Technische Universität München
Lehrstuhl für Sicherheit in der Informationstechnik
an der Fakultät für Elektrotechnik und Informationstechnik

Breaking and Restoring Embedded System Security

From Practical Attacks to Novel PUF-Based Physical Security Enclosures

Johannes Obermaier

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:		Prof. Dr.-Ing. Ralf Brederlow
Prüfer der Dissertation:	1.	Prof. Dr.-Ing. Georg Sigl
	2.	Prof. Dr.-Ing. Rainer Kokozinski

Die Dissertation wurde am 20.03.2019 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 16.09.2019 angenommen.

Johannes Obermaier
mail@obermaier-johannes.de
https://www.obermaier-johannes.de
https://orcid.org/0000-0001-8021-6132

Abstract

Embedded systems play a major role in our everyday life. They are, for instance, deployed in automotive applications as electric control units, in most homes as network equipment or surveillance systems, and in high-assurance systems as Hardware Security Modules (HSMs). However, their security is affected by adversaries who try to tamper with these devices to gain unauthorized access or to perform manipulations. In order to enhance their security, security tests have to be performed on these devices by disassembling and analyzing them. Consecutively, weaknesses have to be discovered, understood, and finally exploited to break security. Based on these results, countermeasures directed at the detected issues must be developed in order to restore embedded system security.

For this purpose, I analyze embedded system security from the attacker's and from the developer's perspective to achieve a holistic approach. The first half focuses on attacks on embedded systems whereas the other half presents physical security enclosures as one viable countermeasure against tampering.

My work starts with a security analysis regarding network security of four selected Internet of Things (IoT) devices by the example of cloud-based video surveillance systems. I define two attacker models, physically disassemble the devices, reverse-engineer the underlying security concepts, and investigate them for design errors. As a consequence of discovering weaknesses in every single system, I derive vulnerabilities affecting those systems' confidentiality, authenticity, and availability. Furthermore, I detected several common issues and anti-patterns that are present among different manufacturers. By creating exploits for the vulnerabilities, I demonstrate that the security of the tested IoT devices is insufficient and may not have been sufficiently considered during system design.

Apart from software issues, embedded systems may also be affected by hardware attacks such as clock glitch attacks. I created a setup for embedded device testing to understand the effects and to enhance the effectiveness of clock glitches via a novel approach. In comparison to conventional solutions, the proposed improvement is based on a random and fuzzy glitch that aims at triggering faults of various types in the system under test without requiring in-depth knowledge about its internals. My work demonstrates through a state-of-the-art microcontroller that the fuzzy glitch indeed causes faults that range, for instance, from computation errors over incorrectly executed branches up to early exit from loops. Thus, these results show that the fuzzy glitch

is a viable solution to attack current commercial microcontrollers via fault injection, thereby affecting embedded system security via this attack vector.

Even if both attack vectors mentioned above, present in software and in hardware, were prevented, further device-level interfaces and security measures would have to be tested for vulnerabilities. For instance, most microcontrollers feature firmware readout protection, which prevents an unauthorized user from extracting the firmware via the device's debug interface. To shed light on this topic, I conducted an in-depth analysis of the STM32F0 microcontroller series. After discovering three weaknesses in its security concept, I developed three exploits for the vulnerabilities. First, I created the non-invasive Cold-Boot Stepping (CBS) approach, which enables the extraction of secret information under certain circumstances. Secondly, I describe a semi-invasive attack downgrading the device's security setting, thereby re-enabling previously disabled debug interface features. Thirdly, I present another non-invasive approach that exploits a race condition which allows an attacker to extract the entire firmware within a few hours or even minutes, cf. `CVE-2017-18347` [MIT18]. Altogether, my results show that the debug-interface of state-of-the-art microcontrollers constitutes a considerable attack vector that could lever out firmware protection and thereby leak intellectual property.

To counteract adversaries depending on physical access, i.e., all the attack approaches presented, physical security enclosures exist that entirely surround the device and prevent direct access. In this work, I analyze several currently available solutions relying on integrity verification based on a conductive mesh as a security enclosure. However, some of these solutions appear outdated and I, therefore, discuss security enclosures which are based on Physical Unclonable Functions (PUFs). I.e., B-TREPID is one such solution that is developed at the Fraunhofer Institute AISEC and is discussed in detail in this dissertation.

In my research, I focus primarily on the measurement aspect of B-TREPID's PUF, which lies in the variation of capacitive coupling between its enclosure's conductive traces. A cryptographic key is generated from the PUF that encrypts the entire system under protection. Thus, if the enclosure is violated, the key and thereby all Critical Security Parameters (CSPs) in the system become inaccessible. To understand the details of the physical enclosure, I present a Finite Element Analysis (FEA) of the proposed enclosure to estimate its electrical properties and the influence of material parameters. Additionally to these results, my measurements show that significant parasitic capacitances up to the nanofarad range are present and the variation in capacitive coupling is only a few femtofarads.

After that, I present the associated measurement system combining integrity verification and PUF evaluation in just one circuit. My novel measurement approach achieves the PUF evaluation via *DFT-based in-enclosure differential capacitance measurement* that directly yields the differential measurement results without requiring explicit subtraction in analog or digital circuitry. I prove the feasibility of my concept via two prototype implementations for two different enclosure technologies. Each enclosure consists of 128 differential capacitive nodes that are successfully tested for integrity and then evaluated as a PUF by my measurement system in less than 50 ms. Finally,

to investigate the system from the attacker's perspective, I execute a physical signal tapping attack targeting the enclosure. Its results show that although the PUF-based solution may provide a high level of security, the enclosure's material properties must be enhanced to hamper physical tampering. Altogether my measurement system provides a building block for PUF-based physical security enclosures by proving the B-TREPID concept to be functional and by demonstrating that capacitive variations in the femtofarad range can be reliably extracted.

To complete the presented concept for physical tamper protection, I developed the Embedded Key Management System (EKMS) concept, which enables easy integration of this PUF-based solution into conventional PUF-agnostic systems by providing a PUF abstraction layer. The EKMS provides a cryptographic Application Programming Interface (API) to the system under protection, which can request operations on the keys derived from the PUF, e.g., to encrypt and sign data. To implement the proposed concept, I extended FreeRTOS, which is a real-time operating system, by additional security features such that it constitutes a basis for the system. The EKMS was successfully tested in practice on the measurement system platform and an enclosure prototype. This proves that PUF-based solutions are conceptually compatible to current embedded devices via my EKMS concept.

Finally, I discuss possible future work for the next steps towards an increased Technology Readiness Level (TRL) of B-TREPID. This takes into account possible future applications, i.e., in automotive or military systems. Altogether, my dissertation covers several successful and novel approaches to break embedded security but then presents a sophisticated PUF-based solution to counteract such attacks.

Zusammenfassung

Eingebettete Systeme spielen eine bedeutende Rolle in unserem täglichen Leben. Sie sind beispielsweise in den Steuergeräten von Fahrzeugen enthalten aber auch in den meisten Haushalten in Form von Netzwerkequipment oder Sicherheitssystemen wie Kameras. Darüber hinaus bilden sie den Kern von Hochsicherheitssystem wie beispielsweise von Hardware Security Modules (HSMs). Jedoch ist ihre Sicherheit von Angreifern bedroht, die versuchen, unauthorisierten Zugang zu erlangen oder anderweitige Manipulationen beabsichtigen. Um ihre Sicherheit zu verbessern, ist es notwendig, eingebettete Systeme zu zerlegen und eingehend bezüglich ihrer Sicherheit zu analysieren. Dabei sollen Schwachstellen aufgedeckt, verstanden, und letztendlich ausgenutzt werden, um die Sicherheit dieser Geräte zu brechen. Basierend auf diesen Ergebnissen sind Gegenmaßnahmen zu entwickeln, welche diese Schwachstellen absichern und somit die Sicherheit der eingebetteten System wiederherstellen.

In meiner Disseratation analysiere ich daher die Sicherheit von eingebetteten System einerseits aus dem Blickwinkel des Angreifer als auch aus der Sicht eines Entwickelers, um eine ganzheitliche Betrachtung zu erreichen. Die erste Hälfte meiner Arbeit setzt den Schwerpunkt auf Angriffe auf eingebettete Systeme wobei die andere Hälfte Gegenmaßnahmen präsentiert die Manipulationen entgegenwirken.

Ich beginne mit einer Sicherheitsanalyse bezüglich der Netzwerkkommunikation von vier ausgewählten Internet of Things (IoT) Geräten, welche cloudbasierte Videoüberwachungssysteme darstellen. Hierzu definiere ich zwei Angrifermodelle, analysiere die Geräte, ermittle die zugrundeliegenden Sicherheitskonzepte per Reverse-Engineering, und untersuche diese auf Designfehler. Nachdem ich in jedem untersuchten Gerät Schwachstellen aufdeckte, entwickelte ich diese zu Verwundbarkeiten weiter, welche die Vertraulichkeit, Authentizität, und Verfügbarkeit der Geräte stark beeinträchtigen. Zusätzlich zeige ich einige gemeinsame Schwachstellen und Anti-Patterns für Sicherheitskonzepte auf, welche über mehrere Hersteller hinweg präsent sind. Anhand den entwickelten Exploits zeige ich, dass die Sicherheit der getesteten eingebetteten Systemen unzureichend ist und vermutlich nicht die notwendige Beachtung während der Geräteentwicklung erhielt.

Eingebettete System sind, abgesehen von den zuvor genannten Softwareschwachstellen, auch von Hardwareangriffen betroffen, beispielsweise per Manipulation des Taktsignals. Hierfür erstellte ich einen Testaufbau für eingebettete Systeme, um einerseits die Auswirkungen dieser Angriffe zu verstehen und andererseits aber auch

die Effizienz von Clock Glitches mit einem neuartigen Ansatz zu verbessern. Das vorgestellte Konzept basiert, im Vergleich mit üblichen Ansätzen, auf einem zufälligen und wirren Clock Glitch Signal, dem *Fuzzy Glitch*. Dieser zielt darauf ab, verschiedene Typen von Fehlern im System zu erzeugen, wobei allerdings kaum Vorwissen über dessen interne Struktur notwendig ist. Meine Experimente anhand eines state-of-the-art Mikrocontrollers demonstrieren, dass der Fuzzy Glitch tatsächlich die erwarteten Fehler auslöst, beispielsweise Rechenfehler, fehlerhafte Ausführung von Sprungbefehlen, bis hin zu einem verfrühten Verlassen von Schleifen. Anhand dessen zeigen meine Ergebnisse, dass der Fuzzy Glitch ein geeignetes Werkzeug für Fehlerangriffe auf Mikrocontroller ist und es somit ihre Sicherheit über diesen Angriffsvektor beeinträchtigen kann.

Für den Fall, dass beide zuvor genannten Angriffsvektoren über Software sowie über Hardware abgesichert wurden, müssen dennoch gerätespezifische Schnittstellen und Sicherheitsmechanismen auf Verwundbarkeiten geprüft werden. Beispielsweise unterstützen die meisten Mikrocontroller Sicherheitsmechanismen, die ein unautho-risiertes Auslesen der Firmware über die Entwicklungsschnittstellen verhindern. Um dieses Thema näher zu beleuchten, untersuchte ich die STM32F0 Mikrocontrollerserie im Detail. Nachdem ich drei Schwachstellen im Sicherheitskonzept aufgedeckt hatte, entwickelte ich drei Methoden, um dies auszunutzen. Der erste Ansatz ist Cold-Boot Stepping (CBS), welches das Auslesen von Informationen unter bestimmten Randbe-dinungen erlaubt. Eine weitere Methode nutzt einen semiinvasiven Angriff, um die Sicherheitskonfiguration des System derart herabzusetzen, dass Entwicklungsschnittstel-len wieder aktiviert werden. Der dritte Ansatz ist hingegen nichtinvasiv und nutzt eine Race Condition aus, welche einem Angreifer die vollständige Extraktion der Firmware innerhalb weniger Stunden oder gar Minuten erlaubt, cf. `CVE-2017-18347` [MIT18]. Zusammengefasst zeigen meine Ergebnisse, dass die Sicherheit von Entwicklungsschnitt-stellen von Mikrocontrollern ein betrachtenswerter Angriffsvektor ist, welcher den Firmwareschutz aushebeln und damit geistiges Eigentum gefährden kann.

Da alle vorgestellten Angriffe physischen Zugang zu den Systemen voraussetzen, lassen sie sich durch eine spezielle Gegenmaßnahme verhindern: Schutzumhüllungen, welche das Gerät vollständig umfassen. Daher analysiere ich im Anschluss verschiedene existierende Lösungen, welche die Systemintegrität mithilfe eines leitfähigen Geflechts überwachen. Jedoch erscheinen diese Lösungen nicht mehr zeitgemäß und ich beziehe Lösungen, basierend auf Physical Unclonable Functions (PUFs), in diese Betrachtung ein. Dabei stellt B-TREPID eine mögliche Lösung dar, die am Fraunhofer AISEC entwickelt und in meiner Arbeit im Detail betrachtet wird.

In meiner dahingehenden Forschung konzentriere ich mich primär auf den Aspekt der Auswertung dieser PUF, welche in der Variation der kapazitiven Kopplung zwi-schen den Leiterbahnstrukturen der Schutzumhüllung liegt. Aus der PUF wird ein kryptographischer Schlüssel abgeleitet, welcher das gesamte System verschlüsselt. Falls die Schutzumhüllung verletzt wird, führt dies zur Zerstörung des darin enthaltenen Schlüssels und alle Critical Security Parameters (CSPs) werden somit unzugänglich. Hierzu stelle ich eine Finite Element Analysis (FEA) der Schutzumhüllung vor, um dessen elektrischen Paramter und den Einfluss von Materialparametern zu ermitteln.

Weitere Messergebnisse zeigen, dass vergleichsweise große parasitäre Kapazitäten im Nanofaradbereich vorhanden sind, wobei die Variation der kapazitiven Kopplung lediglich wenige Femtofarad beträgt.

Anschließend stelle ich das zugehörige Messsystem vor, welches die Integritätsprüfung zusammen mit der Auswertung der PUF in einer einzigen Schaltung vereint. Mein neuer Messansatz realisiert die Auswertung der PUF per *DFT-basierter Umhüllungsinterner differentieller kapazitiver Messung*, welche das differentielle Messergebniss direkt erzeugt, ohne dass explizit Subtraktionen im analogen oder digitalen Teil der Schaltung ausgeführt werden müssen. Ich beweise die praktische Umsetzbarkeit meines Konzepts durch zwei prototypische Implementierungen für zwei unterschiedliche Typen von Schutzumhüllungen. Jede Schutzumhüllung enthält 128 differentielle kapazitive Knoten, welche das Messsystem erfolgreich auf Integrität prüft und anschließend deren PUF innerhalb von 50 ms vollständig auswertet. Im Anschluss daran führte ich, um das System vom Standpunkt eines Angreifers zu betrachten, einen physischen Angriff auf die Schutzumhüllung aus, indem ich entsprechende interne Signale der Umhüllung abgriff. Die Ergebnisse zeigen, dass, obwohl PUF-basierte Lösungen hohe Sicherheit bieten können, die Materialien der Schutzumhüllung verbessert werden müssen, um derartige Angriffe zu erschweren.

Acknowledgements

I want to thank the Fraunhofer Institute for Applied and Integrated Security AISEC for enabling and supporting this work. I am especially thankful for the support by Prof. Dr.-Ing. Georg Sigl, who supervised this dissertation, gave helpful advice, and sparked discussions on the focus and direction of my work. Furthermore, I want to thank the further committee members, Prof. Dr.-Ing. Rainer Kokozinski and Prof. Dr.-Ing. Ralf Brederlow, for supporting my examination.

I want to express my gratitude to my superior, Bartol Filipovic, for providing time and support for all the projects that this dissertation is built upon. Additionally, I want to acknowledge the excellent cooperation with all my colleagues, which have all contributed in their individual way, from being a coauthor, over discussing interesting aspects, up to just bringing joy into every day's work. The space of this work would not suffice to list any and all events.

First of all, I want to express my most profound respect to Vincent Immler who put invaluable efforts into the enclosure project and I want to thank him for our excellent collaboration. Next, I thank Matthias Hiller, who took the role of being my mentor, who guided the security enclosure topics in his group at the institute, and supported the creation of my papers. Additionally, I acknowledge the valuable contributions of my further scientific coauthors at Fraunhofer institutes, in alphabetic order: Florian Hauschild, Martin Hutle, Martin König, Marc Schink, Bodo Selmke, Robert Specht, and Stefan Tatschner. Additionally, I want to thank my further colleagues for fruitful discussions, especially Maxim Hennig, Robert Hesselbarth, Ludwig Kürzinger, Dieter Schuster, and Philipp Zieris. A special thanks goes to Kathrin Garb for the proofreading of the greater part of my dissertation. Furthermore, my students Martin Achtner and Qiyi Li helped me create valuable results. Additionally, I value the support by Michael Hani for supplying decommissioned but helpful lab equipment for some of my offensive security experiments. In addition, I want to thank my parents for considerably supporting my entire education.

A special thanks goes to the institutes, organizations, and their employees who supported my work. The Fraunhofer Research Institution for Microsystems and Solid State Technologies EMFT and also the Fraunhofer Institute for Microelectronic Circuits and Systems IMS provided invaluable capabilities and support throughout the entire COPYCAT project — and beyond. This ranged from manufacturing multiple COPYCAT enclosures, enabling validation of the basic concept, up to creating an

ASIC for enclosure evaluation. The Chair of Security in Information Technology at the Technical University of Munich made valuable research possible by providing the COMSOL simulation environment. Last but not least, I want to especially express my gratitude to Singapore's DSO National Laboratories for the fruitful collaboration on security enclosures that significantly contributed to the results presented in this dissertation.

My work was partially funded via the IUNO project by the German Federal Ministry of Education and Research, under Grant No. KIS4ITS0001, which supported my work presented in Chapter 2. Additionally, this work was supported by the Fraunhofer Internal Programs under Grant No. MAVO 828 432. It allowed me to work on the COPYCAT project detailed in Chapter 5 to 8.

Contents

Chapter 1

Introduction

Today's world is powered by countless embedded devices operating covertly in the background. Users are often oblivious of the existence, capabilities, and internal functionality of such systems — until they stop operating. The reasons are manifold, ranging from inevitable technical defects up to intentional manipulations by adversaries such as malicious hackers who want to repurpose devices for their botnet.

One such awakening took place at the end of 2016 when a variant of the Mirai worm suddenly knocked almost one million German Internet users offline [Kre16]. Criminals tried to take control over Internet routers by exploiting a vulnerability in a publicly reachable interface; however, the unprofessional execution took the devices offline, thereby unintentionally raising attention to the attack. Although quite inconvenient for the users and vendor of the routers, the consequences were minimal as no new botnet was created and no damage was caused to external services.

In contrast to this, the original Mirai botnet comprised several hundreds of thousand devices and had a devastating effect on the availability of some selected web services [AAB+17]. However, the infection of embedded devices with such malware is not evident, thereby leaving users unaware of the threat emerging from their networks towards others and towards themselves. Manipulated devices whatsoever affect all three aspects of security, the so-called CIA-triad, which comprises confidentiality, integrity, and availability. A device infected with malware violates confidentiality, which demands that data is only accessible for the designated entities, as any data processed on the device may be exfiltrated and leaked to an adversary. The same is especially true if access to the local network is viable. This impedes confidentiality also of the data therein if no further protection schemes are in place. Furthermore, data integrity, which requires the data to be unaltered, is also endangered since the malware may manipulate data on the device and possibly other data passing through the network. Finally, availability, which demands the continuous operation of the device and its services, is affected on multiple levels. Malware can affect the correct functionality of the device, may spread across the network infecting other devices which will then fail, or may repurpose the device for Denial-of-Service (DoS) attacks against other systems.

Altogether, securing embedded devices, especially when connected to a network, is of utmost importance.

Even if embedded devices are not interconnected, security may still play a major role. In everyday products, the manufacturer is very interested in protecting the devices' firmware and intellectual property contained therein since there are usually large investments in software developments. Thus, a competitor on the market must not be able to simply clone the device with little effort. Additionally, manipulated devices may be a safety risk, for instance, in automotive systems, or customers may hold the manufacturer responsible for failures caused by unofficial firmware.

1.1 The Need for a Holistic Security Approach

As a countermeasure, a holistic security concept is required that integrates hardware, firmware, and software into the protection concept. Security is one significant aspect of every device and must accordingly be treated right from the beginning of product design and development throughout the entire product life cycle.

For special applications such as high-assurance systems, fulfilling the CIA-triad may not be sufficient. These devices require tamper-evidence, thus, attempts to manipulate the system must result in a measurable and irrecoverable change of properties that will irrecoverably indicate the occurrence of tampering. Such devices must not remain operable if physical manipulations occurred that affected the system's integrity. Thus, tamper-resistance describes the ability of a mechanism to physically prevent manipulations and to initiate countermeasures, such as self-deletion. This is the standard in current high security devices such as Hardware Security Modules (HSMs).

However, increased security of products and services is often accompanied by a reduction in user experience and technical reliability. A trade-off exists between usability and security since increasing security towards a maximum level will usually burden the developers and users with additional tasks and more complexity. Additionally, there is another trade-off between tamper sensitivity and system availability. On the one hand, if a system is highly sensitive, its availability decreases due to a larger number of false alarms. On the other hand, if sensitivity is low and availability is optimized, overall system security, i.e., integrity, confidentiality, and authenticity, is endangered as attacks may go undetected.

Tamper detection is a demanding challenge. When a physical quantity is measured, e.g., the resistance of a conducting mesh that surrounds an embedded device, the system must then decide whether the measurement results hint at an ongoing physical tamper attempt. Thus, there is room for interpretation on how an expected attack looks like and which effects may be attributed to common fluctuations emerging from environmental conditions. Developments have gone into the direction of intrinsically interlinking physical integrity and device functionality. Thus, tampered devices automatically become technically and irreversibly defective, thereby obliterating any secret information stored therein. Thus, if such a concept is implemented correctly, there is very little to no room left for interpretation of measurement results.

Such a security concept may be realized via tamper-resistant Physical Unclonable Function (PUF)-based solutions. A PUF exploits inherent physical manufacturing variations, which emerge from nearly every manufacturing process. After measuring and digitizing this variation, a unique and device-specific cryptographic key is being derived. Manipulations will alter the device's physical properties, modify the PUF, and thereby destroy the device's key. These are topics that are currently being researched in order to create a protection scheme that provides excellent usability while providing high confidentiality, authenticity, and availability. Since these solutions are comparably expensive, they are only feasible for high-security applications where the protection of assets outweighs the additional costs.

Thus, many devices, ranging from IoT camera systems down to standard commercial microcontrollers, are protected via other means that shall provide security that is adequate to their associated threat and attacker model. Thus, my thesis will start with an analysis of such protection technologies and concepts and will then dive deep into the details of novel high-security PUF-based enclosures.

1.2 Contributions and Structure of this Dissertation

Chapter 1 presents background information on the topic of embedded system security. The first half of the dissertation continues with an analysis of existing security concepts, implementations, and attacks thereon. The second half focuses on B-TREPID, a PUF-based physical security enclosure that protects an embedded system in its entirety.

In Chapter 2, I present a thorough security analysis of cloud-based embedded devices by the example of four smart home camera systems. In the analysis, I discovered several common design issues in embedded device security and my results demonstrate that software security is one essential factor in overall system security. However, it is not sufficient for a holistic approach as hardware security must also be taken into consideration. Hence, Chapter 3 demonstrates a hardware attack based on the novel fuzzy glitch and focuses on the glitch resilience of embedded systems. My results demonstrate that apart from software vulnerabilities, an embedded system must also be protected against adversaries with the capability to perform hardware attacks. Nevertheless, even when taking software and hardware attacks into account, system security still depends on the strength of debug interface security of the underlying microcontroller or System On Chip (SoC). Many of them feature a readout protection mechanism that aims at preventing an adversary from extracting and manipulating the firmware. In Chapter 4, I analyze state-of-the-art readout protection mechanisms in a group of microcontrollers and demonstrate by non-invasive and semi-invasive attacks that this feature is often not as secure as specified. Altogether, these vulnerabilities in software and especially in hardware demonstrate that secure system engineering might still not suffice to achieve satisfactory overall system security, as major parts of the system are out of the developer's control.

To overcome this issue, physical security enclosures provide additional and strong protection for high-assurance embedded systems. In Chapter 5, I present a survey on past, present, and future solutions for physical security enclosures. After analyzing the technology of past solutions, I conclude that they may not be able to keep up with today's and especially the future's attack tools; thus, there is a strong need for progress and a new generation of security enclosures. I fill this currently existing gap with a PUF-based solution. For this purpose, I present B-TREPID that is a novel PUF-based security enclosure that exploits capacitance variations in a conductive mesh embedded in the enclosure. The development of this solution has proven to be very demanding; hence, I focus on three central aspects.

First, I outline the general concept, describe its design, and provide a simulation model thereof in Chapter 6. Following an introduction of the concept, the Finite Element Analysis (FEA) presented therein investigates the influence of enclosure material parameters on the capacitive characteristics.

I cover this aspect in detail in Chapter 7, which comprises the enclosure integrity verification and the measurement of the PUF, which emerges from minuscule capacitance variations in the femtofarad range. I present a measurement system that combines all the required functionality in one circuit: It can verify the enclosure's integrity and evaluate the entire 128-node enclosure PUF in 50 ms. My results demonstrate that the overall security system can work in practice and that my concept is well suited to extract capacitance variations reliably.

System integration is the third aspect since combining a PUF with an existing system is accompanied by additional efforts such as PUF readout, PUF data processing, and key management. As a consequence, I developed the Embedded Key Management System (EKMS), presented in Chapter 8, that provides a PUF abstraction layer for the concept and thereby eases system integration. To implement such a system in practice, I extended the real-time operating system FreeRTOS by additional security features. Based on this platform, I show that the EKMS provides an adequate interface between the PUF and the system under protection but without significantly increasing its complexity.

Finally, I sum up the outcome of my thesis and give an outlook into possible future developments in Chapter 9. I discuss several open challenges that have to be resolved for a higher Technology Readiness Level (TRL) of B-TREPID.

Chapter 2

Analyzing the Security and Privacy of Cloud-Based Video Surveillance Systems

This chapter covers an analysis of typical Internet of Things (IoT) devices of the year 2015. During this time, especially cloud-based camera surveillance systems became more and more ubiquitously available for industrial and private environments. However, the sensitive nature of the surveillance use case imposes high requirements on security, i.e., confidentiality, authenticity, and availability of such systems — whose need for correct realization has often been emphasized [FWKM15, ZCW+14, MYAZ15]. In this work, I investigate whether the underlying security concepts of four mass-market camera systems can, in fact, comply with these requirements. Considering two attacker models, I evaluate the concepts for theoretical weaknesses, check whether the concepts have been correctly implemented, and analyze the impact of possible vulnerabilities. For this purpose, I manually reverse-engineer the security concept and its implementation, thereby discovering numerous vulnerabilities in every tested camera system. I show that those weaknesses impair all three components of the users' security and, as a consequence, have the power to damage the camera system manufacturer's reputation. I demonstrate how an attacker can exploit these vulnerabilities in order to blackmail users and companies by denial-of-service attacks, injecting forged video streams, and by eavesdropping private video data — even without physical access to the device.

Parts of this chapter have previously been presented at the *2nd ACM International Workshop on IoT Privacy, Trust, and Security (IoTPTS)* in 2016 and were published in its corresponding proceedings [OH16]. This publication was created in cooperation with Martin Hutle, who assisted in developing the general storyline, improving the results, and supported writing the paper. Please note that all results were shared with the affected manufacturers via a cooperative disclosure process.

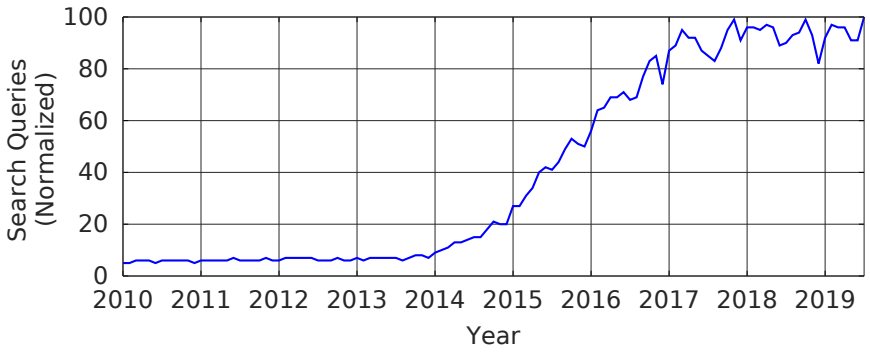


Figure 2.1: According to the number of search queries, the interest in *Internet of Things* began skyrocketing in late 2014. Data source: Google Trends for "IoT" [Goo19]

2.1 Overview

The interest in the *Internet of Things* has skyrocketed in 2015, as displayed in Figure 2.1. Numerous devices have since populated the emerging field of cloud-based video surveillance systems. Decreasing costs and the growing availability of embedded devices enabled Smart Home and Home Automation systems. For instance, typical applications are video surveillance systems for building security. These devices commonly feature a connection to a cloud server or other data relays so that the user has access from all over the world. However, these new possibilities are accompanied by new threats [KLMR04], for the user and for the company.

2.1.1 Related Work

Cameras are often employed in spaces where privacy must be guaranteed. Users acquire those devices for personal use and place them inside their homes, thereby granting them broad access to their daily life, habits, and most private situations. As shown by surveys, users are indeed concerned about preserving their privacy [Del15] and fear unauthorized manipulation [GfK15], especially in cloud-based services. Therefore, the user's expectations of manufacturers concerning data privacy, integrity, and system availability are high. This includes that the users are protected from eavesdropping, that their video streams cannot be manipulated, and that system operation remains uninterrupted [SP08]. Failing to meet these requirements has a severe impact as this is not only disadvantageous for the user but also threatens the manufacturer of the camera system. Both, the user and the manufacturer, have to fear potential blackmailing and loss of reputation.

Manufacturers provide very little information on how a system communicates with its cloud server and whether the connection is well-secured. The security concepts are usually not standardized, are mostly proprietary, and undisclosed, hereby rendering public reviews impossible. Moreover, there is no guarantee that any well-designed

security concept has been deployed on the devices at all, not to mention a technically sound implementation.

The topic of security in surveillance systems received a lot of attention but has seldom been verified for real applications. In 2009, Vagts and Beyerer stated several challenges, for example, data privacy, and conclude that "the issues must be addressed right from the beginning" [VB09]. They emphasize "security by design" and "privacy by design". Winkler and Rinner give an overview of several implementations of data-centric security approaches [WR14]. The concepts are compared regarding their ability to provide integrity, authenticity, data freshness, confidentiality, and related aspects of security. In 2015, they published a prototype for secure embedded visual sensing [WR15]. The work focuses on end-users and especially takes cloud storage of video data into account. There are several proposed methods and concepts, whose realization in currently available end-user devices is unclear and needs further investigation.

For embedded devices in general, automated large-scale firmware analysis has been conducted [CZF+14, CZF16]. This approach is able to extract password hashes and RSA keys from firmware, for example. But these tests only cover specific patterns but may not detect critical conceptual errors or flaws in the system's design.

2.1.2 Contributions

In contrast to the aforementioned automated analysis on a large number of devices, I perform a deep evaluation of selected camera systems. Furthermore, I also reverse-engineer the underlying security concept and investigate the implications of identified weaknesses. All in all, my contributions include:

- An attacker model for cloud-based video surveillance systems.
- An analysis of the security of the *local* network interface.
- An analysis of *remote* attacks on the camera devices and their users.
- A practical verification of the vulnerabilities.
- A discussion of common design issues discovered across different manufacturers.

Related subjects, such as cloud server penetration tests, are not considered in my work. Due to the nature of this work — it was not supported or endorsed by the manufacturers — such tests would most likely cause legal issues.

2.1.3 Structure

In this chapter, I analyze the quality of the employed security concepts in four cloud-based video surveillance systems. After an introduction to the topic of IoT security in Section 2.1, I present the methods and selected devices for analysis in Section 2.2. My approach starts with an analysis of the device hardware and its network services in Section 2.3. It is followed by reverse-engineering the cloud server communication and underlying security concepts. Then, each implementation is assessed regarding its ability to provide *privacy*, *integrity*, and *availability* in Section 2.4. Based on these

results, Section 2.5 discusses the weaknesses and highlights common anti-patterns in IoT security. In Section 2.6, I discuss my results from 2015 concerning recent developments, i.e., the discovery of the Mirai botnet, and I discuss envisaged IoT security regulations. Finally, Section 2.7 presents a summary and gives an outlook into the possible future of IoT security.

2.2 Disclosure Process, Materials, and Methods

The section starts with a discussion of the chosen method for the disclosure of results. Next, the section describes the selection of camera systems, the technical approach to analyzing them, and defines two attacker models.

2.2.1 Coordinated Disclosure

During the analysis, the ethical question emerged whether and to which extent the results should be published. While there are advocates for *full disclosure* [Sch07], others criticize such an approach and endorse *coordinated disclosure*, also known as *responsible disclosure*, or even demand *non disclosure* [Was16, She03]. In the first case, all results are immediately shared with the broad public, the second method concedes several weeks for the manufacturer to fix the issues before publication, and in the last case, nothing is published at all. After careful consideration of options and possible consequences, the decision fell onto a tailored coordinated disclosure approach, which publishes the results after informing the manufacturers, but without naming them explicitly and redacting some information. The decision is based on a manifold of reasons.

When my analysis started in 2015, innumerable IoT camera systems were available on the market and their number was almost increasing daily. Due to the quick developments, the vast number of systems, and the high effort for an in-depth security analysis, covering the entire market even for this single year is practically impossible. Hence, the analysis had to be limited to a small but carefully selected subset. However, this raises the questions, whether it would be fair to blame or praise a few manufacturers that were (un)lucky that their devices were chosen. Additionally, since all tested systems were affected by at least one vulnerability, this might lead someone to the wrong conclusion that this is an exhaustive analysis and all other non-listed devices are secure or insecure.

The publication of vulnerabilities with explicitly naming manufacturers can have unforeseen consequences on them. How such a publication is perceived is out of the author's control, especially if the media tries to convert a fairly complex topic into a short and widely understandable article. These aspects can lead to over-simplification or exaggeration that may have an unintendedly strong negative impact on a manufacturer up to a loss of trust. The concern is once more emphasized by the fact that only devices of a few manufacturers are tested.

The effect on users who unwittingly bought an insecure device must also be taken into consideration. Unfortunately, in the case of home surveillance camera systems, users will be most affected by security vulnerabilities. Those devices may have direct access to their most private moments and in combination with vulnerabilities concerning eavesdropping video streams, this can have disastrous consequences. Especially since several manufacturers showed only very little interest in fixing the vulnerabilities, these problems may persist permanently. This is the strongest proponent against detailed disclosure as it would heavily expose users.

Nevertheless, publishing the results is of major importance. Personal experience has shown that manufacturers are reluctant when it comes to fixing vulnerabilities and instead try to sit it out. The publication of results, however, inhibits such behavior. A publication proves that vulnerabilities were discovered, that the manufacturer was informed, and had the possibility to fix it. This exerts pressure and has, in fact, led to a resolution of such issues.

Even without informing the manufacturers and enabling them to fix their system, the risk persists as vulnerabilities are often rediscovered [Ozm05, AB17] and may be maliciously exploited. This fear is not far fetched, as vulnerabilities that have been described in my publication [OH16] were identically discovered afterward also by other researchers [LLX+17, LLX+18].

The most important step is to inform the manufacturers early and to the full extent. Time frames of a few months, such as 45 days [Man18] or 90 days [EHA+15] are common and each manufacturer was informed by me more than five months prior to the publication.

However, there is a balancing act between publishing sufficient information to convey the vulnerability but still preventing too easy exploitation. With this in mind, I decided to redact the following information:

- **Manufacturers:** As previously stated, the manufacturers' names are withheld to prevent negative consequences on them or on their users.
- **Device names and related information:** The same is true for the name of the device, however, also related information such as Media Access Control (MAC) addresses that can easily be related to a manufacturer, are removed or replaced with `01:23:45:67:89:AB`.
- **Passwords and keys:** Passwords or keys were often shared among several devices. The knowledge of the exact value has no effect on the scientific results but would enable easy exploitation of the vulnerability. Accordingly, they have been substituted with placeholders such as `[...]`.

Even if an adversary discovers the analyzed systems, they will not gain a headstart therefrom since they have to go through most of the time-consuming reverse-engineering process on their own. Altogether, these precautions enable the discussion of real-world security concepts without endangering manufacturers or users and some of the vulnerabilities will already be resolved at the time of publication.

2.2.2 Selection Criteria for Camera Systems

The market of IoT camera systems is large, but only a small fraction of systems can be analyzed in detail due to the high effort required. As a consequence, I chose a set of systems that represent typical devices at the time of the study in 2015. The price of a system might have an influence on its level of security; thus, the selection covers low-cost solutions up to high-priced devices. Several additional technical criteria were defined that the selection of specific devices is based on:

- **Cloud-based streaming and storage:** The device is connected to a cloud service and streaming, as well as the storage of captured videos, if available, is managed externally. Hence, the camera is not accessed directly over the local network but primarily using the provided cloud service, i.e., using a web browser or a mobile phone application.

- **Remote access:** A connection into the local network, e.g., via Virtual Private Network (VPN), is not required for accessing the camera stream, all operations are handled by the cloud service. This means that the device is accessible from all over the world but should be protected by adequate means.

- **Standalone system:** The system should not be part of a more extensive setup and must be able to work on its own, only power and network connectivity into the Internet need to be provided. This excludes professional, large-scale surveillance systems since this analysis focuses on end-user appliances.

2.2.3 Selected Camera Systems

After careful consideration, the following systems were selected. All prices in this section were quoted in mid-2015.

Camera A is a medium-priced camera system in the range between 100 € and 200 €. It supports cloud storage of captured videos and features motion detection, including alerts to the user. The system is either accessed over the web browser or the mobile phone application. The device can be exclusively accessed via the cloud service and local access is not envisaged.

Figure 2.2: Camera A.

Figure 2.3: Camera B.

Camera B is a highly-priced camera system a little above 200 €. Its appearance is widely similar to camera A, but the manufacturers denoted on the devices differ and do not seem to be related. Hence, this system is of particular interest regarding how similar the security concepts and implementations are. The features such as cloud storage, online access, and recording are mostly the same for both systems. This device can also be exclusively accessed via the cloud service without support for local usage.

Figure 2.4: Camera C.

Camera C is a camera system in the price segment below 100 €. The cloud concept is widely reduced, compared to the other systems. It is based on a relay-like cloud infrastructure that is partially involved in stream transmission but does not support storage of recorded video data. While the main use case of the system is remote viewing using the web browser or the mobile phone application, this camera was designed to be also accessible from the local network.

Figure 2.5: Camera D.

Camera D is another camera system of the price segment below 100 €. Similar to Camera C, it only supports cloud-based viewing but no recording of video streams. The main use case of this system is also remote viewing and access from the local network is possible.

2.2.4 Attacker Model

For the security analysis, I define two attacker models of a broad scope. This follows from the nature of IoT devices that may encounter all kinds of attackers due to their

worldwide reachability. The first attacker is named *local attacker* and the second one is denoted as *remote attacker*.

The **local attacker** has access to the local network to which the camera device is also connected. The attacker can communicate with other devices on the same network as a usual node without any restrictions. There is no Network Address Translation (NAT) taking place when communicating with the camera device over the local network. This resembles a typical network configuration in private homes or small businesses. The attacker might be inside the building or access the network from the close vicinity via an accessible Wireless Local Area Network (WLAN). However, there is no physical access assumed to the camera device under attack. Such an attacker is represented, for example, by a malevolent employee or guest.

In contrast to this, the **remote attacker** cannot access the local network but is equipped with an Internet connection. Hence, the attacker is unable to communicate with camera devices directly but can reach the cloud servers. No other limitations apply; thus, the scope of this attacker is very broad, as it matches anyone with an Internet connection anywhere in the world. However, I extend the typical remote attacker by the ability to optionally observe *encrypted* wireless network traffic. While sometimes this ability is attributed to the local attacker, this is considered in the remote attacker model, since no access to the premises is necessary. The extension allows more sophisticated and targeted attacks in close vicinity of wireless camera devices without the need for physical access to the device, building, and internal network.

Both attackers are assumed to be sufficiently experienced in testing embedded devices. They are familiar with the usual procedures and tools to conduct, for example, Man in the Middle (MitM) attacks, hardware manipulations, and software reverse-engineering. They have access to a simple electronics lab, equipped with standard tools such as a soldering iron, screwdrivers, serial interface adapters, and common consumables. They might have a professional background but do not have any abilities typical for state- or military-grade attackers. Thus, attacks requiring faked certificates created by any illegitimate access to a widely deployed Certificate Authority (CA) are explicitly excluded. The financial means are assumed to be limited; thus, a few IoT devices can be acquired but not dozens of them. The same is true for the computational effort that must not exceed the possibilities provided by a standard computer system, e.g., factorization of large RSA keys is excluded.

These two attacker models were defined to create a rather broad adversary. At no point, special qualification or equipment is needed; thus, this attacker model applies even to many hobbyist level engineers, some motivated students, or basically a vast number of people. By choosing this model, I prove that IoT security in 2015 is alarmingly simple to break with a massive impact without requiring the involvement of any highly-trained professional in a high-tech lab.

2.2.5 Technical Approach

Both attacker models comprise an identical technical approach. By acquiring and analyzing *a single* device, the attackers try to gain sufficient knowledge about security

weaknesses and try to find ways to exploit them. In the next step, these exploits are generalized in order to attack *all other* camera devices of this manufacturer and of this type.

For the analysis, I assume a black-box test setup, in which each attacker has little to no initial information about the system; hence, they both spend the majority of their effort on reverse-engineering. The attackers' analysis first aims at reverse-engineering the systems' inner workings, their security concepts, and at understanding their way of implementation. For this purpose, two methods, targeting the firmware of the system and the communication interfaces, are employed: Firmware analysis and traffic interception.

While traffic can be easily assessed since it has to leave the device on its way into the Internet, the firmware remains hidden inside and access must be gained at first. During the initial phase of my analysis, each device is carefully disassembled to collect basic information and to discover potential physical attack vectors for further analysis. I chose this approach since end-user products are usually not protected against physical tampering unless explicitly noted. Although devices are often designed to be challenging to open and an attempt may damage its shiny casing, this is acceptable as long as this has no impact on the device's *technical* functionality.

In the first step, the casing is unscrewed or pried open, and then the circuit boards are removed. Secondly, the main system components, i.e., the SoC, SDRAM, flash memory, and camera module, are identified and datasheets are obtained, if possible. Thirdly, the boards are further inspected for footprints of optional components, debug connectors, and comparable interfaces. The next step is individual for each device and is taken depending on the previous results, e.g., memory chips may be desoldered and debuggers are attached. The obtained firmware images are disassembled and analyzed using the reverse-engineering framework *radare2*, which supports a multitude of architectures.

For the second approach of my analysis, traffic is analyzed and can be modified by an interposed Linux computer system, which acts as a router. For packet recording and forwarding, the tool *Wireshark* and the kernel module *iptables* are employed. The same setup is able to optionally manipulate and redirect traffic to alternative endpoints such as a local cloud-server for MitM experiments.

Although this facilitates a quick look on weakly secured systems, the information gained by traffic analysis may be rather limited if the connection is encrypted. In contrast to this, disassembling firmware reveals the most profound details; however, it comes with the highest effort. Thus, both approaches were not followed separately but in conjunction with each other for best efficiency.

