SoA) CAC logic locking technique is SFLL-rem [105]. Chapter 4 presented how to adapt such a generic logic locking technique in the context of RF transceivers. *SyncLock* compared to SFLL-rem is conceptually different. As it will be discussed in detail in Section 5.5.2-4, SFLL-rem was shown to be vulnerable to a recently developed structural attack [102], while *SyncLock* circumvents successfully these attacks for two main reasons, namely (a) corruption is not a function of the STS_{nom} input which is fixed, i.e., *SyncLock* does not generate Protected Input Patterns (PIPs), and (b) the corrupt unit hidden inside the frame generation block is not linked to the STS_{nom} input.

5.5 THREAT MODEL AND SECURITY ANALYSIS

5.5.1 *Threat model*

We consider the most demanding threat model for a defender. We assume that the attacker is in possession of the netlist and an oracle, i.e., a working chip with the correct key applied into the TPM. The goal of the attacker is either to identify a key that establishes synchronization or, alternatively, remove *SyncLock* while restoring the functionality.

5.5.2 *Resilience to counter-attacks*

Table 13 presents a comparison of *SyncLock* and different anti-piracy defenses, their scope, attack stage, and resilience to known counter-attacks. The defenses presented in Table 13, are introduced in more detail in Section 2.2.

Next, we describe the known counter-attacks and discuss how *SyncLock* achieves resilience against all of them.

1. **Attacks in the analog domain**: These attacks assume the existence of an obfuscated analog component, e.g., through biasing

Table 13: Comparison of different anti-piracy defenses, their scope, attack stage, and resilience to known counter-attacks.

	Defense	Defense applicability	Attack stage			Counter-attacks to defense					
			System Integrator	Foundry	Reverse -engineering	Attacks on biasing locking	Brute force	Optimization -based	I/O query	Structural	KPA
	Split Manufacturing	Fabrication in two different foundries	✗	✓	✗	N/A	N/A	N/A	N/A	N/A	N/A
	Camouflaging	Analog and Digital ICs	✗	✗	✓	N/A	N/A	N/A	N/A	N/A	N/A
	Biasing Locking	Analog ICs	✓	✓	✓	✗	✓	✓	N/A	N/A	N/A
	Calibration Locking	IC with multi-bit tuning, secure calibration algorithm	✓	✓	✓	N/A	N/A	N/A	N/A	N/A	N/A
Logic Locking	Traditional XOR-based	Generic	✓	✓	✓	N/A	✓	✓	✗	✓	N/A
Logic Locking	SAT-attack resilient	Generic	✓	✓	✓	N/A	✓	✓	✓	✗	N/A
Logic Locking	Corrupt-And-Correct	Generic	✓	✓	✓	N/A	✓	✓	✓	✗	N/A
Logic Locking	***SyncLock***	RF transceivers	✓	✓	✓	N/A	✓	✓	✓	✓	✓

✓: Resilience to attack , ✗: vulnerability to attack

locking. They do not apply to *SyncLock* since *SyncLock* is not based on analog component obfuscation.

2. **Brute-force and optimization attacks**: The attacker searches in the key space either randomly in a brute-force manner or more efficiently by employing an optimization algorithm hoping to find a key that enables synchronization. The search is performed by simulating the design at netlist-level where the TPM is circumvented and the key inputs are accessed directly. At each iteration, instead of evaluating synchronization, a faster evaluation criterion may be devised by involving the oracle. For example a simulated transient response can be compared to that of the oracle. Resilience against this attack is achieved since: (a) the key space size, i.e., 2^{512} for a full key size, is huge; (b) a single secret key enables synchronization, thus the optimization function behaves like a delta function on the secret key and an optimization algorithm will "zig-zag" endlessly; (c) a single simulation at netlist-level can be very time-consuming, thus the attacker in practice can perform a very limited number of trials.

3. **Input-output query attacks**: SAT-based attacks compute Distinguishing Input Patterns (DIPs), defined as inputs which produce different output for at least two different keys, and prunes down multiple incorrect keys iteratively using DIPs and querying the oracle. CAC techniques, like SFLL-hd and SFLL-rem, were specifically proposed to push the limits of the SAT attack by eliminating exactly one key per iteration, thus making it equivalent to a brute-force attack in terms of attack time. SAT-based attacks does not apply to *SyncLock* since the inputs to the preamble generation block, i.e., the DATA_i values and SEL, are fixed and hard-coded, thus no DIPs can be generated.

4. **Structural attacks**: Structural attacks, a.k.a. removal attacks, aim at identifying and removing the locking mechanism. The attacker can trace the key-bits from the TPM to straightforwardly identify and remove the first XOR-based stream cipher in the preamble generation block. In this case, the design will be left with a hard-coded error introduced by the second XOR-based stream cipher inside the frame generation block. Thus, the attacker will need to identify this second corrupt unit too to complete the removal attack. However, after logic synthesis this small circuit is immersed in the original design and the two become inseparable. The attacker has at hand a non-annotated netlist, thus identifying this small circuit is puzzling.

   The fact that the corrupt unit is non-identifiable is the hypothesis of SFLL-hd too. For SFLL-hd, however, two specific attacks were developed recently that succeed in identifying the corrupt unit [100, 165]. They perform a structural analysis of the locked netlist to identify PIPs by leveraging the properties of the HD-based corrupt and correct units, and also exploiting the fact that in SFLL-hd the input feeds the corrupt unit. These attacks, apart from being specific to SFLL-hd, are not generalizable for *SyncLock* for two reasons. First, in *SyncLock* corruption is not a function of the STS_{nom} input which is fixed, i.e., *SyncLock* does not generate PIPs or stated differently all input are PIPs. Second, the corrupt unit hidden inside the frame generation block is spatially separated from the correct unit inside the input preamble generation block, thus the corrupt unit cannot be traced from the input.

   Another structural attack was recently proposed that defeats both SFLL-hd and SFLL-rem [102]. It works differently by analyzing the Boolean truth table of the corrupted circuit to extract the PIPs. The SFLL-hd and SFLL-rem techniques essentially construct the corrupted circuit by adding (removing) selected minterm(s) to (from) the original circuit to create the PIPs. Then, the logic synthesis tool synthesizes the resulting corrupted circuit. The attack in [102] aims at recovering the PIPs. It demonstrates how the optimization performed to minimize the PPA overhead executed by the EDA tools may expose the PIPs. This attack is not applicable to *SyncLock* either since *SyncLock* does not employ PIPs to secure the circuit, i.e., it does not add or remove any minterms, and no trace is left in the Boolean truth table. In addition, the correctness of the extracted PIPs is verified by querying the oracle with the PIPs. As explained in Section 5.5.2-3, in the case of *SyncLock*, an attacker cannot query the oracle since the inputs to the preamble generation block, i.e., the DATA_i values and SEL, are fixed and hard-coded.

In short, it is realistic to assume that the corrupt unit inside the frame generation block cannot be distinguished within a "sea" of non-annotated digital gates. However, as we observed with other corrupt-and-correct logic locking techniques [103, 105], it leaves a backdoor that may be exploited to develop an attack based on structural analysis. Although such an attack is not known at this point for *SyncLock*, the possibility cannot be ruled out.