A shortcut in these tasks can be taken by gaining *root* access to the system, which is equivalent to obtaining the highest privilege level on the system. It gives instantaneous access to all the system's configuration, its firmware, to network interfaces, and often to communication data before encryption. Achieving this can significantly shorten the required time for reverse-engineering, especially since a static code analysis on an

extracted firmware image can be replaced with a dynamic analysis on the running system.

Finally, all the obtained information is combined for a thorough security analysis and any findings are investigated for their implications. The results are especially evaluated not only regarding the device's own security but with respect to possible non-technical threats emerging for the manufacturer and for the user.

2.3 Camera Device Security Analysis

For the analysis, I initially disassembled all four camera systems to get hardware access to the system. By dissecting the firmware and analyzing network communication, I gain more information about the inner working of the system and potential attack vectors. These results are then combined to develop powerful exploits to nullify system security.

2.3.1 Hardware Disassembly and Analysis

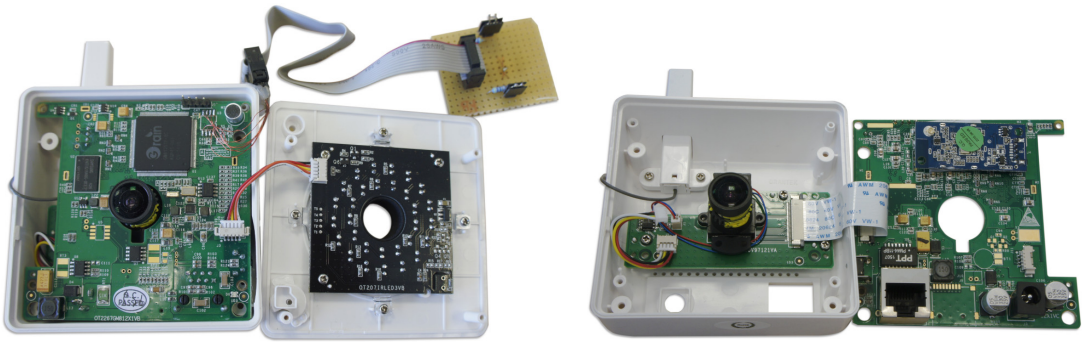
For a first analysis, I disassemble each device. This aims at gaining general knowledge about the system, finding potential attack vectors such as debug interfaces, and getting access to the firmware. The approach for all four systems is similar and only the first one is described in detail since it was the most effortful. The firmware of each system is based on the Linux kernel; thus, also the analysis on the software side can be considered to be analogical.

2.3.1.1 Camera A and B

Camera A and B are based on identical hardware. Figure 2.6 shows the disassembled camera with some of my modifications for analysis. The system comprises three circuit boards. The black quadratic board in Figure 2.6a contains a light sensor and infrared Light Emitting Diodes (LEDs) for night-vision. The innermost board, which is depicted in Figure 2.6b on the left, is entirely dedicated to the camera module and its optics assembly. The largest board is the embedded device, which incorporates a SoC, SDRAM, flash memory, a wired and wireless network interface, and voltage converters for power supply. Thus, the device's IoT capabilities entirely stem from this board while others can be ignored in the security analysis.

The system is based on the GM8125 [Gra11] SoC by Grain Media, which is the largest chip in the upper region of the board in Figure 2.6a. It comprises an FA626TE ARM9 core, which is a 32-bit processor that runs at up to 540 MHz. Furthermore, the SoC features a parallel interface for camera image sensors, hardware video encoders for H.264 and MPEG4, Direct Memory Access (DMA), and crypto accelerators for the Advanced Encryption Standard (AES), Data Encryption Standard (DES), and 3DES.

The SoC contains an Ethernet media access control module and can directly utilize such an interface using only a transformer. For wireless connectivity, an additional board is permanently attached to the bottom side of the SoC board. The blue-colored



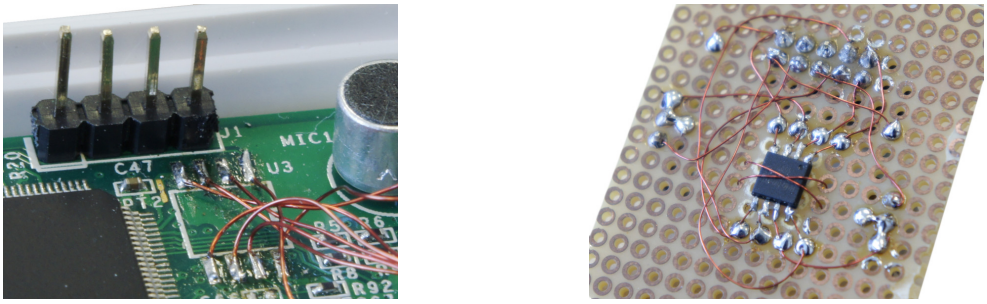
(a) Modified main PCB with external flash memory.

(b) Camera (left) and main PCB (right).

Figure 2.6: The disassembled and modified Camera A/B.

board is visible in Figure 2.6b in the upper region. It looks similar to a standard WLAN dongle but is directly soldered to the SoC's Universal Serial Bus (USB) interface.

The SoC has 128 MiB of external Synchronous Dynamic Random-Access Memory (SDRAM) available. It is implemented as a Samsung K4T1G164QG memory IC which is placed left of the SoC. For non-volatile storage of data, the device contains a Winbond W25Q64 serial flash memory with a capacity of 8 MiB [Win16]. This component resides on the circuit board next to the top right corner of the SoC.



(a) UART interface (top) and wiring to external flash memory board (bottom).

(b) Flash memory IC on a detachable external board.

Figure 2.7: Modified main PCB for easier analysis.

In this region, between the SoC and the upper board edge, an unpopulated 4-pin 2.54 mm pitch footprint is present. Thus, a pin header was soldered in to access the signals, as depicted in Figure 2.7a. This location, pitch, and pin count is common for Universal Asynchronous Receiver Transmitter (UART) debug connectors. Usually, there is one pin for TX, RX, GND, and VCC. Even though there is no standard pin assignment, a multimeter and oscilloscope suffice to associate each pin with its function easily in a few minutes. Alternatively, the FCC ID database contains detailed technical

information about a very similar device, including the UART connector pinout on a schematic [FCC12].

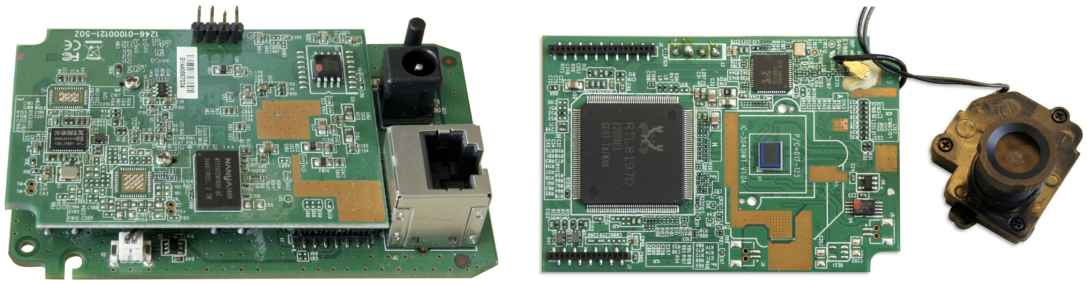
The connection gives access to a serial console on the device with the experimentally detected baud rate of 115.2 kBd. Although the interface allows watching the startup procedure of the device and witnessing debug information while running, any further access is blocked by a password.

To gain access to the firmware nevertheless, I desoldered the 8-pin flash memory IC, which is placed in close vicinity of the debug connector, both shown in Figure 2.7a. Wires were soldered to the footprint on the Printed Circuit Board (PCB) and connected to an extension cable. The flash memory IC was then placed on a detachable board, see Figure 2.7b, which enables an easy readout of the memory. Please note that this is an entirely unprotected memory chip without any hardware security features. For an overview of the whole modified system, refer to Figure 2.6a that shows the extension cable and the yellow external flash memory board attached to the device.

Next, the firmware is analyzed. For this purpose, the flash memory is read out, the partitions are extracted, and the file systems, comprising the operating system data, the application binaries, and configuration files, are unpacked. Although the main partition appears to contain a CRAMFS file system, unpacking fails. The inodes are in an incompatible format and each one is 4 bytes larger than specified. I adjusted the filesystem processing to take this into account, and hence unpacking became successful. All firmware binaries and configuration data became accessible for analysis.

2.3.1.2 Camera C

Camera C follows a similar modular hardware architecture as it consists of two stacked boards, depicted in Figure 2.8a. The lower board comprises the power and Ethernet connector, as well as some supplementary electronics such as voltage converters and infrared LEDs. The upper board contains the SoC and implements the IoT functionality; thus, the further analysis focuses exclusively on it.



(a) Backside of the system's PCBs. **(b)** Frontside including camera module.

Figure 2.8: The disassembled and slightly modified Camera C PCBs.

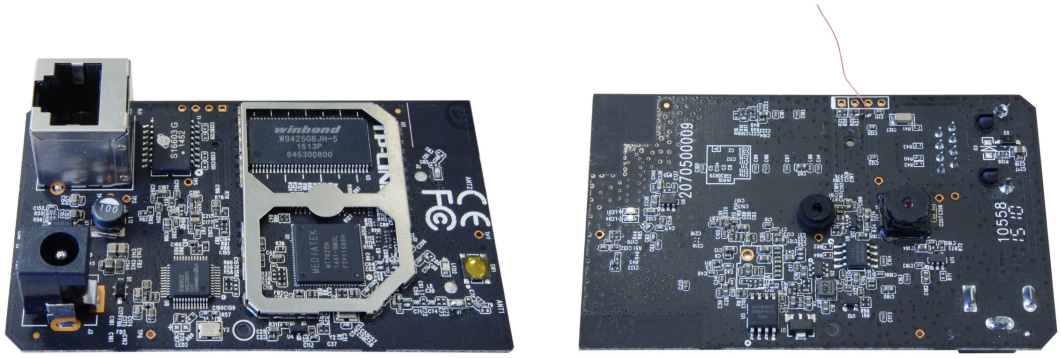
The top side of the SoC board is shown in Figure 2.8b. The optics assembly has been taken off to expose the camera sensor. Apart from it, there is the wireless network chip near the upper edge. All this is controlled by the Realtek RTL8197D SoC, which is the largest chip and resides in the left half of the board's front side. The SoC contains an RLX5281 MIPS core running at up to 660 MHz [Rea18]. According to its datasheet, it was designed to be used in a network switch; hence, there are no camera interfaces or video encoders in hardware. Therefore, the image sensor, which is slightly right to the center of the board and has a light blue tint, is not connected to the SoC directly. Its signal is preprocessed by a SONIX SN9C291A camera-to-USB controller chip, which is placed near the left edge on the backside of the board in Figure 2.8a.

The other ICs on the backside are the SDRAM and the flash memory. The NANYA NT5TU32M16 SDRAM IC is on the lower edge of the board and comes with 64 MiB of memory. The Macronix MX25L64 flash memory IC provides 8 MiB of non-volatile storage. It is located in the upper right corner and marked with a red dot.

A typical 4-pin 2.54 mm pitch debug-connector footprint is located on the upper board edge, as visible on both sides of the board. I soldered in a pin header and gained instantaneous root access to the device. This access allows arbitrary read and write commands using the available root shell, as well as monitoring the startup process and debug output while running. Thus, desoldering the flash memory IC is not necessary in this case.

2.3.1.3 Camera D

Camera D employs a single-board design. Most components are located on the backside of the board, shown in Figure 2.9a. The SoC and the SDRAM are both shielded by a metal High Frequency (HF) casing, which has been partially removed during analysis.



(a) Backside without the removed HF casing. **(b)** Frontside of the PCB.

Figure 2.9: The disassembled Camera D PCBs.

The chip in the lower region is the MediaTek MT7620 SoC, which features a 580 MHz MIPS core [Med12]. Additionally, it contains a wired network interface and an integrated WLAN interface. The SoC was designed as "router-on-a-chip", according to its datasheet [Ral12]; thus, it comprises extended networking capabilities in hardware but cannot interface a camera sensor. Hence, the system has a SONIX SN9C258 camera-to-USB controller chip, which is placed left to the SoC. The system has access to 32 MiB of SDRAM, provided by the Winbond W9425G6 chip above the SoC.

Flash memory is present at the lower edge on the front side of the system, shown in Figure 2.9b. The Winbond W25Q64 contains 8 MiB of non-volatile memory [Win16]. This side of the board comprises the camera sensor, its optics assembly, and a microphone.

The typical 4-pin 2.54 mm pitch debug-connector footprint is present on the upper board edge. Unfortunately, no debug interface is available there. Two resistors in close vicinity to the footprint are intentionally left out to disconnect the UART signals, which "disables" the debug interface thereby — a security measure that was discovered in a few minutes and circumvented with a simple soldering iron. After fixing this issue, root access was gained to the camera and the memory contents were extracted; thus, the flash memory was not required to be desoldered.

2.3.2 Local Network Interface Security

After the firmware has been read from every device and at least partial access to the debug interface has been obtained, the analysis continues on the *local* network interface. An *Nmap* port scan reveals that all devices are reachable via a web server running on port 80 (HTTP). Some devices offer additional local services, e.g., camera D on port 8080 for video streaming. For camera C and D, these are well-documented features as they offer local network access. Both systems are protected against unauthorized local access by requiring user-configurable login credentials. However, port 80 is unexpectedly open for camera A and B and requires further investigation.

2.3.2.1 Weak Factory Interface Protection of Camera A and B

When the user tries to access the camera on port 80 via a web browser, a login prompt appears; thus, this is indeed a web interface. However, it is not meant to be accessed since the user was not informed about the login credentials and the interface's existence is not mentioned in the manual. Trying out common combinations of usernames and passwords was not successful. As a consequence, I searched the previously extracted firmware for login credentials. I discovered the username "admin" and a corresponding password in the firmware and was able to log into the web interface.

The password appears to be of high quality as its length is 36 characters and contains characters of mixed case and numbers. Furthermore, there is an *individual* password for each device, since the password only worked on this single device but not on others. Theoretically, such a password is considered sufficiently secure; however, it turned

out to be very weak. The password is a Base64 encoded string that can be trivially converted into its original format and reverse engineering the password generating function is easy. I found out that the Base64 function is applied to the concatenation of two constant strings with the upper-case and reversed MAC address of the camera's network interface. For a device with the exemplary MAC address `01:23:45:67:89:AB`, the password is `base64encode([...]BA9876543210[...])`. With this knowledge on password generation, an attacker can compute the password for *any* camera, given its MAC address without requiring physical access to the device or its firmware. While in the same network, the MAC address of any device is trivial to obtain using passive listening to network traffic. Thus, the local attacker gains access to any camera's undocumented factory interface on the local network.

2.3.2.2 Missing Factory Interface Input Sanitization in Camera A and B

Camera A and B support mounting an external Server Message Block (SMB) share via its factory interface. The interface has an input field to enter an IP address and path to the share. This was originally intended for an external server where images can be stored.

However, the interface was investigated further and a critical vulnerability was discovered by observing the debug output. When a network drive is mounted, the camera executes `mount.cifs [REMOTE] /home/disk`, where `[REMOTE]` is the user's input. Unfortunately, this input is not sanitized and is therefore vulnerable to command injection. Additionally, the corresponding process runs at the highest privileges, i.e., under the root account, and has full system access. Altogether, this vulnerability can be exploited by the local attacker to gain full device access.

2.3.2.3 Local User Interface of Camera D

I tested the login prompt of the streaming interface on port 8080 of camera D regarding its robustness. When inserting very long usernames of several kilobytes, the camera service tended to crash. This might be a hint on a failed boundary or length check for user input data that might be exploitable. Due to the high effort required and because it is not related to cloud operation, the analysis was not continued in this direction.

2.3.3 Analysis of Communication Encryption

In the next step, the connection between the camera and the cloud service is evaluated. This focuses on the security of the communication protocols, especially regarding their encryption and authentication algorithms.

The first step of the analysis comprises the identification of the connection types and the mechanisms for securing them. Herefore, the traffic between each camera and its cloud is monitored and analyzed with Wireshark. Thereby I observed that all four cameras employ the following three basic types of connections.

The *initial* connection establishes the first communication with the cloud servers. In addition, the connection transports camera-is-online packets for each of the four cameras and also motion-detection notifications for camera A and B. The connection is secured for every camera system but with different means. Camera A and B establish a Transport Layer Security (TLS) 1.2 channel, camera D supports Secure Sockets Layer (SSL) 3.0; thus, standard protocols are employed. In contrast to it, camera C relies on a non-standardized method with a proprietary encryption algorithm and protocol.

The *configuration* connection is designated for setting up video stream parameters and triggering stream transmission. Some cameras follow a slightly different approach. While camera C and D reuse the previously established encrypted channel, camera A and B open a new connection. Although camera A and B are based on the same system, they exhibit different behavior: Camera A utilizes an *unencrypted* human-readable proprietary protocol, but camera B employs an *encrypted* proprietary protocol instead.

Most systems fall back to unsecured connections, which are established on-demand, to transmit the actual *video stream*. Solely camera B reuses its secured proprietary *configuration* connection; thus, this is the only system that does not transmit video stream data over an unsecured channel. Table 2.1 lists a summary of the communication channels and their protection.

Connection	Camera A	Camera B	Camera C	Camera D
initial	TLS 1.2	*TLS 1.2*	*proprietary*	SSL 3.0
configuration	*none*	*proprietary*		
video stream	*none*		*none*	*none*

Table 2.1: Overview of camera protocols. Insecure implementations are in *italics*.

Eavesdropping stream data from camera A, C, and D is therefore trivial and the negative impact on privacy is evident without further need for analysis. Despite the several other secured data channels, they cannot be presumed as being well-protected; hence, a more in-depth analysis of the secured protocols follows. I discovered several severe issues, especially in the TLS *initial* connection and the *configuration/stream* channel of camera B, and additionally in the proprietary *configuration* connection of camera C.

2.3.3.1 TLS Server Certificate Validation Vulnerability in Camera B

Camera A, B, and D have the cloud server's certificate, including its public key stored in flash memory. When the camera establishes an SSL or TLS connection to the cloud server, the server certificate is validated. If the certificate is invalid, the connection should be closed and an attempt to connect may later be retried.

This behavior is correctly implemented in camera A and D, whereas the behavior of camera B was unexpected. When the server presents a self-signed and expired certificate, which indisputably is invalid, the camera falls back to an unsecured Hyper Text Transfer Protocol (HTTP) connection. The camera's debug output, shown in

Figure 2.10, reveals that this is the intended behavior: "SSL failed for HA proxy event. Fallback to HTTP." I verified this further by disassembling the firmware and found the corresponding code that reads the return code of the *curl* tool and initiates a fallback for five TLS/SSL-related error codes.

```
[02-09-2015_17:13:47][HA_Proxy_heartbeat_thread][Line:1381]
    HA has not given application server IP - Restart HA server

[02-09-2015_17:13:48][HA_Proxy_heartbeat_thread][Line:1246]
    SSL failed for HA proxy event.  Fallback to HTTP.
```

Figure 2.10: Camera B debug output during TLS/SSL to HTTP fallback.

The implementation might stem from the idea that the camera should always be able to bootstrap, even if it does not have the latest server certificate, e.g., due to an extended downtime. While the idea appears valid, its implementation is ill-conceived and negates TLS 1.2 security almost entirely. Since the implementation counteracts the idea of validating communication partners and encrypting traffic, typical MitM attacks are viable.

As of 2018, the situation appears to have worsened, according to my repeated tests. The servers are still online and functional but for an unknown reason, the camera shortly communicates with the servers encrypted and then enters the unprotected fallback mode. Furthermore, the server currently supports only four ciphers, the overlap between the server and camera is solely `TLS_RSA_WITH_RC4_128_MD5` which is far from being recommended anymore [GPdM15, ABP+13], RFC 7465 even disallows RC4 in all versions of TLS [Pop15].

Thus, *local attackers* can interpose themselves between the camera and the cloud server, e.g., by downgrading the connection. Despite being based on the same firmware, this behavior is not present in camera A. This behavior was successfully verified in the firmware disassembly in which the aforementioned error code handling and downgrade logic are not present.

2.3.3.2 Weak Proprietary Communication Encryption of Camera B

Next, I analyze the proprietary protocol of camera B that was designed to secure the *configuration* and *data* connections. The communication is initiated by the camera through the transmission of `HELLO` accompanied by its MAC address, to which the cloud server responds with `CHALLENGE` and a 128-bit nonce. The subsequent communication appears to be encrypted.

A visual analysis yields no clues as the data does neither show any structure nor any other anomaly. An entropy test showed a value of approximately 7.99 bits per byte, which is a hint at well-encrypted data. Traffic analysis did not expose any more details and I disassembled the firmware for a deeper insight.

The protocol is built up around the cryptographic primitives Secure Hash Algorithm 1 (SHA-1) [NIS15] and AES128 [NIS01b]. The implementation was found to be very optimized with most loops unrolled, which reduces the number of required branches. Furthermore, this AES implementation is based on T-tables to enhance its performance further.

While analyzing the implementation, I found the function that derives the AES session key. Each time a connection is being established, the server sends the aforementioned 128-bit nonce N. The nonce is concatenated with a *hard-coded* 128-bit pre-shared key K_p which results in the 256-bit intermediate key

$$K_\mathrm{i}[0{:}255] = [K_\mathrm{p}\ N]. \tag{2.1}$$

Please note that K_p is the same among all devices. Next, the SHA-1 function is applied to the intermediate key K_i and the 160-bit hash K_h is generated, thus

$$K_\mathrm{h} = \mathrm{SHA1}(K_\mathrm{i}). \tag{2.2}$$

The hash K_h is truncated to 128 bits by discarding 32 bits at its end. This results in the 128-bit AES session key

$$K_\mathrm{a} = K_\mathrm{h}[0{:}127] \tag{2.3}$$

which is valid as long as the connection persists.

The AES is not applied directly onto the data, encryption and decryption is performed in Cipher Feedback (CFB) mode. Although an initialization vector is required, it is not supported by the protocol. I discovered that the initialization vector is set to zero in the beginning. Furthermore, the cipher context is re-initialized with zero each time the direction of communication changes or when the camera receives a new dataset. This results in the same initialization vector being reused repeatedly.

I discovered two major weaknesses in the protocol design, i.e., the deployment of the same pre-shared key for every device and the reuse of the zero initialization vector in CFB mode. The latter issue weakens the encryption and an attacker may be able to reconstruct parts of the CFB keystream to forge packets under some conditions. However, the most significant issue is the deployment of the same pre-shared key K_p in *every* device. The AES session key can be directly computed with the knowledge of the nonce N and the pre-shared key K_p. If an attacker gets hold of this key by disassembling the firmware of one device, as shown in this section, the encryption of any camera device can be broken easily. Despite finding K_p and the key derivation function in the binary requires some effort, such a task can usually be completed within a few days. This allows a generalization of the vulnerability to all cameras of this type and effectively renders the encryption useless. An attacker is able to encrypt and decrypt arbitrary packets of any system with the knowledge of the pre-shared key; thus, a *local attacker* may eavesdrop and decrypt the connection or even impersonate the cloud server.

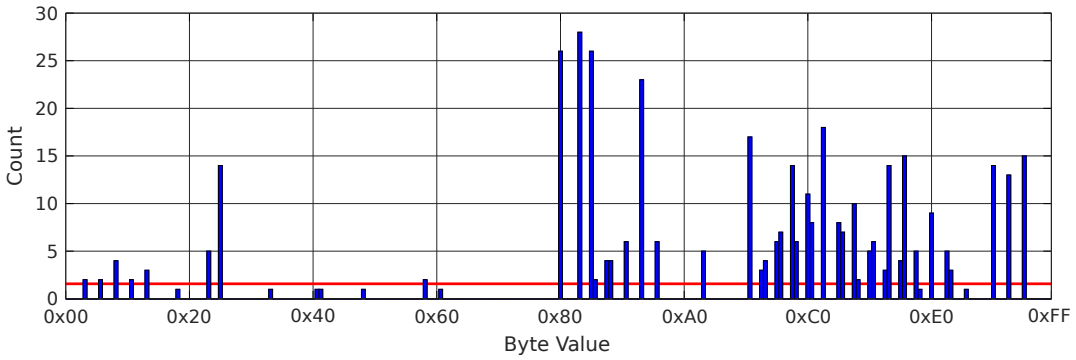


Figure 2.11: Byte value histogram of one 404 byte large encrypted packet of camera C.

2.3.3.3 Weak Proprietary Encryption/Obfuscation of Camera C

The manufacturer of camera C developed a proprietary encryption algorithm for camera and cloud communication. There is no public information available and the protocol does not follow any known standards. The data does not show any human-readable contents, nor is there a typical header, handshake message, or certificate exchange. However, a visual inspection quickly reveals patterns and repeating byte sequences in most packets — a suspicious property that asks for further investigation.

All byte values should be evenly distributed for well-encrypted data; however, a statistical analysis reveals that this is not the case for camera C. Each packet has a significant accumulation of several specific values, while others are heavily underrepresented. This is obvious in the byte value histogram for one packet in Figure 2.11. The plot shows a major imbalance with a maximum of 28 occurrences for byte value `0x85`, whereas the range between `0x62` and `0x7F` is entirely unpopulated. The expected distribution is denoted via a red line. The distribution and its imbalance are similar for other packets, but the horizontal position of the peaks may change. To analyze this further, the entropy of such packets was computed and showed to be only between 5 and 6 bits per byte. Thus, this is far from well-encrypted data that would approach almost 8 bits per byte entropy, cf. Subsubsection 2.3.3.2.

My conjecture of a flawed algorithm was validated by reverse-engineering the encryption algorithm by disassembling the camera's firmware. The binary contains some suspicious symbols such as `rEncodeBytes`, `lDecodeBytes`, `rrotate`, `lrotate`, and `choose_random_shiftnum`. The function `choose_random_shiftnum` is executed first, followed by `rEncodeBytes` which calls `rrotate`. The decryption starts with `lDecodeBytes` which calls `lrotate`. Dissecting these functions reveals that the bits in every byte are circularly right-rotated by a number between one and seven. This number, plus the constant `0x3C` (ASCII character "<"), is the first byte of each packet.

All data is easily decoded by applying the extracted algorithms. Thereby, a human-readable XML dataset is revealed that contains camera control data.

Since there is no key, the knowledge of the algorithm suffices to encode and decode packets. Thus, this encryption is only a very weak obfuscation and no secure encryption algorithm. Having this knowledge, an attacker can decode communication and communicate with the cloud servers. This leads to a severe issue, as shown in Subsubsection 2.4.3.2.

2.3.4 Device Authentication and Access Control

The previous section evaluated the encryption algorithms in detail; however, authentication was omitted. Hence, this section focuses on weaknesses in video stream authentication and the vulnerabilities arising from there.

2.3.4.1 Device and Stream Authentication of Camera A and Camera B

Cameras A and B are based on the same concept of authenticating the camera against the cloud server. Despite encrypting all data, camera B can be regarded as being as secure as camera A, since the encryption of its proprietary protocol was successfully broken.

After the *configuration* connection to the cloud server has been established, the camera identifies itself by transmitting its MAC address. This information is used by the server to recognize the camera device. However, this is the only identifying information that is transmitted by the camera; thus, there is no authentication, but only a very weak identification.

The *initial* connection does not make use of any TLS 1.2 client authentication features. The camera accesses the scripts `httpevent.php` and `httpmotionevent.php` on the cloud server. The first one is polled regularly and notifies the cloud and the user that the camera is online. When the second one is accessed, it causes a motion-event notification, comparable to an alarm, which is sent to the user. All data is transferred via an HTTP GET-request to which the camera appends its MAC address as a request parameter; thus, the MAC address is again used for a weak identification

Altogether, both systems employ no authentication, but only a weak identification. This is a major design flaw that enables a range of attacks for the remote attacker. For example, since there are no means for secure authentication, impersonation of cameras is viable for an attacker.

2.3.4.2 Device and Stream Authentication of Camera C

Camera C has a similar authentication concept that is also based on the camera's MAC address. Even though the procedure takes place inside the obfuscated connection, authentication data is accessible for analysis since the self-made encryption algorithm is futile.

After the camera has established a connection to the cloud server, the camera transmits an XML-encoded dataset. This data includes the field `id`, whose value contains the MAC address. Other data such as serial numbers are also transmitted,

but obviously not checked by the server during authentication. Thus the same issues emerge, enabling an attacker to impersonate the camera.

2.3.5 Server-Side Weaknesses

During the analysis of the camera devices, two cloud server related security issues came to light, which are explained shortly in this section.

2.3.5.1 Weak Geolocation Service Pin Generation of Camera D

Upon startup, camera D queries the cloud server for geolocation information using HTTP. Geolocation services attribute an IP address more or less precisely to a geographic location. The camera requires this information to adjust the timezone setting, for example. Such services are offered by companies that sometimes have a pay-per-use policy; thus, each request generates costs — or, if there are no costs, at least generates server load. Nonetheless, such a service must be protected against abuse.

In this case, the cloud service is protected by a pin code. When the camera sends its request, it must include its own MAC address and the corresponding pin code. Only if the MAC address is authorized and the pin code matches *for this* MAC address, a reply will be delivered.

My analysis discovered that the pin code is generated on-demand on the camera system. Thus, the algorithm is present on the system and was extracted by me via a disassembler. The pin code is the MD5 hash of a 64 byte long string that consists of constant characters and parts of the lower-case MAC address. For example, the geolocation pin code for a camera with the MAC address `01:23:45:67:89:AB` is

$$\text{pincode} = \texttt{MD5([...]01:23:45[...]01:23:45:67:89:ab[...])}. \tag{2.4}$$

With this knowledge, the remote attacker can generate valid pairs of *arbitrary* MAC addresses and pin codes. Abuse of the cloud service is viable that can easily cause high system load and costs.

2.3.5.2 Camera A Cloud Server Data Leakage

Even though no penetration testing was performed on the cloud servers due to legal reasons, a data leak was discovered nevertheless. As the cloud server of camera A was accessed through a web browser to test connectivity, a directory listing appeared. When a folder is accessed that has no webpage associated with it, the web server may give a list of all files and folders contained therein. This is a configurable option that may be helpful during development but is disabled in well-secured production servers.

The directory listing exposed a binary named `passtool_vf` that was accessible without restriction. The file is a 32-bit x86 binary that is intended to run on a Linux operating system. I ran the binary in an isolated virtual machine and discovered that this is a password generation tool that takes the camera's MAC address as an input and

returns the factory interface's password. The disassembly of the binary shows exactly the same algorithm that was previously reverse-engineered in Subsubsection 2.3.2.1.

Due to this information leakage, an attacker would have acquired knowledge of the password generation algorithm even if the initial reverse-engineering was not successful. Thus, the configuration error is critical since it even nullifies any secure password derivation algorithm.

2.4 Local and Remote Attacks

This section demonstrates how the local and remote attackers can exploit the previously described security design and implementation weaknesses. Due to the variety of potential attacks, only the most severe ones are described. All the mentioned attacks were successfully verified in practice.

2.4.1 Summary of Discovered Issues

A summary of the discovered vulnerabilities is given in Table 2.2. Each camera is affected by at least one remote or local vulnerability.

Attack	Camera A	Camera B	Camera C	Camera D
Camera impersonation	R+L	R+L	R+L	–
Server impersonation	L	L	L	–
Manipulation of the camera	L	L	–	–
Stream eavesdropping	L	L	R+L	L
Attack on cloud server	–	–	–	R+L

Table 2.2: Overview of vulnerabilities against the local (L) and remote attacker (R).

2.4.2 Local Attacker

The local attacker represents a major threat against camera A, B, and C, due to their insufficiently secured factory or communication interfaces. Furthermore, the cloud server's identity is not sufficiently verified, which constitutes another weakness. Each camera, including camera D, has no or only weak stream encryption that allows for eavesdropping. Please note that my attacker model assumed that solely the local attacker has access to the camera. If devices are exposed to the Internet without NAT and firewalls, the described approaches become feasible also to a remote attacker.

2.4.2.1 Camera A and B Factory Interface Exploitation

Factory interface abuse is trivial for camera A and B. After computing the password, the local attacker gains admin access to the camera's factory interface. Despite not being intended to be used during cloud operation and suffering from several software bugs, the interface is still alarmingly powerful. The attacker can view and record the

current video and audio stream without proper authorization. Additionally, continuous surveillance is achieved by configuring the camera to transmit images to any external SMB server periodically. Such a configuration persists until the next reboot and might remain active several months without requiring the presence of the attacker. Setting up surveillance is completed within a few minutes and cannot be noticed by the typical user. Hence, this type of attack has the potential to impair the user's privacy heavily.

The local attacker is able to destroy camera A and B through the factory interface. It provides a feature to save a camera configuration to a file and to restore it later. When a manipulated invalid configuration file is uploaded, the camera will not start up correctly anymore and remains in an infinite reboot loop. The user cannot fix the issue as reprogramming of the non-volatile flash memory chip is necessary. This vulnerability enables a very effective denial-of-service attack within minutes that practically destroys the system and impedes its availability. Additionally, the manufacturer is harmed as such damage is hard to attribute to the user and a replacement device may have to be provided due to warranty.

The local attacker gains root access to the camera by combining factory interface access with a second vulnerability. For that, the attacker opens the SMB mount configuration page on the factory interface. Thereon, an SMB share can be entered that becomes the unsanitized input to a command that is executed as root, as described in Subsubsection 2.3.2.2. An input, such as

```
//192.168.1.2/k /home/disk && /home/disk/j.sh ; #
```

results in the following command:

```
mount.cifs //192.168.1.2/k /home/disk && /home/disk/j.sh ; # /home/disk
```

This mounts the folder `k` of the attacker's system at `192.168.1.2` and then executes the script `j.sh` from the aforementioned folder. Short file and path names are advisable as there is a length limitation for the input and additional characters get truncated. The exploit can set up a reverse root shell on the system that enables an attacker to take over control of the entire device. Furthermore, as the exploit allows mounting an *external* disk, a multitude of tools is made available easily without being limited to the camera's available flash memory. Thereby, an attacker may destroy the system, run privacy-infringing operations, or may use the system for other malicious purposes such as (D)DoS [MR04].

These issues are especially severe as the users are neither able to detect an attack nor able to protect their camera. The existence of the factory interface is not documented and it is unknown to the user who is exposed to potential privacy infringement and availability limitations.

2.4.2.2 Camera A and B Backdoor Setup Page

A page called *Backdoor setup* was discovered in the factory interface of camera A and B. There was no link to this page; it was found in the extracted firmware

and is accessed via `http://[IP]/debug825/`. Despite its name, it allows no critical operations and is directed to a more sophisticated system debugging. Moreover, most operations require additional passwords to activate them; however, they were all found within minutes in a firmware binary.

From the *technical* point of view, there is no noteworthy weakness. But regarding *publicity*, disclosing that a system has a *Backdoor setup* functionality could quickly diminish the manufacturer's reputation and loss of trust may follow — most end users will not be able to tell apart a real backdoor from a misnamed debug interface. Thus, the manufacturer is more affected than the user. An attacker can leverage such a hidden page to harm the manufacturers or blackmail them by threatening disclosure of this information.

2.4.2.3 Camera A, B, and C Cloud Server Impersonation

Cloud server impersonation is feasible for camera A, B, and C. While the *configuration* connection of camera A is unsecured, B and C implement some sort of encryption. However, since their security has been broken, all connections are regarded as being unsecured. When impersonating the cloud server, the attacker chooses a MitM approach. The approach for each system is slightly different, but the outcome is similar. The attacker obtains control of most of the camera's functions and can manipulate it. In the most simple form, this results in a denial-of-service attack as the connection to the cloud is interrupted and the system's availability is damaged. The attacker might also eavesdrop the video stream and affect the user's privacy thereby.

2.4.3 Remote Attacker

All four cameras and the cloud infrastructure are affected by the remote attacker. While there is only a minor vulnerability in the cloud concept of camera D, the other three cameras show major issues. For cameras A, B, and C, impersonation is possible and an attacker may inject forged video streams. For camera C, even unauthorized remote access to the camera and its stream is possible by executing a more complex approach.

For each attack, knowledge of the camera's MAC address is a prerequisite. Depending on the intention of the attacker, two methods are suited to obtain MAC addresses. If one *specific* camera is attacked, a single, maybe encrypted, packet from the camera's wireless interface will be sufficient. The MAC address is included in the unencrypted packet header of each WLAN packet. In the other case, if one *random* or *many* cameras are attacked, MAC addresses can simply be guessed. Each manufacturer is allocated only a limited range of MAC addresses and they are usually assigned to each manufactured device in an incremental manner. Based on a single known MAC address, the attacker can enumerate a large number of further addresses that likely exist.

2.4.3.1 Camera Impersonation of Camera A and B

Camera A and B can be impersonated towards the cloud server because there is no secure authentication, but only a weak identification through MAC addresses.

The first attack targets the motion detection system. When the feature has been activated and the camera senses motion, it transmits a TLS HTTP GET-request to the cloud server, similar to the *initial* connection. The request accesses the script `httpmotionevent.php` and the request contains the camera's MAC address for identification: `https://[...]/httpmotionevent.php?[...]&mac=0123456789AB&[...]`. The user will be alarmed about a detected motion and a push notification on the smartphone might be shown.

Although the camera's channel is TLS-protected, an attacker can establish a connection into the cloud and transmit an identical GET-request. Any warnings regarding an unknown server certificate can be disregarded. Since no client certificates are deployed, the cloud server cannot verify the client. Thus, the remote attacker can easily trigger such an event, and this degrades the authenticity of a motion detection notification.

The second attack targets the camera's availability and cloud connection by a low-effort denial-of-service approach. Camera A and B are continuously connected to the cloud over the *configuration* connection. It is unsecured for camera A and secured for B, but due to the successfully broken encryption, both connections are considered insecure in the following. Due to the encryption, the attack is slightly more effortful for camera B. The remaining approach is similar for both cameras.

Initially, an attacker establishes the control connection to the cloud server and impersonates the camera by using its MAC address. This action immediately terminates the cloud server's connection to the corresponding *real* camera, as the server preserves only the latest connection for each device. After losing the connection, the legitimate camera will try to re-establish the communication channel after some time. I observed that repeating the attack every 40 to 50 seconds disconnects the camera permanently and it remains inaccessible for the user.

The attack requires packets that sum up to less than one kilobyte, making the attack well scalable. Although not tested for legal reasons, an attacker may easily disconnect numerous cameras with a single Internet connection. Thus, this is an advanced denial-of-service attack that scales from a single user to up to hundreds or even thousands of devices. It impacts the system's availability and is a potential blackmailing threat for the manufacturers.

The most powerful attack is the injection of a forged video stream. Instead of only blocking the camera, the attacker behaves like a real camera via the configuration and video stream data connection. When the connection is being established, a video stream request will not end up at the legitimate camera but at the attacker. Later, when the user requests the stream via the cloud server, the attacker may reply with an arbitrary video which will then be displayed to the user and/or recorded in the cloud. The attacker is free to choose any video, ranging from a black still image up to intimidating and disturbing video streams. This attack targets the availability but

even more the authenticity of the user's video stream data. Such an attack is an even larger threat for manufacturers because their reputation is easily impaired thereby. Due to the high visibility and significant effect, this attack would be very promising for blackmailing the manufacturer.

The presented vulnerabilities are combinable. For example, the attacker may trigger a motion detection event which causes the cloud to request a video stream. The stream may then be chosen by the attacker, transmitted into the cloud, stored there, and then presented to the user. Despite requiring some effort, I demonstrated this attack successfully on my camera. Altogether, these vulnerabilities enable an alarming variety of potentially devastating attacks.

2.4.3.2 Camera C Impersonation and Stream Eavesdropping

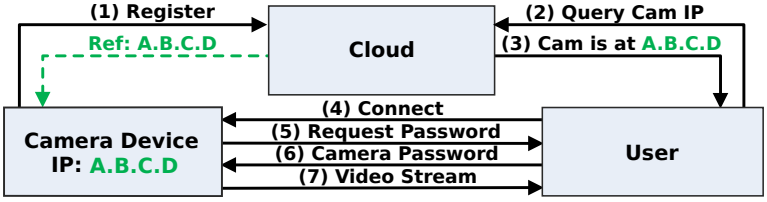


Figure 2.12: The mode of operation of the undisturbed system.

Camera C is affected by similar issues as in cameras A and B. Since the system misses secure authentication, denial-of-service attacks and the injection of forged video streams are possible. The availability and authenticity of video streams cannot be guaranteed. Thus the system suffers from similar vulnerabilities, which may harm the user and especially the manufacturing company.

There exists a related method exclusively for camera C that allows an attacker to eavesdrop the original video stream. The problem emerges from the communication concept between cloud, camera, and user. The procedure for video stream access is depicted in Figure 2.12. Initially, the camera registers in the cloud using its MAC address and the cloud stores the corresponding IP address. When the user decides to watch a live video stream, the IP address is requested from the cloud and a *direct* connection to the camera is being established. Next, the camera requests a password from the user and authenticates the user by validating the password.

The attacker can exploit this behavior to gain knowledge of the camera's password, as depicted in Figure 2.13. In order to do so, the camera is impersonated and the cloud server stores the attacker's IP address. When the user wants to view the video stream, the connection is not established to the legitimate camera, but to the attacker instead. The attacker can request the password from the unaware user. Since the attack is not visible, the user will enter the password. It is then Base64 encoded and transmitted to the attacker who gains knowledge of the camera's password. At this point, the attacker may decide to inject a forged video stream; however, the vulnerability is more severe.

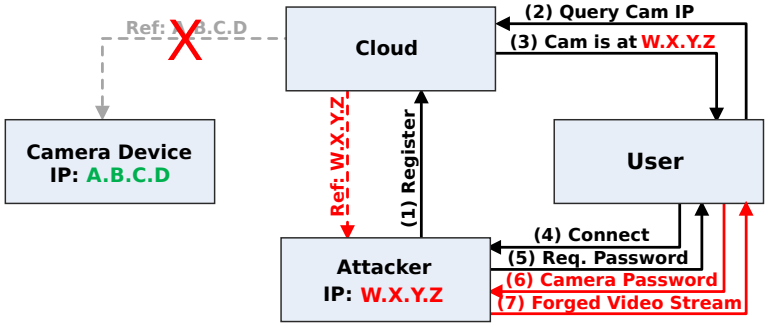


Figure 2.13: The attacker impersonates the camera, obtains the user's passwords and injects a forged video stream.

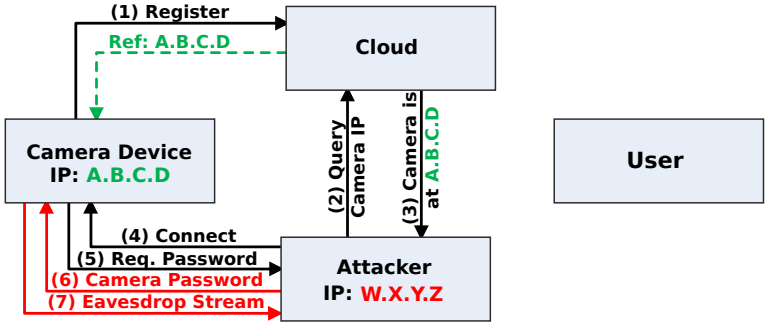


Figure 2.14: The attacker eavesdrops the user's video stream after obtaining the password.

Afterward, the attacker ceases the impersonation and the legitimate camera retakes its place. The attacker is now able to login to the legitimate camera and to eavesdrop the video stream by using the official software and the previously acquired password. These steps are summarized in Figure 2.14. The attack nullifies the confidentiality of the video stream and harms the user's privacy. Additionally, the manufacturer is threatened even more, since such incidents have a significant impact on the reputation. The attack can be performed on any camera of this type, and only the MAC address needs to be known or guessed.

2.5 Discussion of Weaknesses

My analysis discovered significant flaws in the design and implementation of the evaluated devices. Most of the weaknesses stem from widespread mistakes in security design and some of them were found across different manufacturers.