5. **Known-Plaintext Attack (KPA)**: It applies to stream ciphers with symmetric encryption, i.e., when the encryption and decryption processes are performed using the same encryption key. In our context, the plaintext is STS_{nom} and is known to the attacker since it is published in the IEEE 802.11 standard [124]. Let us consider the first *SyncLock* implementation in [121]. The attacker applies a trial key, denoted by key_{trial}. From Eq. (16), using the associative and self-inverse properties of the XOR function, we obtain

$$\mathrm{key}_{h-c} = \mathrm{key}_{trial} \oplus \mathrm{STS}_{nom} \oplus \mathrm{STS}_{out}. \tag{17}$$

Knowing STS_{nom} and selecting any key_{trial}, the attacker can successfully recover key_{h-c} and, thereby, the secret key since $\mathrm{key} = \mathrm{key}_{h-c}$, provided that STS_{out} is measured accurately. In the oracle chip, the attacker cannot re-write the TPM to apply a trial key, thus STS_{out} has to be extracted by simulation. The attacker can simulate a transmission and try to extract STS_{out} from the transmitted frame. A loopback connection to the receiver can be used to analyse the transmitted signal. There are three different loopback modes, as shown in Fig. 33, namely (a) the AFE RF loopback that connects the output of the transmitter's PA to the input of the receiver's LNA, (b) the AFE baseband loopback right before the RF mixers, and (c) the digital baseband loopback between the DSP output and subsequent AFE data converters. In all three scenarios, the attacker should be able to manually locate the received STS_{out} bits. However, with loopback modes (a) and (b), some bits of STS_{out} will be corrupted due to analog impairments, quantization noise, and nonlinearities introduced throughout the signal processing at transistor-level. To demonstrate this, we implemented this attack using loopback mode (b) on our hardware platform described in Section 5.2. Fig. 38 shows the amplitude values of the first 32 samples of the transmitted and received (measured) STS_{out} for a given key_{trial}. As it can be seen, for every sample the amplitude values differ between the two STS_{out} signals. To quantify the number of bit errors, we translated the floating-point values into binary fixed-point values and obtained a HD

of 1902 bits out of 5120 bits between the two signals. Thus, the transmitted $\mathrm{STS}_{\mathrm{out}}$ will be extracted with errors and the computed $\mathrm{key}_{\mathrm{h-c}}$ from Eq. (17) will be incorrect.

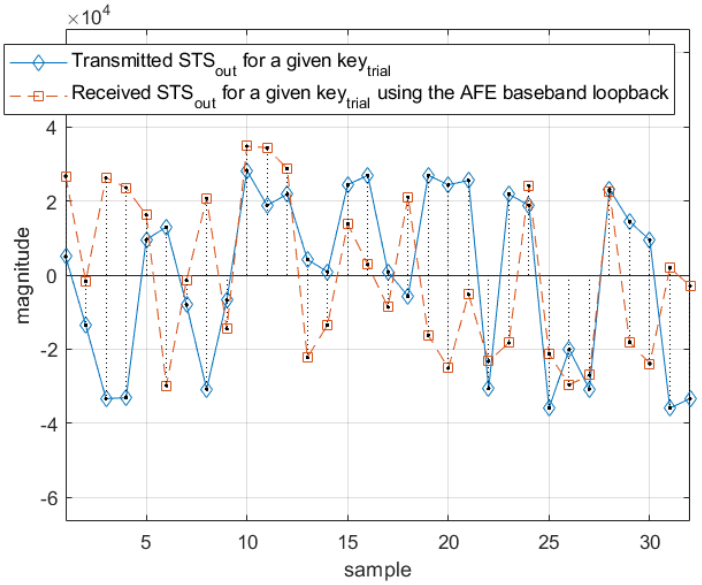


Figure 38: Amplitude values of the first 32 samples of the transmitted and received $\mathrm{STS}_{\mathrm{out}}$ for a given $\mathrm{key}_{\mathrm{trial}}$ using the Analog Front-End (AFE) baseband loopback.

In contrast, if the attacker can locate the boundary between the DSP and AFE to implement loopback mode (c), then $\mathrm{STS}_{\mathrm{out}}$ can be extracted accurately and the KPA is completed successfully. This is a security breach of the first *SyncLock* implementation in [121] that is addressed with the second *SyncLock* implementation in this chapter. As shown in the comparison Table 12, this is the main differentiation between the two implementations and the motivation for complexifying *SyncLock* giving rise to the second implementation.

In particular, introducing the nonlinear feedback inside the frame generation block helps thwarting the KPA even when $\mathrm{STS}_{\mathrm{out}}$ is correctly extracted using a purely digital baseband loopback (c). The reason is that the identity from Eq. (5) now becomes

$$\mathrm{STS}_{\mathrm{out}} = (\mathrm{STS}_{\mathrm{nom}} \oplus \mathrm{key}_{\mathrm{trial}}) \oplus \mathrm{f}(\mathrm{STS}_{\mathrm{out}}, \mathrm{key}_{\mathrm{h-c}}) \quad (18)$$

making it impossible to de-embed $\mathrm{key}_{\mathrm{h-c}}$ since both the nonlinear function f and $\mathrm{key}_{\mathrm{h-c}}$ are unknown to the attacker. In the implementation presented herein, $\mathrm{f}(\cdot)$ is an XOR followed by a circular shift operation. However, as mentioned in Section 5.1.4, any shift value can be used and any other function performing

bitwise operations can be used instead. Thus, we presented one out of the countless implementations without endangering the security of *SyncLock*.

5.6 CONCLUSION

We presented *SyncLock*, an anti-piracy design technique for RF transceivers. *SyncLock* makes the synchronization between the transmitter and receiver key-dependent, while there is a single valid key that can have up to 512 effective key-bits. The *SyncLock* mechanism is hidden inside the DSP resulting in an overhead of around 1.22% of the DSP, which is negligible when projected to the entire RF transceiver since the area is dominated by the AFE. *SyncLock* is non-intrusive to the RF transceiver operation when applying the correct key incurring no performance penalty. No changes in the AFE design or the analog design flow are needed. The *SyncLock* mechanism is a simple plug-in to the DSP. *SyncLock* is a generic approach applicable to any RF transceiver architecture, communication protocol, and modulation scheme. Finally, it is shown to be resilient against any known counter-attack aiming at finding the secret key or removing the lock mechanism. The research results presented in this chapter were published in [121, 122].

6

CONCLUSION AND PROSPECTIVE RESEARCH

6.1 CONCLUSION

Today's globalized and outsourcing-based IC supply chain does not allow any steps backward. The present threats in the IC supply chain should not be solved through deglobalization but through novel prevention technologies. Semiconductor industry stakeholders are recognizing, and several are experiencing, the consequences of escalating IC supply chain attacks. The commercialization of HT-infected ICs could permanently damage a company's reputation and trust. In that sense, counting on HT detection mechanisms is imperative. Moreover, in this dynamic and strategic industry, the secrecy of semiconductor companies' IP is a major advantage, therefore, having tools to protect their assets is paramount.

The findings of this thesis contributed substantially to the state-of-the-art and fulfilled the objectives stated in its motivation. This thesis has gone a long way towards enhancing our understanding of RF transceiver security and trust. The HT study findings suggest a course of action that needs to be taken to ensure that the vulnerability exposed in current communication standards is addressed in the upcoming standards that will govern future communication technologies. In parallel, the findings of this thesis suggest that generic solutions are not always the most suitable for specific challenges. In that sense, we have established the guidelines to properly secure RF transceivers. We proposed the first domain-specific logic locking technique for securing RF transceivers, thus laying the foundation of a new research niche.

6.2 CONTRIBUTIONS OF THE THESIS

The findings of this thesis that contribute to the state-of-the-art are presented below by chapter. Also, a summary of the publications is presented in the section after the thesis abstract.

Chapter 3: The research results presented in this chapter were published in [55]. The work presented in this chapter made the following contributions to the state-of-the-art:

1. A comprehensive study on the evolution of thinking in HT attack models.

2. A novel post-silicon, run-time, and low-cost correlation-based defense to detect HT activity hidden in the synchronization data.

3. A novel HT attack model than can thwart any post-silicon defense mechanism. We proposed the AM STS HT attack for leaking sensitive data out of wireless ICs. The AM STS HT attack acts on the synchronization preamble, does not affect the normal RF transceiver operation or link performance, and has a tiny footprint.

4. Our experiments ascertained the strength of the proposed AM STS HT attack calling for the development of a specific practical defense so as to ensure the security of existing and upcoming wireless communication standards. To this end, we discussed several known generic HT countermeasures that could be potentially applicable and should be further evaluated.

5. We demonstrated with hardware measurements the AM STS HT attack from the attacker's perspective where an encrypted message with a 128-bit secret key is being leaked. We analyzed the reliability of the covert channel and we demonstrated that the key can be successfully recovered after less than 10 key transmissions even in the most unfavorable SNR scenario.

Chapter 4: The research results presented in this chapter were published in [120]. This work made the following contributions to the state-of-the-art:

1. We demonstrated for the first time locking against piracy of entire RF transceivers at the system-level. The methodology is based on logic locking of digital blocks in the signal processing chain. The methodology is virtually applicable to any RF transceiver architecture. It can be seamlessly integrated into the RF transceiver design flow, since its analog section is left intact and logic locking is fully automated.

2. We showed that generic logic locking techniques cannot be blindly applied in the context of A/M-S ICs. To this end, we employed the state-of-the-art SFLL-rem logic locking technique and showed how to fine-tune it for effective RF transceiver locking.

3. A proof-of-concept is demonstrated with hardware measurements using a SDR board. Hardware experiments demonstrated effective BER degradation for incorrect keys while achieving zero performance penalty when applying the single correct secret key and negligible area and power overheads.

4. We analyzed the resilience against all foreseen key-recovery attacks.

Chapter 5: The findings presented in this chapter were published in [121, 122]. This work made the following contributions to the state-of-the-art:

1. We presented *SyncLock*, an anti-piracy design technique for RF transceivers. *SyncLock* is an RF transceiver-specific logic locking technique that acts on the synchronization of the transmitter with the receiver. Upon application of an incorrect key, *SyncLock* disables the synchronization, thus the wireless communication link crashes.

2. We presented a novel and innovative RF transceiver-specific logic locking technique. It is based on two spatially separated hardware-level mechanisms. The first mechanism hides a hard-coded error into the design of the data frame generator corrupting the preamble of the data frame. The second mechanism is located upstream in the signal processing chain into the preamble generator and its goal is to corrupt the preamble so as to cancel out the downstream corruption. The corruption applied by the second mechanism is key-controlled with the key being sourced from the TPM. There exists a single correct key that can counterbalance the two spatially separated preamble corruptions.

3. *SyncLock* is generally applicable to any RF transceiver architecture, for any wireless communication protocol using correlation-based synchronization algorithms, and for any modulation scheme. As the lock mechanism is embedded into the baseband DSP, *SyncLock* can be effortlessly integrated into the digital design flow. On the other hand, the sensitive AFE is left intact which is an essential characteristic of *SyncLock* allowing for its wide adoption by analog IC designers.

4. *SyncLock* elegantly achieves all locking objectives: (a) locking is totally transparent to the RF transceiver operation when the correct secret key is applied; (b) applying any invalid key breaks the RF transceiver operation; (c) incurs minimal area and power overheads; (d) thwarts any known counter-attack in both the analog and digital domains.

5. We present the *SyncLock* hardware implementation in detail showing also how the secret key is computed for a given hard-coded error.

6. *SyncLock* is demonstrated with hardware measurements using the SDR bladeRF board.

7. We demonstrate experimentally that *SyncLock* is independent of the modulation method used by the RF transceiver.

6.3 PROSPECTIVE RESEARCH ON RF TRANSCEIVER SECURITY AND TRUST

The perspective of future work aims to arm the upcoming wireless technologies with the most advance and resilient hardware security defenses.

A natural progression of this work is keep researching on RF transceiver-specific logic locking techniques. The Sorbonne Université and SATT Lutech have decided to patent *SyncLock*'s innovation. The patent was filed in March 2022 [166]. In addition, the *SyncLock* project is one of the laureates of the 2022 i-PhD innovation contest [167]. As a result, *SyncLock* is evolving towards a *Deep tech* start-up.

However, synchronization is not the only essential block in an RF transceiver, therefore, other components in the processing chain could be locked to protect the entire system against piracy attacks. For instance, modern Physical Layer (PHY) implementations are equipped with error correction code algorithms. These components detect and correct bit errors on noisy communication channels. However, if the error correction code algorithm is key-controlled an incorrect key would erroneously map the received bits, creating more errors and degrading the RF transceiver performance.

More experiments, using a broader range of communication standards, could shed more light on common properties in synchronization processes of wireless devices. These findings could lead to a comprehensive synchronization locking technique, allowing *SyncLock* to be applied broadly to any wireless communication standard. Additional research in other standards is, therefore, an essential next step to confirm whether the synchronization process or another processing step would be the best locking target.

Another future work idea is making a full Application-Specific Integrated Circuit (ASIC) implementation of an RF transceiver with embedded locking mechanism. A silicon-proven implementation of the *SyncLock* defense could be a very appealing demonstrator for the RF transceiver design community. As well, new attacks will emerge, perhaps in the direction of *SyncLock*, and therefore, new implementations enriched by the scientific community will have to be made.

The emerging IEEE 802.XX communication standard must be continually enriched with security features to meet the new and more critical application demands. Concerning HTs, there is, therefore, a clear need to incorporate more security features in the upcoming wireless communication standards. A key policy priority should therefore be to plan for the long-term care of HT detection and device identification. Wireless device identification is a very important fea-

ture in RF transceiver security. This tool could ensure the resilience of wireless networks. For example, if a wireless device is detected to be HT-infected, it could be identified and banned from the network, mitigating damage. Device identification must be unique and tamper-proof, therefore, adding silicon-level protection is one of the best alternatives to secure wireless devices.

BIBLIOGRAPHY

[1] A. Varas, R. Varadarajan, J. Goodrich, and F. Yinug. *Strengthening the global semiconductor supply chain in an uncertain era*. Report. Boston Consulting Group and Semiconductor Industry Association, 2021.

[2] ASML. *EU chips act, positional paper*. Report. ASML, 2022.

[3] *Design and reuse*. https://www.design-reuse.com/. online.

[4] 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." In: *Proceedings of the IEEE* 102.8 (2014), pp. 1207–1228.

[5] U.S. Department of Commerce and U.S. Department of Homeland Security. *Assessment of the critical supply chains supporting the U.S. information and communications technology industry*. Report. U.S. Department of Commerce and U.S. Department of Homeland Security, 2022.

[6] R. Torrance and D. James. "The state-of-the-art in semiconductor reverse engineering." In: *Proceedings of the 48th Design Automation Conference (DAC*. 2011, pp. 333–338.