Fallback to insecure modes: Being able to downgrade a function or protocol to an insecure method is a very common pattern in attacks, e.g., the FREAK

attack [Gre15]. In the case of camera B, a certificate validation error, which should have closed the connection, caused a fallback to an unencrypted connection.

Proprietary security protocols: Designing your own crypto-algorithms is usually a terrible idea. Well-established cryptographic primitives require years of discussion and analysis to avoid systematic and implementation flaws. Hence, self-made designs will often fail. In the case of the proprietary encryption of camera C, the algorithm does not follow Kerckhoff's principle, and the algorithm is obviously extremely weak. On the other hand, also inappropriate handling of initialization vectors, such as in the proprietary encryption of camera B, is a typical implementation fault of otherwise secure primitives.

Weak passwords: Default passwords and master keys are a significant threat in general, as they can be revealed by analysis of a single device and then used for large-scale and remote attacks. In the case of camera B, a pre-shared key is present for data encryption that is shared among all devices. Furthermore, cameras A and B generate the factory interface password from public information, namely the MAC address of the camera.

No secure authentication: All tested cameras have some kind of authentication weakness. In the case of cameras A, B, and C, the identity is verified by checking only the MAC. Similarly, camera D employs a pin code in addition; however, it can be trivially computed with knowledge of the MAC.

Lack of security engineering: The overall problem is the insufficient consideration of security during the product development cycle. An early and thorough security analysis according to well-established standards would have identified many problems that were discovered in this evaluation. Creating a new secured software revision and deploying it to the customers when the products are already on the market causes significant effort and is accompanied by high risk.

2.6 IoT Security After 2015: The Mirai Botnet and Upcoming Regulations

The goal of my work was to raise awareness for the dangers of insufficiently protected IoT devices of 2015 on a *conceptual level*. The analysis shows that all four tested IoT devices lack an adequate security concept and are alarmingly vulnerable even to attackers with little financial means and little experience. After publishing my results, the security of IoT devices soon came into public mind at the end of 2016 as the Mirai botnet [KKSV17] was discovered. It had recruited more than 600 000 IoT devices and weaponized them for taking major websites offline using Denial-of-Service (DoS) attacks [AAB+17].

However, there are differences between my analysis and the goals pursued by Mirai as none of my tested devices was affected. Mirai aimed at building a Botnet-of-Things for DoS attacks on external targets, whereas my work analyzed the security

measures deployed in embedded devices. While the vulnerabilities initially exploited by Mirai were mostly trivial, such as well-known standard passwords and externally reachable configuration interfaces [AAB+17], my analysis dug deeper and was directed at discovering flaws in the security *concepts* of the devices under test. Thus, the issues exploited by Mirai were out of the scope of my work as weakly protected and publicly reachable interfaces — i.e., configuration errors by the user — are obvious weaknesses that require no in-depth analysis. Their exploitation was only a matter of time, and the vulnerabilities could mostly be fixed by the *user* via a proper configuration. In contrast to this, the more critical weaknesses discovered in my work have shown to be on the *manufacturer's* side and cannot be resolved by the user, leaving even tech-savvy users oblivious of the danger.

Nevertheless, a new variant of Mirai appeared shortly after, which exploited flaws in TR-064 implementations of routers. Thereby, the responsibility has been shifted back to the manufacturer to provide security updates for the affected devices. Experience shows that there is currently no large incentive to invest further resources into fixing products, which might sometimes already be discontinued; thus, devices remain vulnerable [TO15].

Fueled by the Mirai botnet discussions, researchers are increasingly discussing the need for market or regulatory solutions against IoT-related security issues [Jer17]. While there is a large number of standards primarily for network aspects of IoT [SYY+13, Med16], security aspects are starting to gain more attention since 2016, according to Walker et al. [WSNS18]. In the EU, the General Data Protection Regulation (GDPR) [Eur16], which became effective in May 2018, aims at increasing privacy protection and requires to notify security breaches to the data protection authority, for example. Additionally, there are movements by the European Commission to create an "EU cybersecurity certification framework" which shall be handled by the EU Agency for Network and Information Security (ENISA). The US Congress is considering similar bills as of 2018. For instance, the "Cyber Shield Act" proposes a voluntary certification program, whereas the "IoT Cybersecurity Act" enforces encryption, the absence of hard-coded passwords, and patchable devices. Finally, Walker et al. note that China has shown activity in a similar direction to increase IoT security. However, they list several risks for IoT development, such as a fragmentation between markets due to incompatible regulations.

Despite the noteworthy ambitions to regulate the IoT, there are several reasons to doubt their effectiveness.

- **Legacy products:** IoT devices of the past and present are not subjected to any security certification. Creating new standards will not retroactively make these devices more secure, they will still be in use, connected to the Internet, and sometimes lacking sufficient security.

- **Global regulations:** Local regulations cannot solve the global IoT security problem, as unsecured devices will prevail in the unregulated fraction of the Internet that can still affect any other region, e.g., via distributed DoS attacks.

- **Out of business:** Although regulations foresee security updates, companies might go out of business with their consumer products still connected to the Internet; however, no security update will be developed and deployed.
- **Withdrawal of devices:** Withdrawing devices that cannot be fixed, e.g., due to unresolvable firmware issues or when the manufacturer has gone out of business, imposes significant effort and might not be practical, especially on a global scale.
- **Counterfeit (security) certifications:** Although currently existing regulations enforce the adherence to standards and a Communauté Européenne (CE) sign on products sold in the EU, there exist faked markings that allow products to enter the EU despite not meeting the requirements [Ver08].

The regulation of IoT devices is a difficult task that might not be sufficiently addressable via local regulations and certifications, but it has to be tackled on a global scale due to the border-less structure of the Internet. However, all these discussions assume that there is an *inevitable need* for every device to be exposed to the Internet — an assumption that should especially be questioned by end-users before acquiring their next devices.

2.7 Conclusion

The preceding sections presented a security analysis of four video surveillance systems and a discussion thereof. Due to their deployment in private houses, they have a significant impact on privacy. Furthermore, manufacturers are highly vulnerable to blackmailing due to an insufficient security concept. During the analysis, I found strong indications that some of the manufacturers reused the vulnerable software components also in other IoT devices besides camera systems — thus, this may be just the tip of the iceberg. This was verified by Ling et al. in 2017, who successfully applied the exploit described in Subsubsection 2.4.3.2 on smart plugs [LLX+17].

My attacks demonstrate the criticality of the problems and emphasize the need for a secure engineering process for IoT devices. Although only four cameras were tested, my findings indicate that IoT devices in the year 2015 did often not follow basic techniques of state-of-the-art security. With respect to future developments, especially in the direction of highly-interconnected production in industry, there is a need for a thorough security development process for embedded systems [WK16, Wie15].

My analysis was only successful since the cameras were physically accessible and could be opened without significantly damaging the system. This enabled me to extract their firmware, to discover the weaknesses as mentioned above during an in-depth evaluation, and to develop them into vulnerabilities. Since all vulnerabilities are network- and software-related, analyzing the firmware has shown to be expedient and the correct approach. Exploitation is possible only when knowing the associated keys, passwords, and algorithms to manipulate communication or to gain system access. Without access to the firmware, finding most weaknesses exclusively via the network interface might have been difficult, if not impossible.

Thus, the presented approach is only viable if there is no protection against physical tampering that prevents manipulations, such as desoldering a flash memory IC or attaching a debugger. However, tamper protection mechanisms will not solve *software* related weaknesses, but they still raise the bar for an attacker significantly.

Chapter 3

Fuzzy-Glitch: A Practical Ring Oscillator Based Clock Glitch Attack

The previous chapter demonstrated that software vulnerabilities could nullify the security of an embedded system. Most software weaknesses can be resolved, e.g., by fixing the corresponding code or by improving the underlying concept. However, secure software is one requirement but is on its own not sufficient for a secure embedded system. Even if there are no *software* related weaknesses in a system, it is still exposed to other classes of attacks, such as *hardware* related ones.

There exist several types of hardware attacks, which will be shortly discussed in Subsection 3.1.1. Clock glitches are one attack class, on which this chapter focuses on and presents a novel approach that employs a *fuzzy glitch*. This method and parts of this chapter have previously been presented at the *22nd International Conference on Applied Electronics* in 2017 and were published in its corresponding proceedings [OSS17]. The research was done in collaboration with Robert Specht; Qiyi Li implemented the Field-Programmable Gate Array (FPGA)-based design during his research internship. Later, Florian Hauschild improved the glitch generating system and performed further experiments on microcontrollers and their Phase-Locked Loops (PLLs). These results were published in 2019 and demonstrate that clock glitch attacks can still affect systems that derive their clock via a PLL [SHO19].

3.1 Overview

There exist several categories for physical attacks on microcontrollers, e.g., active and passive attacks. For the latter one, an attacker exploits a side-channel via the observation of physical quantities, such as power consumption [KJJ99], local electromagnetic fields [HMH+12], or timing [OLR14] while the system runs as intended. Thereby an attacker may extract secret information such as cryptographic keys. Active

attacks, however, rely on the manipulation of the system, e.g., by injecting faults during execution.

When discussing active attacks on a system, three related but different terms must be defined beforehand: Fault, error, and failure [GC15]. Initially, a *fault* is the outcome of an attack that influenced the system. It usually affects a single system element, such as a variable in memory or a register, by changing its value. For example, the fault affected a Boolean value, which was altered from false to true. A fault will only develop into an *error* if the value is processed and thereby propagates further into the system, for example, the incorrect value causes the unintended execution of an if-clause. Please note that not every fault leads to an error, e.g., if a new and correct value overwrites the incorrect value before it is read, the fault vanishes without an error. If an error is present in the system, it may further propagate and lead to the *failure* of the system. This may be observable from the outside, if the system returns an incorrect result, leaks critical data, or ceases operation entirely. However, various reasons exist why an error might not lead to a failure. High-reliability and security-critical systems are often designed to incorporate countermeasures targeting the propagation of errors via specialized algorithms [ABF+03]. Altogether, an attack always starts with inducing a fault that is intended to become an error and shall finally cause the system's failure.

3.1.1 Related Work

In security, fault attacks aim at causing erroneous operation of the processor and provoke failures that leak critical information [TMA11]. For example, clock glitches are employed in hardware security tests, which evaluate systems regarding vulnerabilities that emerge from fault attacks. Clock glitches belong to the class of non-invasive attacks for which the chip's package does not have to be modified. Compared to semi-invasive laser fault attacks, the chip does not need to be decapsulated and only a few wires need to be attached; thus, fault injection via clock glitches is more comfortable to perform and less sophisticated equipment is required. This is especially true when compared with very effortful invasive attacks, which manipulate the chip in hardware, i.e., break and add traces on the silicon.

In general, a glitch is a short and unwanted change of the logic level of a digital signal. An example is depicted in the upper plot in Figure 3.1. Glitches can occur on every signal; however, the clock signal is usually manipulated for attacks. Usually, the clock signal is temporarily manipulated by inserting a precisely timed and controlled glitch. This quick change in its logic level carries much higher frequencies, compared to the original clock signal. Thus, the device is now operated at frequencies out of its specification, comparable to a massive but time-limited overclocking. This leads to timing violations of registers and results in undefined behavior [ADN+10]. Such an attack usually applies a precisely timed and controlled glitch.

The effects are divided into two groups, control flow manipulations and data changes. If the control flow is diverted, this may result in an early exit from loops or system crashes [CT05, BGV11]. Effects on data are also especially critical, e.g., for cryptographic algorithms such as AES, since well-placed bit flips lead to powerful

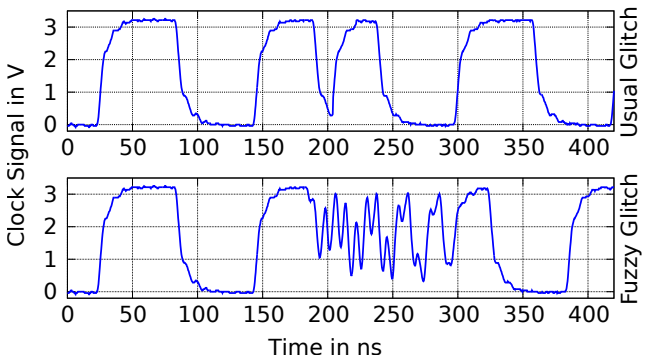


Figure 3.1: Depiction of a usual (top) and a fuzzy (bottom) clock glitch.

attacks that can extract the key [SMC09]. Hence, securing devices against fault attacks is very important and powerful tools are required to test systems against them.

Regarding clock glitch attacks, numerous implementations have been published that follow the same baseline for signal generation. In all cases known to the author, the system creates a precisely timed glitch signal [MSI16, KH14, KHEB14, ESH+11]. The method creates well-reproducible results, has proven to be adequate for an in-depth analysis, and provides high success ratios [KH14, ESH+11]. Thus, there exist powerful methods to attack well-known systems and algorithms precisely by having the possibility to target even a single instruction [BGV11].

Nonetheless, the method is accompanied by some drawbacks. It works best if the system under attack is well-known and was analyzed beforehand — a precondition that cannot be fulfilled in many cases. Additionally, the glitch generation circuitry is often built up around FPGA-specific hardware such as Digital Clock Managers (DCMs) [MSI16]. Their number is limited and their capabilities depend on the specific device and vendor, which may be a restriction for small and simple devices. Thus, such an attack requires complex glitch generators and deep knowledge about the system under attack to apply a well-chosen glitch during execution. Moreover, the generated glitch is still a fairly deterministic digital signal without intended randomness; thus, a variation of the attack parameters, e.g., glitch frequency, rise and fall time, has to be applied externally. In most cases, the evaluation of *all* possible parameter combinations is not feasible in reasonable time [PBBJ15] and only a subset can be tested in practice.

3.1.2 Contributions

To overcome these issues, I propose an alternative concept on the generation of clock signal glitches that breaks with the common approach of injecting precisely generated clock glitches. Instead, I present a novel method that generates a random, non-deterministic, and fuzzy clock glitch signal, which is exemplarily depicted in the lower region of Figure 3.1.

My approach aims primarily at embedded systems that have not been analyzed in detail, yet, but shall be tested regarding their susceptibility to glitch attacks. Thus, I trade in attack reproducibility for the ability to reduce the parameter space, thereby allowing a simpler and quicker evaluation of a system. Additionally, the concept reduces the requirements on the FPGA.

In detail, the presented contributions include:

- A novel concept to create a clock signal glitch by combining two length-adjustable ring oscillators via an XOR gate.
- The use of a *fuzzy* random and non-deterministic glitch signal instead of an exactly timed glitch for security testing.
- An implementation of the proposed concept on a Spartan-3E FPGA platform based exclusively on standard logic.
- A demonstration of a successful fuzzy clock glitch attack on the state-of-the-art ARM Cortex-M0 based microcontroller STM32F030.

3.1.3 Structure

After describing the general idea in Section 3.1, an explanation of the technical concept and its implementation follows in Section 3.2. The next section continues with an evaluation of the generated glitches in Subsection 3.2.3. Next, in Section 3.3, the fuzzy glitch carries out an exemplary attack on a microcontroller and the characteristics and effectiveness of the glitch are analyzed. Section 3.4 discusses further developments and attacks on microcontrollers clocked via a PLL. All results are summarized with respect to embedded system security in Section 3.5.

3.2 Glitch Generation Concept and Implementation

The fuzzy glitch aims at achieving a random, non-deterministic, and fuzzy clock glitch signal. This requires three components: A signal source that generates one or more frequencies, a source of entropy for randomness, and means to inject the glitch.

3.2.1 Fuzzy Glitch Concept

Since common oscillators, such as crystal-based ones and PLLs, target high stability and signal quality, they are no viable source for multi-frequency and random glitches. Instead, I deploy multiple configurable Ring Oscillators (ROs) [Eil90] for the first two requirements.

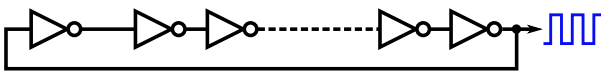


Figure 3.2: A simplified ring oscillator with an odd number of inverters.

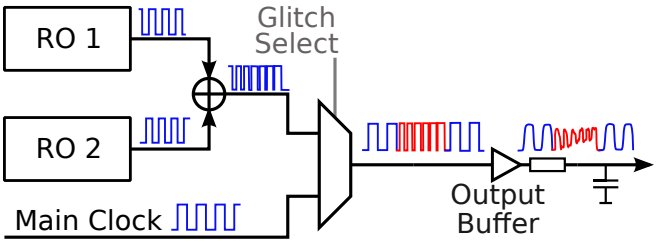


Figure 3.3: Glitch insertion circuitry, output buffer, and RC load.

ROs are special oscillators, which are implemented by simply interconnecting an odd number of inverters in a circular manner, as shown in Figure 3.2. The circuit is an unstable combinatorial loop that oscillates without requiring any external input despite power. On an FPGA, ROs are implemented using standard slice logic; thus, these oscillators do not need special FPGA-specific blocks such as DCMs. Their oscillation frequency depends primarily on the length of the inverter chain because each inverter adds additional delay and consequently increases the signal period.

Besides that, ROs fulfill a dual purpose as they are not only frequency generators but also provide randomness to the system. The frequency and phase of the output signal are subjected to jitter [Abi06] — a property that is also exploited in RO-based true random number generators [FBBV08]. This contributes to an intrinsic variation of the glitch since the attack parameters are automatically altered over time.

The third required component of the system generates the glitch, as a single RO is not able to create such a signal by itself. If two ROs of different frequencies f_1 and f_2 are combined by applying them to the inputs of an XOR gate, the resulting signal is toggled on every edge of each RO, hence creating numerous glitches. This corresponds to mixing the output of both ROs, resulting in $|f_1 - f_2|$ and $|f_1 + f_2|$, a new signal that contains various high- and low-frequency components in its spectrum. Since the output of an RO is not a sine wave but a digital square wave signal, it contains harmonics that contribute additional frequency components, especially after mixing. The FPGA's output impedance and load capacitance limits the signal's bandwidth and causes distortion.

During an attack, a glitch is injected only for a limited amount of time. For this purpose, the system has a multiplexer to choose between the unmodified clock signal and the fuzzy glitch. An adjustable timer controls the duration.

The simplified glitch generation and injection logic are depicted in Figure 3.3. The multiplexer, controlled by the Glitch Select signal, switches between the original clock and the glitch signal. In the default state, the system outputs the unmodified clock signal.

3.2.2 Practical Implementation

The proposed system was implemented on a Nexys 2 board comprising a Spartan-3E FPGA. The basic structure of Figure 3.3 was implemented straightforward.

In contrast to this, I chose a reconfigurable implementation for the ROs. Their exact frequency and their other parameters are difficult to predict since they are free-running oscillators that are not coupled to any FPGA clock. Their frequency depends on the speed grade of the device, the placement and routing of the inverters, and local internal variations of the FPGA [MCMS10]. Hence, a solution was developed that allows adjusting the RO frequency over a wide range.

The oscillation frequency of an RO depends on the length of its loop, including on-chip routing and especially the number of inverters. Each inverter consumes time during switching and the routing creates an additional delay. Thus, an increase in the number of inverters raises the loop length and consequently the oscillation period, thereby lowering the RO's frequency. Hence, the oscillation frequency is controlled by scaling the length of the inverter chain and I created a design to fulfill this requirement.

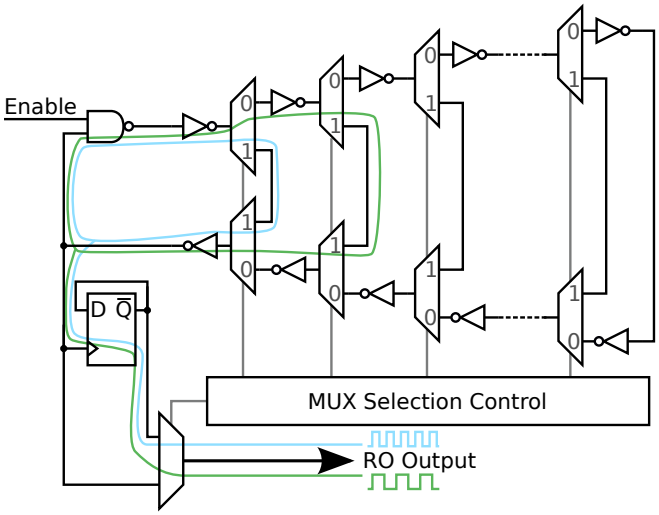


Figure 3.4: The adjustable RO design with demultiplexers (top) and multiplexers (bottom) inverter chain length adjustment. Two configuration examples are shown in blue and green.

Figure 3.4 shows the scalable RO module. The RO is formed by an adjustable number of inverter pairs and one additional inverter, implemented as a NAND gate. Thus, the number of inverters is guaranteed to be odd in any configuration. Apart from two inverters, each sub-module comprises the corresponding demultiplexer and multiplexer logic for length adjustment. The MUX Selection Control block implements a one-hot encoding that modifies the inverter chain length by bypassing inverters. Although the demultiplexers are optional in the design, they shut down unused inverters, so no dangling logic exists whose high switching frequency may interfere with neighboring ROs. If a demultiplexer is disabled, its output is set to a constant logic level of `0`.

The Enable input of the NAND gate allows to start and stop the oscillation by establishing and interrupting the inverter chain. If the Enable signal is `0`, the NAND gate's output is forced to `1`, independent of the other input, thus preventing the RO from oscillating. When the Enable signal is set to `1`, the NAND gate becomes sensitive to its other input, implements an inverter, and closes the loop. The additional inverter ensures that the number of inverters is odd. That leads to instability of the combinatorial loop and hence oscillation begins.

The D-flipflop at the output, wired as toggle flipflop, is optional. It divides the output frequency by two and may improve the duty cycle if selected. The frequency can either be halved by doubling the length of the inverter chain or by activating this register. The first method requires more resources compared to the second method as the register comes at little or even no additional slice usage.

The glitch circuitry is configured via the system's UART interface. It allows adjusting most parameters quickly without executing the cumbersome synthesis process repeatedly. The glitch system implements two ROs that can be configured independently of each other. This includes the length of each RO, which can be set in odd steps between 3 and 255. Additionally, the toggle flipflop at each RO output can be enabled and disabled as required. The glitch insertion module is also configurable. The duration of the clock glitch is set in steps of 10 ns between 10 ns and up to several microseconds. This parameter defines the duration during which the original clock signal is replaced by the fuzzy glitch.

The design was created with flexibility and ease of experimentation in mind. Solely the number of implemented sub-modules, defining the frequency span which the RO can deliver, is fixed during the synthesis of the VHDL design.

3.2.3 Experiment Results

This section evaluates the implemented system. First, the performance of the scalable RO is measured in terms of its output frequency and adjustability. Second, the overall system is evaluated and the generated glitch waveform is assessed.

Two scalable ROs, named RO 1 and RO 2, were analyzed. First, by measuring RO 1 while RO 2 remained disabled. Next, the experiment was repeated vice versa by measuring RO 2 while RO 1 was disabled. The result was expected to show a linear mapping of the inverter chain length onto the RO oscillation period.

The results are plotted in Figure 3.5. The period rises approximately linearly with the number of active inverters. However, for a three inverter RO, which is the shortest possible configuration, the period is slightly lower than expected. This minor deviation is attributed to the irregular structure of the first inverter group, which has fewer multiplexers and demultiplexers per inverter, compared to the remaining circuit. Altogether, the measurement matches the expectation and is in accordance with the design. The design shows good adjustability, as increasing the length by two inverters will raise the period by 6.5 ns on average.

The slightly increased frequency of RO 2 compared to RO 1 is a noteworthy detail. Despite both ROs being based on the same VHDL design [MCMS10], they commonly

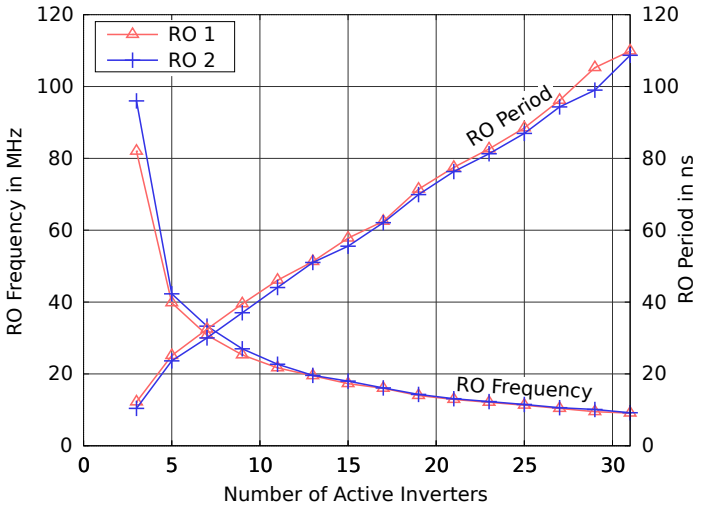


Figure 3.5: RO output frequency and period vs. inverter chain length.

exhibit frequency variations. This characteristic is beneficial for fuzzy glitch generation since the mixing of two not-precisely matching signals creates a suitable glitch signal.

After the verification of the scalable RO design, the resulting fuzzy clock glitch signal is assessed. I configured the ROs to a length of three and five inverters, corresponding to 82 MHz and 42 MHz, respectively. The glitch duration is set to 100 ns. The untampered output clock frequency is 8 MHz. For the experiment, an oscilloscope is attached to the FPGA output, the glitch is triggered, and recorded.

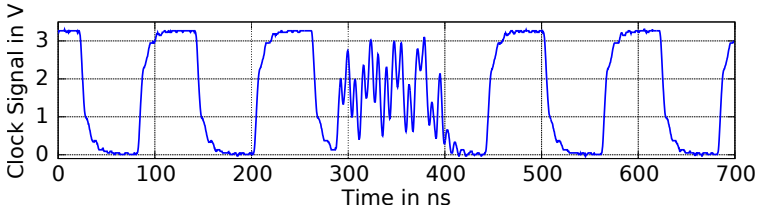


Figure 3.6: A fuzzy glitch with a duration of 100 ns, interrupting an 8 MHz clock signal.

Figure 3.6 displays the generated fuzzy glitch. As long as the glitch signal is not active, the 8 MHz clock remains unaltered. Upon the trigger event at about 280 ns, the fuzzy glitch signal replaces the clock signal for 100 ns. As expected, the signal is not a square wave anymore and has no regular shape. Furthermore, it does not fully span between 0 V and 3.3 V and is far from being a high-quality *digital* clock signal. The signal is distorted because the FPGA board cannot drive a signal with such high-frequency signal components sufficiently well. Upon repeating the experiment,

the glitch is different each time, since the internal state of each RO diverges over time and both RO oscillations run independently from each other.

3.2.4 Limitations

Giving a definitive statement regarding the quality of randomness contained in the fuzzy glitch is difficult if not impossible. In general, there is no metric which allows a straightforward evaluation of glitch randomness quality since it is a highly complex *analog* signal with different requirements compared to random numbers. Standard random number quality tests usually operate on digital data and may not yield valuable results in this case. For example, if the next glitch is sometimes predictable based on previous glitches, the glitch setup is not limited, but randomness tests would likely fail.

Furthermore, evaluating the glitch quality by means of a theoretical model appeared to be unsuitable since the glitch generation system was intentionally designed to exhibit chaotic behavior which can hardly be modeled. Both ROs are free-running oscillators whose frequencies fluctuate quickly, causing significant jitter. Moreover, environmental conditions, such as temperature, have shown to affect the glitch, making it even less predictable. Altogether, the fuzzy glitch contains randomness created by a chaotic source, which is highly influenced by external and internal parameters.

3.3 Exemplary Attack on an ARM Cortex-M0 Based Microcontroller

The effectiveness of the implemented clock glitch setup against embedded systems was proven in practice. To demonstrate this, I set up a sample system based on an ARM Cortex-M0 microcontroller. The microcontroller runs a firmware that allows assessing the system's reaction to glitches. The glitch duration was the primary parameter that was varied during the test.

3.3.1 Experiment Setup

The experiment's setup is shown in Figure 3.7. The Spartan-3E FPGA board, which generates the fuzzy glitch, is on the right-hand side. The glitch parameters are configured via a UART interface of the FPGA. The red-black colored twisted pair of wires at the bottom of the image connects the FPGA to the microcontroller's clock input.

The STM32F0 Discovery evaluation board, placed on the left-hand side, resembles an embedded system under attack. It comprises an STM32F030 microcontroller based on an ARM Cortex-M0 core. The effects of the glitch on the microcontroller are examined via the microcontroller's UART interface.

The microcontroller is configured to run on the external 8 MHz clock source, controlled by the FPGA. The PLL is disabled; thus, the external clock drives the Cortex-M0 core directly.

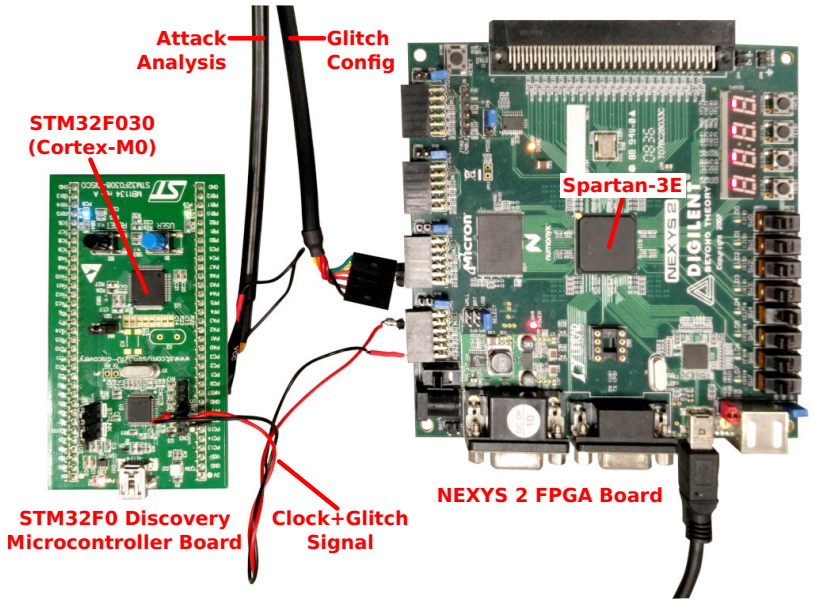


Figure 3.7: Test setup for fault injection on embedded systems via the fuzzy glitch.

3.3.2 Experiment Firmware Under Attack

In most systems, an embedded system performs several tasks in quick succession, but this impedes a thorough analysis of glitch effects. Although faults can still be injected, their external visibility might be limited. For example, if a glitch modifies processor flags, the next instruction might overwrite them without any effect taking place. A similar problem may occur when a variable is changed to a different value having the same interpretation. E.g., `1` and `2` both represent the boolean value `TRUE` and such a fault may not cause any effect. Thus, the external visibility of glitch-induced faults highly depends on the currently executed instruction, task, and firmware. As a consequence, results may not be conclusive, their interpretation becomes difficult, and reliable reproduction is hardly possible.

To overcome these issues, I developed a tailored bare-metal firmware without an operating system. The firmware has an infinite loop that continuously runs the code to be tested; hence, there is only a single task. This ensures that the code under test is executed during the glitch and the firmware additionally helps to collect information on faults afterward. The body of the loop can be chosen freely, in this case I included instructions for computations, comparisons and branches — basic building blocks of common control flow statements like loops and conditional expressions. The firmware was implemented in a way that an erroneous execution of any instruction of the loop will have a recognizable effect. This part of the firmware was manually implemented in assembly language [STM12] to maintain full control over the generated instructions.

Listing 3.1: Excerpt from the microcontroller code under attack.

```
movs r0, #0x01 ;  r0 = 1
mov r10, r0    ; r10 = 1
loop1:
        movs r5, #0x00 ; r5  = 0
        add r5, r10    ; r5 += 1
        add r5, r10    ; r5 += 1
        add r5, r10    ; ...
        add r5, r10
        add r5, r10
        add r5, r10
        add r5, r10
        add r5, r10
        add r5, r10    ; ...
        add r5, r10    ; r5 += 1
        cmp r5, #0x0A  ; (r5 == 10)?
beq loop1 ; if true → goto loop1
```

An excerpt from the ARM thumb assembly code is shown in Listing 3.1. The code is specially targeted towards `add` instructions that implement an addition of two 32-bit registers. To be able to differentiate between glitch effects on `add` instructions and loop-related instructions, only the latter ones shall modify the processor flags. Therefore, the code requires the `add` instruction instead of `adds` which modifies the flags. Due to an instruction set limitation, only the `adds` instruction is available when using two low registers (`r0` to `r7`). In contrast to this, the `add` instruction which does not modify any flags exists solely for an addition between one low and one high register, e.g., `r5` and `r10`.

All this is taken into account by the code under test. First, the registers `r0` and `r10` are set to `0x01`. Due to an instruction set limitation, a move of an immediate value into `r10` is impossible, thus, the value is loaded into `r0` and then moved from `r0` into `r10`. Next, the code enters the loop, which resets `r5` to `0x00` and updates the flags to establish a well-defined initial state. The following ten instructions are identical `add` instructions that implement an addition of `r5` and `r10` whose result is written back into `r5`. Each instruction increments the register `r5` by `0x01` without updating the processor flags. After these ten instructions were executed, `r5` can be expected to equal ten. This is verified in the next step, which compares `r5` to `0x0A`. If both values are equal, the branch-if-equal (beq) instruction will jump back to the beginning of the loop, denoted by the label `loop1`. This is an infinite loop as `r5` will always equal ten after it has been incremented ten times.

The loop has a very high sensitivity to incorrect instruction execution due to glitch-induced faults. For example, if register `r5` is not correctly reset to `0x00` or if one incrementation is missed or executed twice, the final value will not equal `0x0A`. The

subsequent compare will fail, the branch will not be executed, and the control flow is diverted out of the infinite loop. Similar effects may happen, if the compare fails or if the branch is not correctly taken.

3.3.3 Glitch Effects and Analysis

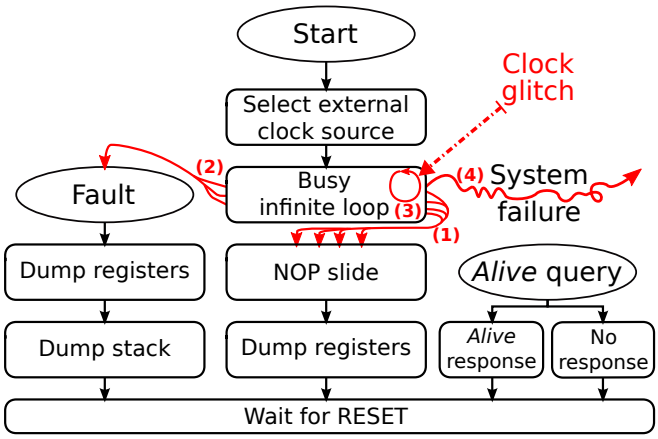


Figure 3.8: The microcontroller firmware under attack including possible glitch-induced faults.

The microcontroller runs an extended firmware, which allows a deep analysis of the glitch effects on the infinite loop. The firmware structure is depicted in Figure 3.8. When the system starts, it selects the external clock source and enters the infinite loop. The glitch can cause four different effects on the firmware:

1. **Success:** The system leaves the infinite loop and executes code following the loop.
2. **Hard fault:** A fault occurs and the system jumps to the corresponding fault handler.
3. **No effect:** No effect on the system is visible and execution continues unaffected.
4. **Crash:** The system crashes and code execution stops immediately.

In the **success** case, the system leaves the loop and drops into the No Operation (NOP) slide that follows the infinite loop. The NOP slide is a large region of code containing tens of subsequent NOP instructions. Such code is necessary to correctly catch a perturbed control flow even if multiple instructions are skipped on exiting the loop. Following the NOP slide, the microcontroller dumps all registers via its UART interface.

In the **hard fault** case, the glitch causes the system to jump into the fault handler or another interrupt handler. These cases are caught by a default-handler, redirecting

each such event to the dumping function. Upon a fault or interrupt, this function is called and creates a register dump that is transmitted over the UART interface. Additionally, a stack dump is extracted, showing the triggering instruction and provides extended information.

In the success and hard fault case, the microcontroller automatically transmits a register dump upon exiting the infinite loop. However, if the microcontroller sent no status message, it either was unaffected by the glitch and is still running correctly or it has crashed and ceased operation. Both cases are identified by sending an "Alive?" query to the microcontroller and checking whether a reply is sent.

In the **no effect** case, the microcontroller was not affected by the glitch and continues operating. Thus, it responds to the "Alive?" request and proves that it is still fully functional.

In the **crash** case, when the system has ceased operation, no response will be received on an "Alive?" request. This shows that the system is in the lockup state; hence, the processor has stopped code execution and requires a hard reset.

3.3.4 Evaluation and Results

In this section, the clock glitch attack is executed on the embedded system and the effects are analyzed. For the attack, the ROs are configured to a length of three and five inverters, which is the same setting presented in Subsection 3.2.3. In the experiment, the duration of the fuzzy glitch was varied, starting with 20 ns. It was then increased in steps of 20 ns up to a maximum duration of 800 ns, totaling in 40 different settings. For each of the 40 settings, I repeated the glitch attack 800 times and evaluated the result for each repetition.

The register and stack dumps show various effects caused by the fuzzy clock glitch. In most cases, register `r5` was affected, because the loop operates primarily on this register. Sometimes the incrementation of `r5` fails and the register remains at a number less than ten, e.g., `0x06`. Thus, either several instructions were skipped or the incrementation failed. I observed also the opposite, when `r5` was increased above the expected value, e.g. up to `0x14` or `0x0C`. This may have been caused by a skipped reset of `r5`, by a multiple execution of the `add` instruction, or by miscalculation. In a few cases, the `r5` register value was inconclusive, e.g., `0x08000606` or `0x00800000`. This is a hint on a major malfunction of the processor due to the glitch that lead to a fault in `r5`.

There were some special cases when `r5` equaled ten, but nevertheless, the system left the loop. An analysis of the status register reveals that the compare instruction erroneously reported `r5` as being different from ten, although this was not the case. This caused the flags to be set incorrectly, the following branch was not taken, and the loop was left.

The branch instruction has also failed on some occasions. Although `r5` equals ten and the status register shows that the compare of `r5` was successful, the branch is not taken.

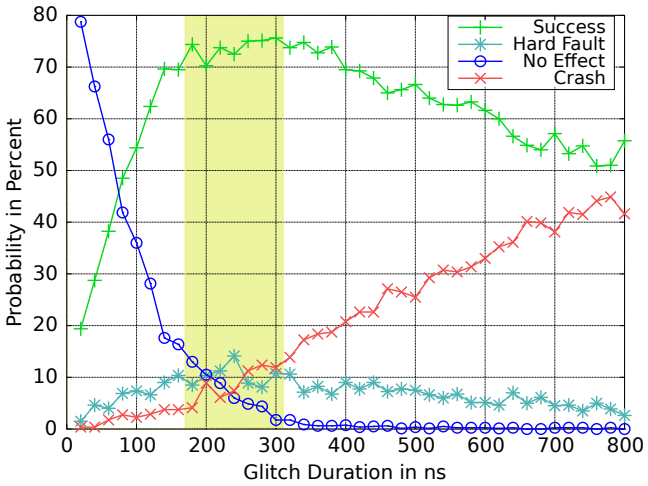


Figure 3.9: Fuzzy glitch effects vs. glitch duration.

The NOP slide after the infinite loop has proven to be very useful. The code was manually instrumented and the results showed that the glitch sometimes causes the first few instructions of the NOP slide to be executed incorrectly. Nevertheless, the analysis of these cases was still successful, since a fault caused in the NOP slide does not cause any effect. NOP is implemented via `mov r8, r8` and `r8` is a register whose value is irrelevant.

In a few cases of around 10 % on average, the glitch causes a hard fault. I observed no other spurious interrupts during my experiments. An examination of the stack and the return pointer in the case of a hard fault showed that there is no particular instruction that causes the hard fault during a glitch. Instead, the hard fault happens randomly. The reason for this behavior is unknown but can be speculated. One possibility is an access to an invalid memory address, caused by a corruption of the program counter which holds the currently executed memory address. Another explanation is an error during instruction fetch when an invalid instruction is given to the processor.

The fuzzy glitch can also crash the system. This was observed in less than 5 % up to 45 % of all cases, depending strongly on the glitch duration. A lockup may result from overstressing the system via the glitch. In this case, the state is unrecoverable since there were multiple or severe data faults.

3.3.5 Statistical Evaluation

A plot of glitch effectiveness vs. duration is depicted in Figure 3.9. The plot shows that very short glitch durations have **no effect** on the system in about 80 % of all cases. Upon increasing the glitch duration, the ratio of unsuccessful attempts declines quickly. At 200 ns duration, the glitch causes an effect in approximately 90 % of the cases.

As the glitch duration approaches 180 ns, the probability for **success** rises quickly. The optimal glitch duration spans from 180 ns up to 310 ns, where success is achieved in about 75 % of the attacks while maintaining little crash probability. This region of optimal glitch effectiveness is highlighted in Figure 3.9. If the glitch duration is further increased, the success ratio begins to decline again as the system crashes more often.

The probability for a system **crash** increases approximately linearly from 12 % at 300 ns to up to 45 % at around 800 ns. This increase takes place at the cost of the success ratio, thus operating in this region is not advised for this microcontroller. The prolonged glitch leads to an increased number of faults that finally crash the system.

The probability for a **hard fault** stays below 15 % in most of the cases. Since an *add* instruction can never cause a hard fault, the error must emerge somewhere else. One reason might be an erroneous instruction decoding which, for example, could lead to an unaligned or otherwise illegal memory access.

The plot shows that although a random process generates the glitch, its effects on the microcontroller are still under control and are adjustable by varying the glitch duration. The results of the measurements are conclusive and match the expectation: Neither very short nor excessively long fuzzy glitches have high success; there is an optimum in between. In this setup, the region of optimal glitch effectiveness spans from 180 ns to 310 ns.

3.4 Future Developments

My experiments were limited to a system that directly feeds the external clock to the processor. However, PLLs are widespread in microcontrollers and experience shows that they are usually activated to achieve a higher clock frequency and more computational power. This introduces a filter between the clock input and the processor; thus, clock glitches cannot propagate through the system directly. The PLL comprises a Voltage-Controlled Oscillator (VCO) and a control loop that ensures that the PLL's output clock is phase and frequency locked to the input clock.

Additional experiments were conducted on microcontrollers that were running on an external clock signal, which was fed to a PLL before reaching the core. The expectation was that a *single* clock glitch could slightly disturb the system; however, the PLL's limited control loop bandwidth prevents any large deviations in clock frequency or phase that would cause malfunctions. This assumption was confirmed for the STM32F030 and for the Infineon XMC4500 microcontroller, which both contain an ARM Cortex-M core.

As an alternative to a short glitch, a several hundred microseconds long fuzzy glitch was applied to the system. Since a glitch is a high-frequency signal, the control loop will increase the VCO's output frequency until it exceeds the core's specified maximum clock frequency. This causes faults in the processor similar to the faults described in this chapter; thus, clock glitch attacks are also viable on systems running with an externally fed PLL. Although the temporal location of the fault cannot be controlled precisely, the experiment shows that a PLL will not fully protect an embedded system

against clock glitch attacks. Nevertheless, such attacks are strongly hampered by the PLL.

Due to these results, future work should continue the topic and focus on increasing glitch efficiency on PLLs by combining such attacks with supply voltage glitches and extreme environmental conditions, i.e., low or high temperatures. Additionally, countermeasures have to be evaluated and developed with respect to the fuzzy glitch.