[7] B. Lippmann, M. Werner, N. Unverricht, A. Singla, P. Egger, A. Dübotzky, H. Gieser, M. Rasche, O. Kellermann, and H. Graeb. "Integrated flow for reverse engineering of nanoscale technologies." In: *Proceedings of the 24th Asia and South Pacific Design Automation Conference*. 2019, pp. 82–89.

[8] M. Holler, M. Guizar-Sicairos, E. H. Tsai, R. Dinapoli, E. Müller, O. Bunk, J. Raabe, and G. Aeppli. "High-resolution non-destructive three-dimensional imaging of integrated circuits." In: *Nature* 543.7645 (2017), pp. 402–406.

[9] D. Takahashi. "Feds close huge chip counterfeiting case." In: *VentureBeat*. 2011. URL: https://bit.ly/3JjgoHf (visited on 09/25/2011).

[10] European Union Intellectual Property Office. *Intellectual Property Crime Threat Assessment 2022*. Report. EUROPOL, 2022.

[11] S. Smith. "SEMI: Innovation Is at Risk, Losses of up to $4 Billion Annually due to IP Infringement." In: *Nanotechnology Now*. 2008. URL: https://bit.ly/3bKz1r0 (visited on 08/08/2022).

[12] T. Linton and B. Vakil. "Coronavirus Is Proving We Need More Resilient Supply Chains." In: *Harvard Business Review*. 2020. URL: https://hbr.org/2020/03/coronavirus-is-proving-that-we-need-more-resilient-supply-chains (visited on 04/20/2022).

[13] W. Debby, L. Yoolim, and N. Yantoultra. "Semiconductor, Chip Shortages to Worsen as Covid Worsens in Malaysia." In: *Bloomberg*. 2021. URL: https://bloom.bg/3OOa8II (visited on 08/23/2021).

[14] Y. Stephanie and S. Jiyoung. "Global chip shortage is far from over as wait times get longer." In: *The Wall Street Journal*. 2021. URL: https://on.wsj.com/3zNXK7a (visited on 10/29/2021).

[15] JABIL. "Why The Chips Are Down: Explaining the Global Chip Shortage." In: *Jabil*. 2022. URL: https://www.jabil.com/blog/global-chip-shortages.html (visited on 08/01/2022).

[16] I. King. "Wait Times for Chips Grow Again in March as Shortages Drag On." In: *Yahoo Financial*. 2022. URL: https://yhoo.it/3SjLgLC (visited on 04/05/2022).

[17] N. Flaherty. "Counterfeit chips flood in to exploit chip shortage." In: *Electronics Europe News*. 2021. URL: https://www.eenewseurope.com/en/counterfeit-chips-flood-in-to-exploit-chip-shortage/ (visited on 04/20/2022).

[18] *ERAI: Counterfeit Electronics Database*. https://www.erai.com/. online.

[19] A. Shah. "Europe, US warn of fake-chip danger to national security, critical systems." In: *The Register*. 2022. URL: https://www.theregister.com/2022/03/18/eu_us_counterfeit_chips/ (visited on 04/20/2022).

[20] *Internet of Things statistics for 2022 - Taking Things Apart*. https://dataprot.net/statistics/iot-statistics/. online.

[21] M. Tehranipoor and F. Koushanfar. "A survey of hardware trojan taxonomy and detection." In: *IEEE Design & Test of Computers* 27.1 (2010), pp. 10–25.

[22] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor. "Trustworthy hardware: Identifying and classifying hardware trojans." In: *Computer* 43.10 (2010), pp. 39–46.

[23] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan. "Hardware Trojan attacks: Threat analysis and countermeasures." In: *Proceedings of the IEEE* 102.8 (2014), pp. 1229–1247.

[24] K. Xiao, D. Forte, Y. Jin, R. Karri, S. Bhunia, and M. Tehranipoor. "Hardware trojans: Lessons learned after one decade of research." In: *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 22.1 (2016), pp. 1–23.

[25] S. Bhunia and M. M. Tehranipoor. *The Hardware Trojan War: Attacks, Myths, and Defenses*. Springer, 2017.

[26] A. Jain, Z. Zhou, and U. Guin. "Survey of Recent Developments for Hardware Trojan Detection." In: *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. 2021, pp. 1–5.

[27] Y. Shiyanovskii, F. Wolff, A. Rajendran, C. Papachristou, D. Weyer, and W. Clay. "Process reliability based trojans through NBTI and HCI effects." In: *2010 NASA/ESA Conference on Adaptive Hardware and Systems*. IEEE. 2010, pp. 215–222.

[28] L. Lin, T. G. M. Kasper, C. Paar, and W. Burleson. *Trojan Side-Channels: Lightweight Hardware Trojans through Side-Channel Engineering*. Berlin, Germany: Springer, 2009, pp. 382–395.

[29] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson. "Stealthy dopant-level hardware trojans: extended version." In: *Journal of Cryptographic Engineering* 4.1 (2014), pp. 19–31.

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

[31] X. Guo, H. Zhu, Y. Jin, and X. Zhang. "When capacitors attack: Formal method driven design and detection of charge-domain trojans." In: *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE. 2019, pp. 1727–1732.

[32] Y. Kim, R. Daly, J. Kim, C. Fallin, J. H. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu. "Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors." In: *ACM SIGARCH Computer Architecture News* 42.3 (2014), pp. 361–372.

[33] C. Kison, O. M. Awad, M. Fyrbiak, and C. Paar. "Security implications of intentional capacitive crosstalk." In: *IEEE Transactions on Information Forensics and Security* 14.12 (2019), pp. 3246–3258.

[34] K. Nagarajan, M. N. I. Khan, and S. Ghosh. "ENTT: A family of emerging NVM-based trojan triggers." In: *2019 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE. 2019, pp. 51–60.

[35] Z. Liu, Y. Li, Y. Duan, R. L. Geiger, and D. Chen. "Identification and break of positive feedback loops in Trojan states vulnerable circuits." In: *2014 IEEE International Symposium on Circuits And Aystems (ISCAS)*. IEEE. 2014, pp. 289–292.

[36] X. Cao, Q. Wang, R. L. Geiger, and D. J. Chen. "A hardware Trojan embedded in the Inverse Widlar reference generator." In: *2015 IEEE 58th International Midwest Symposium on Circuits And Systems (MWSCAS)*. IEEE. 2015, pp. 1–4.

[37] Q. Wang, R. L. Geiger, and D. Chen. "Hardware Trojans embedded in the dynamic operation of analog and mixed-signal circuits." In: *2015 National Aerospace and Electronics Conference (NAECON)*. IEEE. 2015, pp. 155–158.

[38] C. Cai and D. Chen. "Performance enhancement induced Trojan states in op-amps, their detection and removal." In: *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. 2015, pp. 3020–3023.

[39] Q. Wang, D. Chen, and R. L. Geiger. "Transparent side channel trigger mechanism on analog circuits with PAAST hardware trojans." In: *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. 2018, pp. 1–4.

[40] M. Elshamy, G. Di Natale, A. Pavlidis, M.-M. Louërat, and H.-G. Stratigopoulos. "Hardware trojan attacks in analog/mixed-signal ICs via the test access mechanism." In: *2020 IEEE European Test Symposium (ETS)*. IEEE. 2020, pp. 1–6.

[41] M. Elshamy, G. Di Natale, A. Sayed, A. Pavlidis, M.-M. Louërat, H. Aboushady, and H.-G. Stratigopoulos. "Digital-to-Analog Hardware Trojan Attacks." In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 69.2 (2021), pp. 573–586.