3.5 Conclusion

In this chapter, I presented a novel class of clock glitch fault attacks that employs a fuzzy clock signal. I demonstrated that the proposed design could be implemented on an FPGA. The experiment showed that the fuzzy clock glitch could successfully attack an embedded system that comprises the ARM Cortex-M0 based microcontroller STM32F030. The injected faults were analyzed in detail with the help of a custom microcontroller firmware that was optimized towards an easy analysis. The results demonstrate that the fuzzy glitch reaches up to 75 % glitch effectiveness in the optimal case and remains at over 50 % across a wide range of parameters. Altogether, my results prove that this alternative approach on clock glitch attacks is valid and feasible.

The fuzzy glitch setup has demonstrated to be helpful for embedded system penetration testing. As previously mentioned, only very little or no information is publicly available on the concrete implementation of security features of an embedded device or a microcontroller. Tests become even more demanding since black box tests are often explicitly requested and very prevalent. Thus, choosing a good set of parameters for *precise* clock glitch attacks is often difficult. This is not a purely theoretical consideration — the fuzzy glitch has successfully proven its power in a penetration test during an industry project. Therein, it helped to circumvent a fundamental security feature, the data readout protection of an embedded device with a state-of-the-art microcontroller.

Thus, a secure embedded system also needs to be protected against hardware attacks, such as glitches. These non-invasive attacks are still comparably simple to perform as no major physical manipulation on the system is required. However, many devices are not prone to such an attack, for example, if they run from an internally generated clock that is not accessible from the outside. Nevertheless, these devices can still be broken by semi-invasive and invasive hardware attacks. Thus, a deeper look into the hardware is required for holistic embedded device security.

Chapter 4

Shedding too much Light on a Microcontroller's Firmware Protection

The previous chapters demonstrated that software vulnerabilities and non-invasive hardware attacks could undermine embedded system security. For the cloud camera systems in Chapter 2, software vulnerabilities were exploited and access to communication channels and the cloud system was gained. In Chapter 3, faults were injected during the execution of the firmware, leading to undesired behavior. In both cases, the presented attacks target the firmware by exploiting vulnerabilities or causing undesired behavior.

However, any security features of the underlying hardware have not been analyzed so far. Even if a system is well-protected in software and its hardware is not susceptible to clock glitches, e.g., it runs solely on an internal clock, such a system may still not be secure. Most systems feature some sort of debug interface that allows full read and write access to the entire device. This interface is only active during development and disabled in the final product to prevent unauthorized access. Nevertheless, attackers put great efforts into detecting flaws in debug interfaces, as re-enabling the interface immediately nullifies device security. The firmware, often containing cryptographic keys or similar data, becomes extractable, thereby exposing such secret data to an attacker.

This chapter presents an evaluation of the debug interface of the state-of-the-art STM32 microcontroller family and focuses especially on the STM32F0 series. Altogether, three design and implementation weaknesses were discovered in the STM32F0 series, which are described in detail. I presented parts of this chapter previously at the *11th USENIX Workshop on Offensive Technologies (WOOT)* in 2017 and the related paper was published in its proceedings [OT17]. Additionally, I requested a CVE-ID for one of the discovered vulnerabilities, resulting in `CVE-2017-18347` being assigned to the issue [MIT18].

These results have sparked interest also in the non-scientific community via online discussions and articles, e.g., in Hackernews [kw718], in talks about microcontroller security [Tie18], and in discussions of secure key cards [Bea17]. The results were also mentioned by Roth et al. who discovered additional vulnerabilities in another STM32 family deployed in hardware Bitcoin wallets [RND18]. Altogether, this underlines the impact and practical relevance of the topic.

To spark further work, I provide the tools developed for this chapter as supplementary material at `https://science.obermaier-johannes.de/firmwareProtection`. The basic parts, comprising my own code, are provided under the MIT license. The material includes all required scripts, ELF, and source code files for the Cold-Boot Stepping and debug interface exploit Proofs of Concept (PoCs), as well as sample firmware to be attacked.

This work was done in cooperation with Stefan Tatschner. He supported writing the paper and the creation of the PoC for each vulnerability.

Please note that some of the experiments pose inherent hazards due to high energy pulse lasers, Short-wave Ultraviolet (UV-C) radiation, hot surfaces, aggressive chemicals, and especially the combination thereof. The experiments were conducted with the utmost care, sufficient practical experience, and adequate safety measures, including protective gear.

4.1 Overview

Microcontrollers are deployed in countless embedded systems, ranging from industrial systems over automotive control units up to end-user devices. As their capabilities increased over the last few years, the complexity of their tasks rose and hence their firmware got more sophisticated.

While systems in the past were deployed mostly in stand-alone applications, current devices may be part of a sensor network and may interact with the Internet-of-Things. These systems contain valuable Intellectual Property (IP), such as sophisticated measurement and control algorithms. Furthermore, devices may be license-locked and contain cryptographic material such as keys. Creating those devices and their firmware is accompanied by substantial investments in hardware and software development.

At the same time, gaining access to these assets became more worthwhile for an adversary. Product piracy has emerged to a large threat, where competitors precisely clone products and cause damage to the affected company [Ign16]. This is not only limited to financial damage, but counterfeit products of lower quality may also impact the reputation of a manufacturer.

To counteract product piracy, most manufacturers of microcontrollers added firmware readout protection to their devices that is a component of hardware security. When activated, this feature prevents an unauthorized party from obtaining the contents of the device's internal memory. In most implementations, the system's programming interface is disabled entirely, or access is at least strictly limited, e.g., by blocking access to flash memory. For some devices, such a configuration may be

permanent; for others, access can be re-enabled at the cost of losing all memory contents. There exist countless flavors in between, depending on the targeted level of security which the manufacturer tries to achieve.

4.1.1 Related Work

However, these mechanisms are not always able to keep up with novel attacks and are broken by attackers. Hobbyists, as well as professional researchers, have circumvented several systems in the past, often due to the underlying insufficient hardware security [SDK+13, Var11]. Especially, Skorobogatov et al. have shown that many security concepts of hardware manufacturers may often not cover all corner cases [SA03], have weaknesses [Sko10], hidden functions, or even backdoors [SW12].

Apart from scientific research on hardware security measures in microcontrollers some companies also try to circumvent readout protection. There exist online offers for firmware extraction for a multitude of microcontrollers. In contrast to researchers, these malicious attackers operate covertly without publishing their exploits; thus, vulnerabilities often survive in the long-term. As the manufacturer might not be aware of the underlying issue, no fix can be developed. In contrast to this, if vulnerabilities are communicated to the manufacturer in a professional manner and with enough detail, those issues can be fixed in the next chip revision and, in some cases, workarounds can be published.

Many older microcontrollers were extensively tested for security and often exploited [Sko00]. As a consequence, the industry shows a growing interest in recent microcontrollers, for example, the ARM Cortex-M based STM32 series. Their wide deployment finally raised interest in their level of security, mostly in terms of firmware protection.

Unfortunately, there were no security testing results for the STM32 series publicly available in 2017. Giving a statement regarding the protection was impossible, despite often being requested. Therefore, I undertook a thorough security analysis of the STM32F0 series regarding its security concept for firmware protection. I focus on the achieved level of security and on the effort for the exploitation of any discovered weaknesses. To evaluate the device, I start with a high-level conceptual analysis of the available security configuration options and gradually dig deeper into the hardware implementation. In doing so, I cover a broad spectrum of attacks that combine software-based and hardware-based approaches.

Since the family of STM32 devices is large and contains several sub-groups, I limit my analysis primarily to the STM32F0 series. It features entry-level ARM Cortex-M0 microcontrollers at moderate cost for a wide range of commercial products. Wherever possible, an estimation of the impact on other STM32 series is given. All findings were shared to the full extent with STMicroelectronics before publication via a coordinated disclosure approach.

4.1.2 Contributions

This chapter presents my following contributions:

- A conceptual analysis of STM32F0 security core components.
- The discovery of three weaknesses and a presentation of corresponding ideas for their exploitation, i.e.,
  - **Cold-Boot Stepping:** Enforcement of single-stepping under limited debugging capabilities in Readout Protection (RDP) level 1.
  - **Security Downgrade:** Leveraging a lock-level design issue to downgrade the firmware protection setting.
  - **Debug Interface Exploit:** Discovery of a race condition in the debug interface in RDP level 1 to read out the entire firmware.
- A discussion of the impact of each vulnerability on the overall system security and possible countermeasures.
- A PoC is provided for each weakness which develops them into three vulnerabilities.

To encourage discussion and to ensure reproducibility of my results, the source code files for all PoCs and additional materials are publicly provided.

4.1.3 Structure

This chapter gives an introduction to the open questions of the STM32F0 firmware protection strength in Section 4.1. Next, Section 4.2 continues with a summary of the STM32F0 security concept. The security analysis and its detailed results are presented in Section 4.3. It comprises three subsections, each presenting one conceptual analysis, the identification of a weakness, and a corresponding PoC that demonstrates the vulnerability. Therein, Subsection 4.3.1 presents the cold-boot stepping method, Subsection 4.3.2 reveals the security downgrade issue, and Subsection 4.3.3 explains the debug interface exploit. The conclusion summarizes the results and puts them into the context of holistic system security in Section 4.4.

4.2 STM32 Firmware Security Concept

The STM32's flash memory readout protection is the key component of its security concept [STM17] and is incorporated in every device. It protects the system's firmware, stored in flash memory, against an unauthorized readout. Depending on the chip series, additional security mechanisms such as a Memory Protection Unit (MPU) and privileged/unprivileged execution modes [STM16b] are available. Altogether, such technologies aim at enhancing the system's security.

The flash memory readout protection constitutes the root of system security since it prevents an attacker from extracting the firmware. If this mechanism fails, the firmware can be read out, manipulated, and copied onto a new device. Hence, breaking the readout protection mechanism will fully compromise security.

4.2.1 Flash Readout Protection Levels

The flash memory readout protection concept of most STM32 microcontrollers comprises the three selectable RDP levels 0, 1, and 2 [STM17]. A higher level corresponds to increased protection, while the capabilities of the device's debug interface become more and more limited. The levels have the following characteristics:

- **RDP level 0** is the initial configuration and imposes no restrictions. The debug interface is active and allows full access to the device. This level is intended to be selected during development and testing.
- **RDP level 1** keeps the debug interface active but restricts access to the flash memory. As soon as a debugger connects to the system, the flash memory is locked. It can neither be read out via the debugger nor indirectly via DMA nor is the processor able to execute code from it. An upgrade to RDP level 2 is possible at any time to increase protection. Downgrading to RDP level 0 is foreseen but comes at the cost of losing all flash memory contents.
- **RDP level 2** is the most restricted one but provides the highest security. Debug access is entirely disabled by shutting down the debug interface permanently. Hence, no debugging functions are accessible and the flash memory cannot be read out. This configuration is irreversible and cannot be downgraded to any other level.

Although RDP level 2 offers the highest level of protection, RDP level 1 is still in use. Experience shows that companies dislike the idea of locking down their devices completely since it prevents analyzing buggy and failed devices. Furthermore, even STMicroelectronics warns in their datasheet [STM17] that defective part analysis is impossible on devices set to RDP level 2. Additionally, RDP level 2 is not available on all devices, e.g., the STM32F1 series supports only RDP level 0 and 1. Altogether, this results in devices set to RDP level 1. This protection level raises particular interest regarding whether this configuration can be considered secure or insecure — a question that is investigated in detail in Subsection 4.3.1.

4.2.2 Readout Protection Design

The RDP level is part of the microcontroller's system configuration that is stored in the dedicated *option byte* section [STM17]. Therein, the three available RDP levels are persistently encoded via 16 bits of flash memory.

Since 16 bits are more than sufficient to encode three states, there is room for redundancy. A well-chosen encoding is crucial for maintaining security even under attacks. It raises the bar significantly because manipulating a single bit will not suffice

to alter the protection level. Several bits will have to be flipped during an attack to overcome a configuration storage that includes redundancy. For example, other already broken systems have only a single bit to distinguish between locked and unlocked configurations [SDK+13].

In the STM32 series, those 16 bits are represented by two bytes, which are named *RDP* and *nRDP*. In every intended configuration, nRDP equals the bitwise complement of RDP. Table 4.1 shows the mapping of each RDP and nRDP setting to the corresponding RDP level. RDP level 0 and 2 are each represented by exactly one pair of bytes. All other configurations, including non-complementary pairs of bytes, default to RDP level 1.

nRDP	RDP	Resulting protection
`0x55`	`0xAA`	RDP level 0
Any other combination		RDP level 1
`0x33`	`0xCC`	RDP level 2

Table 4.1: Flash readout protection settings of the STM32F0 series [STM17].

RDP level 2 and 0 have a Hamming distance of eight, thus, an attacker would have to modify eight bits to switch from the most-secured state to the unsecured state. Although this mapping appears to be robust at first glance, I discover a design issue and defeat the concept in Subsection 4.3.2.

4.2.3 Flash Protection Logic

While RDP level 0 and RDP level 2 are straightforward in their documentation, i.e., the debug interface is either fully accessible or not accessible at all, the technical details of RDP level 1 are only sparsely described. According to the datasheet, the microcontroller supports two modes of operation in RDP level 1, named "User mode" and "Debug [...] mode" [STM17].

The microcontroller initially runs in unrestricted user mode and switches over to debug mode when a debugger is attached to the device. Once the system enters debug mode, any access to the flash memory is denied for the debugger and for the microcontroller. The datasheet also claims that flash memory is then "totally inaccessible" and "[...] even a simple read access generates a bus error and a Hard Fault interrupt" as a consequence [STM17]. The datasheet does not contain any information about the inner workings of the flash memory protection mechanism and under which conditions *exactly* debug mode is entered. Furthermore, it does not mention accessibility to any other modules, such as the Static Random-Access Memory (SRAM) and peripherals, which are present on the same bus as the flash memory. Altogether, this blurry image of the flash memory readout protection implementation encourages a deeper analysis and reverse-engineering — whose alarming results are presented in Subsection 4.3.3.

4.3 Attacking the Security Concept

The key components of the STM32 security concept require a detailed investigation. Subsection 4.3.1 presents an analysis of RDP level 1 security, Subsection 4.3.2 deals with the strength of the physical implementation of RDP level 2, and Subsection 4.3.3 covers limitations of the flash memory protection logic.

4.3.1 Cold-Boot Stepping

The datasheet assures that the flash memory is read-protected in RDP level 1 while a debugger is attached [STM17]. One may easily overlook that this statement refers only to *flash* memory but not to SRAM or peripherals. No information is given regarding access permissions on these components in RDP level 1.

A microcontroller is programmed with a sample firmware and set to RDP level 1 to investigate these missing details. The firmware contains a loop that toggles an LED; hence, as long as the program is running, the LED is blinking. Next, a debugger is attached to the microcontroller and the blinking stops immediately. Hence, the microcontroller was halted, since its core could no longer fetch any instructions from the flash memory. This behavior is in accordance with the datasheet and expected so far.

Nevertheless, the debugger has some access to the system. Although the flash memory is not readable, the entire SRAM is accessible with application data still in place. Attaching the debugger does not trigger zeroization or any comparable protection mechanism; thus, the data in SRAM remains intact. Consequently, the question arises to what extent the exposed data in SRAM poses a threat to system security.

4.3.1.1 Concept of SRAM Snapshot Generation

The readable SRAM with all data still in place can be considered a Cold-Boot scenario [HSH+09]. The system has ceased operation, but data of previously running applications still resides in memory.

If secret data is present in the SRAM, it is exposed immediately; thus, the threat is self-evident. As a countermeasure, typical implementations of cryptographic algorithms keep keys in RAM only during usage down to a few milliseconds or even less. Keys or related data cannot be retrieved by *manually* attaching a debugger, as hitting the right moment in time is practically impossible. Furthermore, the microcontroller has several kilobytes of SRAM whose memory map is usually unknown to the attacker and hard to reconstruct from scratch. Stepping through the firmware is also impossible in RDP level 1; hence, the internal control flow remains hidden.

As a countermeasure against this protection, I present Cold-Boot Stepping (CBS), a method to precisely take snapshots of the system's SRAM at the moment of choice. The idea is that the system is effectively halted every few clock cycles, the SRAM is read out, and a snapshot is created thereby. This general idea is partially derived from a very specific approach [Var11] applied on a PIC microcontroller in 2011.

The approach is outlined as follows. At first, CBS brings the system into a well-defined initial state, e.g., by causing a power-on reset. Next, the firmware is allowed to run for a precisely controlled duration and is then stopped. Then, the debugger reads out the entire SRAM and stores a snapshot. Afterward, the next iteration starts with an increment in the execution duration of a few additional clock cycles.

By tracking the changes in SRAM between snapshots, one can reverse-engineer the basic control flow, find out the usage of specific addresses, and may also discover briefly visible secret data. This is similar to stepping through the firmware, since only a few additional instructions are executed from one SRAM snapshot to the next one.

This general CBS approach can be adapted to STM32 microcontrollers by applying several tricks to control code execution and SRAM data freezing.

1. **System Reset:** At the beginning of each iteration, the system is brought into the initial state. A full reset, i.e., a power-on reset, is triggered by a power cycle to re-enable access and execution from flash memory.
   (a) **Power OFF:** Initially, the system is powered down.
   (b) **Assert Reset:** Reset is asserted, i.e., pulled low, before the system is powered up. This allows the system to power up without starting firmware execution immediately.
   (c) **Power ON:** The system is powered up under reset. The system's internal circuitry is ready to start execution, but is inhibited by the reset signal.
2. **Run System for** $(n \cdot T)$**:** The firmware is allowed to run for a duration of exactly n steps of duration T.
   (a) **Deassert Reset:** Releasing the reset signal, i.e., pulling it high, starts firmware execution.
   (b) **Wait for** $(n \cdot T)$**:** The firmware runs for a duration of exactly $(n \cdot T)$. The firmware will advance to the moment of interest.
   (c) **Assert Reset:** As soon as the time is up, reset is applied again by pulling it low. This stops code execution and resets the processor, but data in SRAM persists and is frozen.
3. **Dump Memory:** The SRAM is read out by the debugger and is written to a file.
   (a) **Start Debugger:** The debugger attaches to the microcontroller under reset. Next, the debugger takes control and forces the processor into halt mode.
   (b) **Deassert Reset:** Reset is deasserted to startup the Advanced High-performance Bus (AHB) [ARM06] again, thereby enabling SRAM access. The SRAM's state is preserved because the system was halted by the debugger and did not continue execution.

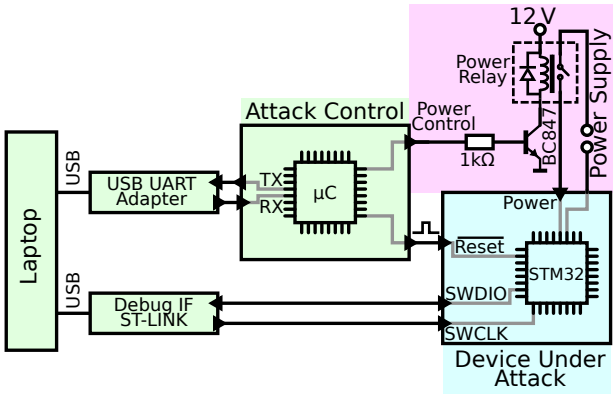


Figure 4.1: The Cold-Boot Stepping setup schematic.

(c) **Dump SRAM:** The debugger reads the SRAM and writes the data into a file.

4. $\mathbf{n = n + 1}$**:** After these steps have been completed, the setup is prepared for the next iteration. To do so, the duration of $(n \cdot T)$ is increased by one step to $(n + 1) \cdot T$.

5. **Repeat:** Each iteration is started with an increased execution duration. Iterations are performed until all instructions of interest have been covered.

The CBS setup schematic is shown in Figure 4.1. The laptop on the left-hand side orchestrates the attack. An *ST-LINK* debugger connects the laptop with the debug interface of the device under attack. Additionally, the laptop has a UART connection to the attack control board. The microcontroller on the attack control board handles the reset signal and the power supply of the device under attack. Switching the power on and off can be difficult, e.g., if the device under attack is an AC powered system. To achieve a general setup, I employ a relay that enables the attack control microcontroller to switch high voltages such as 230 V AC power.

CBS depends on precise timing with microsecond accuracy that cannot be provided by a standard computer system. An experiment showed that even the Linux kernel with the real-time patch exhibits too much timing jitter, resulting in an unsatisfying reproducibility. Consequently, the duration of execution cannot be controlled by the debugger itself, as neither the computer system nor its USB interface guarantees the required timing accuracy.

The attack control board, shown in Figure 4.1, is employed for accurate timing. Its microcontroller provides a low-jitter timing source as the system is solely dedicated to duration control. All jitter-generating features, such as interrupts, are disabled, thereby enabling fully deterministic timing behavior. At the beginning of each iteration, the computer system transmits the desired execution duration to the attack control board. Next, the attack control board *autonomously* starts up the device under attack, waits

for the configured duration, and halts execution again. Thus, all timing-critical tasks are exclusively handled by the attack control board and any delay from the computer system has no influence. This setup provides sub-microsecond accuracy that allows precisely stepping to the attacker's moment of choice.

4.3.1.2 Proof of Concept: CBS Firmware Extraction

As the attack remains theoretical up to this point, a PoC is developed. A variant of the attack was created in which a fully automatized setup indirectly extracts firmware from a device in RDP level 1. This anonymized example is based on a real case, in which I partially extracted the firmware from a commercial product in a similar configuration.

In many embedded systems, a bootloader verifies the application firmware integrity before its execution. The integrity check is commonly based on a checksum algorithm, such as the Cyclic Redundancy Check (CRC) [PB61]. The algorithm iterates over each byte of the firmware and may, depending on the implementation, store intermediate CRC results in SRAM.

If an attacker gains knowledge of subsequent intermediate results of a CRC computation, the input data can be reconstructed trivially. At first, the attacker must guess the CRC polynomial and related parameters of operations, such as its initial value. Since there are only a few common CRC configurations, an attacker can manually find the correct one.

In order to compute one byte of the original input data, the two corresponding CRC intermediate values are required. Either the CRC can be inverted or a brute-force search over all possible 256 input values can be performed. Both solutions are computationally cheap and come with negligible effort. The burst-error detection capability [War13] of the CRC32 algorithm guarantees that the solution is unique and that one skipped byte, e.g., due to incorrect snapshot timing, does not yield a solution.

This procedure is repeated until all data of interest has been obtained. The CBS method steps through the computation of the CRC, gaining access to intermediate CRC results. Based on these results, the input data of the CRC is reconstructed what reveals the flash memory contents.

Figure 4.2 displays the attack setup which is the practical realization of Figure 4.1. The device under attack is an STM32F0 discovery board set to RDP level 1. The microcontroller contains a sample firmware that computes the CRC over itself. The surrounding devices execute the attack, as previously described. For simplicity, the attack control board is based on the same hardware.

The attack runs autonomously and contains an intelligent algorithm for execution duration control since timing jitter limits the achievable timing accuracy. The laptop dynamically adjusts the execution duration, depending on the result of the last extraction iteration. If a solution for the input byte was found, the execution duration is increased as usual. If the CRC value did not change, the duration is increased by a small amount. If no solution was found since the step included more than one CRC

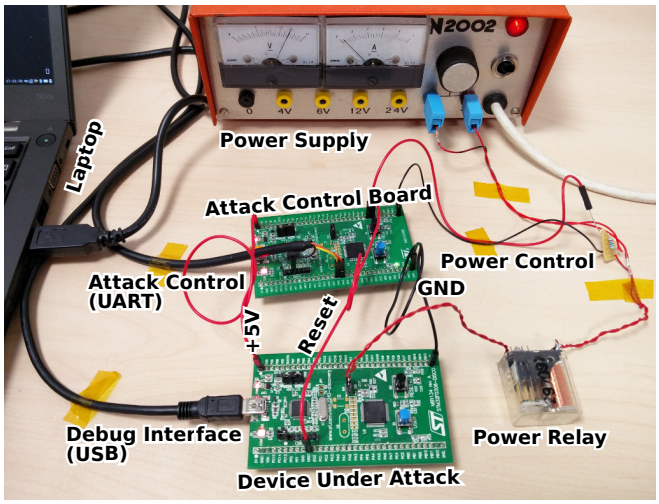


Figure 4.2: The PoC setup for CBS.

cycle, the duration is decreased a little. This constitutes a simple control loop that adjusts and corrects the execution duration on the fly.

The attack performance is limited by the several required data transfers between the laptop, the attack control board, and the device under attack. This limits the readout speed to approximately seven bytes per minute. Despite the low extraction speed, the attack is practically executable in a few days on most devices. Microcontrollers come with comparably small flash memory, such as 64 KiB for the STM32F051R8T6. For example, the small PoC firmware image was completely and correctly extracted in a few hours.

The PoC demonstrates that although RDP level 1 prevents direct reading of flash memory, unlimited read-access to SRAM serves as an invitation to many attacks. I suspect that this attack might work on any STM32 system set to RDP level 1 since the underlying concepts are very similar. Hence, the protection provided by RDP level 1 was shown to be futile under these conditions. Please note that this method emerged from an attack on a commercial product; thus, the described approach has already been successfully applied to break firmware protection in practice.

4.3.1.3 Performance Optimizations

The attack can be parallelized for speed optimization if more than one device under attack is available. Firmware extraction can be split up between them since the bytes are extracted independently from each other and the attack can be started at any memory address. Afterward, the firmware chunks are combined to obtain the entire memory image. This will reduce the required time to a fraction, e.g., approximately to 25 % if four identical devices are available.

Another optimization targets the delay introduced by the debugger and data transfer. The debugging software OpenOCD and ST-LINK require up to several seconds each time to connect to the device under attack. To counteract this issue, the attack could entirely be run on the attack control board that autonomously performs the attack and also replaces the ST-LINK debugger. Thereby, the overhead would be vastly reduced and extraction performance will increase. The computational effort to reconstruct the input value for two consecutive CRC intermediate results is negligible. Only one table search with a maximum of 256 elements and a few shifts and XOR operations are required, which are trivial even for a microcontroller.

4.3.1.4 Countermeasures Against CBS

The developer has to set the device into the fully-locked RDP level 2 for the best protection against this attack. This disables the debug interface completely and without access to SRAM, the attack is infeasible. Except for the STM32F1 series, RDP level 2 is supported by most STM32 microcontrollers.

Please note that the attack does not depend on the reversibility of the CRC algorithm. The approach might even work more efficiently on hash functions, which often store the result of each round in SRAM due to the higher demand on memory. Data hashed with only one round of a hash function is usually still reversible and such intermediate values would also be accessible via CBS. Since hash functions are more computationally expensive relative to a CRC, timing accuracy requirements might even be relaxed, thereby easing the attack.

Operating exclusively on the processor's registers complicates the attack, but cannot entirely prevent it as the approach can be adjusted. An advanced debugger can stop the system with precisely timed debug commands but without asserting the reset signal. This will leave the processor registers intact and they can be read out similarly as SRAM. The approach is feasible using the debugger that I developed for another attack, described in Subsection 4.3.3.

To increase the attacker's effort further, the system can insert random delays during CRC computations. This will hinder the acquisition of memory dumps with precise timing; however, random number generation is tricky on a microcontroller. Furthermore, the state of a software-based Random Number Generator (RNG), could again be exposed in SRAM. Altogether, there is no way around RDP level 2, if the microcontroller must be well-protected as there are no comparably strong countermeasures available.

In a personal discussion with embedded firmware developers, I became aware of an additional semi-technical issue. Developers tend to have large trust in their toolchain for device protection. In 2017, the popular debugging software OpenOCD offered only a single command named "lock" to "Lock the entire flash device". Although the developers assumed to have their devices fully secured, the "lock" command does not support RDP level 2 and only activates RDP level 1. Unfortunately, this is not evident, neither in the software documentation nor in the tool's output. To counteract this issue, I submitted a patch [Obe17], which adds RDP level 2 support to OpenOCD, extends its documentation, and explicitely displays the configured RDP level.

4.3.2 Security Downgrade

The previous section demonstrated that the debug interface should not be accessible; thus, it has to be entirely disabled by setting RDP level 2 for best security. As the attack surface is thereby reduced, an attacker might then target the RDP level to downgrade security and re-enable the debug interface. According to the datasheet, this cannot be achieved using *official* means since "level 2 cannot be removed at all" and the configuration is "irreversible" [STM17].

4.3.2.1 Concept of RDP Level Downgrade

The RDP level is stored in the RDP and nRDP bytes located in the option byte memory region. At the power-on event, the option bytes are automatically loaded from flash memory and the corresponding RDP level is set.

At first glance, the physical RDP level storage seems to be robustly implemented as it employs 16 bits to store three protection levels. In theory, only two bits are necessary to encode three protection levels; thus, having 14 additional bits available for redundancy would strengthen system security. However, a closer look reveals that this is no typical error-correcting code and it has only the capability for very basic error-detection. This leads to a non-optimal mapping between the RDP levels and the option bytes, which weakens security.

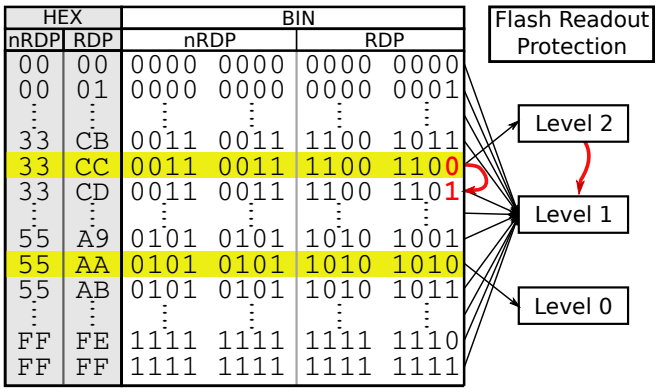


Figure 4.3: A security downgrade from RDP level 2 to 1 is achieved by flipping a single bit.

For an adversary, a downgrade from RDP level 2 to 1 or 0 is of particular interest. Figure 4.3 lists the hexadecimal and binary representation of each RDP level configuration and their corresponding option bytes. While RDP level 0 and 2 map to exactly one setting, RDP level 1 covers the remaining ones.

The most powerful attack is a transition from RDP level 2 to 0 as this entirely removes any protection and re-enables the debug interface. Since the Hamming distance of both settings is eight, an attacker has to modify eight bits in total without affecting

any neighboring bits. More precisely, four specific bits have to be flipped from 0 to 1 and four other bits must be changed from 1 to 0 while the remaining eight bits must remain intact. Such a precise modification to the option bytes is difficult, most likely only realizable with highly specialized equipment, and therefore rather expensive and effortful. Altogether, this results in a high-security margin against such an approach.

A security downgrade from RDP level 2 to 1 is an alternative by which the attacker enables the debug interface again but will only gain limited access to the system. This transition is less challenging to achieve as 65534 of $2^{16} = 65536$ possible configurations, i.e., all except two, map to RDP level 1. Due to this mapping, the minimum Hamming distance between RDP level 2 and 1 is one; thus, a single bit flip causes a downgrade in protection and there is no security margin. Furthermore, not a specific bit needs to be targeted, flipping *any* of the 16 bits suffices. Even accidentally flipping multiple bits will still downgrade security; thus, the approach for such an attack is allowed to be more coarse. Altogether, this imposes a significant weakness because the design undermines RDP level 2 security and places a minimal security margin towards RDP level 1.

4.3.2.2 Proof of Concept: UV-C Security Downgrade

Next, I demonstrate by experiment that this weakness evolves to a vulnerability. As downgrading security involves the manipulation of data, a technique is required to induce data faults [BECN+06]. The startup process of the microcontroller runs on an internal clock; thus, non-invasive clock glitch based attacks, such as presented in Chapter 3, are impossible. Two alternative semi-invasive options are electromagnetic [MDH+13] and optical fault injection [SA03] which both require access to the device's die. The latter method projects light onto the chip to introduce energy at the desired location. As the photons interact with the semiconductor, the targeted region on the chip starts to malfunction. Depending on the attack parameters, this causes temporary or permanent data faults.

Figure 4.4: Decapsulated STM32F051R8T6 with the exposed die at the package's center.

Before an optical fault injection is possible, the device has to be decapsulated. In this process, the package above the chip is partially removed by chemical etching. This exposes the chip's die, shown in Figure 4.4, for semi-invasive and invasive attacks.

The first experiment targets the questions about whether RDP level 2 is, in fact, irreversible. At the same time, this also reveals the memory technology for RDP level storage. In contrast to flash memory, other technologies, such as eFUSE [KII02] memory, can implement real one-time programmable memory, which is based on physically making or breaking an electrical connection. For example, the Xilinx 7-series FPGAs store cryptographic keys and chip settings in an eFUSE memory [DTB+15, Xil18]. While manipulating a flash cell requires only UV-C light, resetting an eFUSE needs more sophisticated equipment that is not easily accessible. Thus, the experiment will show whether the security setting is stored in flash memory and whether a financially limited attacker is able to achieve a security downgrade.

In flash memory technology, data storage is based on electrons, which are trapped on the floating gate of a Metal-Oxide-Semiconductor Field-Effect Transistor (MOSFET) [KS67]. When UV-C light with a wavelength of approximately 254 nm illuminates the floating gate, electrons are ejected, the cell is discharged, and the stored bit permanently flips from 0 to 1 [BCMV03]. Compared to other optical fault injection approaches, this manipulation on flash memory is permanent. Please note that the method is limited to discharging cells and can only flip bits from 0 (charged) to 1 (uncharged). Bits that are 1 are not affected and retain their value. Thus, if the microcontroller's configuration can be manipulated via UV-C light, flash memory technology is used for storage since other types such as eFUSEs are not affected.

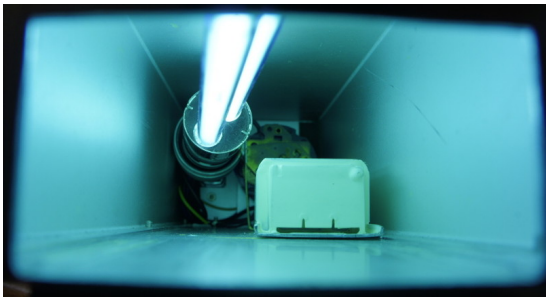


Figure 4.5: Inside of the UV-C EPROM eraser with the active mercury lamp.

For the experiment, a chip is decapsulated, checked for correct operation, locked to RDP level 2, and then exposed to UV-C light. Initially, the debugger cannot attach to the microcontroller, as expected, since the debug interface is disabled. Then, the chip is illuminated by an EPROM-eraser, shown in Figure 4.5, which is based on a UV-C mercury lamp with an emission line at 254 nm. Every hour, a power cycle is performed so that the microcontroller reloads its configuration from memory. After a few hours of UV-C illumination, the debugger finally attaches to the system successfully as its protection level has degraded and the device is no longer fully locked. The "read

protection level status" in the `FLASH_OBR` register reveals that RDP level 2 is no longer active and protection was downgraded to RDP level 1. As expected, a single bit in the option byte was flipped from 0 to 1 as the cell became discharged. In this one experiment, the configuration changed from `0x33CC` to `0x33CD`. Although this is an invalid configuration because `0x33` is not the inverse of `0xCD`, security is still downgraded. The experiment proves that no permanent locking mechanism exists, the security setting is stored in flash memory, and RDP level 2 can, in fact, be downgraded by flipping a single bit.

Besides that, I observed that the chip's current temperature has a major influence on the number of observable bit flips. A higher temperature is accompanied by more bit faults and increased instability of the data during readout. This is presumably caused by the larger thermal noise that occasionally pushes the affected cells slightly over the threshold between zero and one. Adjusting the temperature may be a viable option to fine-tune the results of such an attack.

After illumination, access to SRAM is possible since the device is in RDP level 1, but the CBS approach is not feasible. The attack did not only flip bits of the RDP setting but has broader effects. Even though the RDP level is downgraded, such a coarse full-chip illumination causes extensive damage also to other data in flash memory, such as the firmware. Thus, the firmware was also partially erased by the UV-C light and became unusable.

In order to exclusively target the option bytes, their location was determined by reverse-engineering the flash memory layout. Flash cells are usually arranged inside a regularly structured region on the chip and the cells are surrounded by read and write circuitry. For the chip under analysis, this narrows down the search to the rectangular section in the lower center region, marked in Figure 4.6. I apply a bisection approach to this region to find the underlying memory cell layout. This technique uses the UV-C-sensitive flash memory as an image sensor. At first, the chip is left unsecured in RDP level 0 and loaded with an all-zero firmware image. This makes all cells sensitive to incoming UV-C light. At next, a simple plastic mask is created that covers a specific region, e.g., the upper half of the flash memory module region. Then the chip is illuminated for a few hours with the UV-C light source placed atop of the device at a distance of a few centimeters.

The flash memory is read out every few minutes; the resulting bit flips are recorded and analyzed. Only regions that are not covered by the mask are exposed to UV-C light and will exhibit data faults. In order to map the faults to a physical location, the flash cell layout is guessed manually. Flash memory is arranged in vertical bit lines and horizontal word lines, which create a matrix containing a flash cell on each intersection. As this is a digital system, the number of bit lines and word lines is usually a multiple of two and their product must equal the flash memory size. However, this precondition on its own does not yield a unique solution; thus, additional trial-and-error is necessary. The bit flips are arranged according to the supposed cell layout to solve this riddle. A correct guess will result in a bitflip-pattern that resembles the structure of the mask.

Figure 4.6: Annotated die of the STM32F051R8T6 (Approx. 2700 µm × 2700 µm).

For example, if a mask covers the upper half, the resulting pattern must show bit flips solely in the lower half.

Several bisection steps were necessary to reconstruct and validate the flash cell arrangement. The successful guess for the flash cell layout of an STM32F051R8T6 is depicted in Figure 4.7. The orientation of this figure matches the orientation of the flash cell region in Figure 4.6. I discovered that the *firmware* flash memory region has 1024 vertical bit lines and 512 horizontal word lines, both spanning across the entire cell region. The flash memory has a data width of 32 bits; thus, there are $1024/32 = 32$ bit lines for each bit. The bits are marked in red in Figure 4.7. Each column contains the aforementioned 32 parallel vertical bit lines. The page size of the flash memory is $1\,\text{KiB} = 8 \cdot 1024\,\text{bits}$. This corresponds to the memory cells, which are addressed by exactly eight word lines. Thus, there are no contradictions and the determined cell layout is plausible.

Please note that this evaluation was done for the firmware flash memory region only, the overall flash memory is larger since there is additional memory for configuration

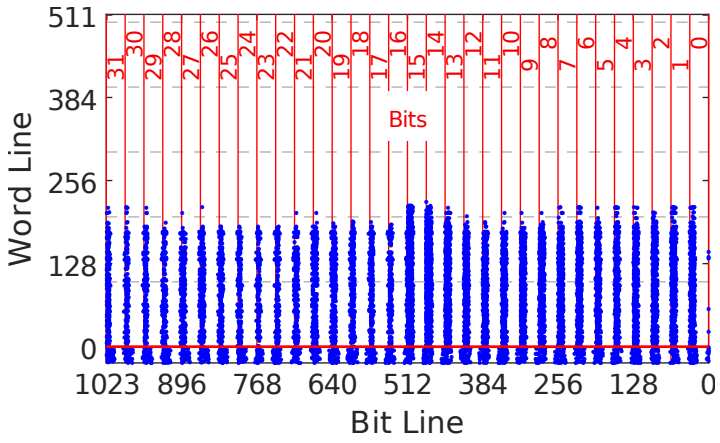


Figure 4.7: Flash cell layout with marked flipped bits (blue) after UV-C irradiation; the mask covered the upper half.

and system purposes. The datasheet shows that the STM32F0 possesses additional 3 KiB as *system memory* which contains a bootloader [STM17]. Furthermore, there are a few additional option bytes in another flash memory region. Thus, the flash memory contains three regions, the firmware flash memory, the system flash memory, and the option byte flash memory.

Further UV-C illumination experiments show that the system flash memory and option byte flash memory regions are similar in structure to the firmware flash memory. Both regions are placed beneath word line 0, indicated by a red horizontal line in Figure 4.7. The bit flips in the system memory are denser compared to the other regions since the system memory cannot be reset after each test and faults accumulate rapidly. The option byte and system memory regions consist of the same number of bit lines as the firmware flash memory, but they have only a few word lines since the regions are much smaller. The RDP and nRDP bytes, which are placed in the 16 LSBs of the option memory, have been located in the bottom right half of the flash cell region. This information allows crafting a mask that covers and protects the firmware flash memory region but leaves the option bytes exposed.

4.3.2.3 Device Analysis by Delayering

The attack does not require more detailed information, but to strengthen my results, I delayered the chip to analyze its internal flash memory layout. In this process, the metal layers of the chip are subsequently removed, starting with the outer layer, thereby making the next internal metal layer visible. Common chips are built in an interleaved manner with a transparent isolating layer, made from materials such as SiO_2, on top of each metal layer.

There are several methods to perform delayering, e.g., by mechanical polishing using slurry [SHL+13] or chemical etching. Professional labs make use of more than

one hundred recipes, tailored to the materials of each layer [TJ09]. These methods require extensive effort and experience and can precisely delayer the chip; however, a coarse result is sufficient for my analysis. I decided to use only equipment that is already available and that I am familiar with as the alternatives mentioned above are impractical.

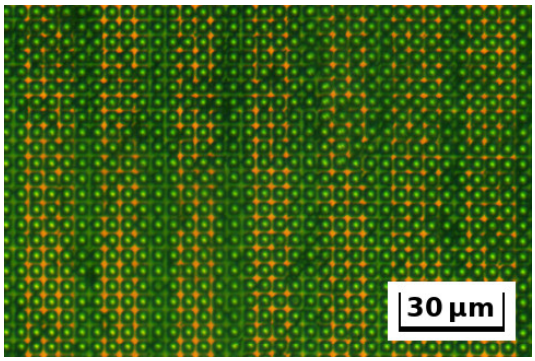
In order to delayer the chip, a single metal layer and one isolating layer must be removed subsequently without causing major damage to any other layers below. This requires two methods that each aim at *one* of the materials to be removed. The resulting method combines a two-step approach based on laser ablation and hot sulfuric acid (H_2SO_4) etching. While the laser is directed at destroying the isolating layer, sulfuric acid aims at removing the metal layer. In contrast to this, the laser cannot remove the metal layer and sulfuric acid does not attack an inert SiO_2 isolation layer. Thereby, this method achieves a high selectivity between materials.

In the first step, the laser, a *NEW WAVE QuikLaze*, is focused onto the metal layer that is covered by the isolating layer. When a short laser pulse is fired, the metal heats up, melts, and partly evaporates, thereby cracking open the rigid insulation above and exposing the metal layer. The metal is mostly melted or becomes scattered around in the vicinity, but it is not entirely removed as the amount of energy is insufficient. The laser is applied to the entire region of interest, removes the isolating layer, and exposes the metal to be removed.

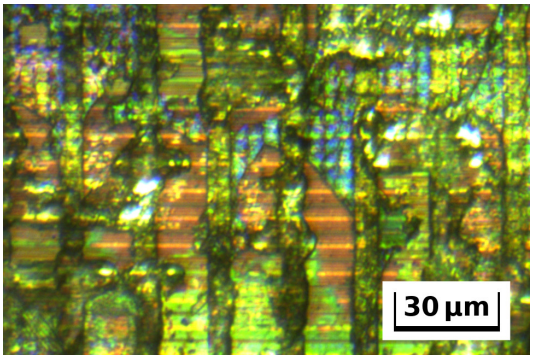
Next, sulfuric acid of a suitable concentration, e.g., 37 %, is poured into a glass container. The chip is placed on top of a hot plate that is set a little above the boiling point of water. With the help of a pipette, a small drop of sulfuric acid is applied onto the chip. The high temperature causes the water, contained in the sulfuric acid, to evaporate, thereby raising the acid's concentration. At the same time, the temperature speeds up the reaction between the exposed metal layer and the acid. Thereby, the exposed region of the metal layer is removed cleanly. However, any isolating layer below is not attacked by the acid, since this inert material is resistant against sulfuric acid. Thereby, a single metal layer can be removed from the chip. Finally, the chip is rinsed with water to remove any residual acid. It is now ready for optical inspection.

These steps are repeatedly executed and allow to dig deep into the chip. Nevertheless, this method required some experimentation with parameters, e.g. laser pulse power, but finally allowed me to capture an image of the inner layers of the flash cell region. While the structure is clearly visible, the colors may not be precisely represented since they are influenced by illumination, microscope optics, and camera adjustments.

The untampered flash cell region is shown in Figure 4.8a, the topmost layer appears in a green tint and contains only a fill pattern. The subjacent layer, appearing in orange, is slightly visible and it comprises comparably wide vertical traces that run in parallel. After these two layers have been removed, the next layer, shown in Figure 4.8b, becomes visible. This layer appears mostly in orange and brown; however, the delayering approach is rather coarse and insulating material from the other layers remained on the surface.

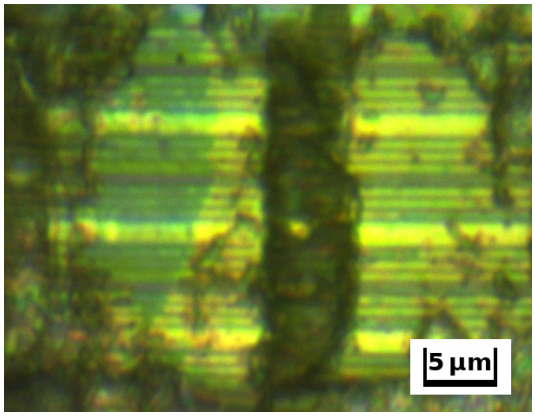


(a) Unmodified chip, Fill pattern (green, top layer) and traces (orange, next inner layer).

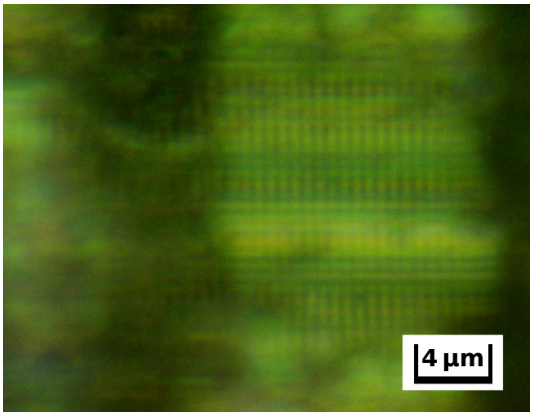


(b) Inner layers after partial removal of two top layers, horizontal word lines are visible.

Figure 4.8: Top metal layers of the flash cell region of an STM32F051R8T6.



(a) Horizontal word lines on the inner layer.



(b) Vertical bit lines on the next inner layer.

Figure 4.9: Inner metal layers of the flash cell region of an STM32F051R8T6.

The magnification of the microscope was increased to gain a better view. Due to adjusted lighting, the previously orange/brown layer appears in green. Figure 4.9a shows a magnified image of the aforementioned inner layer. The figure shows several horizontal traces running in parallel, which are the word lines of the flash cell region. They were manually counted and their number falls into the expected range around $512 + 8 \cdot 3 + 1 = 537$, corresponding to firmware, system, and option byte flash memory. Either by delayering or by carefully adjusting the focus, the next subjacent layer becomes visible, shown in Figure 4.9b. These vertical traces are the bit lines of the flash cell region. Their number also matches the expectation of about 1024 bit lines.

These results verify the supposed flash cell arrangement. Furthermore, the reason for the vertical pattern of non-flipped bits in each bit-column in Figure 4.7 was also discovered: Several large vertical metal traces on the second inner chip layer block the UV-C light before it reaches the actual cell.

In another short experiment that was targeted towards injecting temporary faults into flash memory, the same layout was verified. This test employed the same *NEW WAVE QuikLaze* station; however, the power setting was significantly reduced to cause no structural damage. Temporary faults were successfully injected when the cell was read and the laser was fired at the same time.

4.3.2.4 PoC with Increased Precision

After the position of the RDP bytes is known, I target this memory more precisely via a carefully crafted mask that covers the firmware flash memory. By placing the mask diagonally, only the lower right corner of the flash cell region is exposed where the LSBs of the RDP bytes reside. Depending on the positioning accuracy, I achieved a security downgrade while the number of accidentally flipped firmware bits varied between a few hundred down to none. My best results yielded a security downgrade to RDP level 1 with no firmware bit faults at all.

Altogether, the exploitation of a severe weakness in the security design was demonstrated. Having very few or even no firmware bit faults enables the practical use of RDP level 1 attacks such as CBS. This vulnerability applies to all STM32 microcontrollers having such a security level design. However, the attack's feasibility strongly differs between the controllers, depending on the physical implementation of the flash memory.

4.3.2.5 Countermeasure: RDP Downgrade Detection

No countermeasure exists, which can prevent a security downgrade since the issue resides in hardware. Developers have to rely on the manufacturer to enhance the concept, e.g., by switching RDP level 1 and 2 so that RDP level 2 becomes the fallback setting.

Nevertheless, the effect of the attack can be mitigated in firmware. Upon startup, the firmware should immediately check whether the expected protection level is still selected. This includes the "read protection level status" in the `FLASH_OBR` register and the option bytes themselves. In case of a mismatch between the setting and the expectation, the RDP level 2 setting should be rewritten to the option bytes as this restores the missing charge in the flash cells and re-enables security. Depending on the required reliability of the system and the bearable risk, zeroization, i.e., self-deletion of the entire firmware, may also be an option.

4.3.3 Debug Interface Exploit

The STM32F0 microcontroller series does not have a JTAG interface but is equipped with a Serial Wire Debug (SWD) interface for debugging and flash memory programming. This interface is disabled in RDP level 2 and flash memory access is restricted in RDP level 1, as described in Subsection 4.2.3. When a debugger is attached to the SWD interface in RDP level 1, the flash memory protection logic cuts any access

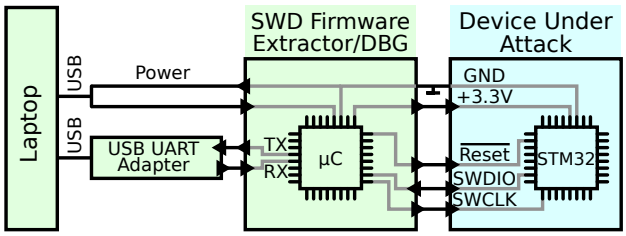


Figure 4.10: Schematic of the SWD experiments for debug interface exploitation.

to flash memory. This sparsely documented mechanism raises interest and asks for a detailed analysis.

4.3.3.1 Protection Logic Reverse Engineering

The *ST-LINK* debugger, which is a standard off-the-shelf device, has shown to be inadequate for my analysis. As soon as the debugger is connected to the microcontroller and before voluntarily transmitting any command, the flash memory is immediately locked down. This issue is caused by the debugger, which performs several SWD requests automatically upon connecting and thereby triggers the protection mechanism. As the immediate lockup prevents a precise analysis, this overly verbose debugger is of no use for the experiment.

To gain control of the *physical layer* of the SWD communication, I implemented my own rudimentary but versatile debugger. This is viable since the SWD interface is well-documented [ARM09a]: SWD uses two signals, which are named SWDIO and SWCLK, for synchronous communication between the debugger and the microcontroller. SWDIO is the bidirectional data signal, whereas SWCLK is the clock for communication.

Figure 4.10 shows the setup for the analysis. The laptop controls the debugger (DBG) via its UART interface. The debugger is connected to the power, reset, and SWD signals of the device under analysis and attack. The debugger's firmware provides full access to all SWD communication layers; thus, a fine-grained analysis becomes possible.

The first experiment investigates which SWD transactions trigger the flash memory protection logic. For this purpose, the device under attack is set to RDP level 1, the firmware is run from the flash memory, and switches an LED on and off. The blinking LED shows that the system is currently running. When flash memory access is blocked, the processor fails to fetch any further instructions and the blinking of the LED stops.

The experiment shows that the protection mechanism is not triggered by all SWD transactions, but only if a system bus (e.g., AHB-Lite [ARM06]) is accessed. Reading and writing SWD interface-internal registers has no consequences and the system remains running. However, as soon as the debugger interacts with the system bus to access any other module, such as peripherals, SRAM, and flash memory, debug mode is entered and the flash memory becomes locked down. More precisely, the debugger

triggers a transfer on the system bus by reading from or writing to the SWD AHB Access Port (AHB-AP) Data Read/Write register. A physical layer analysis reveals that the AHB transaction and the following lockdown is triggered by the last rising SWCLK edge of the corresponding SWD packet transmission. These investigations were only possible by developing one's own debugger, which allows these types of experiments.

4.3.3.2 Discovery and Analysis of the Vulnerability

The SWD communication is reduced to its essentials to analyze the behavior of the flash memory protection logic in detail. Most SWD transactions were removed which leads to a minimal configuration that fully initializes the SWD interface, but has not triggered the flash memory protection, yet. In the next step, a bus access is triggered by a read request from a flash memory address. This action is expected to trigger the flash memory protection logic. This happens in fact — but in rare cases, some data ends up in the SWD read buffer. To my surprise, this is data from the actually read-protected flash memory.

If at all, this deviation from the specification is observed only on the *first* read access on the bus. Any subsequent read attempts fail and return an SWD ERROR response, as specified. The flash memory protection logic must be reset by a power-cycle to repeat the experiment.

Read access shows to be only randomly successful. As this is typical for a timing-based race condition, I experimented with the bus load and timings to investigate this issue further. I implemented different firmwares using assembly language, one containing mostly no-operations (NOPs) and another one featuring long sequences of store word (STR) instructions. Altogether, the bus load varied from roughly 50 % for the NOP firmware to almost 100 % for the STR firmware. The access succeeds in every case while running the NOP firmware, but the access always fails for the STR firmware. This shows that the issue is related to the current bus load.

A look into the Cortex-M0 specification [ARM09b] supports the conjecture of a timing-related issue. Instruction and data fetches have priority over debug access at the moment of bus arbitration. Thus, the debugger has to wait for a free cycle on the bus to place its transaction. If the debugger gains instant access to the bus during the experiment, the transaction takes place before the flash memory protection locking is triggered — thus, a typical race condition occurs. In the other case, when the debugger's access is delayed by an ongoing instruction or data fetch, the protection logic wins the race and the flash memory module will reject the debugger's later-arriving read attempt.

I investigated this issue further by linearly adding bus load by increasing the number of wait states of the flash memory module. The wait state settings define the *additional* number of clock cycles between the read request and the availability of data at the flash memory module's output. At zero wait states, a read access requires two clock cycles on the AHB, one for the address phase and one for the data phase [ARM06]. Taking the NOP firmware as a basis, I observed that adding one wait cycle causes one

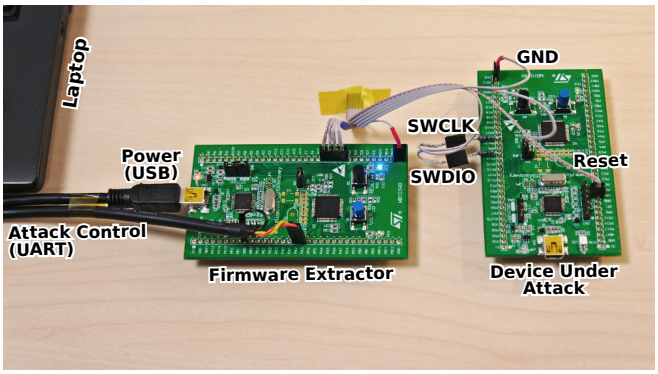


Figure 4.11: PoC setup for firmware extraction from a device in RDP level 1.

out of three read accesses to fail. By gradually increasing the wait states w up to the maximum of seven, I was able to verify that the read success probability p_s adheres to the following equation:

$$p_\mathrm{s} = 1 - \frac{w}{2+w} = \frac{2}{2+w} \tag{4.1}$$

The rearranged term on the right-hand side of the equation allows giving an educated guess about the internal timing issue. The denominator $(2 + w)$ represents the total number of bus cycles for the AHB access to the flash memory. The numerator 2 is the number of *vulnerable* bus cycles. Hence, I conclude that the flash memory protection logic is triggered two cycles too late.

Experiments with the STR firmware revealed that the flash memory module itself manages the access restriction. This firmware rapidly toggles an LED via the bus; thus, any other transactions will cause a measurable timing jitter on the toggling frequency. Such variations cannot be perceived visually, but an oscilloscope can display them precisely. At a failed access to the flash memory, the bus is blocked for three clock cycles in total, which matches the duration of an error response on the AHB [ARM06]. Thus, the access is actively denied by the flash memory module, which generates an error response. Furthermore, this shows that the protection logic makes the decision about the access directly after the data phase as no wait states have ever been observed during a failed access, independent of the wait state configuration.

The reason for the protection delay of two clock cycles lies in the chip's hardware, but the exact cause remains blurred. One possibility may be an incorrect implementation of clock-domain crossing between the debug clock domain [ARM09a] and the remaining logic of the chip. Altogether, these experiments showed that there is a major hardware issue, that entirely annuls the device's content protection mechanism in RDP level 1.

4.3.3.3 Proof of Concept: Code Extraction

The following PoC demonstrates that the vulnerability is not limited to a single bus access, but the full firmware can be extracted from the microcontroller. The setup is

presented in Figure 4.11 which is the practical realization of Figure 4.10. The debugger was extended to perform the aforementioned exploit autonomously; thus, it is named *Firmware Extractor* in the following. This attack tool reads out the entire firmware of the device under attack in RDP level 1.

The PoC comprises two STM32F0 Discovery boards with STM32F051R8T6 microcontrollers. The laptop controls the Firmware Extractor via its UART interface. This interface configures parameters such as the start address and data length for extraction. During readout, the data is sent to the laptop via the same UART interface.

Compared to the underlying concept, the attack is implemented straightforwardly with one extension. As the read access works only once before the system is locked up, the Firmware Extractor cycles power to the device under attack after each readout. Altogether, the following steps are executed:

1. **System Reset:** Initially, a power cycle is applied to reset the system, including the flash memory protection logic.

2. **Debugger interface initialization:** The steps, described in the datasheets of the microcontroller and its SWD interface [STM17, ARM09a], are followed. First, the SWD interface is reset by applying the reset pattern to SWDIO and SWCLK. Secondly, the debugger reads the interface's Identification Code Register (`IDCODE`) from the SWD Debug Port (SWD-DP). Thirdly, the debugger sets the System power-up request (`CSYSPWRUPREQ`) and Debug power-up request (`CDBGPWRUPREQ`) in the Control/Status register (`CTRL/STAT`) of the SWD-DP to fully initialize the debug interface.

3. **Set the access width to 32 bit:** Although optional, switching to 32-bit mode is recommended, as a full word is extracted in each step, which speeds up firmware extraction. Thus, the size field in the AHB Access Port (AHB-AP) Control/Status Word Register (`CSW`) is set to `0x02`.

4. **Set flash memory source address:** The address to be read next is written into the AHB-AP Transfer Address register (`TAR`).

5. **Trigger the flash memory read:** The debugger performs a read access from the AHB-AP Data Read/Write register (`DRW`). This triggers the AHB transaction that reads the *protected* flash memory.

6. **Read the extracted data:** The result of the access is read from the DP Read Buffer Register `RDBUF`. On success, if flash memory access was granted, the interface returns an `SWD OK` response. On access failure, `SWD ERROR` is returned. In the case of `SWD OK`, the word was correctly extracted from flash memory and is sent to the laptop.

7. **Iterate:** Upon success, this procedure is restarted with (address + 4) to read the next word from flash memory. On failure, the address is not incremented and the read access is retried.

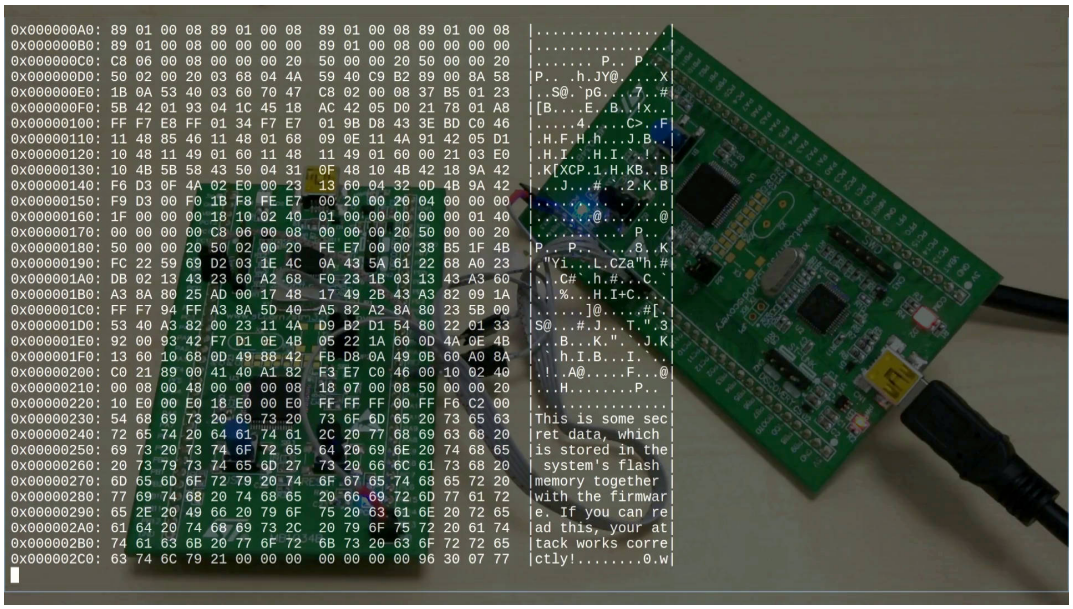


Figure 4.12: Snapshot of the Firmware Extractor demonstration video.

The PoC was successfully tested on an STM32F051R8T6 and STM32F030R8T6 microcontroller. Due to the very similar chip functionality, I expect the entire STM32F0 series to be affected. Nevertheless, experiments on a few samples indicate that other *series*, such as the STM32F1, are not vulnerable.

The PoC extracts the firmware at a speed of roughly 45 bytes per second on average. The largest STM32F0 microcontroller has 256 KiB of flash memory; thus, it could be read out entirely in less than two hours. Hence, the attack is practically feasible. The code and a demonstration video of the attack are available at `https://science.obermaier-johannes.de`. A snapshot thereof is shown in Figure 4.12.

4.3.3.4 Discussion of Countermeasures

The attack can be hampered by setting the device to the higher RDP level 2 since this disables the debug interface entirely. Nevertheless, this does not fully resolve the issue because the protection level is not stored immutably, as demonstrated in Subsection 4.3.2. In comparison to the CBS attack, the requirements on the security downgrade are significantly relaxed. While the firmware must remain intact and executable for CBS, the debug interface exploit works also on damaged firmware as the firmware will not be executed. Thus, correctly placing the mask for a security downgrade is less demanding. Even if data errors occur, they are usually limited to a small range of addresses, hence, destroying cryptographic material or other IP is rather unlikely. If multiple devices with slightly damaged contents are available, their readout

results can be combined to obtain an error-free flash memory dump via methods such as majority voting. Hence, there is no reliable protection, especially if combined attacks are also considered. This issue can solely be fixed by the manufacturer who can fix the underlying hardware issue and provide an updated microcontroller revision.

4.4 Conclusion and Outlook

This chapter presents three weaknesses in the STM32F0 series that leverage firmware content protection. First, the readability of SRAM in RDP level 1 emerges to a vulnerability, as it may allow reading out cryptographic material or even the entire firmware. Secondly, the stronger RDP level 2 incorporates a design error, which significantly weakens the security concept and unintentionally simplifies a security downgrade. Thirdly, the SWD hardware implementation contains a race-condition that completely annuls device security in RDP level 1 and exposes the firmware to the attacker. Thus, I was able to develop all weaknesses to vulnerabilities by demonstrating their practical exploitability. The demonstration was conducted via three PoCs, one for each vulnerability. They show the practical relevance of my work and present novel methods for attacks and reverse-engineering of security measures in embedded devices.

Altogether, these vulnerabilities entirely break security for RDP level 1 and alarmingly weaken security for RDP level 2. Since the presented attacks are possible with low-cost lab equipment, the impact increases. Especially a combination of vulnerabilities, e.g., downgrading security from RDP level 2 and using the SWD exploit to read the firmware pushes the complexity of attacks down to the hobbyist-level. Cold-Boot Stepping and the security downgrade might also work on other STM32 chip series, as they are based on a similar concept. Thus, the impact of this analysis is severe and might be extendable to more systems as well.

In 2019, Marc Schink and me continued working on this topic, again focussing on firmware protection. In this case, we investigated another security concept, named Execute-Only Memory, which prevents readout and only allows execution. However, we were able to demonstrate the conceptual weakness and demonstrated reading out code from protected memory [SO19].

These results demonstrate that even with an error-free software and immunity against non-invasive fault injection, as assumed in this chapter, a system may still not be secure. Developers have to rely on the manufacturer of their embedded devices to sufficiently secure their hardware interfaces; otherwise security will be broken via such an attack vector. However, these kinds of tests are very effortful and are not guaranteed to detect every design and implementation flaw. Thus, the question arises, how system security can be guaranteed if commercial-off-the-shelf components are present with an unclear level of hardware security.

Chapter 5

The Past, Present, and Future of Physical Security Enclosures

The three previous chapters demonstrated that embedded systems are subjected to a variety of attacks. Even if there are no software vulnerabilities, attackers might try to exploit hardware-related vulnerabilities. For example, attackers might inject faults via clock glitches or physically manipulate the chip via invasive attacks. But even if this is not successful, an insecure debug interface can immediately nullify security, as shown in the previous chapter. Hence, sufficiently securing an embedded system might be impossible for its developers since the most problematic issues, such as debug interface vulnerabilities, are out of their control.

High-security applications do not only rely on the protection of each component but primarily on additional *physical* security measures. A tamper-sensitive security enclosure, surrounding the entire device, is one possibility to hamper tampering or related hardware-based attacks. Thereby, physical system security does not depend on each individual chip anymore and is condensed into the security enclosure. However, there exist several concepts for security enclosures ranging from permanently monitored covers with conductive traces up to novel Physical Unclonable Function (PUF)-based solutions.

This chapter presents an overview and discusses past, present, and potential future hardware security enclosure solutions. Parts of this chapter were previously presented at the *IEEE International Workshop on Physical Attacks and Inspection on Electronics (PAINE)* in 2018. The corresponding position paper titled *The Past, Present, and Future of Physical Security Enclosures: From Battery-Backed Monitoring to PUF-Based Inherent Security and Beyond* was published in the *Special Issue: Hardware Reverse Engineering and Obfuscation* of the *Journal of Hardware and Systems Security (HaSS)* [OI18]. This work was created in cooperation with Vincent Immler, who acquired all evaluated modules, contributed to their analysis, and to writing the paper.

5.1 Overview

Several standards exist for the security certification of embedded devices, such as FIPS 140-2 [NIS01a] and Common Criteria (CC) [KLR08]. They demand that every compliant device is well-protected against tampering, e.g., by means of a physical security boundary. Such a boundary aims at separating the secure domain of a system from the insecure one, thereby protecting the device against physical attacks, such as drilling and probing [Wei00]. This applies to single-chip devices and to multiple-chip systems. In contrast to single-chip devices, such as smart cards, which are protected in silicon, the protection of multiple-chip embedded systems is more challenging. For these embedded systems on PCBs, boundaries are implemented as security covers, housings, or envelopes [IMJFC13, ES05]. These generic countermeasures make a wide range of invasive, semi-invasive and non-invasive attacks [Sko05] more difficult to perform, as most of them require hardware access which is prevented by the security enclosure.

To protect an embedded device, physical access needs to be prevented by adequate measures such as the aforementioned security boundaries. Vendors developed a range of solutions that allow creating an enclosed space wherein critical data can securely be processed. These boundaries are commonly either implemented via an *envelope* or a *cover*. Envelope-based solutions wrap the embedded device entirely into a sensor foil whose integrity is monitored from the inside. Thus, there are no unprotected spots as the module is entirely surrounded and thereby protected. Compared to this, cover-based solutions usually protect only the critical region of the embedded device and leave the remaining circuitry unprotected. The protection is achieved via two sensitive covers on the top and bottom of the critical region wherein the critical data is processed and the covers are monitored for integrity. This chapter presents examples for the envelope-based and cover-based solutions: The envelope-based solution is deployed in the IBM Crypto Coprocessor, whereas a cover protects the HP Atalla Cryptographic Subsystem. Both solutions are disassembled and analyzed in detail in Section 5.2.

5.1.1 Related Work

The company GORE offered solutions that comply with FIPS 140-2 level 3 and 4, the two highest levels of protection. These products, which have been discontinued, were based on a cover [GOR07] or envelope [IMJFC13] that encloses the Module Under Protection (MUP) with an electrically conductive mesh of a well-defined ohmic resistance. Attempts to penetrate the mesh are likely to result in open circuits or will at least alter the resistance of the traces therein. A monitoring circuit inside the protected space of the system, which continuously measures the trace resistance, senses the change in resistance due to tampering and consequently triggers an alarm. This causes the zeroization of all Critical Security Parameters (CSPs), such as cryptographic keys, which are thereby irrecoverably destroyed. When the carrier system is powered off, a battery supplies energy to the monitoring system for uninterrupted operation. The CSPs are stored in a *volatile* Battery-Backed Random-Access Memory (BBRAM), powered from the same source as the monitoring circuit; thus, data is only retained as

long as power is available. Furthermore, BBRAM enables instantaneous zeriozation by turning off its power upon detection of physical tampering, thereby deleting all secrets. The steps from noticing a manipulation to the deletion of CSPs are known as tamper-detection and tamper-response. However, they require *explicit* action by the monitoring circuit as they are not interlinked with each other.

As an alternative, PUFs provide key storage without dedicated memory. PUFs exploit inherent physical manufacturing variations to derive a unique and device-specific key. Such properties appear in nearly every manufacturing process and affect, e.g., semiconductors [HYKD14, MCMS10], optic materials [VWNU16], PCBs [WSL+15], and large-scale thin-film devices [IOK+18a]. These properties need to be measurable to digitize and derive a key from them. However, PUFs may be of a different statistical quality, which might affect their cryptographic strength when being used as key source.

To analyze such PUFs, several concepts for performance evaluation have been proposed [MGS13, HYKS10]; however, the parameters need to be selected according to the target application. Some crucial metrics for PUFs in security enclosures are reliability, uniqueness, randomness, and tamper-evidence. Reliability describes the ability of the PUF to provide the same response under different environmental conditions, such as temperature and humidity, that the device is expected to operate in. Uniqueness is a measure for the difference in the PUF between different devices, i.e., each device has its own individual key. Randomness aims at the unpredictability of the PUF and the resulting key, ensuring that it cannot be guessed by an adversary or easily controlled during manufacturing. Tamper-evidence is a metric especially important for security enclosures as a manipulation of the system must cause measurable evidence of tampering.

Since most PUFs are only one component in a larger system, they cannot protect the system as a whole and such a solution cannot be considered tamper-evident [GCDD02, HYKD14, HNT+13]. However, some PUFs can provide real tamper-evident key storage by inextricably linking physical integrity to CSPs — a topic that has gained attention from 2015 onwards [VNK+15, IOK+18a, OIHS18, ION+18].

5.1.2 Contributions

This chapter provides a review of past, present, and future hardware security enclosure technologies, covering the aforementioned protection technologies. In this chapter, I present the following contributions:

- An analysis of **past** trace resistance-based security solutions, including
  - a detailed dismantling of the physical security enclosure or cover.
  - an evaluation of their enclosure integrity monitoring circuits.
- An overview of **present** PUF-based solutions which are available or in development, including

  - a discussion of their underlying basic concepts, advantages, and potential weaknesses.
  - an introduction of B-TREPID on which this dissertation focuses on.

- A motivation for **future** developments towards PUF-based security enclosures.

5.1.3 Structure

This chapter starts with an introduction to the topic in Section 5.1. For a fair comparison between battery-backed and PUF-based enclosures, this chapter first surveys several commercial tamper-respondent designs in Section 5.2. As the details of such solutions are typically not publicly available, two important battery-backed solutions were analyzed in more detail, namely the IBM Crypto Coprocessor in Subsection 5.2.1 and the HP Atalla Cryptographic Subsystem in Subsection 5.2.2. Subsequently, Section 5.3 summarizes novel concepts for the next generation of security enclosures that are based on PUFs and no longer require a battery. Next, a set of open challenges is presented and discussed in Section 5.4 to provide guidance for follow-up work. Finally, Section 5.5 concludes this chapter with an outlook into the possible future of security enclosures.

5.2 The Past: Battery-Backed Enclosures

There are several concepts to create tamper-respondent enclosures. The fringe-effect capacitive proximity sensor [ES05] is one solution that claims FIPS 140-2 level 4 conformance. Its security concept is based on electrodes whose capacitive coupling is monitored continuously. As long as the system is not under attack, the capacitive coupling remains constant. If the enclosure is tampered with, the intruding object interferes with the electric field and causes a change in capacitance. This is detected by the monitoring circuit, which will consequently trigger the zeroization of all CSPs. The system incorporates a battery for retaining the CSPs in BBRAM and for continuous monitoring, also if the carrier system is powered off.

To the best of the author's knowledge, most alternative approaches exclusively monitor the ohmic trace resistance. One such example is the Security Housing manufactured by Bourns [BOU07, BQ04]. It is based on plastic or ceramic covers that comprise one or more layers of electrically conductive traces. This cover is then mounted atop a PCB after manufacturing to enclose and protect the components underneath. The former commercial brochure [BOU07] from 2007 states a protection against drill diameters down to 500 µm. To resist further types of tampering, the traces are reportedly manufactured such that they cannot be electrically contacted easily.

No real-world applications are known to the author that make use of the aforementioned approach; however, there are protected systems that are classified, e.g., in military applications, for which any information is practically unobtainable. In general, public documentation on *commercial* Hardware Security Modules (HSMs)

also does not include specific information on their tamper-respondent enclosures. This might be owed to the fact that the Joint Interpretation Library (JIL), which assesses devices regarding their attack potential, grants points based on the lack of publicly available information on security mechanisms [SOG13]. As passing the security certification process depends on the achieved score, withholding information on anti-tamper mechanisms is encouraged, resulting partially in a *security-by-obscurity* principle.

Due to the unavailability of public documentation on battery-backed tamper-responding technologies, two differently protected HSMs were acquired and analyzed in detail by means of destructive disassembly. First, this section analyzes the IBM Cryptographic Coprocessor and then continues with the HP Atalla Cryptographic Subsystem. The choice of these two systems is based on their strong protection, i.e., the IBM device complies with FIPS 140-2 level 4 and the HP device fulfills the requirements for FIPS 140-2 level 3. Additionally, decommissioned devices of both systems are readily available via an online marketplace at a fraction of the original price. Other devices, such as Point of Sale (PoS) terminals, which adhere to the Payment Card Industry (PCI) standards [Pay13], were not tested since they are less sophisticated in their protection and some information is already available online [DMA08, Jer11, Jon14]. The same is true for gambling machines that comprise tamper-protected modules that have also been publicly analyzed [Jon16, Mas13].

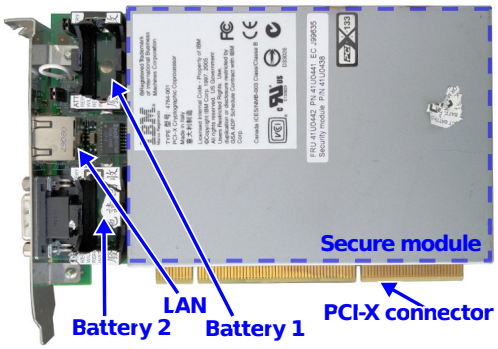
5.2.1 IBM Crypto Coprocessor

Figure 5.1a shows an IBM HSM which is based on an embedded system which is enclosed by the GORE tamper-respondent envelope [IMJFC13]. The envelope comprises several delicate layers of electrically conductive carbon traces. A battery-backed tamper-detection monitoring circuit measures the ohmic resistance of those traces, thereby checking the system's physical integrity. This circuit is powered by two redundant 3 V lithium batteries residing outside the HSM, marked in Figure 5.1a.

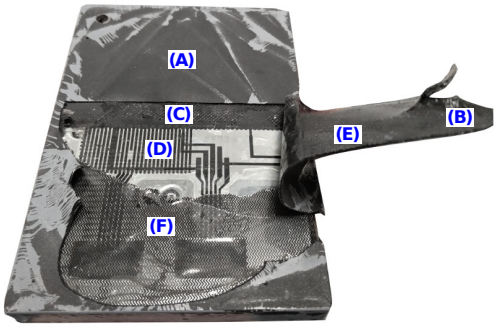
Apart from this main protection feature via the enclosure, I discovered several additional tamper-detection mechanisms. However, the following sections focus on the most relevant subset and my manual reverse-engineering might not have discovered every last detail of anti-tamper mechanisms. Please note that my analysis is primarily based on *destructive* disassembly and does not represent a successful attack on the system, but provides ideas for potential attack vectors. Nevertheless, this brief analysis emphasizes the need for batteryless security enclosures as I stumbled across several drawbacks of this technology.

5.2.1.1 Physical Properties of the HSM

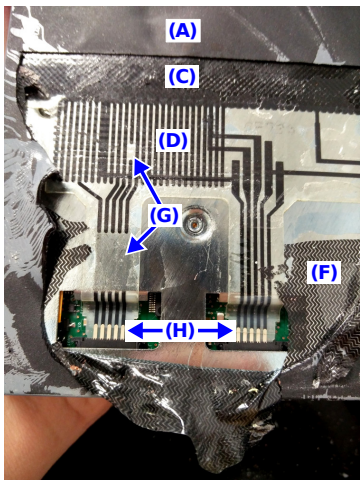
Figure 5.1a shows the intact IBM HSM with the secure module, which is internally protected by the GORE envelope. For analysis, I removed the outer metal casing and gained access to the potted HSM and its envelope. The potting and envelope were partially opened for a more detailed evaluation, as shown in Figure 5.1b and 5.1c.



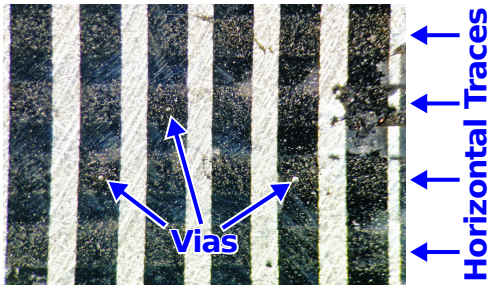
(a) The IBM Cryptographic Coprocessor HSM which was chosen for analysis.



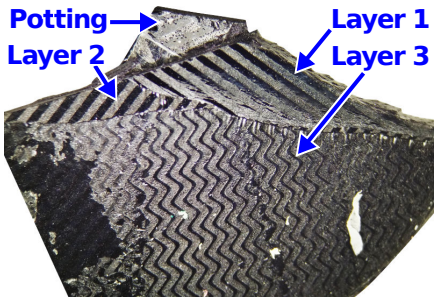
(b) Partially opened wrapped and potted module.



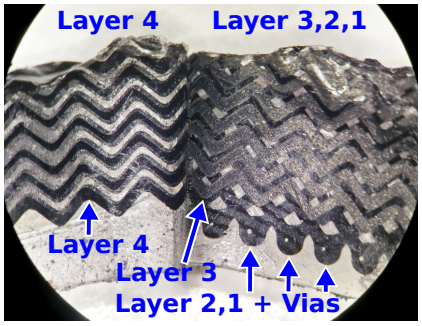
(c) Randomization and connector area.



(d) Vias in region (D), residue of the horizontal traces is slightly visible.



(e) Three of four carbon trace layers, enabling a detection of 300 µm holes.



(f) The four trace layers that have partially been pulled apart.

Figure 5.1: Disassembly and analysis of an IBM HSM and its tamper-respondent envelope by GORE [IMJFC13].

The potting (A) is a dark, rubbery, and opaque material with a strong smell. It completely surrounds and conceals the wrapped envelope, preventing optical analysis. When peeling off potting material around the center of the module, a large section (B) becomes loose and applying further force tears the envelope apart. This reveals the underlying structures such as the beginning of the envelope's sensory region (C), where the envelope overlaps with its own end for a little more than roughly one centimeter. In this region, diagonally running traces are visible, which span across the entire intact module, as visible especially in Figure 5.1c.

The envelope incorporates means for an electrical permutation of its traces. This allows obfuscating the routing inside the envelope individually for each envelope without the need to modify the measurement circuit. The feature is implemented in region (D), which comprises *vertical* traces which connect this area to the envelope's sensory region. Additionally, region (E) contains *horizontal* traces that span across the entire width of region (D). Since (D) and (E) are separated by a carrier substrate, connections are created by adding vias in the insulating layer. Figure 5.1d displays a close-up of region (D), which shows three vias and some residue of the horizontal traces of region (E). In this case, the second and fifth vertical trace, counted from the left, were connected by a horizontal trace through two of the vias. Some via drilling processes, such as laser-drilling, are not based on a fixed toolset and can be reconfigured on demand; thus, they support device-individual via positions. Although the concrete manufacturing method was never disclosed, this is one possible realization of layout randomization, which appears likely, according to own discussions with manufacturers.

In general, the envelope's materials are very sensitive to physical tampering. Regions (D) and (E) in Figure 5.1b and 5.1c were designed to separate from each other under mechanical force, thus, disassembling the envelope without permanently damaging it is highly improbable. The same is true for the sensory region, starting at (C), which comprises four layers of resistive carbon traces. When trying to pull the envelope off the device, these sensor layers usually separate, e.g., (F) remains on the inner casing while the upper three layers stick together. The layers are further electrically and mechanically interlinked, e.g., when the envelope's bottom layer (F) is pulled off the casing, parts of the underlying traces of (G) are also torn off. This material property is a crucial element for the envelope's security as it prevents a damage-free disassembly on the physical level and renders this attack vector impractical.

A close-up of the envelope is shown in Figure 5.1e and Figure 5.1f as seen from inside the device. Three out of four layers of the mesh are visible in the first figure, whereas the two innermost layers are a zig-zag pattern and the two outer layers show a diagonal structure that is orthogonal to each other. The traces are made from a soft carbon-ink material, which becomes quickly scrapped off with very little force and sharp tools. Moreover, its chemical properties are expected to closely resemble the potting material such that applying solvents only to remove the potting is assumed difficult, if not impossible. Attempts to contact these carbon-ink traces using standard lead-based solder failed; instead, conductive silver or similar materials must be used. Similarly, directly probing these traces is difficult as they become easily damaged

by standard metal multimeter probes. Separating the substrate layers, as shown in Figure 5.1f will mechanically destroy the traces since they partially detach from the substrate, thereby altering their resistance. However, the mesh is comparably coarse compared to the trace width achievable in PCB and flexPCB technologies.

5.2.1.2 Enclosed HSM System and its Envelope Monitoring Circuit

Two flat cables that are part of the envelope connect the envelope to the battery-backed supervisor circuit. The cable and its corresponding connector are designated with (H) in Figure 5.1c. Seven signals are present in total on both connectors: ground (GND), the supply voltage (V_{CC}), and five voltage sense signals (V_{S1} to V_{S5}). The envelope's traces are configured as five voltage dividers, which are tapped at their center where the sense signal is measured. Since each voltage divider is connected between GND and V_{CC}, the center voltage of each trace is $V_{S1} = V_{S2} = ... = V_{CC}/2$ in the untampered state. All five voltages are monitored continuously by the evaluation circuit.

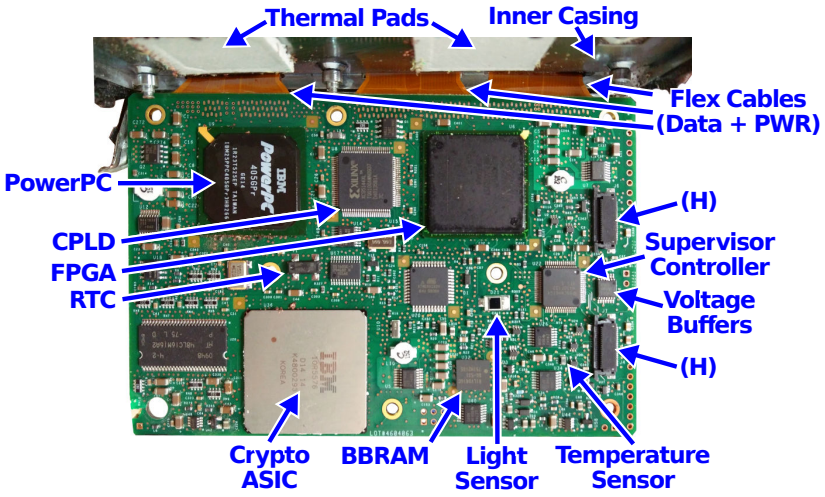


Figure 5.2: The module under protection including the measurement circuit.

The actual HSM board that comprises the cryptographic circuits and the monitoring circuit is shown in Figure 5.2. Reverse engineering of the system yielded the majority of expected components [ABC+12, And10]. Due to the limited energy available in the batteries, the monitoring system was apparently built with minimizing power consumption in mind. Consequently, it uses a low-power MSP430 microcontroller as the supervisor circuit controller. In addition, the operational amplifiers run on a low supply current. The envelope's conductive traces are estimated to have a resistance in the range of several megaohms, minimizing the currents. Though, such a high-ohmic voltage divider is designated to be susceptible to external influences that might erroneously trigger the tamper detection. This statement is supported by the fact that

the envelope's output voltages V_{S1} to V_{S5} are stabilized with 10 nF capacitors. Please note that low-power designs comprising such large capacitances and resistances are known to be comparably slow, thereby limiting the circuit's response time to physical tampering.

The enclosure monitoring seems to be implemented twofold with different concepts. Each signal is initially fed into an operational amplifier configured as a voltage follower, which generates a buffered copy of each input voltage to be used in the following circuitry. In the first monitoring concept, the voltages are fed to analog inputs of the supervisor microcontroller, which most likely checks their levels during operation. The second method is purely analog without any microcontroller. The five buffered signals are combined via diodes, thereby outputting the minimum and maximum voltage of the five voltages. A window comparator checks if they exceed the lower or upper bound of the specified range and can raise a tamper alarm that causes zeroization. The comparator's thresholds were experimentally verified to lie at 1.04 V and 1.98 V for this device, thus, the circuit was most likely designed to accept voltages between 1 V and 2 V for the untampered state. This is a noteworthy broad range since the supply voltage is only 3.0 V and the voltage dividers output is 1.5 V in the untampered state, thus, deviations of −33 % and +33 % are considered acceptable.

As the system works solely with static voltage levels, the envelope's output remains constant as long as it has not been tampered with. While this minimizes power consumption, it comes with the conceptual weakness that an attacker could force the expected voltage from an external source into the circuit. Alternatively, a similar attack could be performed through minimalist holes or attempted repairs with conductive silver. In either case, creating a suitable contact to the connector pads directly at region (H) may be comparably easy, as they are not made from carbon-ink and relatively large compared to the size of the mesh. Since there is no dynamic monitoring signal for the envelope, the attacker does not need to synchronize to it, further simplifying the attack. I verified experimentally on the disassembled HSM that the monitoring circuitry does not trigger an alarm when a constant voltage is applied from an external source.

Apart from enclosure monitoring, the evaluation circuit additionally performs system-level checks to detect other adversarial operating conditions. This includes, for example, checking the battery's voltage level which must not be outside a specified range. The analog circuitry will trigger a tamper alarm if the battery voltage drops below 2.5 V or exceeds 3.7 V. The same is true for the temperature, as the evaluation unit also comprises an internal temperature sensor, marked in Figure 5.2. The low-temperature alarm threshold lies at approximately −23 °C while the high-temperature alarm threshold is at about +99 °C. I experimentally determined these values in a temperature test chamber.