[42] M. Portolan, A. Pavlidis, G. Di Natale, E. Faehn, and H.-G. Stratigopoulos. "Circuit-to-Circuit Attacks in SoCs via Trojan-Infected IEEE 1687 Test Infrastructure." In: *2022 IEEE International Test Conference (ITC)*. 2022.

[43] N. Kiyavash, F. Koushanfar, T. P. Coleman, and M. Rodrigues. "A timing channel spyware for the CSMA/CA protocol." In: *IEEE Transactions on Information Forensics and Security* 8.3 (2013), pp. 477–487.

[44] A. Dutta, D. Saha, D. Grunwald, and D. Sicker. "Secret Agent Radio: Covert Communication through Dirty Constellations." In: *Information Hiding*. Ed. by M. Kirchner and D. Ghosal. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 160–175.

[45] J. Classen, M. Schulz, and M. Hollick. "Practical covert channels for WiFi systems." In: *2015 IEEE Conference on Communications and Network Security (CNS)*. 2015, pp. 209–217. DOI: 10.1109/CNS.2015.7346830.

[46] Z. Hijaz and V. S. Frost. "Exploiting OFDM systems for covert communication." In: *2010-Milcom 2010 Military Communications Conference*. IEEE. 2010, pp. 2149–2155.

[47] S. Grabski and K. Szczypiorski. "Steganography in OFDM symbols of fast IEEE 802.11 n networks." In: *2013 IEEE security and privacy workshops*. IEEE. 2013, pp. 158–164.

[48] K. S. Subraman, A. Antonopoulos, A. A. Abotabl, A. Nosratinia, and Y. Makris. "Demonstrating and mitigating the risk of an FEC-based hardware Trojan in wireless networks." In: *IEEE Transactions on Information Forensics and Security* 14.10 (2019), pp. 2720–2734.

[49] Y. Jin and Y. Makris. "Hardware Trojans in wireless cryptographic ICs." In: *IEEE Design & Test of Computers* 27.1 (2010), pp. 26–35.

[50] Y. Liu, Y. Jin, A. Nosratinia, and Y. Makris. "Silicon demonstration of hardware Trojan design and detection in wireless cryptographic ICs." In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 25.4 (2016), pp. 1506–1519.

[51] K. S. Subramani, N. Helal, A. Antonopoulos, A. Nosratinia, and Y. Makris. "Amplitude-modulating analog/rf hardware trojans in wireless networks: Risks and remedies." In: *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 3497–3510.

[52] D. Chang, G. Bhat, U. Ogras, B. Bakkaloglu, and S. Ozev. "Detection mechanisms for unauthorized wireless transmissions." In: *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 23.6 (2018), pp. 1–21.

[53] K. Sankhe, F. Restuccia, S. D'Oro, T. Jian, Z. Wang, A. Al-Shawabka, J. Dy, T. Melodia, S. Ioannidis, and K. Chowdhury. "Impairment shift keying: Covert signaling by deep learning of controlled radio imperfections." In: *MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM)*. IEEE. 2019, pp. 598–603.

[54] C. Kapatsori, Y. Liu, A. Antonopoulos, and Y. Makris. "Hardware dithering: a run-time method for trojan neutralization in wireless cryptographic ICs." In: *2018 IEEE International Test Conference (ITC)*. IEEE. 2018, pp. 1–7.

[55] A. R. Díaz Rizo, H. Aboushady, and H.-G. Stratigopoulos. "Leaking Wireless ICs via Hardware Trojan-Infected Synchronization." In: *IEEE Transactions on Dependable and Secure Computing* (2022), pp. 1–16. DOI: 10.1109/TDSC.2022.3218507.

[56] P. Clark. "Apple sues processor startup for theft of trade secrets." In: *eeNews - Analog*. 2022. URL: https://www.eenewsanalog.com/en/apple-sues-processor-startup-for-theft-of-trade-secrets/ (visited on 05/03/2022).

[57] D. P. Affairs. "DARPA Selects Teams to Increase Security of Semiconductor Supply Chain." In: *DARPA*. 2020. URL: https://www.darpa.mil/news-events/2020-05-27 (visited on 08/08/2022).

[58] S. Leef. *In Pursuit of Secure Silicon*. Report 1. Mentor Graphics, 2017.

[59] R. W. Jarvis and M. G. Mcintyre. *Split manufacturing method for advanced semiconductor circuits*. US Patent 7,195,931. Mar. 2007.

[60] T. D. Perez and S. Pagliarini. "A Survey on Split Manufacturing: Attacks, Defenses, and Challenges." In: *IEEE Access* 8 (Oct. 2020), pp. 184013–184035.

[61] Y. Bi, J. S. Yuan, and Y. Jin. "Beyond the interconnections: split manufacturing in RF designs." In: *Electronics* 4.3 (Aug. 2015), pp. 541–564.

[62] A. Vijayakumar, V. C. Patil, D. E. Holcomb, C. Paar, and S. Kundu. "Physical design obfuscation of hardware: A comprehensive investigation of device and logic-level techniques." In: *IEEE Transactions on Information Forensics and Security* 12.1 (2016), pp. 64–77.

[63] R. P. Cocchi, L. W. Chow, J. P. Baukus, and B. J. Wang. *Method and apparatus for camouflaging a standard cell based integrated circuit with micro circuits and post processing*. US Patent 8,510,700. Aug. 2013.

[64] A. Ash-Saki and S. Ghosh. "How multi-threshold designs can protect analog IPs." In: *Proc. IEEE Int. Conf. Comput. Design (ICCD)*. Oct. 2018, pp. 464–471.

[65] J. Leonhard, A. Sayed, M.-M. Louërat, H. Aboushady, and H.-G. Stratigopoulos. "Analog and mixed-signal IC security via sizing camouflaging." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 40.5 (2020), pp. 822–835.

[66] A. Chakraborty, N. G. Jayasankaran, Y. Liu, J. Rajendran, O. Sinanoglu, A. Srivastava, Y. Xie, M. Yasin, and M. Zuzak. "Keynote: A disquisition on logic locking." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39.10 (2019), pp. 1952–1972.

[67] V. V. Rao and I. Savidis. "Protecting analog circuits with parameter biasing obfuscation." In: *2017 18th IEEE Latin American Test Symposium (LATS)*. IEEE. 2017, pp. 1–6.

[68] V. V. Rao and I. Savidis. "Performance and security analysis of parameter-obfuscated analog circuits." In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 29.12 (2021), pp. 2013–2026.

[69] J. Wang, C. Shi, A. Sanabria-Borbon, E. Sánchez-Sinencio, and J. Hu. "Thwarting analog IC piracy via combinational locking." In: *2017 IEEE International Test Conference (ITC)*. IEEE. 2017, pp. 1–10.

[70] G. Volanis, Y. Lu, S. G. R. Nimmalapudi, A. Antonopoulos, A. Marshall, and Y. Makris. "Analog performance locking through neural network-based biasing." In: *2019 IEEE 37th VLSI Test Symposium (VTS)*. IEEE. 2019, pp. 1–6.

[71] D. H. K. Hoe, J. Rajendran, and R. Karri. "Towards Secure Analog Designs: A Secure Sense Amplifier Using Memristors." In: *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*. July 2014, pp. 516–521.

[72] N. G. Jayasankaran, A. Sanabria-Borbón, A. Abuellil, E. Sánchez-Sinencio, J. Hu, and J. Rajendran. "Breaking analog locking techniques." In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 28.10 (2020), pp. 2157–2170.

[73] R. Y. Acharya, S. Chowdhury, F. Ganji, and D. Forte. "Attack of the Genes: Finding Keys and Parameters of Locked Analog ICs Using Genetic Algorithm." In: *Proc. IEEE Int. Symp.Hardw. Oriented Secur. Trust (HOST)*. Dec. 2020, pp. 284–294.

[74] J. Leonhard, M. Elshamy, M.-M. Louërat, and H.-G. Stratigopoulos. "Breaking analog biasing locking techniques via re-synthesis." In: *Proceedings of the 26th Asia and South Pacific Design Automation Conference*. 2021, pp. 555–560.

[75] N. G. Jayasankaran, A. S. Borbon, E. Sanchez-Sinencio, J. Hu, and J. Rajendran. "Towards Provably-Secure Analog and Mixed-Signal Locking Against Overproduction." In: *Proc. 18th Int. Conf. Comput.-Aided Design (ICCAD)*. Nov. 2018.

[76] M. Elshamy, A. Sayed, M.-M. Louërat, A. Rhouni, H. Aboushady, and H.-G. Stratigopoulos. "Securing programmable analog ICs against piracy." In: *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE. 2020, pp. 61–66.

[77] M. Elshamy, A. Sayed, M.-M. Louërat, H. Aboushady, and H.-G. Stratigopoulos. "Locking by Untuning: A Lock-Less Approach for Analog and Mixed-Signal IC Security." In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 29.12 (2021), pp. 2130–2142.

[78] M. Tlili, A. Sayed, D. Mahmoud, M.-M. Louërat, H. Aboushady, and H.-G. Stratigopoulos. "Anti-piracy of analog and mixed-signal circuits in FD-SOI." In: *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE. 2022, pp. 423–428.

[79] S. G. R. Nimmalapudi, G. Volanis, Y. Lu, A. Antonopoulos, A. Marshall, and Y. Makris. "Range-controlled floating-gate transistors: A unified solution for unlocking and calibrating analog ICs." In: *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE. 2020, pp. 286–289.

[80] J. A. Roy, F. Koushanfar, and I. L. Markov. "Ending piracy of integrated circuits." In: *Computer* 43.10 (Oct. 2010), pp. 30–38.

[81] K. Vättö. "Intel to Offer CPU Upgrades via Software for Selected Models." In: *AnandTech*. 2011. URL: https://bit.ly/3AvQMUG (visited on 08/18/2022).

[82] B. Olney and R. Karam. "Tunable FPGA bitstream obfuscation with boolean satisfiability attack countermeasure." In: *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 25.2 (2020), pp. 1–22.

[83] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri. "Security analysis of logic obfuscation." In: *Proceedings of the 49th Annual Design Automation Conference*. 2012, pp. 83–89.

[84] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri. "Fault analysis-based logic encryption." In: *IEEE Transactions on computers* 64.2 (2013), pp. 410–424.

[85] B. Colombier, L. Bossuet, and D. Hély. "From secured logic to IP protection." In: *Microprocessors and Microsystems* 47 (2016), pp. 44–54.

[86] M. Yasin, J. J. Rajendran, O. Sinanoglu, and R. Karri. "On improving the security of logic locking." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 35.9 (2015), pp. 1411–1424.

[87] K. Juretus and I. Savidis. "Reduced overhead gate level logic encryption." In: *2016 International Great Lakes Symposium on VLSI (GLSVLSI)*. IEEE. 2016, pp. 15–20.

[88] N. Limaye, E. Kalligeros, N. Karousos, I. G. Karybali, and O. Sinanoglu. "Thwarting all logic locking attacks: Dishonest oracle with truly random logic locking." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 40.9 (2020), pp. 1740–1753.

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

[90] K. Z. Azar, H. M. Kamali, H. Homayoun, and A. Sasan. "SMT attack: Next generation attack on obfuscated circuits with capabilities and performance beyond the SAT attacks." In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* (2019), pp. 97–122.

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

[92] Y. Shen and H. Zhou. "Double DIP: Re-evaluating security of logic encryption algorithms." In: *Proceedings of the on Great Lakes Symposium on VLSI 2017*. 2017, pp. 179–184.

[93] 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)*. 2016, pp. 236–241. DOI: 10.1109/HST.2016.7495588.

[94] Y. Xie and A. Srivastava. "Anti-SAT: Mitigating SAT Attack on Logic Locking." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 38.2 (2019), pp. 199–207. DOI: 10.1109/TCAD.2018.2801220.

[95] R. Canetti. "Towards realizing random oracles: Hash functions that hide all partial information." In: *Annual International Cryptology Conference*. Springer. 1997, pp. 455–469.

[96] M. Yasin, B. Mazumdar, O. Sinanoglu, and J. Rajendran. "Removal attacks on logic locking and camouflaging techniques." In: *IEEE Transactions on Emerging Topics in Computing* 8.2 (2017), pp. 517–532.

[97] M. Li, K. Shamsi, T. Meade, Z. Zhao, B. Yu, Y. Jin, and D. Z. Pan. "Provably secure camouflaging strategy for IC protection." In: *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. ACM. 2016, pp. 1–8.

[98] X. Xu, B. Shakya, M. M. Tehranipoor, and D. Forte. "Novel bypass attack and BDD-based tradeoff analysis against all known logic locking attacks." In: *International conference on cryptographic hardware and embedded systems*. Springer. 2017, pp. 189–210.

[99] P. Chakraborty, J. Cruz, and S. Bhunia. "SAIL: Machine learning guided structural analysis attack on hardware obfuscation." In: *2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*. IEEE. 2018, pp. 56–61.

[100] F. Yang, M. Tang, and O. Sinanoglu. "Stripped functionality logic locking with Hamming distance-based restore unit (SFLL-hd)–unlocked." In: *IEEE Transactions on Information Forensics and Security* 14.10 (2019), pp. 2778–2786.

[101] D. Sirone and P. Subramanyan. "Functional analysis attacks on logic locking." In: *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 2514–2527.

[102] Z. Han, M. Yasin, and J. J. Rajendran. "Does logic locking work with EDA tools?" In: *30th USENIX Security Symposium (USENIX Security 21)*. 2021, pp. 1055–1072.

[103] M. Yasin, A. Sengupta, M. T. Nabeel, M. Ashraf, J. Rajendran, and O. Sinanoglu. "Provably-Secure Logic Locking: From Theory To Practice." In: *Proc. ACM SIGSAC Conf. Comput. and Commun. Security*. Oct. 2017, pp. 1601–1618.

[104] A. Sengupta, M. Nabeel, M. Yasin, and O. Sinanoglu. "ATPG-based cost-effective, secure logic locking." In: *2018 IEEE 36th VLSI Test Symposium (VTS)*. IEEE. 2018, pp. 1–6.

[105] A. Sengupta, M. Nabeel, N. Limaye, M. Ashraf, and O. Sinanoglu. "Truly stripping functionality for logic locking: A fault-based perspective." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39.12 (2020), pp. 4439–4452.

[106] B. Shakya, X. Xu, M. Tehranipoor, and D. Forte. "Caslock: A security-corruptibility trade-off resilient logic locking scheme." In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* (2020), pp. 175–202.