Having a successful attacker in mind, the circuit comprises a large-area photodiode that senses even the smallest amounts of light inside the enclosure. The device resides in the center of the board next to the monitoring circuitry, as marked in Figure 5.2. Although no part number is present, this might be a BPW34 or comparable photodiode.

Its signal is amplified with a gain of 1 MΩ; thus, a current of 200 nA creates a voltage of 2 V which, when exceeded, causes a tamper alarm. For reference, adding a single layer of black isolating tape atop the photodiode is not sufficient and an intensive light source, such as a LED flashlight, can still trigger the tamper detection.

There might be more protection mechanisms present that have not been reverse-engineered. I discovered some additional comparators which might, for example, sense the supply voltage, applied via the PCI-X interface. Although a large-area photodiode can detect x-rays in some special configurations, an x-ray sensing feature seems not to be present, similar to the related IBM HSMs [IBM16].

For each tamper detection mechanism, there is a comparator which generates a digital output. These signals are combined via diode logic, implementing a *logical OR*. This signal is connected to the input of a flip-flop that generates the tamper-detection status signal. Hence, even if the microcontroller is not active or defective, the alarm as part of the tamper-response is triggered anyway. In case of an alarm, the power supply to the CSP-storage BBRAM marked in Figure 5.2 is cut and its power rail is actively pulled to ground by a crowbar circuit, thereby zeroizing all data.

The remaining components are presumably not related to tamper detection but implement the HSM functionality. These components are powered only by the main supply but not by the battery. The board comprises, for example, an IBM crypto Application-Specific Integrated Circuit (ASIC), a Xilinx FPGA, a Xilinx Complex Programmable Logic Device (CPLD), a Real-Time Clock (RTC), and a PowerPC chip. The FPGA and PowerPC appear to have a high power dissipation; thus, they are thermally connected to the inner casing via thermal pads. Temperature hot spots can be an issue for security enclosures since they regionally increase the ohmic resistance of the trace, similar to tamper attempts.

5.2.1.3 Analysis Summary

Taking into account the previously described countermeasures, only the most sophisticated attackers are expected to attempt breaking into the system and only very few, if any, would succeed. Despite being discontinued, this physical security enclosure technology has understandably been the de facto standard for many years throughout the industry. Still, I identified *potential* drawbacks of the system's measurement setup, i.e., static signals and the need for a battery, resulting in limited responsiveness of the design. In addition to its restricted operating temperature range, this strongly supports the need for batteryless tamper-resistant enclosures.

5.2.2 HP Atalla Cryptographic Subsystem (ACS)

HP's HSM as shown in Figure 5.3 is enclosed by a silver-colored cover on top and bottom of the PCB. The top cover comprises a heat sink and is screwed to the PCB and to the bottom cover. Since the device is not entirely enclosed or potted, an attacker might consider cover penetration or cover removal. If the device is fully assembled, no openings are present that allow getting access to the inside of the secure compartment.

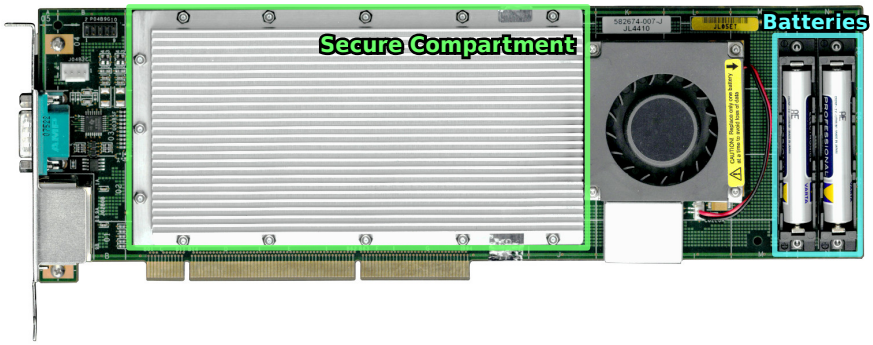


Figure 5.3: The intact HP Atalla Cryptographic Subsystem.

Nevertheless, reverse-engineering the device is simplified when compared to the previously analyzed device. Since there is no potting material and no adhesives bond the covers to the PCB, it can simply be unscrewed. I expect that this shall allow authorized servicing and maintenance, as the cover can be removed and refitted quickly.

If the cover is removed, a tamper alarm will be caused, since there are several tamper-detection sensors. Both covers comprise the following layers: An outer metal heat sink, a thermal pad, a flexible sheet comprising a mesh, and an inner shell. Please note that removing the heat sink will not cause instant zeroization, as long as the inner shell remains pressed onto the circuit board.

5.2.2.1 Cover Removal and PCB-Based Tamper Sensors

The disassembled HP Atalla Cryptographic Subsystem is shown in Figure 5.4. Cover removal sensors are present on the top and bottom on the PCB. They consist of two pads that are shorted by a thin conductive tape that is attached to the rim of the inside of each cover. The conductive tape is most likely an EMI gasket, which is a flexible, spongy, and fine metal mesh that, for example, connects several case components in HF devices. In this HSM, when the pressure holding the covers together is relieved, the conductive tape no longer shorts the pads on the PCB, which is detected by the monitoring circuit. This will trigger the zeroization of CSPs, which are again stored in a BBRAM. Please note that for a successful top cover removal, the bottom cover must be removed first, as the top cover's screws directly feed into the PCB. Hence, access to these screws is prevented as long as the bottom cover is attached, thereby hindering unauthorized disassembly.

In contrast to the top cover, the bottom cover comprises an additional very thick conductive tape in its center that creates a connection between the PCB and the bottom cover's metal shell. This connection is checked via an additional signal present at the electrical connector of the cover. Hence, a step-wise and careful disassembly needs to focus on these specific countermeasures, as the cover removal sensors are exposed once the outer bottom shell or the heat sink atop are removed.

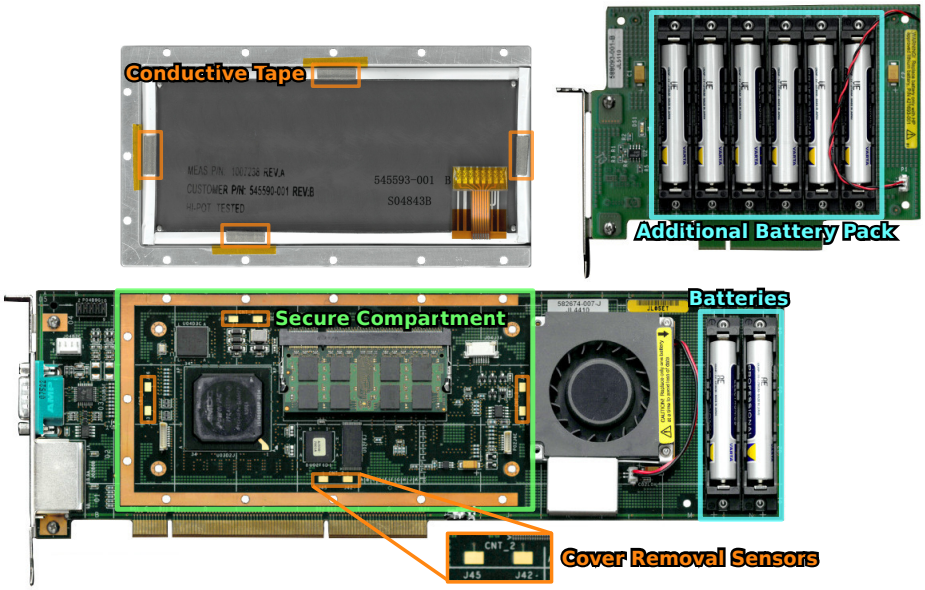


Figure 5.4: The disassembled HP Atalla Cryptographic Subsystem.

5.2.2.2 Cover Penetration Considerations

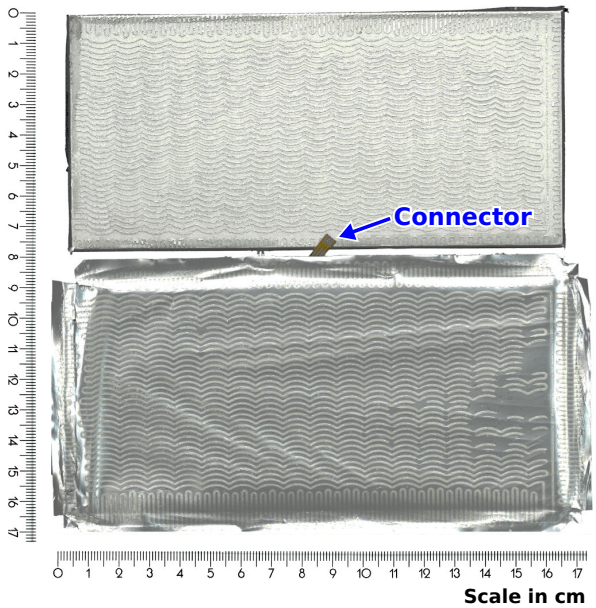


Figure 5.5: Dismantled bottom cover showing the resistive sensor mesh. Its connector only carries the three signals CoverGND, mesh input, and mesh output.

This cover's mesh is relatively coarse with a trace width of approximately 1 mm and equally sized space in between, as shown in Figure 5.5. Its structure size is about three times larger than the mesh of the GORE envelope. The cover comprises only a single trace loop, which is implemented as one long track in a serpentine pattern, which spans across the entire surface. Since the cover can be taken off without causing any permanent damage, I was able to measure the loop's electrical resistance, which is 300 Ω. The relatively large gap between the traces allows drill diameters of up to 1 mm to go undetected. Furthermore, the overall track length may add uncertainty to the exact resistance value since it is subjected to manufacturing variations. Altogether, this cover technology appears to have limited resistance against physical penetration.

5.2.2.3 Device Analysis Summary

This device seems to provide only a moderate level of protection, owed to the less complex set of countermeasures and a comparably coarse mesh. Nevertheless, breaking this protection still appears to be very challenging. Attacks either could penetrate the mesh and probe signals or someone could try to disable each countermeasure individually and disassemble the system. For this step-by-step approach to succeed, defeating the bottom cover's connection through the conductive foam in its center appears to be the most challenging task due to its distance from the outer rim. Taking into account the previous considerations and the lack of, e.g., a light sensor, an attack is more probable to succeed when compared to the GORE envelope of IBM's HSM.

5.2.3 Drawbacks of Battery-Backed Solutions

Battery-backed mechanisms entail several drawbacks due to the continuous monitoring of the enclosure, even when the system is powered off. Practical challenges arise from added bulk and weight that limits the deployment of such systems for mobile applications. Moreover, such a battery-powered concept increases cost in addition to the enclosure itself. On a technical level, batteries are subject to self-discharge over time, which limits the device's shelf-life. Prolonged storage may fully discharge the batteries, which causes the loss of all CSPs, rendering the device inoperable. This is a severe limitation that is tackled by regular maintenance, i.e., personnel has to manually inspect and replace batteries before they are fully depleted.

However, an insufficient battery voltage is not the only reason that causes the tamper detection mechanism to trip. The shipping process is yet another significant obstacle since the mechanism is armed at a trusted facility and then exposed to uncontrolled temperature, mechanical shocks, and vibration during transportation. As these devices implement Environmental Failure Protection (EFP) to limit physical tampering with the materials, e.g., melting potting away or literally freezing BBRAM contents [Sko02], their environmental operating window is typically small. For example, the upper ambient temperature limit for the similar IBM 4765 HSM is 60 °C during shipping, already taking into account that the HSM is in a "thermally insulated box with gel packs" [IBM12]. Since the monitoring circuit is permanently in operation

exceeding environmental limits even for a short time during transport may result in the inadvertent detection of an attack.

In contrast to these issues, batteryless PUF-based security enclosures are not prone to such problems as they are entirely powered down when not in use, e.g., during transport and storage. This makes them mostly immune to environmental conditions during shipping, including temperature and mechanical shocks. Due to their batteryless design, their shelf-life is also not inherently limited by a battery and regular maintenance can be omitted. This leads to the consideration of the next technological step of batteryless PUF-based security enclosures that were designed to supersede the aforementioned solutions.

5.3 The Present: PUF-Based Enclosures

When PUFs were first introduced in 2002 [PRTG02], their properties included tamper-resistance, i.e., attempted probing or an intrusion inevitably causes a change in its physical parameters. However, subsequent publications soon shifted that focus towards secure key derivation based on PUFs, but without the necessity of tamper-resistant properties [GCDD02]. This is a viable approach for an Integrated Circuit (IC) design comprising PUFs, since they are typically just another component in a larger system such as a complex SoC. Among different implementations, such as optical PUFs [EFK+12], silicon-based PUFs are very prevalent. In general, they offer only limited tamper-resistance as long as no countermeasures are employed, since their secrets can often be extracted during runtime by probing. This has been demonstrated by invasive [HNT+13] methods, by semi-invasive [MSSS11a] attacks, and by non-invasive approaches exploiting side-channel leakage [MSSS11b]. Still, a large body of work [HYKD14, MV10] then concentrated on this research direction, while only very few continued working on PUFs with tamper-evident properties.

5.3.1 Coating PUF

The Coating PUF [TSŠ+06] is one example of tamper-evident PUFs. It protects an IC by covering its top with capacitive sensors, which are again coated with a material of randomly varying physical properties. The material contains particles of two different dielectric constants, which are randomly positioned during manufacturing. During operation, the chip autonomously extracts the material's local unique properties by measuring the capacitances and finally derives the secret key. If an attack has occurred previously and the coating or sensors have been damaged, reconstructing this key is infeasible.

Despite this concept can, in fact, provide sophisticated protection, it is accompanied by several drawbacks. First, covering *every* IC of an embedded device with capacitive sensors and a coating requires a fully customized and therefore costly manufacturing of every IC under protection. Secondly, such an approach depends on the support of the IC's manufacturer, who must be willing to integrate those extensions. Thirdly, not

all attacks require access to the top side of the chip, e.g., laser fault injection applies light to the backside of the chip [SHS16]. Additionally, access to the PCB would still be possible and thereby allows various attacks, such as glitch or side-channel attacks. This is the most severe disadvantage since this concept allows protecting only single chips and a holistic approach, including the entire embedded system, is infeasible.

Due to this, I focus on non-silicon, *system-level* PUFs, i.e., tamper-evident PUFs that enclose a significant portion of the system, thereby obstructing physical access. Based on this requirement to protect a system as a whole, I first discuss a solution based on an Optical Polymer Waveguide and finally present the concept of B-TREPID.

5.3.2 Optical Waveguide Polymer PUF

Vai et al. presented an optical PUF [SFIC14] with its corresponding system architecture [VNK+15]. Their system employs LEDs that send light through a polymer waveguide that covers the critical area of the system. An imager receives the scattered light and the key can be successfully generated, as long as the waveguide has not been tampered with.

As the waveguide solely covers the top of the system, its edges and bottom remain unprotected and accessing critical signals on the PCB is not prevented. Moreover, the paper omits whether the waveguide is continuously monitored after power-on or not; thus, such a system could perhaps be attacked during runtime to extract keys in volatile memory. Unfortunately, the publication also does not include a statistical evaluation or results from attacks, which makes assessing its provided security very difficult. Nevertheless, I consider the presented concept as an important first step towards PUF-based security enclosures.

5.3.3 Batteryless Tamper-Resistant Envelope with a PUF and Integrity Detection (B-TREPID)

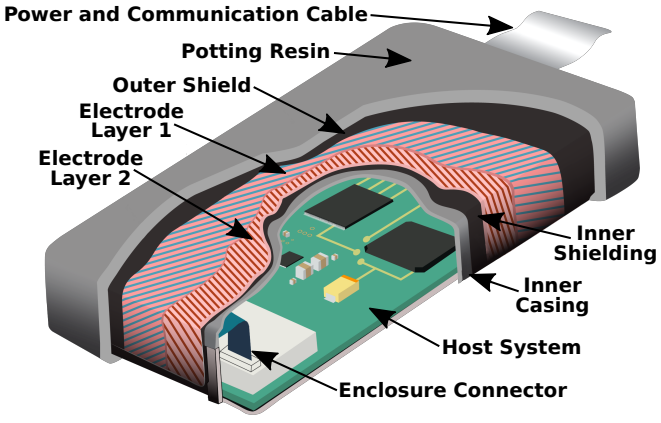


Figure 5.6: Host system protection by B-TREPID.

B-TREPID is another PUF-based security enclosure solution which was presented in 2018 [IOK+18a] and its details were discussed in several publications [IHL+17, IOK+18b, OIHS18, OHHS18, ION+18]. The concept was developed in a joint work of the Fraunhofer Institutes AISEC, EMFT, and IMS. My research contributed to this project and I focused on the secure measurement and system integration aspects.

Similarly to the GORE envelope, the B-TREPID enclosure is created from a flexible foil containing conductive traces, which is wrapped around the entire device, as sketched in Figure 5.6. This concept combines the advantages of PUFs and integrity-oriented enclosures — but without requiring battery-backup. The system's PUF is the capacitive coupling between the traces of the mesh, whose trace width and thickness are subjected to random manufacturing variation during production. A measurement system extracts the capacitance variation PUF from the enclosure and additionally verifies its integrity by checking the traces for open and short circuits.

The basic startup procedure is as follows. First, the measurement system verifies the enclosure's integrity and continues if and only if there are no faults. Next, the same measurement system evaluates the capacitance PUF and passes the resulting data into a key derivation function. Subsequently, the PUF key is further processed by applying error-correcting codes to counteract effects such as noise and environmental influences. Afterward, the system and its CSPs can be decrypted with the derived PUF-based key. The startup procedure is complemented by several runtime tamper-detection mechanisms that not only supervise the mesh's integrity but limit the allowed change in capacitance during operation as well. If at any stage an attack is detected, zeroization of all CSPs and related critical measurement data is triggered to secure the system and to render it inoperable.

When the system is powered down in a well-controlled manner, all data in volatile memory is wiped and only the data, encrypted by the PUF key, remains present on the system. Thus, battery backup is not required anymore and there is no maintenance effort during the storage period. During the next startup, the data is again decrypted with the key derived from the PUF, given that the enclosure is still intact. If an adversary manipulated the system while being powered off, the physical properties of the enclosure and ultimately the PUF would have changed. This causes either the initial enclosure integrity verification to fail or the PUF key derivation not to succeed. Thereby, the system's decryption key is inherently linked to the enclosure's integrity. This is a powerful feature that forces an adversary to circumvent both security measures, which are founded in different physical concepts, at the same time.

The mesh comprises four layers, whereas the inner two layers contain a fine mesh of orthogonal traces in a meandering layout, which are protected by two shielding layers covering the enclosure. The mesh's traces have a width and spacing of 100 µm each. They span across the entire surface so that there are no blind spots with a gap of more than 100 µm. Due to practical considerations regarding entropy and measurement range, the traces are grouped into 16 electrodes per layer. The measurement system evaluates the coupling between traces on both layers in a differential, as well as absolute, manner, resulting in 128 and 256 measurement values, respectively. The absolute measurement

results and the integrity verification data is solely used for intrusion detection. Only the differential measurement yields sufficient entropy and contributes to the PUF-based key generation. Statistical evaluations have shown that the differential capacitance values follow a Gaussian distribution, which is backed by the underlying stochastic model [Imm19].

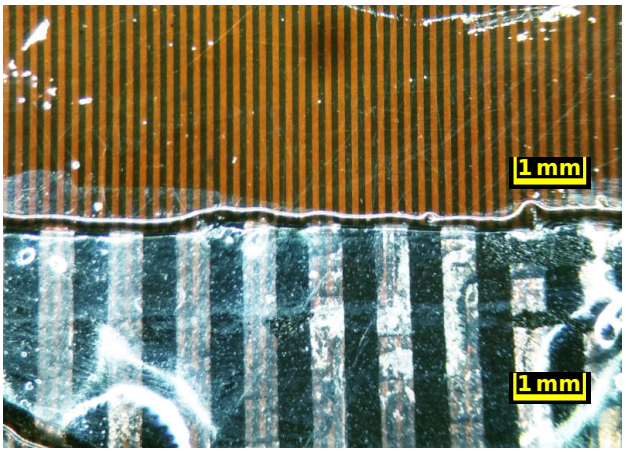


Figure 5.7: Comparison between B-TREPID (top) and GORE (bottom) trace dimensions.

To compare the structure size of the GORE envelope to the solution provided by B-TREPID, I took a microscope image for comparison, shown in Figure 5.7. The image comprises one trace layer of each enclosure immersed into liquid isopropanol to increase contrast while reducing surface reflections. The large difference in structure size becomes evident. The solution by GORE is about a factor of five larger.

There is a trend going to smaller structure sizes as requested by customers. For example, an attack performed with a 0.3 mm drill is guaranteed to be detected by B-TREPID, which was the design goal for this system. It was verified practically that the current layout and structure size fulfills this requirement. However, although the B-TREPID *concept* is mostly complete, it is not ready to replace solutions such as the GORE envelope, yet, since further optimizations of material properties, manufacturing yield, and enclosure size are necessary.

5.4 The Future: Open Challenges for Security Enclosures

Considering my previous analyses and related work, I identified several open challenges. This is true in particular for the targeted market of low to mid-volume products ranging between 1 000 and 100 000 devices per year. The development and production of a security-hardened ASIC for such a product is usually unprofitable due to the associated high fixed costs. In contrast to this, a well-designed security enclosure could be used for multiple low to mid-volume products. Especially some mask-less

manufacturing technologies, such as inkjet printing, allow the profitable production of individual enclosure sizes and designs, especially for low-volume products.

All this underlines the demand for PUF-based security enclosures as they could fill the gap in the market of security solutions for multiple-chip systems with low to medium production volumes. However, before PUF-based security enclosures can become the next standard for device protection, some more issues have to be resolved.

- Smart materials with a strong physical avalanche effect are needed that cause a propagation effect upon tampering, which thoroughly destroys the key contained therein. This should be especially achieved without hazardous materials such as explosives, acids, and bases, which could interfere with the safety of the system and the user.
- Scalable solutions for PUF-based enclosures are required that provide well-understood building blocks that serve as the basis for larger-scale enclosures with an almost arbitrary size or form-factor.
- For entirely enclosed systems, similar to GORE envelope-based devices, the secure routing of signals between the protected inside and the unprotected outside is still a partially open question. Seams or openings have to exist that allow power signals and communication interfaces to reach into the inside. However, adversaries could abuse those openings to gain access. Thus, they need to be thoroughly protected especially for enclosures targeted on an even smaller drill protection.
- In contrast to fully-enclosed systems, devices such as the HP Atalla, which are only protected by a cover-solution, need PCB edge protection. Thus, a structure must be designed and inserted into the PCB, which prevents drilling along the PCB into the protected space. This is especially important for smaller drill sizes and more sophisticated attackers.
- For both solutions, covers and full enclosures, manufacturing capabilities must be improved such that they provide a higher yield at a lower cost per piece. The goal should be manufacturing costs in the single-digit US dollar range.
- While there are standards for the security assessment of integrity-based enclosures, no explicit systematic exists for PUF-based solutions. This is especially true for technologies whose sensitivity might depend on the location of the attack on the enclosure and attempted repairs since the system could be powered-off during an attack. Furthermore, a statistical analysis of the underlying PUF data is currently difficult since there are no standardized tests such an enclosure has to fulfill.
- Since many companies doubt the security of battery-backed integrity-only approaches and have requested PUF-based solutions, real-world attacks need to be published that expose weaknesses and allow an honest evaluation of the security provided by previous solutions.

5.5 Conclusion

In this chapter, I surveyed existing commercial solutions of battery-backed enclosures. Then, I analyzed two implementations in more detail to support the argument that battery-backed approaches might not necessarily represent the best solution for tamper-resistance. In addition to that, several potential weaknesses in both analyzed systems were identified and described. Subsequently, I summarized PUF-based security enclosures, which were developed and presented in the last few years up to 2018. However, most of them suffer from one or more technical issues, which prevents them from replacing battery-backed solutions; thus, I discussed open challenges that will have to be resolved in the future.

Based on the experience with customers, novel solutions are desperately requested to fill the void that was left by discontinued commercial solutions. Moreover, even within the context of commercially available battery-backed solutions, many find themselves in applications where requirements are incompatible with batteries. Hence, there is a great demand for the development and *publication* of secure and reasonably-priced physical security enclosures.

Chapter 6

B-TREPID: Batteryless Tamper-Resistant Envelope with a PUF and Integrity Detection

The previous chapter showed that there are several outdated security enclosures and ongoing development is directed towards PUF-based solutions. One of those new systems is B-TREPID, which was summarized in Subsection 5.3.3. The concept and a practical verification were initially presented at the *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)* in 2018 [IOK+18a]. Additionally, the concept was filed as EU patent, has been published under `EP3550466`, and is awaiting decision as of 2019 [IO19]. An extended paper about the system and attacks thereon is scheduled to be presented at the *Conference on Cryptographic Hardware and Embedded Systems (CHES)* in 2019 [ION+18].

The concept's underlying idea was created by Hennig et al. in 2013 [HSZS13, HSZF13], but they focused primarily on product protection, i.e., the enclosure aimed to prevent unauthorized reverse-engineering and cloning of devices. Developments continuing in 2015 moved the focus away from product protection into the direction of physical security for high-assurance systems. Since this has shown to be a viable approach, the remaining dissertation focuses on the technical advances in this project until the end of 2018.

The scientific aspects of the B-TREPID security envelope were worked on mainly by two PhD students, i.e., Vincent Immler and myself, who split up the topic in the following way. The PUF measurement system, including the analog circuitry, the digital signal processing, the firmware, and the corresponding embedded system architecture, which eases integration into a larger system, were primarily handled by myself. Vincent Immler focused on the overall system design and especially the information technology aspects of the project, ranging from statistical evaluation of the PUF data, over quantization, up to key derivation functions [IHKS16, IHL+17] and the optimization of the enclosure electrode layout.

We defined the interface between our works at the digitized raw PUF data, i.e., the differential capacitance, which is passed by the measurement system to the preprocessing and key derivation functions. More information on the latter topics can be found in the corresponding dissertation [Imm19]. This separation has proven to be an adequate choice; however, close collaboration on these and all other, not explicitly mentioned aspects, was consistently performed.

6.1 Security Enclosure Technologies

As described in the previous chapter, security enclosures can be classified into two major types: Enclosures are either based on the monitoring of trace resistance [IMJFC13] or alternatively on PUFs [VNK+15, TSŠ+06]. The first technology is solely based on the verification of the enclosure's integrity by detecting open and short circuits of its incorporated traces. As a drawback, continuous operation relies on a battery and the monitoring system needs to be in operation even during storage and transport. Since this causes issues due to, for example, inherently limited shelf-life, maintenance effort, and sensitivity to rough transportation, alternatives were developed. In contrast to the systems mentioned above, novel tamper-evident PUF-based enclosures provide inherent security without a battery. These batteryless technologies are based on the repeatable generation of the system-individual cryptographic key from the enclosure's unique physical properties. The derived key encrypts either the critical security parameters or the entire system, depending on the specific use-case. Any intrusion destroys parts of the enclosure, its physical properties are altered and key derivation fails, thus, the cryptographic key is permanently lost and the adversary cannot decrypt the HSM's data.

Apart from the general concept, the Fraunhofer project COPYCAT proposed a specific architecture of a *capacitive* PUF-based enclosure [HSZS13, IOK+18a, ION+18] which exploits the variation in capacitive coupling between two layers of electrodes, i.e., the PUF. The electrodes serve a dual-purpose, as they are checked for integrity and are evaluated as a PUF; hence, the concept combines advantages from trace-monitoring approaches and PUF-based enclosures.

6.1.1 The COPYCAT and COVER Project

The concept of B-TREPID was primarily developed in the COPYCAT project. The project spanned across four years between 2015 and 2018 and was funded by the MAVO grant, which is one of the Fraunhofer internal programs. The team comprises three Fraunhofer Institutes, which contributed their own expertise to realize this project.

The Fraunhofer Institute for Microelectronic Circuits and Systems (IMS) was leading the overall project and focused on developing and manufacturing an ASIC that provides a single-chip measurement system. The Fraunhofer Research Institution for Microsystems and Solid State Technologies (EMFT) physically manufactured the envelopes in a unique process, tailored for the project. The Fraunhofer Institute for

Applied and Integrated Security (AISEC) developed methods for integrating the system, i.e., this includes the creation of the software and hardware of the measurement system, key generation algorithms, and analyzing the statistical properties. Since these tasks cannot be independently handled as they depend on each other, collaboration and exchange of ideas were a major part throughout the entire interdisciplinary project.

In addition to the COPYCAT project, the developments at the Fraunhofer Institute AISEC were strongly supported by Singapore's DSO National Laboratories in the COVER project. This project was based on the same PUF-based approach but with different constraints regarding foil manufacturing technology and the enclosure shape and type. Since the underlying concept is identical, the projects were able to contribute to each other and generate results that apply to both projects, especially in the measurement and integration aspects on which this dissertation focuses on. Thus, unless otherwise noted, descriptions and results in the remaining chapters apply to both projects.

6.1.2 Contributions

Throughout these two projects, my work focused on the following aspects:

- **The development of simulation models and optimizations** of the enclosure. Among the general concept, this chapter contains the results of this research.
- **The measurement system** that verifies the enclosure's integrity and extracts the PUF property. Additionally, the impact of attacks is demonstrated from the measurement system's point of view. The results are presented in detail in Chapter 7.
- **The system and firmware architecture** that enables the integration of the enclosure and its measurement system into an HSM. Chapter 8 describes the firmware architecture and the required security extensions that were developed to create security extensions for the FreeRTOS operating system.

6.1.3 Structure

First, Subsection 6.1.1 gives a short introduction to the projects that were involved in the development of B-TREPID. At next, B-TREPID is presented, starting with a comparison of envelope versus cover-based solutions in Section 6.2. Afterwards, Section 6.3 continues with a detailed description of the enclosure layout and Section 6.4 follows describing the layer stacks. At next, the physical properties of the enclosure and its traces are investigated in Section 6.5. Finally, a Finite Element Analysis (FEA) of the enclosure is presented in Section 6.6, which explains the influence of possible optimizations of the enclosure.

6.2 Envelope Versus Cover Solutions

Akin to the protection strategies for battery-backed enclosures, listed in Section 5.2, an embedded device can either be covered partially or entirely by a PUF-based protection system. In general, I observed two prevalent methods: A device is either surrounded by a *full enclosure* implemented via an envelope, such as the GORE solution shown in Subsection 5.2.1, or only the critical regions are protected by a top and a bottom *cover*, shown by reference to the HP Atalla system in Subsection 5.2.2. In contrast to the aforementioned concepts which support only one type, B-TREPID can be implemented as an envelope and a cover-based solution. For COPYCAT, the envelope option was chosen, while the COVER project preferred the cover-based solution. Each of the two enclosure options targets a specific use-case and is accompanied by specific advantages and disadvantages.

6.2.1 Full Enclosure via an Envelope

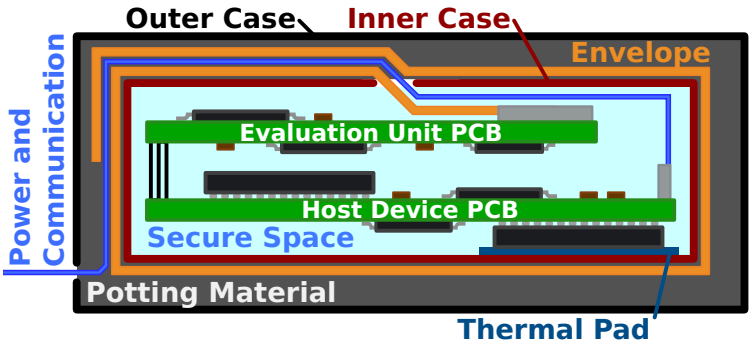


Figure 6.1: The primary implementation option of B-TREPID encloses the embedded device entirely via an envelope.

Figure 6.1 shows B-TREPID's concept which entirely wraps the embedded system into the security enclosure. The host device, implementing the HSM functionality, is in the center of the system. The enclosure's monitoring circuit, i.e., the evaluation unit, is either placed on the same PCB or alternatively atop and separately. Both devices are enclosed by an inner casing, which gives mechanical stability and a thermal interface to the outside. If required, thermal pads can be added between the individual ICs and the casing for improved power dissipation.

The casing is wrapped into the security enclosure; only a small opening remains, which allows the measurement, power, and communication signals to enter the inside of the secure space. The enclosure surrounds the entire device and has a significant amount of overlap at the region of the opening, thereby preventing an adversary from easily penetrating the protected space. The overlap's design goal is that there must be at least two changes in the direction of 90° or more when an adversary tries to

penetrate the system along the power and communication lines, thereby making such an attack very difficult if not impossible. To further counteract attacks, the enclosed system is embedded into potting material that hampers disassembling the enclosure or laser drilling attempts. Finally, an outer casing is added for eased handling and stability reasons.

The concept is currently only compatible with envelopes manufactured by the Fraunhofer Research Institution EMFT since other technologies, such as FlexPCB cannot provide the required mechanical flexibility. The envelope can be scaled and is able to enclose multiple-board systems or components of various heights as implied by Figure 6.1. Apart from the increased demand in enclosure size, there are no conceptual drawbacks or limitations; thus, this enclosure concept gives high flexibility to the system designer. Additionally, system security can be further strengthened, for example, by adding potting around or inside of the secure space. If weight is of more concern than security, such as in constrained mobile applications, the system might be implemented without any potting. However, the concrete implementation must be adjusted to the system's requirements, but this available flexibility eases the integration of B-TREPID into an embedded system.

6.2.2 Cover-Based Solution

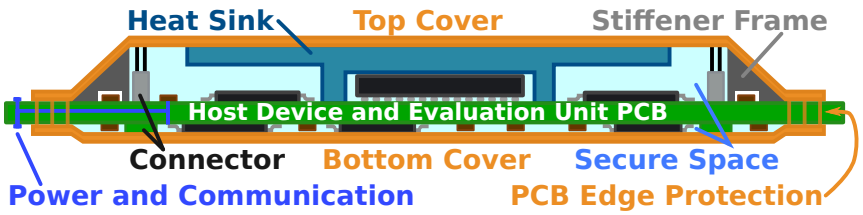


Figure 6.2: The second implementation option of B-TREPID covers the top and bottom of the device only.

Figure 6.2 shows the COVER project's alternative implementation of B-TREPID, which protects the top and bottom of an embedded device with two security covers. The PCB is the center of the system; it comprises the host system and the evaluation circuit. For this option, only a single PCB is supported, which also has to provide mechanical support to the enclosure. The PCB can contain any components; however, some height restrictions apply. Each side can either be flat or concave, usually, both sides are of a different type to include high components while maintaining a low overall device height. In this example, relatively high components may be placed on the top side of the PCB. The components on the bottom side must be less than one millimeter in height. For heat dissipation, a large heat sink is recommended, which creates a thermal interface between the IC and the enclosure. Since the concept does not employ an internal casing, this role is partially taken over by the heat sink, which spreads the heat along a large region.

The enclosure consists of two components, the top and the bottom cover. To connect the covers to the PCB and its evaluation unit, a connector is attached, which plugs into a socket on the PCB. Since the top cover has a comparably large height, it must be mechanically enhanced by a stiffener frame that helps to maintain its shape even under shocks and light force.

Both covers are manufactured in FlexPCB technology. This gives high flexibility since this is a standardized process that is offered by multiple vendors; however, the cost for such a cover is comparably high due to its multiple layers and dimensions. While the bottom cover is simply a rectangular and flat design, the top cover requires means to enable its mounting, i.e., well-placed cuts and overlapping regions allow bending the cover into its final shape. More details are available in Subsection 7.7.2 and in the associated publication [ION+18].

Mounting the covers onto the PCB is another demanding task. In order to prevent intrusion through the edge of the board, the seams of the covers must be interlinked with each other and the PCB. This structure, which must be integrated into the PCB, e.g., by vias and traces on inner layers, was named vertical protection structure and protects the PCB's edge. The structure has shown to be comparably complex to design, especially since communication and power traces must be able to pass it. Since this was not the focus of the project, its development has been postponed.

Altogether this concept is potting-free and has a considerably lower weight, enabling its deployment in special applications such as airborne systems. Furthermore, its reliance on standard FlexPCB technology gives higher flexibility and allows several manufacturers. However, cover-based solutions and its edge protection are commonly believed to deliver less security, compared to full enclosures.

6.3 Enclosure Layout

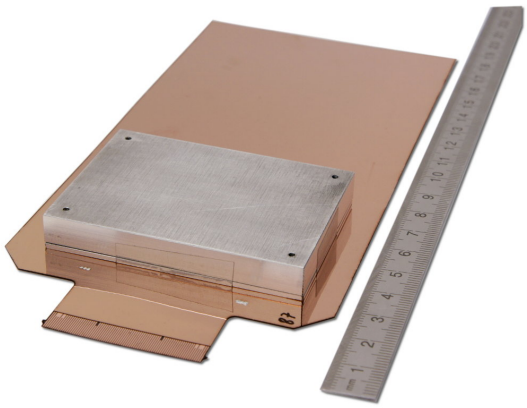


Figure 6.3: The COPYCAT envelope and a silver box, representing a sample device, which will be wrapped into the enclosure.

Figure 6.3 shows the COPYCAT envelope in its initial shape before it is wrapped around the device under protection. The enclosure is of larger size since it has to surround the device entirely, including some overlap. A wide 80-pin connector is present at the front side of the enclosure, which gives access to the internal traces, representing the PUF.

Altogether, this enclosure comprises four electrical layers, whereas only the shielding layers are visible from the outside. Two inner layers, containing a matrix of electrical capacitances, create the PUF. While there are different options of implementing such a layout, this section presents the solution chosen for the COPYCAT project. The COVER project is based on a conceptually similar but different layout [ION+18].

6.3.1 Electrode and Trace Structure

In general, a capacitor requires two electrodes, which are separated by an insulator, such as a dielectric, e.g., polyimide. For this project, which is based on a four-layer design, one electrode is on the inner top layer while the other electrode is on the inner bottom layer. To prevent interference and fringing fields, a grounded shield on the top and bottom surrounds the enclosure. The enclosure's electrode layout was developed to be scalable to the module's dimensions. To obtain multiple PUFs and sufficient entropy thereby, both layers are partitioned into several separate electrodes, e.g., 16 per layer, which are routed in a meander-style to fill the entire area.

The design also takes into account that it should not only allow for capacitance measurement but also for integrity verification of the traces; thus, both ends of each electrode must be accessible to check for defects. I.e., An open circuit due to an interrupted trace and a short circuit resulting from an unintended connection to any other traces must be detectable.

Additionally, the electrode layout was chosen accordingly to the capabilities of the manufacturing process. Due to the large enclosure size and fine trace width and spacing, usually 100 µm or even less, the manufacturing processes are at their limits and an optimized design significantly increases the yield. For example, the reliability of the laser via process, which creates connections between both inner layers, was a source of concern initially. As a consequence, a layout with a low via count was chosen for the COPYCAT project.

The resulting electrode layout is depicted in Figure 6.4. The connector is placed on the left-hand side while the enclosure spans further into the bottom and right-hand side direction.

Reflecting their function, the traces on the top layer are named TX electrodes, since they carry the excitation signal during the capacitance measurement. Similarly, the traces on the bottom layer are the RX electrodes as they receive the excitation signal. Each electrode has an index running between 1 and the number of electrodes on the layer, N and M for the TX and RX electrode count, respectively. This results in a total number of $N \cdot M$ capacitors inside the enclosure. For integrity verification, both ends of each electrode are accessible, whereas a subset "R" suffix designates the rear end, e.g., $\mathrm{RX_{1R}}$ and $\mathrm{TX_{1R}}$.

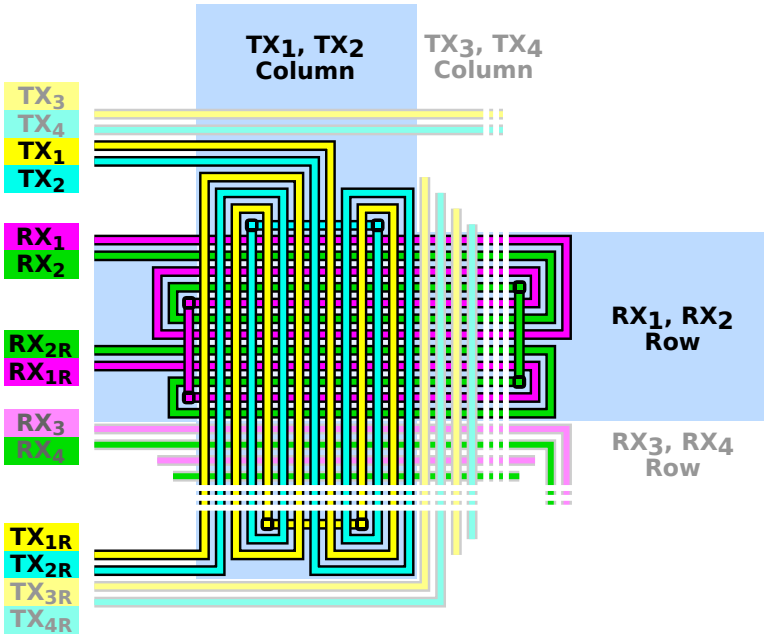


Figure 6.4: The COPYCAT envelope electrode layout concept.

Since the electrode width is low compared to the enclosure size, the electrodes are routed in a meander-layout to fill the entire surface. The layout, which is spanning the electrodes across the device under protection's entire surface in a meandering style, guarantees that there are no unprotected spots. For increased security, the layout is based on a pairwise routing, which guarantees that an electrode is never a direct neighbor to its own since this would cause some short circuits to be undetectable, thereby creating weak spots.

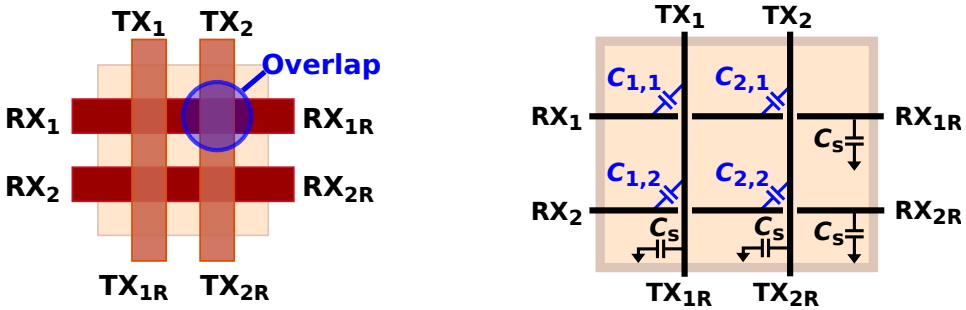
Additionally, the pairwise routing enables the possibility of a differential capacitance evaluation and avoids several disadvantages. Statistical analyses have demonstrated that the absolute capacitance variation, directly taken from the coupling between two electrodes, is unsuited as a PUF [IHOS17]. Trace thickness and width variations usually show a circular pattern, due to the manufacturing process [Bri04]. This causes a correlation between the location of the electrodes and their PUF property. As this contradicts the PUF's unpredictability, a *differential* capacitance measurement is employed, which cancels global offsets but reduces the number of available measurement values to $M \cdot N/2$.

Since the electrode layout concept is not bound to any structure size, it can be chosen accordingly to the manufacturing process and the targeted level of security. The trace width and spacing defines the minimum dill diameter, which is guaranteed to be detected. For example, to detect a common 300 µm drill, a trace width and spacing of 100 µm or less is required. These were the parameters chosen for COPYCAT since they are also near the process limit of the Fraunhofer Research Institution EMFT.