[107] J. Leonhard, M. Yasin, S. Turk, M. T. Nabeel, M.-M. Louërat, R. Chotin-Avot, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos. "MixLock: Securing Mixed-Signal Circuits via Logic Locking." In: *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 2019, pp. 84–89. DOI: 10.23919/DATE.2019.8715043.

[108] J. Leonhard, M.-M. Louërat, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos. "Mixed-signal hardware security using MixLock: Demonstration in an audio application." In: *2019 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*. IEEE. 2019, pp. 185–188.

[109] J. Leonhard, N. Limaye, S. Turk, A. Sayed, A. R. Díaz Rizo, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos. "Digitally Assisted Mixed-Signal Circuit Security." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41.8 (2022), pp. 2449–2462. DOI: 10.1109/TCAD.2021.3111550.

[110] A. Kahng, J. Lach, W. Mangione-Smith, S. Mantik, I. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe. "Watermarking techniques for intellectual property protection." In: *Proceedings 1998 Design and Automation Conference. 35th DAC. (Cat. No.98CH36175)*. 1998, pp. 776–781. DOI: 10.1145/277044.277240.

[111] F. Koushanfar, G. Qu, and M. Potkonjak. "Intellectual Property Metering." In: *Information Hiding*. Ed. by I. S. Moskowitz. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 81–95. ISBN: 978-3-540-45496-0.

[112] M. Lecomte, J. Fournier, and P. Maurine. "An on-chip technique to detect hardware trojans and assist counterfeit identification." In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 25.12 (2016), pp. 3317–3330.

[113] Y. M. Alkabani and F. Koushanfar. "Active Hardware Metering for Intellectual Property Protection and Security." In: *16th USENIX Security Symposium (USENIX Security 07)*. Boston, MA: USENIX Association, Aug. 2007. URL: https://www.usenix.org/conference/16th-usenix-security-symposium/active-hardware-metering-intellectual-property-protection.

[114] F. Koushanfar. "Provably Secure Active IC Metering Techniques for Piracy Avoidance and Digital Rights Management." In: *IEEE Transactions on Information Forensics and Security* 7.1 (2012), pp. 51–63. DOI: 10.1109/TIFS.2011.2163307.

[115] I. Polian, F. Regazzoni, and J. Sepulveda. "Introduction to hardware-oriented security for MPSoCs." In: *2017 30th IEEE International System-on-Chip Conference (SOCC)*. 2017, pp. 102–107. DOI: 10.1109/SOCC.2017.8226017.

[116] A. Sanabria-Borbon, N. G. Jayasankaran, S. Lee, E. Sánchez-Sinencio, J. Hu, and J. Rajendran. "Schmitt trigger-based key provisioning for locking analog/rf integrated circuits." In: *2020 IEEE International Test Conference (ITC)*. IEEE. 2020, pp. 1–10.

[117] C. Herder, M.-D. Yu, F. Koushanfar, and S. Devadas. "Physical Unclonable Functions and Applications: A Tutorial." In: *Proceedings of the IEEE* 102.8 (2014), pp. 1126–1141. DOI: 10.1109/JPROC.2014.2320516.

[118] J. Sepulveda, F. Willgerodt, and M. Pehl. "SEPUFSoC: Using PUFs for Memory Integrity and Authentication in Multi-Processors System-on-Chip." In: GLSVLSI '18. Chicago, IL, USA: Association for Computing Machinery, 2018, pp. 39–44. ISBN: 9781450357241. DOI: 10.1145/3194554.3194562. URL: https://doi.org/10.1145/3194554.3194562.

[119] C. Marchand, L. Bossuet, and K. Gaj. "Area-oriented comparison of lightweight block ciphers implemented in hardware for the activation mechanism in the anti-counterfeiting schemes." In: *International Journal of Circuit Theory and Applications* 45.2 (2017), pp. 274–291.

[120] A. R. Díaz Rizo, J. Leonhard, H. Aboushady, and H.-G. Stratigopoulos. "RF Transceiver Security Against Piracy Attacks." In: *IEEE Transactions on Circuits and Systems II: Express Briefs* (2022), pp. 1–1. DOI: 10.1109/TCSII.2022.3165709.

[121] A. R. Díaz Rizo, H. Aboushady, and H.-G. Stratigopoulos. "SyncLock: RF transceiver security using synchronization locking." In: *2022 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. IEEE. 2022, pp. 1153–1156.

[122] A. R. Díaz Rizo, H. Aboushady, and H.-G. Stratigopoulos. "Anti-Piracy Design of RF Transceivers." In: *IEEE Transactions on Circuits and Systems I: Regular Papers* (2022), pp. 1–14. DOI: 10.1109/TCSI.2022.3214111.

[123] M. Fyrbiak, S. Wallat, P. Swierczynski, M. Hoffmann, S. Hoppach, M. Wilhelm, T. Weidlich, R. Tessier, and C. Paar. "HAL—the missing piece of the puzzle for hardware reverse engineering, Trojan detection and insertion." In: *IEEE Transactions on Dependable and Secure Computing* 16.3 (2018), pp. 498–510.

[124] IEEE. "IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications." In: *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)* (2016), pp. 1–3534.

[125] T. M. Schmidl and D. C. Cox. "Robust frequency and timing synchronization for OFDM." In: *IEEE Transactions on Communications* 45.12 (1997), pp. 1613–1621.

[126] Nuand. *Open-source IEEE 802.11 compatible software defined radio VHDL modem (bladeRF-wiphy)*. https://github.com/Nuand/bladeRF-wiphy/. Online.

[127] Nuand. *SDR bladeRF 2.0 micro xA9*. https://bit.ly/3z2QV1N. Online.

[128] X. Zhang and M. Tehranipoor. "Case study: Detecting hardware Trojans in third-party digital IP cores." In: *2011 IEEE International Symposium on Hardware-Oriented Security and Trust*. IEEE. 2011, pp. 67–70.

[129] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia. *MERO: A Statistical Approach for Hardware Trojan Detection*. Berlin, Germany: Springer, 2009, pp. 396–410.

[130] 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*. 2013, pp. 697–708.

[131] H. Salmani. "COTD: Reference-free hardware trojan detection and recovery based on controllability and observability in gate-level netlist." In: *IEEE Transactions on Information Forensics and Security* 12.2 (2016), pp. 338–350.

[132] M. Nourian, M. Fazeli, and D. Hély. "Hardware trojan detection using an advised genetic algorithm based logic testing." In: *Journal of Electronic Testing* 34.4 (2018), pp. 461–470.

[133] S. K. Haider, C. Jin, M. Ahmad, D. M. Shila, O. Khan, and M. van Dijk. "Advancing the State-of-the-Art in Hardware Trojans Detection." In: *IEEE Transactions on Dependable and Secure Computing* 16.01 (2019), pp. 18–32.

[134] V. R. Surabhi, P. Krishnamurthy, H. Amrouch, K. Basu, J. Henkel, R. Karri, and F. Khorrami. "Hardware Trojan Detection Using Controlled Circuit Aging." In: *IEEE Access* 8 (Apr. 2020), pp. 77415–77434.

[135] V. R. Surabhi, P. Krishnamurthy, H. Amrouch, J. Henkel, R. Karri, and F. Khorrami. "Exposing hardware Trojans in embedded platforms via short-term aging." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39.11 (2020), pp. 3519–3530.

[136] M. Hicks, M. Finnicum, S. T. King, M. M. Martin, and J. M. Smith. "Overcoming an untrusted computing base: Detecting and removing malicious hardware automatically." In: *2010 IEEE Symposium on Security and Privacy*. IEEE. 2010, pp. 159–172.