Additionally, manufacturing structures near the process limit has shown to lead to more relative variation, thereby increasing the entropy of the PUF. Regarding manufacturing, this layout has only two mandatory vias per electrode, which is significantly less compared to alternative solutions [ION+18]. Thereby, vias are not any longer the yield-limiting factor in manufacturing.

6.3.2 Equivalent Circuit and Nomenclature

Each capacitor is built from hundreds up to thousands of tiny trace overlaps connected in parallel, each individually contributing to the total variation in capacitance. Since the electrical resistance is considered being negligible, those little capacitors are assumed to be connected in parallel, thereby contributing their capacitance and variation to one large capacitor per TX-to-RX combination.



(a) Mesh layout level. **(b)** Mesh circuit level.

Figure 6.5: Simplified model of a 2×2 electrode design.

For the following considerations, I reduce the enclosure to a simplified 2×2 model with a single overlap depicted in Figure 6.5. Despite the model is significantly reduced compared to the COPYCAT implementation with $N = M = 16$, there is no limitation of generality.

The following nomenclature is defined for the remaining dissertation: As stated beforehand, N and M correspond to the number of TX and RX electrodes, respectively. While the primary ends TX_a and TX_c provide access for integrity verification and capacitance measurement, the other ends denoted by $\mathrm{TX}_{a\mathrm{R}}$ and $\mathrm{TX}_{c\mathrm{R}}$ are solely for integrity verification. The subset index denotes the electrode number and ranges in $[1, N]$ for TX electrodes and $[1, M]$ for RX electrodes.

For general considerations which do not refer to a specific electrode or capacitance, C_m denotes the mutual capacitance between one TX and one RX electrode. The same applies to C_s, which describes the capacitance of a TX or RX electrode towards the shielding, i.e., the parasitic capacitance. The differential capacitance of two TX electrodes towards one RX electrode is denoted as ΔC.

For cases that refer to a specific set of electrodes, capacitances are denoted by C with two or three subset indexes for the electrode numbers. If two indexes are

present, $C_{a,c}$ denotes the *absolute/mutual* capacitance between TX electrode a and RX electrode c, e.g., $C_{1,2}$ in Figure 6.5. If three indexes are present, $C_{a,b,c}$ denotes the *differential* capacitance of the TX electrodes a and b towards the RX electrode c. The differential capacitance is defined as

$$C_{a,b,c} = C_{b,c} - C_{a,c}. \tag{6.1}$$

A special case is the parasitic capacitance C_s, which is assumed to be approximately identical for each electrode due to their similar size, thus, they do not need to be differentiated.

The measurement system supplies additional information, such as the phase of the measured signal whose expression is $\varphi_{a,c}$ and $\varphi_{a,b,c}$ for the absolute and differential measurement, respectively. When referring to a measurement value that was obtained under special conditions, this is denoted by a superset annotation. For example, the evaluation of the phase of the second harmonic at $f = 66.6\,\mathrm{kHz}$ for $C_{1,2,3}$ is expressed as $\varphi_{1,2,3}^{f=66.6\,\mathrm{kHz}}$.

6.3.3 Electrode Arrangement

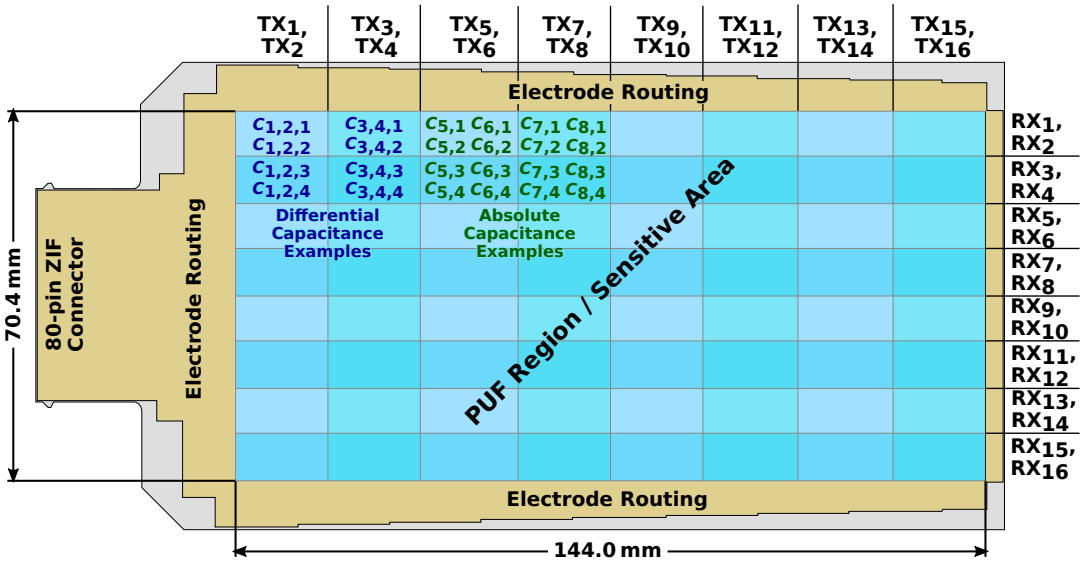


Figure 6.6: True to scale drawing of the COPYCAT envelope including electrodes.

The COPYCAT envelope is based on a design with $N = M = 16$, resulting in 16 electrodes per layer, as shown in the true to scale drawing in Figure 6.6. The electrodes span a total area of $144.0\,\mathrm{mm} \times 70.4\,\mathrm{mm}$, which is the secure area that contains the PUF and is sensitive to tampering. However, the total enclosure is larger since additional routing is required, e.g., for the rightmost TX traces, which are wired to the connector on the left edge.

In the following, I define a *trace segment* as the part of a trace that runs straight from one edge of the sensitive area to the opposite edge, e.g., TX_1 has seven segments in the layout example of Figure 6.4. In the PUF region, the TX electrodes of the enclosure are running vertically while the RX electrodes are running horizontally. For COPYCAT, the sensitive area of the enclosure is entirely filled with traces and has 16 TX and 16 RX electrodes whose layout is similar to the figure above. In the prototype, the width and spacing of the electrode trace segements is 100 µm each. This results in

$$\frac{1}{16} \cdot \frac{144.0\,\text{mm}}{100\,\mu\text{m} + 100\,\mu\text{m}} = 45 \tag{6.2}$$

vertically running trace segments per TX electrode on the top layer and

$$\frac{1}{16} \cdot \frac{70.4\,\text{mm}}{100\,\mu\text{m} + 100\,\mu\text{m}} = 22 \tag{6.3}$$

horizontally running trace segments per RX electrode on the bottom layer. Thus, each TX-to-RX combination, corresponding to a single absolute capacitance, is composed of

$$45 \cdot 22 = 990 \tag{6.4}$$

tiny capacitors, created by trace overlaps. Since the enclosure comprises $16 \cdot 16 = 256$ absolute capacitances, it contains a total number of 253 440 trace overlaps and the total trace length is more than 100 m in the PUF region alone.

Figure 6.6 shows the specific location of some absolute and differential capacitances for COPYCAT. The number of independent differential capacitances is half the number of absolute capacitances; hence, there are 128. The connector on the left edge of the enclosure has 80 pins which comprise $2 \cdot 2 \cdot 16 = 64$ pins solely for the electrode inputs and outputs, i.e., TX_1, TX_{1R}, ..., TX_{16}, TX_{16R}, RX_1, RX_{1R}, ..., RX_{16}, and RX_{16R}. The remaining pins connect the shielding and provide spacing between TX and RX signals on the connector.

6.4 Enclosure Layer Stack of COPYCAT

In both implementations, the enclosure consists of four electrically conducting layers that are separated by an insulator. Since the underlying layer stack of the COVER project is conceptually similar to the one of COPYCAT, only the latter one is explained for brevity.

6.4.1 Current Layer Stack

The layer stack of the COPYCAT envelope is depicted in Figure 6.7. The electrodes for the capacitive PUF structure are built up around a layer of UPILEX 50s, which is a 50 µm thick polyimide substrate which acts as an insulator. The electrode and shielding layers consist of copper, which is placed atop a few nanometers thin layer of chromium that increases the adhesion to the polyimide. Since the thin chromium layer

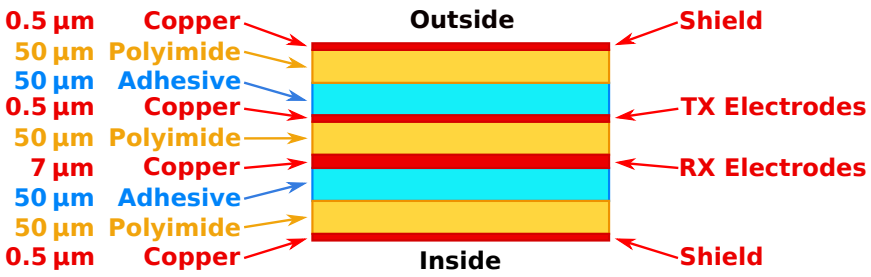


Figure 6.7: The COPYCAT layer stack, containing two electrode layers, covered by two shielding layers which are all built up around a polyimide substrate.

is negligible from an electrical point of view and is only required for mechanical reasons, it is not taken into account in any further considerations. The TX and RX copper layers are created by sputtering and have an initial thickness of 500 nm. A galvanic process strengthens the RX layer until its thickness reaches 7 µm, creating a more rigid layer that can finally act as an electrical connector without getting mechanically damaged. Throughout these steps, both inner copper layers are structured via a lithography process to create a specific electrode layout for the capacitive PUF.

After the inner layer setup was created, two shielding layers are added at their outside. They are made from similar materials, i.e., a 500 nm copper layer atop a thin chromium layer which is placed on a 50 µm polyimide substrate. These layers do not become lithographically structured as the inner and outer shielding are continuous copper planes.

These three components, the inner layers and two outer shields are then glued together by a 3M VHB9460 adhesive tape with a thickness of 50 µm. Altogether, this results in a total thickness of around 250 µm. While such a layer stack can be manufactured, its mechanical flexibility is rather limited; hence, wrapping the enclosure around the device under protection is difficult and damages the enclosure's electrodes, i.e., electrodes break and create open circuits.

6.4.2 Improved Layer Stack

Due to the limited mechanical flexibility, the Fraunhofer Research Institution EMFT investigated an enhanced layer stack as an alternative, shown in Figure 6.8, which shall provide more flexible envelopes. The innermost two layers and their substrate remains the same, but the shielding is modified. This structure is no longer added separately afterward, the dielectric and shield are now directly deposited onto the inner layers. This is accomplished via a screen printing process that uses comparably cheaper masks and manufacturing compared to lithography technology. After this step has been finished, the shielding is also applied via a screen printing process, as shown in Figure 6.9. Both materials, the dielectric and the shielding material, were selected with respect to high mechanical flexibility so that the envelope is less rigid, compared

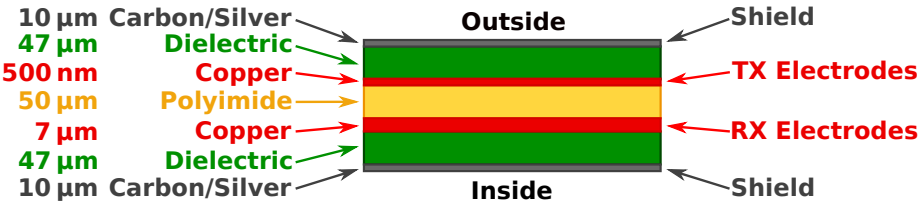


Figure 6.8: The enhanced COPYCAT layer stack, containing two electrode layers and the screen printed dielectric and shielding layers.

Figure 6.9: Two technology samples of the screen printed COPYCAT shielding using carbon (left) and silver (right) pastes.

to the first layer stack. Different options were tested for the shielding, including carbon-based and silver-based materials; however, the latter one has shown significantly lower resistance, which improves the shielding. My estimations and measurements have shown that a thinner dielectric leads to a comparably larger parasitic capacitance; thus, there is a trade-off between flexibility and electrical properties. Altogether, due to the thinner and more flexible materials, this results in improved mechanical parameters and the technology supports enclosure thicknesses lower than 250 µm.

At the time of writing of this dissertation, this experiment and further optimizations were ongoing. The first results are auspicious as the manufacturing of samples was successful and the envelopes appear to have much higher flexibility.

6.5 Envelope Physical Properties

The first available samples originated from COPYCAT, which were utilized to verify the concept. However, some technical issues arose from the copper-based connector surface, which had to be resolved first. Additionally, the existence of variation in the trace's physical properties was another open question.

6.5.1 Connector Material Issues

For experimentation, the envelopes are connected to the measurement system's 80-pin Zero Insertion Force (ZIF) socket, which should allow a quick, easy, and repeatable exchange of envelopes. However, during experimentation with the first samples, I noticed that their performance gradually declined as the measurement system incorrectly reported more and more electrodes as being interrupted. As the failures were intermittent and could be partially resolved by applying additional force on the ZIF connector, I attributed this to a connector-related issue. Unfortunately, it caused hardly reproducible measurements of unacceptable quality up to entirely failing envelopes.

The reason quickly detected as the dark color of the envelope's connector gave a strong clue for a chemical degradation. Cleaning is difficult as the very delicate 7 µm thin connector surface could easily be damaged, e.g., by sandpaper. Hence, I developed an appropriate and gentle chemical cleaning procedure.

As the connector is made from copper, a thin but non-conducting layer of copper(II) oxide (CuO) forms on its surface, which has a detrimental effect on the electrical connection. The oxide layer is insoluble in water and alcohols. One chemical method, recommended by the Fraunhofer Research Institution EMFT, removes the oxide layer by hydrochloric acid (HCl). Since the handling of this acid requires safety measures and is unsuited for the electronics lab, I had to find an alternative.

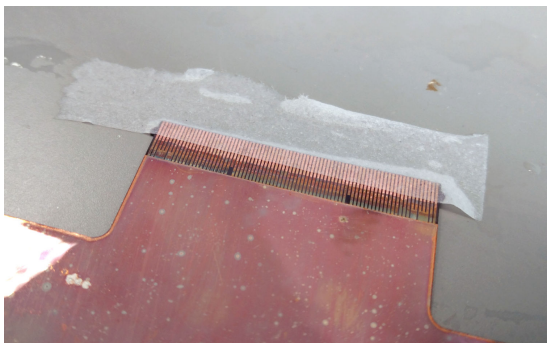


Figure 6.10: Cleaning an envelope connector with a tissue soaked in citric acid.

My method is based on citric acid ($C_6H_8O_7$) that dissolves the copper(II) oxide layer in a few minutes. This chemical is readily available at sufficient quality, e.g., for

removal of limescale in households. It is neither toxic nor does it have an immediate effect on the human skin, thus, it is comparably safe to handle; nevertheless, eye protection should be worn at all times and general safety measures apply. Since this chemical requires several minutes to sufficiently clean the connector, a tissue is soaked with citric acid and put on the front region of the connector, as shown in Figure 6.10. The tissue prevents spillage of acid and keeps it safely in place. Please note that the tissue must *not* touch the shielding as there is a risk that acid creeps in between shielding and electrode layer, thereby causing unwanted conductivity. After a few minutes, the tissue is taken away, the connector is rinsed with water, and then dried. A comparison between two envelope connectors, one before and one after citric acid cleaning, is shown in Figure 6.11.

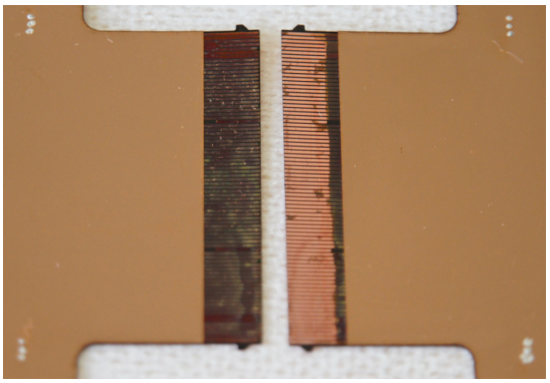


Figure 6.11: Enclosure connectors before (left) and after (right) citric acid cleaning.

Experience shows that precautionary cleaning before *each* measurement yields the best results and nullifies connector-related issues. The procedure is safe, cost-efficient, and very gentle to the sensitive envelope. During the project, many envelopes were treated multiple times, but none of them was destroyed thereby, proving the suitability of my method.

However, for a commercial application, this issue must be solved differently. If connectors are still to be used instead of soldering, the envelope's connection area could be made immune to oxidation by applying a thin layer of gold, similar to Electroless Nickel Immersion Gold (ENIG) in PCB manufacturing [Mil08].

6.5.2 Trace Width Variation

In order to analyze the manufacturing variation, one envelope sample was supplied without shielding layers. It was analyzed with the help of a microscope to make trace width variations visible. However, variations in the dielectric or trace thickness are not quantifiable with this method. Nevertheless, this gives a first insight into the process variation and shows that there is indeed some variation.

Please note that, despite their yellow appearance, the traces consist of copper. The deviation in color is caused by the artificial light source in combination with the microscope optics; however, this provides a very high contrast.

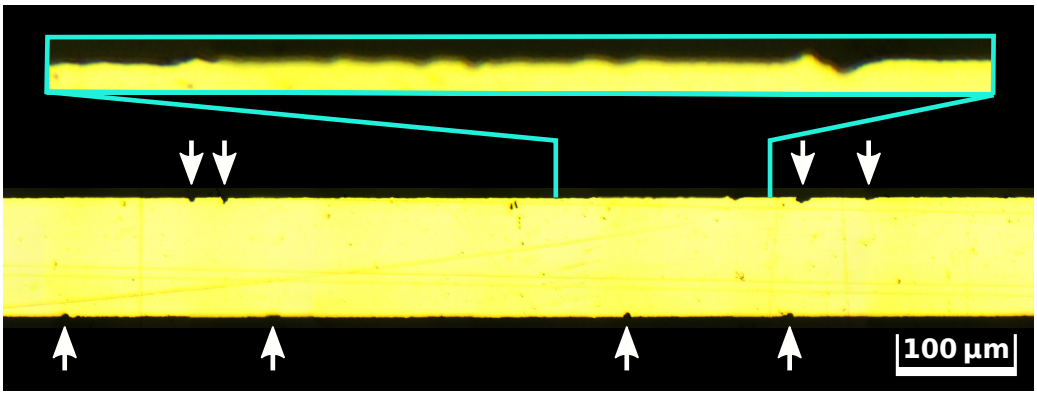


Figure 6.12: Microscope image of a TX electrode with clearly visible variation in width due to edge roughness.

Figure 6.12 shows one 500 nm thin TX electrode of the envelope which was stitched together from several images. Several noteworthy indents are marked with a white arrow, and these locations have a reduction in trace width of several micrometers. An additionally zoomed part of the trace in the upper region shows the edge roughness. Hence, there is variation in the trace's width, which is apparent even on such a small sample trace with a length of less than 1 mm in total.

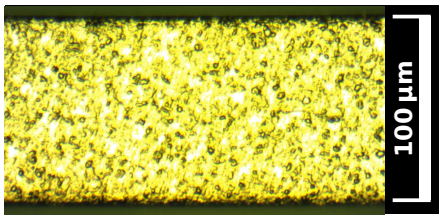


Figure 6.13: Microscope image of an RX electrode with porous surface which is another form of manufacturing variation.

The galvanically reinforced RX trace with a thickness of 7 µm is depicted in Figure 6.13. This trace shows less width variation at its edge, presumably since the galvanic process tends to fill out those gaps. However, the process causes a very uneven surface that has countless indents on its top and appears porous. This also seems to be a plausible source of variation.

Although dielectric and trace thickness variations could not be assessed, this small example already shows that there is some variation in the enclosure originating from

manufacturing which can partially explain the PUF behavior. Modifications to the manufacturing process can increase such variation further, e.g., by intentionally adding particles similar to the approach used for the coating PUF [TSŠ+06].

6.6 Finite Element Analysis of the Enclosure Design

While the first revision of the COPYCAT envelope was targeted toward easy manufacturing, high security, and proper drill protection, other aspects such as capacitance design were of less priority at this time. Another reason was that the enclosure is a complex device consisting of multiple materials and a sophisticated electrode structure; hence, quick and straightforward estimations are hardly possible. To counteract this drawback, I conducted a FEA of the enclosure design in cooperation with the Technical University of Munich (TUM), who provided the COMSOL Multiphysics simulation environment. Thereby, the influence of parameters such as substrate thickness and materials shall be identified, which enables an optimization of future enclosure designs. Furthermore, the results obtained via the measurement system shall be compared against the simulations to verify its correct functionality and to validate the enclosure FEA model.

The COPYCAT envelope comprises a 0.1 mm fine 16×16 electrode mesh with 990 trace overlaps per TX-RX combination and a total trace length of a little more than 100 m. Simulating such a structure with sufficient accuracy easily exceeds the capabilities of conventional computer systems; thus, simulations were performed on designs with heavily reduced complexity. By choosing appropriate simulation models, the results are still valid.

6.6.1 Insufficiency of Simple Approximations

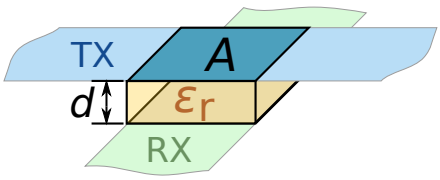


Figure 6.14: Simplified approximation of a single trace overlap as parallel plate capacitor.

A first attempt to estimate the electrodes' mutual capacitance was based on an approximation of a single overlap as a parallel plate capacitor. However, the result yielded only a fraction of the real capacitance. To investigate this further, the approximation is compared to an FEA of the electrode layout.

For this example, a simplified electrode arrangement was created, shown in Figure 6.14 with the TX electrode on top and the RX electrode on the bottom. The traces are assumed to have a negligible thickness, have the distance $d = 50\,\mu\text{m}$, and an

overlapping area of $A = (100\,\mu\text{m})^2$. The room in between is filled with an insulator with a dielectric constant of $\epsilon_\text{r} = 1$, i.e., air, for simplicity.

The mutual capacitance C_m for a parallel plate capacitor is given as

$$C_\text{m} = \epsilon_0 \epsilon_\text{r} \frac{A}{d} \tag{6.5}$$

with ϵ_0 being the vacuum permittivity [Bax97]. This results in the approximation yielding

$$C_\text{m} = 8.85 \cdot 10^{-12} \frac{\text{As}}{\text{Vm}} \cdot 1 \cdot \frac{100\,\mu\text{m} \cdot 100\,\mu\text{m}}{50\,\mu\text{m}} = 1.76\,\text{fF}. \tag{6.6}$$

However, the formula disregards fringing fields [Bax97] and any outer boundary conditions imposed by shieldings and should only be used for arrangements in which the plate size is large compared to the distance between them.

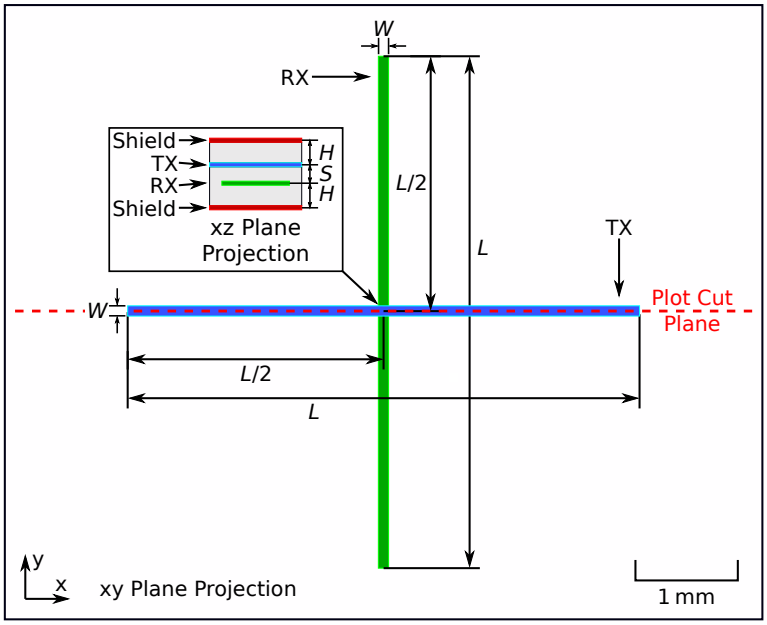


Figure 6.15: Simulation model for a comparison between simple approximations and FEA.

To estimate the error of this approximation, the electrode arrangement was constructed in COMSOL Multiphysics. The corresponding drawing is shown in Figure 6.15 with $S = 50\,\mu\text{m}$ for the substrate thickness, $H = 100\,\mu\text{m}$ for the shield spacing, $W = 100\,\mu\text{m}$ for the trace width, and air as dielectric. The simulation varies the trace length L from 100 µm up to 300 µm, corresponding to an exact overlap and both traces overhanging by 100 µm each, respectively.

The computed capacitance is compared to the FEA result in Figure 6.16. As long as there is no overhang, i.e., at 100 µm trace length, the approximation formula, drawn as red dashed line, is relatively close to the FEA result. Further simulations showed

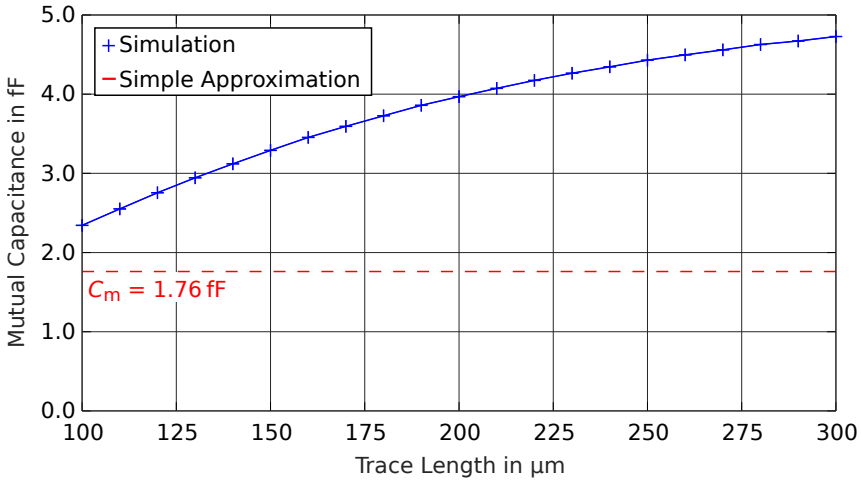


Figure 6.16: Comparison of a parallel plate capacitor approximation to FEA results for different trace lengths and overhangs.

that the present deviation stems partially from fringing fields due to a relatively high trace distance compared to their dimensions. When the trace length is increased and overhang is created, the mutual capacitance rises due to additional fringing fields. For $L = 300\,\mu m$, which corresponds to $100\,\mu m$ overhang at each side similarly to the COPYCAT layout, the mutual capacitance approaches 4.67 fF. Compared to the approximation with 1.76 fF, the correct value is larger by a factor of more than 2.6. Thus, detailed FEA is required as the approximation of the enclosure's structure via a parallel plate capacitor is insufficient and suffers from an extreme error.

6.6.2 Influence of Shield Distance

I noticed that the distance between the traces and the shield has a significant effect on the resulting mutual capacitance. This is unexpected at first glance as the shield is outside the overlapping region and any further coupling effects are unknown. Thus, I conducted additional analysis via the finite element method of this observation to identify the reason.

For the analysis, the same model of Figure 6.15 is applied whereas the trace length is now fixed at $L = 5\,mm$ and the simulation varies the distance between the traces and the shield between $H = 8\,\mu m$ and $H = 750\,\mu m$. The results, depicted in Figure 6.17, clearly show that increasing the shield distance will multiply the mutual capacitance. As long as the shield is close to the electrodes, a minor distance variation will have a significant influence on the mutual capacitance. This effect is reduced for higher distances and the slope of the curve decreases.

As the influence of the shielding is comparably large, plots of the resulting electric field distribution and potential were created for selected simulations with $H = 50\,\mu m$,

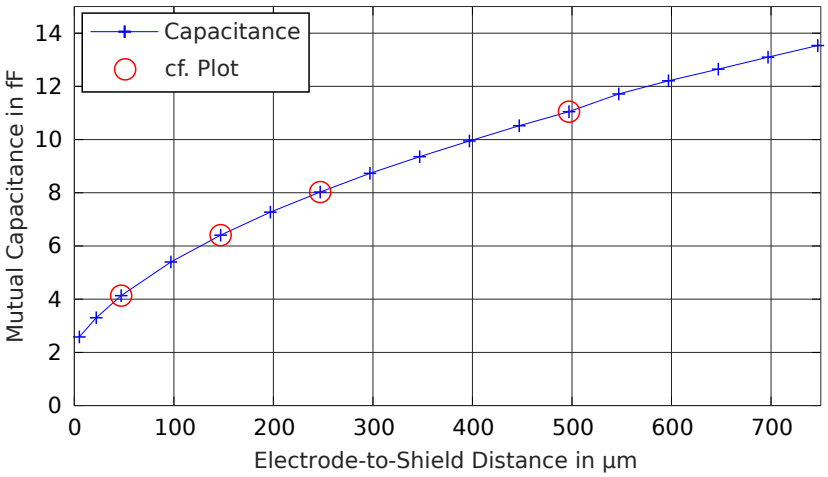


Figure 6.17: Influence of the electrode-to-shield distance on the mutual capacitance, detailed electric field plots are provided in Figure 6.18 and 6.19.

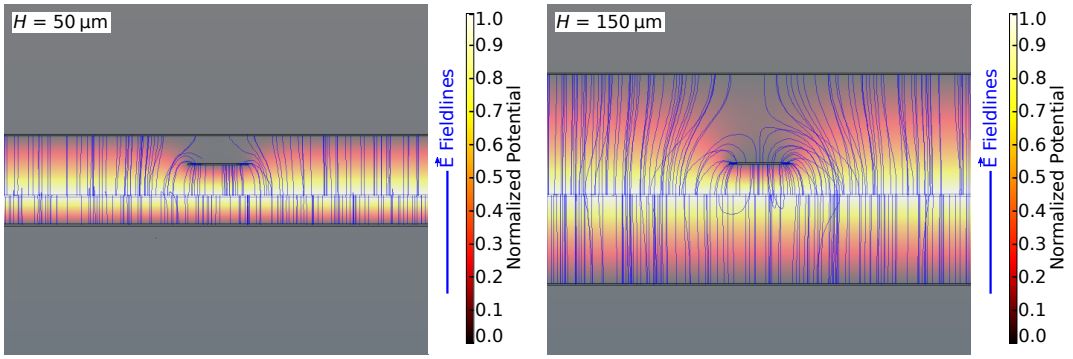


Figure 6.18: Electric field and potential plot for $H = 50\,\mu m$ and $H = 150\,\mu m$.

150 µm, 250 µm, and 500 µm which are shown in Figure 6.18 and 6.19. The cut plane for the potential was chosen as depicted in Figure 6.15, i.e., the viewing direction points along the RX electrode and the TX electrode runs horizontally. The background color indicates the normalized local potential ranging between 0.0 at the grounded shielding and RX electrode up to 1.0 at the TX electrode. The electric field lines, drawn in blue, originate from the TX electrode and either run to the RX electrode or the shielding.

The figures clearly explain the reason for the increased mutual capacitance. As long as the shield is close to the electrodes, the electric field is mostly limited to the overlapping region only with very few fringing fields. However, as the shield is gradually moved into distance, more fringing fields start to develop, which also start and end at the overhanging regions of the traces. Additionally, the potential in the region

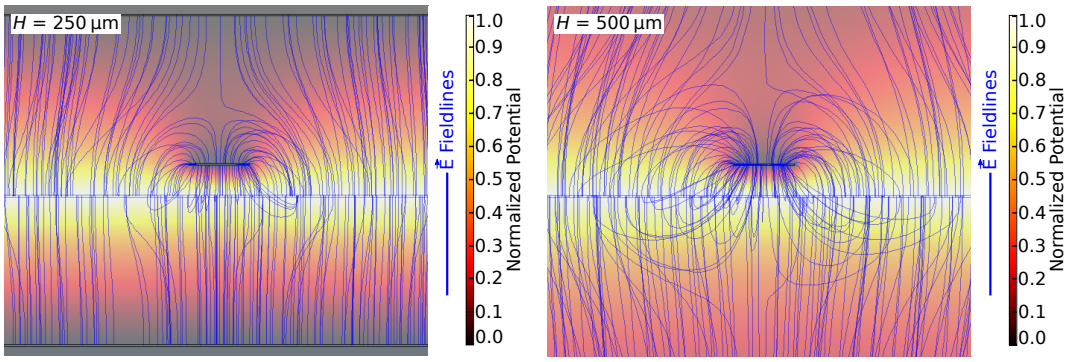


Figure 6.19: Electric field and potential plot for $H = 250\,\mu\text{m}$ and $H = 500\,\mu\text{m}$.

above the RX electrode begins to increase, thereby attracting more field lines and increasing capacitive coupling. Furthermore, while the field lines are rather straight for $H = 50\,\mu\text{m}$, they become distorted and start to bend around the RX electrode in the other plots.

Hence, moving the shield away allows more fringing fields to develop and adds coupling between TX and RX electrodes. Altogether, these effects multiply the mutual capacitance between two electrodes in the simulated arrangement and the results show that the shielding must not be ignored when simulating the enclosure.

6.6.3 Dependence on the Electrode's Polyimide Substrate Thickness

At next, the previous simulation model is improved to match the COPYCAT layer stack and electrode arrangement. This will yield the envelope's actual mutual capacitance and allows experimentation with its material parameters and electrode layout.

The simulation model is shown in Figure 6.20 and is based on actual layer stack with $H = 50\,\mu\text{m}$ for the thickness of the dielectric material and adhesive, $W = 100\,\mu\text{m}$ for the trace width, $D = 100\,\mu\text{m}$ for the electrode spacing, and $L = 5\,\text{mm}$ for the trace length. All conducting layers are assumed to be made from copper, the dielectric constant ϵ_r is 3.4 for the polyimide substrates [UBE18] and 4.7 for the adhesive [Lai11]. Four electrodes are present on each layer to generate the typical mesh structure. The outer two electrodes are grounded to limit fringing fields, similarly to other electrodes inside the enclosure, whereas the inner two electrodes form a pair of TX or RX electrodes.

Adding more TX and RX electrodes, e.g., six or eight per layer, has shown only a negligible change in simulation results, although simulation time grew strongly. Thus, this model is sufficiently accurate for the purpose of estimating the mutual capacitance. The FEA yielded a value of 16.9 fF for the mutual capacitance between two electrodes for a single overlap.

In the first part of the experiment, I modified the thickness of the TX and RX electrode substrate in the center of the layer stack, whereas all other parameters

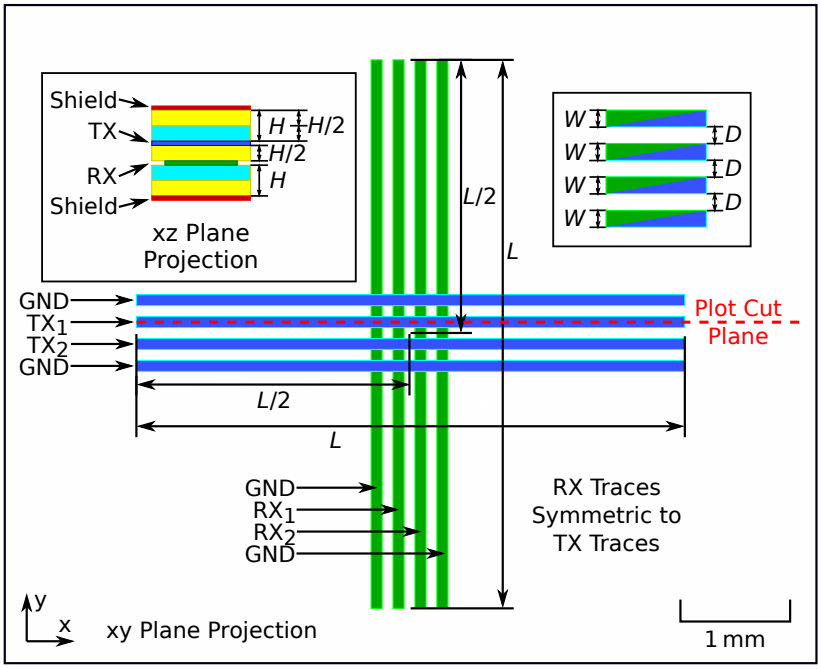


Figure 6.20: The COPYCAT envelope simulation model, including the layer stack and trace layout.

remained constant. The simulation sweeps this value between 5 µm and 300 µm. Please note that thicknesses below 25 µm may not be processable during manufacturing and are present only for completeness of the simulation.

The results depicted in Figure 6.21 show that the mutual capacitance decreases when the substrate thickness increases. The slope follows roughly an $1/x$ behavior as expected when the distance between two electrodes of a capacitor is increased. The results show that mutual capacitance and thereby partially the visibility of variations could be increased by reducing the thickness of the substrate layer. Additionally, this would also reduce the minimal via size due to the aspect ratio of via laser drilling. However, processing becomes more difficult since the mechanical robustness of the material is decreased.

Thus, the current value of 50 µm appears to be a well-chosen trade-off between mutual capacitance and manufacturability, but future designs could experiment with a reduced substrate thickness.

6.6.4 Dependence on Material Dielectric Constants

The layer stack is not limited to the current materials and will be further optimized. Thus, as a preparation for this next step, the effect of alternative dielectric materials and adhesives on the mutual capacitance is evaluated in this subsection. For this

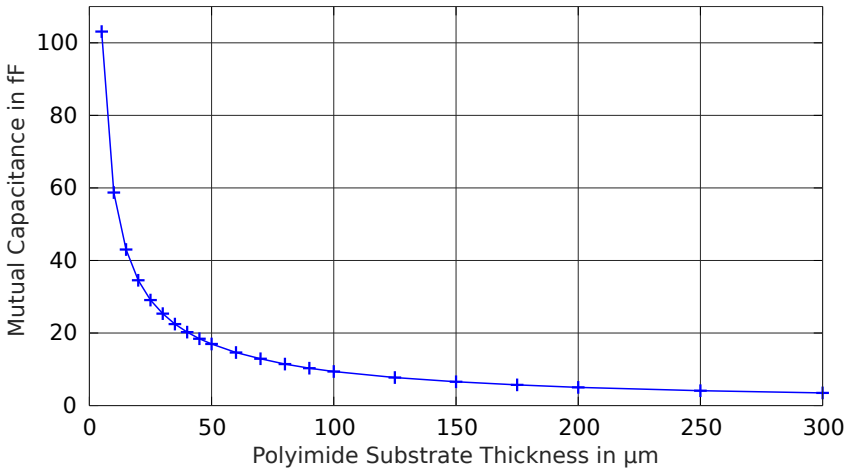


Figure 6.21: Dependence of the mutual capacitance on the polyimide substrate thickness.

purpose, the FEA varies the dielectric constant for the electrodes' substrate, the shields' substrate, and the adhesive. Each simulation is done independently for each material class while the other materials stay at their current properties. The dielectric constant is varied between 1.0 and 5.0 since most materials under consideration lie in this range.

The results are presented in Figure 6.22. The plot shows a reference line for the unaltered layer stack resulting in a mutual capacitance of 16.9 fF. The intersection of each curve with the reference line is at the original dielectric constant, which is additionally denoted.

The plot shows that the mutual capacitance grows approximately linearly with the electrodes' substrate dielectric constant. However, also the adhesive, influencing some of the fringing fields, has a small influence on the resulting capacitance. Surprisingly, the shield substrate has a negative influence on the mutual capacitance, as increasing its dielectric constant causes a slight decrease in capacitance. I attribute this effect to the shield substrate's ability to pull more field lines into its direction and away from the RX electrode, thereby decreasing the coupling. This is similar to reducing the distance to the shielding, which also led to an observable decrease in mutual capacitance.

All in all, these results show that the electrodes' substrate is the dominant factor for mutual capacitance, while the surrounding elements allow only fine-adjustments. This eases optimizations of the outer layers as changes in their dielectric constant do not strongly affect the capacitance.

6.6.5 Influence of Trace Distance

After investigating the influence of the material parameters, the trace layout will be further analyzed in this subsection. I employ the same model, presented previously in Figure 6.20, and adjust it to this experiment. All parameters are set to their default

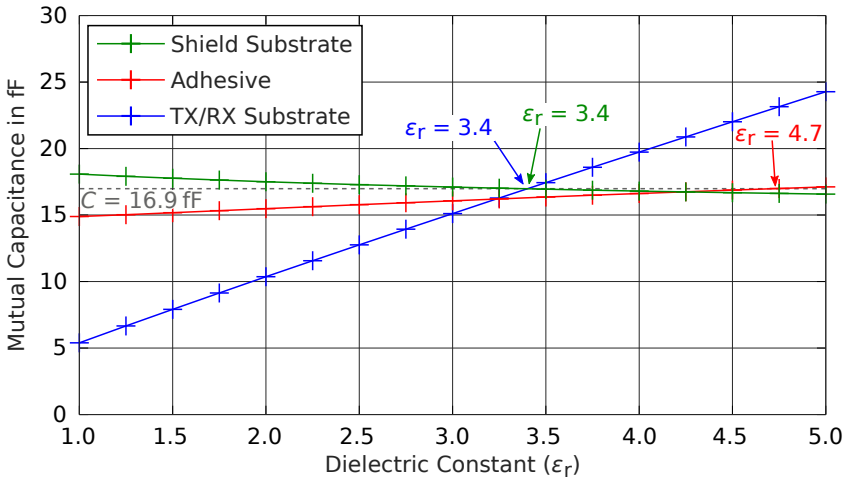


Figure 6.22: Varying the materials' dielectric constant between 1.0 and 5.0 allows a coarse and fine adjustment of the mutual capacitance.

value, representing the COPYCAT envelope, except for the distance between the traces D, also referred to as the electrode spacing. This parameter is swept between 2.5 µm and 350 µm. Please note that small, single figure spacings might be impossible to manufacture with the currently available technology; thus, they are only included for a thorough analysis of the structure on a conceptual level.

Figure 6.23 shows the mutual capacitance in dependence of the electrode spacing D. The results show a strong correlation between both values; a small electrode spacing strongly reduces the mutual capacitance down to around 6 fF at 2.5 µm. With increasing distance, the curve approaches approximately 22 fF. Thus, trace distance can vary the mutual capacitance by over a factor of three.

To understand this effect, plots of the electric field lines and potential were created for selected setups, i.e., $D = 50$ µm, 100 µm, 200 µm, and 300 µm. For the potential, a cut plane was created, indicated in Figure 6.20, with the view along the RX trace direction and the TX_1 electrode running horizontally.

The results are shown in Figure 6.24 to 6.27. The first plot shows a structure of very close traces with a spacing of only 50 µm. There are only very few fringing fields that can pass the gap between the traces. The traces form an approximately closed surface that keeps the electric potential well-contained. When the electrode spacing is increased to 100 µm in the next plot, the electrical field partially protrudes into the space above the RX traces. This increases the potential in this region and moves charge, thereby adding mutual capacitance. For the last two plots of $D = 200$ µm and 300 µm, the RX electrodes are even more surrounded by a high electric potential. In both cases, which yield roughly the same mutual capacitance, the high electric potential has reached the backside of the RX electrode layer and creates even more fringing fields. This increases the mutual capacitance further.