[137] A. C. Myers and B. Liskov. "A decentralized model for information flow control." In: *ACM SIGOPS Operating Systems Review* 31.5 (1997), pp. 129–142.

[138] X. Li, V. Kashyap, J. K. Oberg, M. Tiwari, V. R. Rajarathinam, R. Kastner, T. Sherwood, B. Hardekopf, and F. T. Chong. "Sapper: A language for hardware-level security policy enforcement." In: *Proceedings of the 19th international conference on Architectural support for programming languages and operating systems*. 2014, pp. 97–112.

[139] Y. Jin, X. Guo, R. G. Dutta, M.-M. Bidmeshki, and Y. Makris. "Data secrecy protection through information flow tracking in proof-carrying hardware IP—Part I: Framework fundamentals." In: *IEEE Transactions on Information Forensics and Security* 12.10 (2017), pp. 2416–2429.

[140] M.-M. Bidmeshki, X. Guo, R. G. Dutta, Y. Jin, and Y. Makris. "Data secrecy protection through information flow tracking in proof-carrying hardware IP—Part II: Framework automation." In: *IEEE Transactions on Information Forensics and Security* 12.10 (2017), pp. 2430–2443.

[141] M. M. Bidmeshki, A. Antonopoulos, and Y. Makris. "Proof-Carrying Hardware-Based Information Flow Tracking in Analog/Mixed-Signal Designs." In: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 11.2 (2021), pp. 415–427.

[142] K. Xiao, D. Forte, and M. Tehranipoor. "A novel built-in self-authentication technique to prevent inserting hardware trojans." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 33.12 (2014), pp. 1778–1791.

[143] S. Dupuis, P.-S. Ba, G. Di Natale, M.-L. Flottes, and B. Rouzeyre. "A novel hardware logic encryption technique for thwarting illegal overproduction and Hardware Trojans." In: *2014 IEEE 20th International On-Line Testing Symposium (IOLTS)*. 2014, pp. 49–54. DOI: 10.1109/IOLTS.2014.6873671.

[144] K. Shamsi, M. Li, K. Plaks, S. Fazzari, D. Z. Pan, and Y. Jin. "IP protection and supply chain security through logic obfuscation: A systematic overview." In: *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 24.6 (2019), pp. 1–36.

[145] 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*. 2013, pp. 709–720.

[146] Y. Wang, P. Chen, J. Hu, G. Li, and J. Rajendran. "The Cat and Mouse in Split Manufacturing." In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 26.5 (2018).

[147] T. Sugawara, D. Suzuki, R. Fujii, S. Tawa, R. Hori, M. Shiozaki, and T. Fujino. "Reversing stealthy dopant-level circuits." In: *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer. 2014, pp. 112–126.

[148] F. Courbon, P. Loubet-Moundi, J. J. Fournier, and A. Tria. "A high efficiency hardware trojan detection technique based on fast SEM imaging." In: *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE. 2015, pp. 788–793.

[149] 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)*. IEEE. 2014, pp. 19–24.

[150] O. Söll, T. Korak, M. Muehlberghuber, and M. Hutter. "EM-based detection of hardware trojans on FPGAs." In: *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*. IEEE. 2014, pp. 84–87.

[151] X. T. Ngo, Z. Najm, S. Bhasin, S. Guilley, and J.-L. Danger. "Method taking into account process dispersion to detect hardware Trojan Horse by side-channel analysis." In: *Journal of Cryptographic Engineering* 6.3 (2016), pp. 239–247.

[152] J. He, Y. Zhao, X. Guo, and Y. Jin. "Hardware trojan detection through chip-free electromagnetic side-channel statistical analysis." In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 25.10 (2017), pp. 2939–2948.

[153] K. M. Abdellatif, C. Cornesse, J. Fournier, and B. Robisson. "New partitioning approach for hardware Trojan detection using side-channel measurements." In: *International Symposium on Applied Reconfigurable Computing*. Springer. 2016, pp. 171–182.

[154] Y. Tang, S. Li, L. Fang, X. Hu, and J. Chen. "Golden-chip-free hardware trojan detection through quiescent thermal maps." In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 27.12 (2019), pp. 2872–2883.

[155] L. N. Nguyen, C.-L. Cheng, M. Prvulovic, and A. Zajić. "Creating a backscattering side channel to enable detection of dormant hardware trojans." In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 27.7 (2019), pp. 1561–1574.

[156] A. Stern, D. Mehta, S. Tajik, U. Guin, F. Farahmandi, and M. Tehranipoor. "Sparta-cots: A laser probing approach for sequential trojan detection in cots integrated circuits." In: *2020 IEEE Physical Assurance and Inspection of Electronics (PAINE)*. IEEE. 2020, pp. 1–6.

[157] S. Narasimhan, W. Yueh, X. Wang, S. Mukhopadhyay, and S. Bhunia. "Improving IC security against Trojan attacks through integration of security monitors." In: *IEEE Design & Test of Computers* 29.5 (2012), pp. 37–46.

[158] 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)*. IEEE. 2013, pp. 532–539.

[159] A. Pavlidis, E. Faehn, M.-M. Louërat, and H.-G. Stratigopoulos. "Run-Time Hardware Trojan Detection in Analog and Mixed-Signal ICs." In: *40th IEEE VLSI Test Symposium 2022*. 2022.

[160] B. Razavi. *RF Microelectronics, Second Edition*. Mc Graw Hill, 2012.

[161] S. Spiridon, J. van der Tang, H. Yan, H. Chen, D. Guermandi, X. Liu, E. Arslan, F. van der Goes, and K. Bult. "A 375 mW Multimode DAC-Based Transmitter With 2.2 GHz Signal Bandwidth and In-Band IM3 < -58 dBc in 40 nm CMOS." In: *IEEE Journal of Solid-State Circuits* 48.7 (2013), pp. 1595–1604.

[162] A. Sayed, T. Badran, M.-M. Louërat, and H. Aboushady. "A 1.5-to-3.0 GHz tunable RF sigma-delta ADC with a fixed set of coefficients and a programmable loop delay." In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 67.9 (2020), pp. 1559–1563.

[163] M. C. Hansen, H. Yalcin, and J. P. Hayes. "Unveiling the ISCAS-85 benchmarks: A case study in reverse engineering." In: *IEEE Design & Test of Computers* 16.3 (1999), pp. 72–80.

[164] J. Chacko, K. Juretus, M. Jacovic, C. Sahin, N. Kandasamy, I. Savidis, and K. Dandekar. "Physical gate based preamble obfuscation for securing wireless communication." In: *2017 International Conference on Computing, Networking and Communications (ICNC)*. IEEE. 2017, pp. 293–297.

[165] D. Sirone and P. Subramanyan. "Functional Analysis Attacks on Logic Locking." In: *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*. Mar. 2019, pp. 936–939.

[166] "Method for securing telecommunication transceiver integrated circuit designs against piracy, counterfeiting and unauthorized use." PCT/FR2022/050437. A. R. Díaz Rizo, H. Aboushady, and H.-G. Stratigopoulos. Mar. 12, 2022.

[167] A. R. Díaz Rizo. "SyncLock: Synchronization Locking to secure RF transceivers against piracy and unauthorized use." In: *Concours d'innovation i-PhD, BPI France*. 2022. URL: https://bit.ly/3x4NYvz.