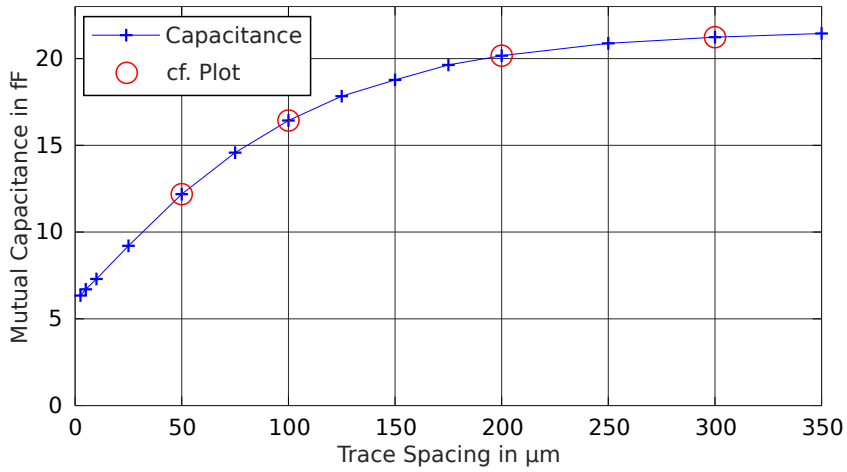


Figure 6.23: Dependence of the mutual capacitance on the spacing between traces.

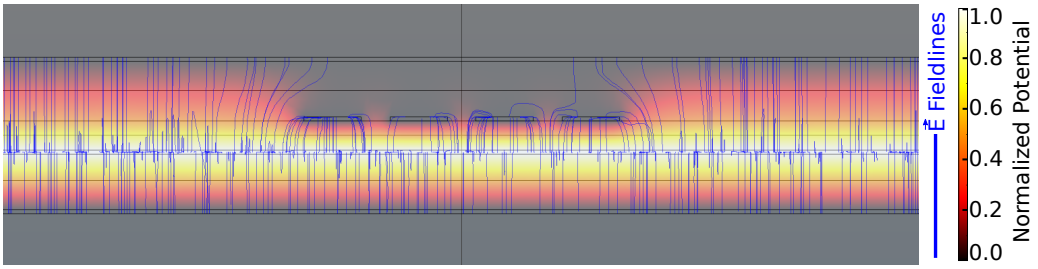


Figure 6.24: Electric fields and local potential for 50 µm electrode spacing.

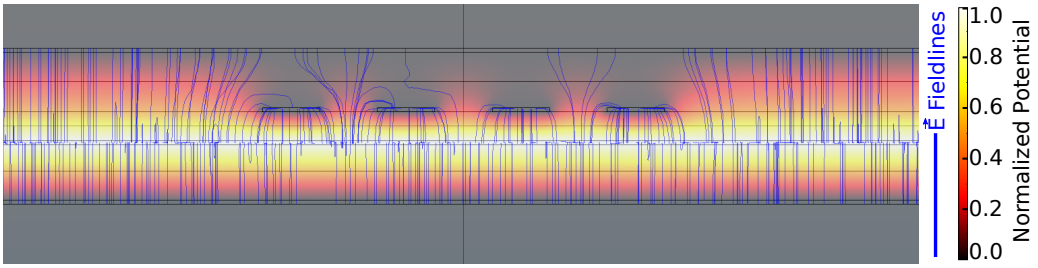


Figure 6.25: Electric fields and local potential for 100 µm electrode spacing.

The results show that the distance between the electrodes has a major influence on the mutual capacitance. Rough approximations without FEA seem to be impossible, only qualitative, but no quantitative estimations can be given. There is still a multitude of other parameters that could be varied, e.g., the thickness of the adhesive layer or

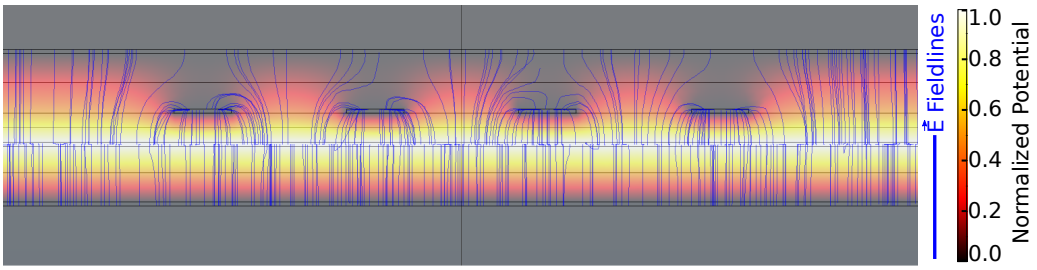


Figure 6.26: Electric fields and local potential for 200 µm electrode spacing.

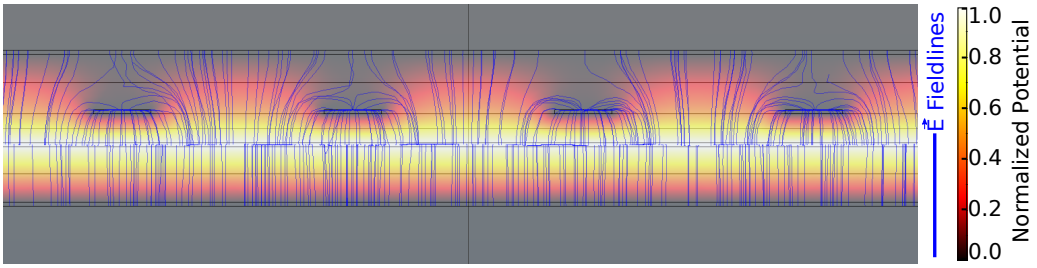


Figure 6.27: Electric fields and local potential for 300 µm electrode spacing.

trace width, and especially a combination thereof. Thus, this section gives a first look into possible optimizations and their effect on mutual capacitance; however, any optimizations regarding the electrode layout must be done with respect to a previously conducted FEA of the designated design. After I directed a majority of the effort towards creating the presented simulation model, running additional computations for other parameters is comparably simple and straightforward as this model can be reused.

6.6.6 Bias Due to Close Trace Routing of TX_1, TX_2, RX_1, and RX_{16}

When the enclosure electrode layout was initially created, the influence of a layout detail on the capacitive coupling was underestimated since there were no FEA results available at this time. The enclosure comprises not only the protected area with capacitive nodes but also requires traces to the connector. They are implemented on the same layers and are placed around the protected area. Since the manufacturing process technically limits the enclosure size, these traces are subjected to a very dense layout requirement to maximize the protected area. This leads to disadvantageous routing of a few traces, resulting in measurement result offsets.

I detected an issue during the statistical evaluation of the envelope prototypes whose results showed an unexpected positive bias for the differential capacitance $C_{1,2,1}$ and a negative bias for $C_{1,2,16}$. The location of the corresponding nodes hints at an

effect only present at the corners of the envelope near the connector's edge. When revising the layout near $C_{1,2,1}$ of an unshielded envelope under a microscope, the issue became apparent, as shown in Figure 6.28. Outside of the protected area, the TX_2 trace of the first TX pair was routed in close proximity to the trace of RX_1, thereby causing unintended crosstalk. The same applies to TX_1 and RX_{16}, which were routed in a similar layout near another corner of the envelope. The layout of the latter issue is not explicitly shown since it is symmetrical and was therefore not analyzed separately. Thus, the following analysis and results are valid for both arrangements but the sign of the bias is inverted for the latter one since TX_1 instead of TX_2 is the offending electrode.

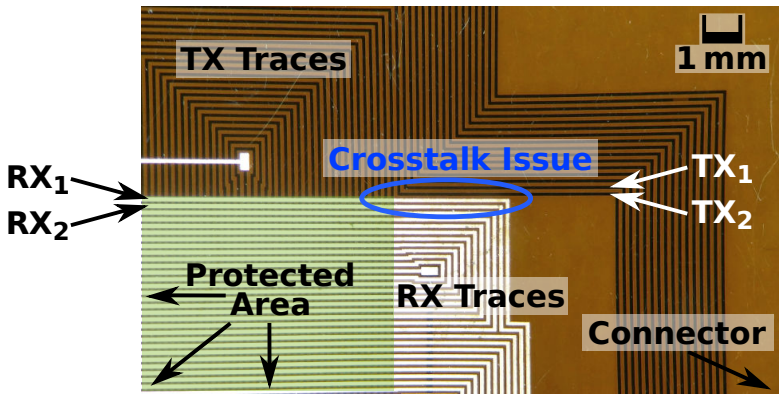


Figure 6.28: Image of the routing of TX_2 and RX_1 traces in close vicinity, causing a bias due to undesired capacitive coupling.

I verified my assumption for the offset via a FEA and derived a model based on Figure 6.28. The RX electrodes on the top layer appear in white, whereas the TX electrodes on the bottom appear in black. Both traces are on different layers, they have a width of $W = 100\,\mu m$, a spacing of $C = D = 100\,\mu m$, and run in parallel for about $P = 3.5\,mm$. The layer stack follows the definition in the previous sections; however, the RX and TX electrodes' positions have been interchanged to match the image in Figure 6.28. The resulting model is depicted in Figure 6.29, representing the two offending electrodes, as well as one neighboring electrode, on each layer.

To assess the amount of coupling, the maxwell capacitance matrix [Max81] is computed and the elements corresponding to the crosstalk are identified. The matrix expresses the amount of charge $\mathbf{Q}$ that is present on electrodes in dependence on the voltage $\mathbf{V}$ of the same or other electrodes, i.e.,

$$\mathbf{Q} = \mathbf{C} \cdot \mathbf{V} \tag{6.7}$$

whereas $\mathbf{Q}$ and $\mathbf{V}$ are vectors and $\mathbf{C}$ is the maxwell capacitance matrix. The capacitance matrix $\mathbf{C}$ has positive values for all diagonal elements, representing the self-capacitance, and negative values for all off-diagonal elements, representing mutual capacitances.

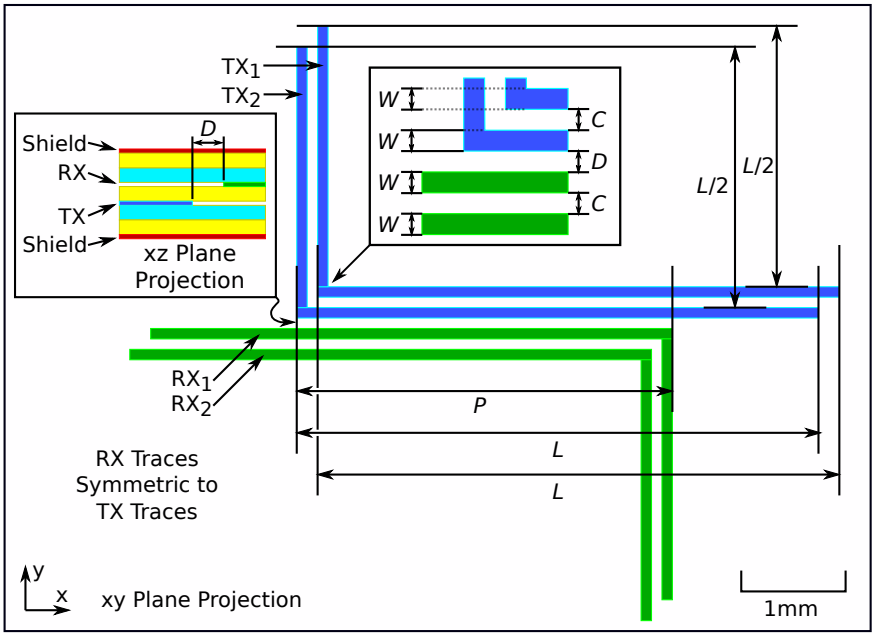


Figure 6.29: Simulation model for analyzing the unwanted coupling due to close routing.

The negative values are caused by conversation of charge, i.e., when positive charge is present on an excitation electrode, the same amount of negative charge must be present on all other electrodes together so that the sum remains zero.

In this case, the matrix has a dimension of 5×5, and the vectors $\mathbf{Q}$ and $\mathbf{V}$ are defined as

$$\mathbf{Q} = [Q_{\mathrm{RX1}} \quad Q_{\mathrm{RX2}} \quad Q_{\mathrm{TX1}} \quad Q_{\mathrm{TX2}} \quad Q_{\mathrm{Shield}}]^{\mathrm{T}} \tag{6.8}$$

and

$$\mathbf{V} = [V_{\mathrm{RX1}} \quad V_{\mathrm{RX2}} \quad V_{\mathrm{TX1}} \quad V_{\mathrm{TX2}} \quad V_{\mathrm{Shield}}]^{\mathrm{T}}. \tag{6.9}$$

These equations can be simplified as the measurement is only sensitive to a *change* in charge δQ_{RX1} of the observed electrode, thus, only the time-dependent elements in $\delta\mathbf{V}$ are non-zero, i.e., the TX electrodes, as the RX electrodes, and shield are at a constant voltage. This reduces $\delta\mathbf{V}$ to

$$\delta\mathbf{V} = [0 \quad 0 \quad \delta V_{\mathrm{TX1}} \quad \delta V_{\mathrm{TX2}} \quad 0]^{\mathrm{T}}. \tag{6.10}$$

Please note that that this requires $\mathbf{C}$ to be constant over voltage and excludes non-linear effects, however, this assumption has shown to be true throughout my measurements. Evaluating Equation 6.7 and 6.10 for δQ_{RX1} yields

$$\delta Q_{\mathrm{RX1}} = -\widehat{C}_{1,1} \cdot \delta V_{\mathrm{TX1}} - \widehat{C}_{2,1} \cdot \delta V_{\mathrm{TX2}}, \tag{6.11}$$

whereas $\widehat{C}_{1,1}$ and $\widehat{C}_{2,1}$ denotes the undesired mutual capacitance of TX_1 towards RX_1 and TX_2 towards RX_1, respecitively.

	RX_1	RX_2	TX_1	TX_2	Shield
RX_1	1041 fF	-129 fF	-1.09 fF	-54.4 fF	-857 fF
RX_2	-129 fF	1032 fF	-0.109 fF	-0.946 fF	-903 fF
TX_1	-1.09 fF	-0.109 fF	983 fF	-115 fF	-867 fF
TX_2	-54.4 fF	-0.946 fF	-115 fF	992 fF	-821 fF
Shield	-857 fF	-903 fF	-867 fF	-821 fF	3448 fF

Table 6.1: Maxwell capacitance matrix for Figure 6.29 with the TX_1, TX_2 to RX_1 coupling $\widehat{C}_{1,1}$ and $\widehat{C}_{2,1}$ due to routing proximity highlighted.

The FEA computes the entire maxwell capacitance matrix, which is given in Table 6.1 along with table headers for the reader's convenience. The parameters of interest are highlighted in blue with $-\widehat{C}_{1,1} = 1.09\,\text{fF}$ and $-\widehat{C}_{2,1} = -54.4\,\text{fF}$. Since the system is based on differential excitation, $\delta V_{\text{TX2}} = -\delta V_{\text{TX1}}$, hence, the charge on the RX electrode is

$$\delta Q_{\text{RX1}} = -\widehat{C}_{1,1} \cdot \delta V_{\text{TX1}} + \widehat{C}_{2,1} \cdot \delta V_{\text{TX1}} \tag{6.12}$$

which can be further simplified to

$$\delta Q_{\text{RX1}} = (\widehat{C}_{2,1} - \widehat{C}_{1,1}) \cdot \delta V_{\text{TX1}}. \tag{6.13}$$

Please note that both, TX_1 and TX_2 have influence on RX_1, however, the effect of TX_1 is opposed to TX_2 and they partially cancel each other out. By applying the definition of the differential capacitance on Equation 6.13, the formula becomes

$$\delta Q_{\text{RX1}} = \widehat{C}_{1,2,1} \cdot \delta V_{\text{TX1}}. \tag{6.14}$$

According to the FEA results,

$$\widehat{C}_{1,2,1} = \widehat{C}_{2,1} - \widehat{C}_{1,1} = 53.3\,\text{fF}. \tag{6.15}$$

Since the system is assumed to be linear, the superposition principle applies; thus, the measurement yields the sum of the differential capacitance $C_{1,2,1}$ and its differential capacitance bias $\widehat{C}_{1,2,1}$. The dynamic range of the measurement system lies in the same order of magnitude as $\widehat{C}_{1,2,1}$ and the error is therefore non-negligible.

To analyze the influence of the prominent electrode TX_2 further, I created a plot of the equipotential lines of the electrode arrangement with the cut plane being the RX electrode layer. In Figure 6.30, the TX_2 electrode is active while all other electrodes are grounded. The equipotential lines show that the electric field can freely extend into the free space in the upper left region, however, RX_1 repels the equipotential lines, and creates a dent, as indicated by two red arrows. Thus, coupling at this edge is the reason for the undesired coupling.

Increasing the space between the TX and RX electrodes would obviously counteract the issue. As space is scarce, knowing the minimum required distance to limit the

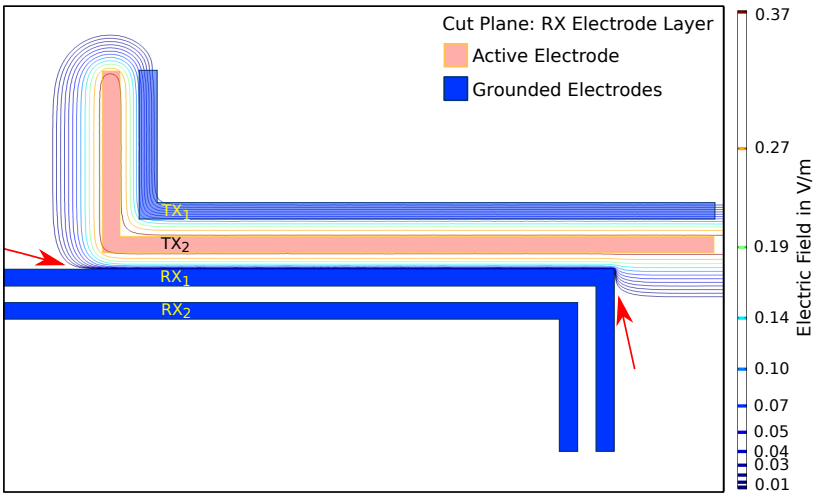


Figure 6.30: Equipotential line plot for TX_2, the RX_1 electrode strongly interferes with the electric field.

bias to negligible values below 0.3 fF, corresponding to the measurement system's accuracy, is essential. To optimize this parameter, I ran another simulation that varies the distance between $D = -0.1$ mm, moving the traces closer until they overlap, and $D = 1.1$ mm where the traces are far apart.

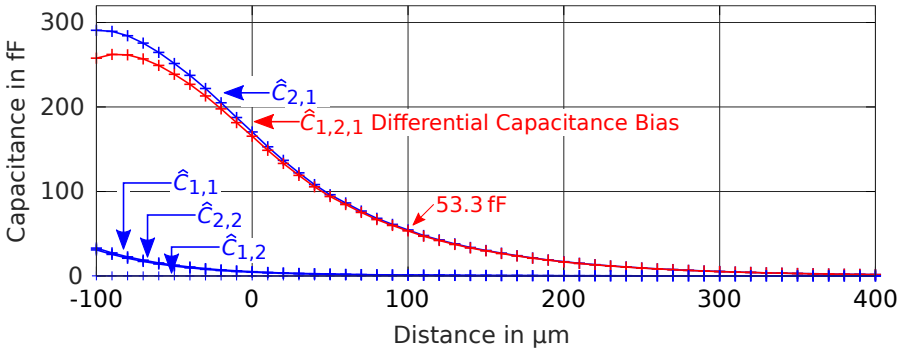


Figure 6.31: Dependence of the capacitance bias from the trace distance D.

The results are plotted in Figure 6.31. As expected, the bias increases when the simulation moves the traces closer, indicating the simulation model works correctly. From a distance of $D = 100\,\mu$m onward, the decrease is comparably small, even doubling the distance to 200 µm only decreases the bias to 16.6 fF.

A logarithmic plot of the results up to $D = 1.1\,\mu$m is depicted in Figure 6.32. The bias sufficiently decreases for higher distances and falls below the 0.3 fF threshold at

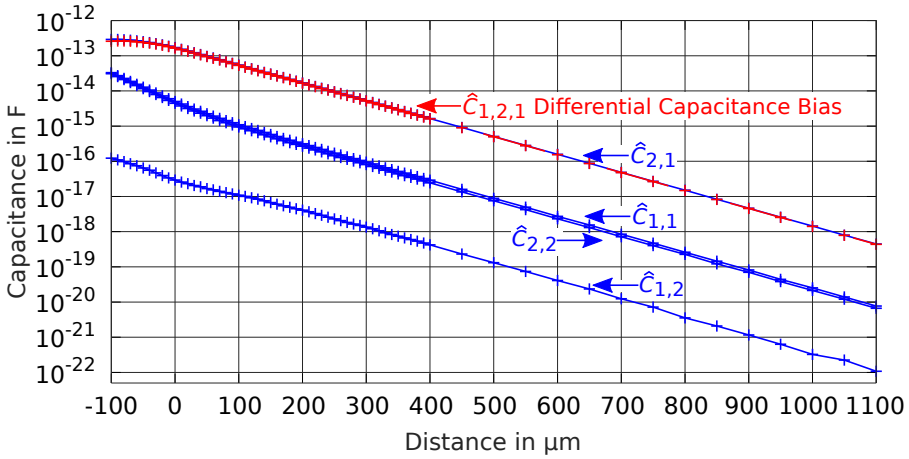


Figure 6.32: Log. plot of the capacitance bias in dependence from the trace distance D.

around $D = 600\,\mu\text{m}$. The determined distance lies well within reasonable ranges, as the envelope has several millimeters of space available at this location, as visible in Figure 6.28. Thus, this issue can be resolved with a small change without a major redesign. However, these results must be treated conservatively because numerical errors might apply for such small capacitances and additional biases due to other layout imperfections might exist that are not reflected by this simulation. Altogether, my results show that enclosure optimization by means of finite element analysis is feasible via the presented simulation model.

6.7 Conclusion

This chapter presented the concept of B-TREPID together with a finite element analysis of the enclosure design. At first, the basic concepts of envelope- and cover-based solutions were explained. Next, the chapter defined the nomenclature for the enclosure's capacitances and gave an overview of COPYCAT's trace layout, including its layer stack. In addition to this, a FEA was presented, which allows an estimation of the enclosure's capacitances and the influence of material parameters such as substrate thickness, dielectric constant, and electrode trace layout. Thereby, the FEA model is the foundation for future work. The effect of layout optimizations can be computed beforehand, which reduces the effort and costs of manufacturing multiple enclosure samples to find the best combination of parameters. Furthermore, as the enclosure may have to be scaled to a larger size in the future, the consequent requirements on the remaining components, such as the measurement system, can be estimated earlier. Altogether, the FEA model provides valuable information regarding the inner working of the enclosure and enables the estimation of properties prior to manufacturing.

In the next step, the enclosure needs to be optimized with respect to the FEA results. However, several aspects, among others, need to be taken into account when the enclosure design is altered: First, manufacturing has several limitations on the materials, for example, the lower bound for the substrate thickness is set by its mechanical robustness. Secondly, the layout must follow security and manufacturing design rules regarding trace width, distance, and the dimension of vias. Thirdly, the measurement system must still be able to handle parasitic capacitances and trace resistances while being capable of extracting a sufficient amount of variation in capacitance. Some of these optimization goals are opposed to each other, e.g., a thinner substrate could increase the measurable amount of variation but would reduce the robustness, which could, in turn, decrease the yield during manufacturing. Thus, each parameter should only be adjusted upon consultation with all involved parties to prevent degradation of the overall system. The results presented in this chapter are only one aspect of the entire system and should thereby not be the sole basis for decisions but a valuable guide, among others.

Chapter 7

A Measurement System for Capacitive PUF-Based Security Enclosures

The previous chapter gave an introduction to the concept of B-TREPID. One of its key components is the measurement system that evaluates the PUF and verifies the enclosure's integrity. However, measuring differential capacitances in the range of several dozens of femtofarads has proven to be a very challenging task — especially since the same circuit has to perform integrity verification as well.

A great effort was spent on developing an adequate measurement system and several researchers at the Fraunhofer Institute AISEC contributed to it. Initially, Vincent Immler and Maxim Hennig experimented with commercial off-the-shelf touchpad controllers that are capable of capacitance measurement. Afterward, a custom measurement circuit was developed; however, the circuit's performance was insufficient and did not feature integrity verification. Starting from these results and utilizing the experience therefrom, I took over this task and developed the current measurement system that measures the capacitances in the femtofarad range and can verify the enclosure's integrity. All topics around this system, from the analog circuitry and its digital signal processing and firmware architecture, was the primary focus of my research.

I presented my measurement system at the *55th Design Automation Conference (DAC)* in 2018 and the corresponding paper is available in the conference proceedings [OIHS18]. Additionally, my measurement concept was filed as EU patent, has been published under `EP3550475`, and is awaiting a decision as of 2019 [OIH19]. This chapter is partially based on the paper and patent application but has been extended with the most recent developments.

This work was done in cooperation with Vincent Immler, who contributed to the development of the measurement concept and to the paper. Furthermore, Robert Hesselbarth supported me in optimizing the measurement system's analog performance. Matthias Hiller provided help in writing the paper.

7.1 Overview

The previous chapter proposed the architecture of B-TREPID, a *capacitive* PUF-based security enclosure [HSZS13, IOK+18a] which is implemented as two-layer mesh of electrodes. The electrodes serve a dual-purpose, they are not only checked to verify the enclosure's integrity, but they are also evaluated as a PUF. However, PUF property extraction is challenging since the capacitance variation lies within the range of only several femtofarads, while parasitic capacitances of hundreds of picofarads or even more are present. Furthermore, any influence of the measurement system on the PUF's output must be minimized to obtain an enclosure PUF but not a circuit PUF. For this purpose, I developed a specialized measurement system that extracts the PUF property precisely and also verifies the mesh's integrity — everything combined within a single system.

To obtain a secure system, both measurements are interlocked. The PUF prevents simple jumper attacks, as additional wiring will have a non-negligible impact on the PUF property. In return, the integrity verification mechanism prevents an attacker from splitting off regions of the capacitive PUF.

7.1.1 Related Work

Despite there are concepts for integrity verification and capacitive PUF sensing in previous work, there is no combination thereof. The resistance-based GORE-envelope [IMJFC13] monitoring system is one example of an integrity-only solution as it provides no PUF readout. Its integrity verification is based on a Wheatstone bridge, which renders a direct integration into a *capacitive* measurement system impossible, as both methods would interfere with each other. The fringe-effect proximity sensor is another tamper-resistant enclosure [ES05], whose evaluation system was analyzed. Despite employing capacitive sensing for detecting intruding objects, it cannot extract capacitive PUF properties in a quantitative manner.

Several more solutions exist which perform capacitive sensing to measure PUF responses. E.g., the Coating-PUF protects a single chip via capacitive sensors integrated in the coated silicon chip [TSŠ+06]. Though this system can extract PUF properties on-chip, it is not compatible with a large-scale enclosure and also does not support integrity verification. Similar issues are observed for the BoardPUF [WSL+15], which is a capacitive PUF implemented in a Printed Circuit Board (PCB) but without means for integrity verification.

7.1.2 Contributions

Since none of these measurement concepts fulfills my requirements, I developed a specialized but scalable measurement system to verify the mesh's integrity and to extract capacitive PUF properties with the same setup. My contributions include:

- A differential measurement concept for capacitive PUF-based security enclosures whose parasitic capacitances are orders of magnitude larger than the PUF's variation.
- A scalable measurement concept enabling enclosure integrity verification and capacitive PUF measurement in the range of milliseconds.
- A proof of concept implementation of the proposed system and a practical verification utilizing enclosure prototypes.
- A practical evaluation of an electrode probing attack on the security enclosure and a discussion of countermeasures.

7.1.3 Structure

The chapter starts with an introduction to the measurement topic in Section 7.1. Section 7.2 continues with an exposition of the measurement system's high-level functionality. Next, the measurement concept explanation follows. Section 7.3 presents the integrity verification concept and Section 7.4 expounds the absolute and differential measurement. Section 7.5 describes the analog circuit and digital signal processing chain design in detail. Following in Section 7.6, I present an optimization of the prototype implementation that cuts down the measurement duration by 40 %. Section 7.7 proves that the proposed system can be implemented and provides the specified functionality. An attack, specially tailored to capacitive PUF-based security enclosures, is presented and evaluated in Section 7.8. Concerning future developments, Section 7.9 presents the basis provided by my work. Finally, Section 7.10 summarizes the presented results and gives an outlook to the next steps in system development.

7.2 High Level System Overview

Before going into the technical details, a summary of the system's specified behavior and its requirements is given.

7.2.1 System Startup, Operation, and Reaction on Tampering

System startup, operation, and its reaction to tamper events are depicted in Figure 7.1. Initially, the measurement and host system are both powered down and no secret data is available in any unencrypted or otherwise unprotected form. When the system is powered on, it starts in the unverified state; thus, no PUF secrets are available and system integrity has not been verified yet. During this phase, the measurement system begins to output a heartbeat signal to the host system, which indicates that the device is operating normally and starting up. Next, the self-verification procedure starts with the first integrity verification of the enclosure and circuit, named *Tamper Detection A*. Among the verification of electrode integrity, this includes several self-checks of the analog circuitry. Upon failure due to manipulations, an interrupted trace or short circuits, an alarm is raised and the heartbeat signal is interrupted, the system stops

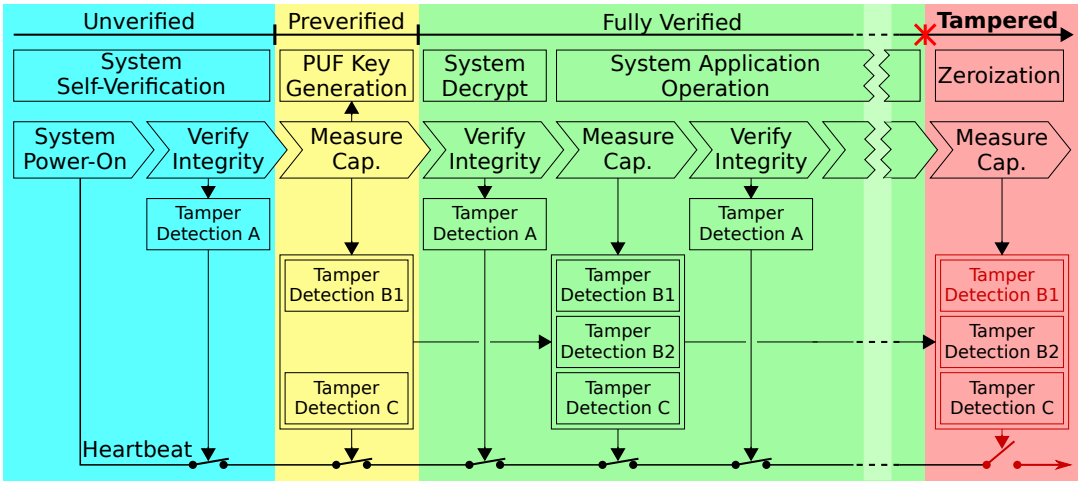


Figure 7.1: System startup, operation, and reaction on tampering.

operation, and switches into the secure state. The same is true for the host system, which monitors the heartbeat signal and will take adequate measures upon an alarm to protect itself and its data. Performing an initial integrity verification reduces the risk of leaking sensitive data to an adversary during the subsequent PUF evaluation if the enclosure is not intact due to tampering.

If the integrity verification was passed successfully, the system proceeds into the preverified state. Therein, the first absolute and differential capacitance measurements are triggered, whereas the latter one extracts the PUF. The result of the measurement is processed by the *Tamper Detection B1* and *Tamper Detection C* modules. Tamper Detection B1 verifies that each differential measurement result lies within a predefined range, i.e., ΔC is correctly read out and the analog circuitry did not clip the value to the positive or negative limit. Tamper Detection C performs a similar operation on the absolute measurement result C_m of each node and checks that it is close to the expected value originating from the enclosure design. Please note that the mutual capacitance is not used as a PUF and its measurement is too coarse to extract variation; thus, storing a blurred reference value to check against does not impact system security. If at least one tamper detection mechanism is triggered, an alarm is raised and the heartbeat signal ceases.

If these steps were successful, the system proceeds into the fully verified state and starts to derive the PUF key from the differential capacitance data [Hil16, Imm19]. After deriving the PUF key from the differential measurement results and deriving the PUF key, the system and optionally CSPs are decrypted. From this point on, the system continues to perform integrity verification and capacitance measurement in an interleaved manner. The tamper detection steps are executed as before; however, *Tamper Detection B2* is added for the capacitance measurement. It computes the

change of the differential measurement data over time and raises an alarm if the rate of change exceeds a predefined limit.

As long as the system is untampered, it remains in the fully verified and operational state. Upon a tampering event, one or more tamper detection mechanisms will be triggered, which raise an alarm, zeroize all CSPs and other critical data, stop the heartbeat, and bring the system into a secure state.

Tamper alarm or a *soft* shutdown, depending on the application profile, can also be triggered by other actions, such as shutting down the system's power supply. Additional tamper sensors, e.g., for light or temperature, can easily be integrated into the system due to its modular concept. Altogether, this illustrates the basic configuration of the system, which has to be adjusted to the application and required level of protection.

7.2.2 System Requirements

The aforementioned concept creates the following requirements on the measurement system. The system must perform differential measurements of capacitive nodes and verify the enclosure's integrity, whereas this functionality is combined in *one* system. Although no time limit is explicitly stated in the standards PCI-HSM [PCI09] and FIPS-140-2 [NIS01a], a time frame of below one second is typical for the startup, measurement, and tamper response [VNK+15]. Thus, I target a time frame of less than 100 ms. To be able to exploit the variation in the femtofarad range, the capacitive measurement must be carried out with a precision of at least 1 fF. With respect to cooling and system integration, the measurement system's power dissipation should not exceed 1 W on average during operation. Furthermore, it should be able to run from commonly available voltages, e.g., 5 V or 3.3 V. Deployment in a security product implicitly requires the measurement concept not to be side-channel prone, i.e., the measurement must avoid leakage and be time-constant. In addition, the chosen concept must be scalable to adapt the enclosure to different form factors of embedded systems under protection. A secondary goal is that the concept supports performance-optimized and area-optimized implementations as this establishes compatibility to more high-security applications.

7.2.3 System Development Process

Three revisions of measurement systems exist, two in the COVER project and one in the COPYCAT project. Due to their different enclosure technologies, the PCBs differ as well; however, there is only a single underlying firmware code repository. This approach has proven to save a significant amount of work, as most improvements to the system are immediately available for *every* board revision. Furthermore, there is only a single codebase that has to be maintained, preventing back-porting issues of improvements and reduces testing and documentation efforts as well.

In order to do so, I set up a development system that separates the revision-specific parts from the general measurement system code. An option allows the developer to

switch between the configuration for each individual board or revision upon compile-time. Configuration files are available that enable the developer to switch specific functionality off and on and provide revision-specific configurations. This concept had especially proven to be advantageous when the code was edited by several other researchers, including working students and other PhD students, as everyone was able to work and contribute to the latest version independently from the underlying hardware revision.

7.3 Measurement Concept: Integrity

Integrity is initially verified during device assembly. Later in the field, the same integrity verification mechanism detects tamper events.

7.3.1 Integrity Violation Types

Short and open circuits are the two basic types of integrity violations. They may either come from manufacturing defects or are later caused by an adversary during a tamper attempt. Since the enclosure contains a multitude of traces, several integrity violations can occur at once and in various combinations, however, they must all be detected by the measurement system.

This is expressed via a simple model of two traces, whereas the first trace is being tested while the second one represents all other electrodes. Each electrode can have zero, one, or two interruptions in its trace, e.g., one at the beginning and one at its end. Additionally, there might exist a short circuit between both traces. Altogether, this sums up to 20 cases ranging from zero up to five faults.

Table 7.1 lists all combinations of faults up to a number of five. Each case is evaluated regarding three aspects: Security, Safety, and Detectability. Only an enclosure with zero faults can be considered *secure*; any fault will disable the PUF partially, create weak spots, and render the system insecure thereby. Despite not relevant for the final product but of utmost importance during development, no faults in the enclosure are allowed to cause any permanent damage to the measurement system. Hence, to ensure a *safe* measurement, short circuits between electrode inputs must always be detected, otherwise, the differential excitation signal source would be shorted during the measurement causing excessive currents. This could destroy the enclosure's traces, might damage the analog circuitry, or might even damage both. Since the enclosure represents an early technology sample, obtaining information about defects is helpful to identify manufacturing issues; thus, the circuit should be able to *detect* defects precisely. Finally, the system returns a list of *valid* traces that, as seen by the integrity verification mechanism, appear to be intact. These measurement goals are of descending priority, with system security on top, measurement safety following closely, and precise detection of error being of lower importance.

Faults	Electrode Status	Secure	Safe	Detect	Valid
0		Y	Y	Y	$\{X_1, X_2\}$
1		N	Y	Y	$\{X_2\}$
1		N	Y	Y	$\{X_1\}$
1		N	N	Y	$\{\}$
2		N	Y	Y	$\{\}$
2		N	Y	Y	$\{\}$
2		N	Y	Y	$\{\}$
2		N	N	Y	$\{\}$
2		N	N	Y	$\{\}$
3		N	Y	P	$\{X_2\}$
3		N	Y	P	$\{X_1\}$
3		N	N	P	$\{\}$
3		N	Y	P	$\{\}$
3		N	Y	Y	$\{\}$
3		N	Y	Y	$\{\}$
4		N	Y	P	$\{\}$
4		N	Y	P	$\{\}$
4		N	Y	P	$\{\}$
4		N	Y	P	$\{\}$
5		N	Y	P	$\{\}$
Y = Yes	**N** = No	**P** = Partially			

Table 7.1: Electrode faults and their effect on enclosure security and measurement.

As depicted in Table 7.1, integrity is unambiguously detectable for zero faults. The same is true for one or two faults, which are detected precisely. This especially includes the dangerous cases where two inputs, X_1 and X_2 are shorted, e.g., for the last two cases with two faults. For three faults and above, the situation is different. Despite the security violation is detectable and safe measurement can be ensured, the errors cannot be precisely identified anymore. In the first two cases, which are comparably unlikely, the short circuits on X_1 and X_2 are not visible to the outside; however, they imply a double interruption on the *neighboring* trace which will reliably cause integrity verification failure. For four errors and above, measurement is always safe and the integrity violation is detectable; however, the traces are in bad condition and not all details are measurable from the outside.

Altogether, the measurement system always detects when a fault is present and safe measurement is also guaranteed under every condition. However, not all faults can be precisely identified; only the information regarding the existence of at least one fault is always available. Although having one or two faults on a trace has been observed, faults of higher count are rather unlikely. Thus, for the most likely faults, the measurement system is able to provide the best information.

7.3.2 Integrity Verification Method

For the purpose of integrity verification, I combined this functionality with the capacitance measurement in just one system. This feature is implemented as follows.

7.3.2.1 TX Electrodes

For TX electrodes, the measurement system verifies their integrity by applying a voltage to each electrode. The system begins with pushing a *single* TX electrode, e.g., TX_1, to a constant voltage. Then, a comparator verifies that the rise in voltage is also present on the output of the electrode, e.g., TX_{1R}. In the case of an open circuit, the voltage does not reach the comparator and no signal is detected. Next, the comparator checks that the signal is not present on any other TX_R outputs; otherwise, a short exists. The system performs these steps for each N^2 TX-to-TX_R combination until the entire TX layer has been verified.

7.3.2.2 RX Electrodes

Integrity verification for RX electrodes cannot be performed the same way due to the capacitance measurement circuitry, which is also present at the RX electrodes. Simply applying a constant voltage would overdrive the analog circuit because it is based on a *current* sensitive design. Thus, instead of applying a voltage, the integrity verification circuit injects a current into the RX electrode via a nanoampere current source that is connected to the RX_R end of the RX electrode. The current flowing from RX_R to RX, for an intact electrode, is amplified at the RX side and then fed to a comparator. If the current passes the electrode successfully, the comparator senses a

signal; otherwise, an open circuit is detected. The comparator must not sense a signal at any other RX electrode; otherwise, a short circuit is detected. The system performs this for each RX_R-to-RX combination, resulting in M^2 steps.

7.4 Measurement Concept: Capacitance

In contrast to the integrity verification, which returns a simple binary value for each electrode, the capacitance measurement yields a floating-point value. This measurement is more demanding due to the quasi-continuous range of values that must be reproduced at each measurement as precisely as possible. As a design guideline, I obtained the following characteristics of common enclosures: In general, I measured typical values between 0.2 nF to 2 nF for C_s, depending on the size and manufacturing technology of the enclosure. C_m is typically three orders of magnitude smaller and ranges from 10 pF to 50 pF. The PUF variation ΔC is again three orders of magnitude smaller with a $1\,\sigma$ interval in the one or two figure femtofarad range.

For the COPYCAT envelopes I obtained a parasitic capacitance of 200 pF. The mutual capacitance C_m between TX and RX electrodes is 18 pF, corresponding to a capacitance per overlap of $18\,\text{pF}/990 \approx 18\,\text{fF}$. This matches the finite element analysis result in Subsection 6.6.3, which yielded approximately 16.9 fF per overlap. The envelope exhibits a capacitance variation with a $1\,\sigma$ interval of $[-6.3\,\text{fF}; +6.3\,\text{fF}]$.

For the COVER project cover, I measured larger values due to an increased enclosure size and different technology [ION+18]. Please note that these values apply solely for the top cover, as this was the primarily investigated part. Its parasitic capacitance C_s is almost 2 nF while $C_m = 50\,\text{pF}$ and the resulting capacitance per overlap is $50\,\text{pF}/1800 \approx 27\,\text{fF}$. For these enclosures, the PUF variation has a $1\,\sigma$ interval of $[-30\,\text{fF}; +30\,\text{fF}]$.

The values for the mutual and parasitic capacitances were obtained in the lab via a measurement setup based on a *EXACT Model 119 Function Generator* for excitation signal generation and a *Fluke 289* multimeter for voltage and current measurement. To minimize local influences and to increase the otherwise minuscule currents, the electrodes were connected in parallel during measurement.

7.4.1 Previous Measurement Approaches

Throughout the project, different approaches were followed for capacitance measurement; however, they were unsuccessful. This subsection gives a short summary of failed approaches.

V1.0 was tried out by other project members who employed commercial off-the-shelf capacitive touchpad controllers. They could extract measurement values from a mesh-like structure; however, they are not suited since there is no integrity verification or *differential* measurement mode. Furthermore, the tested devices are targeted towards detecting changes in capacitance caused by user interaction but will not deliver a precise capacitance measurement result since this is not required in the application.

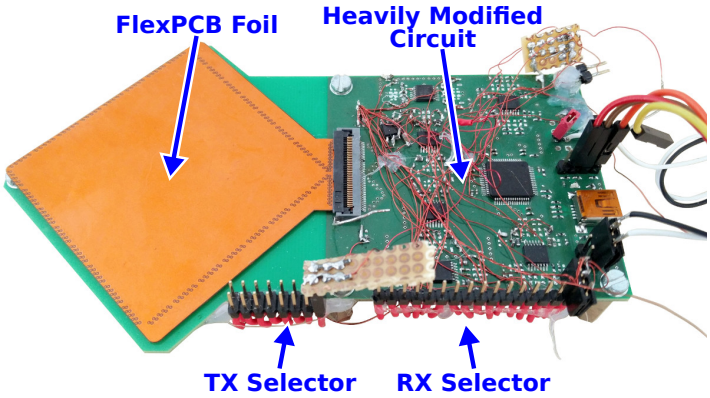


Figure 7.2: V3.0, the first functional differential measurement approach, reusing the V2.0 device by heavily modifying it.

V2.0 was one approach at the beginning of the COPYCAT project. It was a full-custom PCB with a discrete component-based analog circuit and an STM32 microcontroller. The circuit was targeted towards differential capacitance measurement only without taking integrity verification into account, yet. The employed method, named *TX pulse excitation, RX charge subtraction* is explained in the next subsection; however, the device was finally not functional due to several reasons.

V3.0 scavenged the board and components of V2.0, which I decided to reuse for experimentation on an entirely new differential measurement approach. Although the new circuit design was still not able to perform integrity verification due to limitations of the enclosure prototype that missed TX_R and RX_R outputs, at least the functionality of my new differential measurement approach could be successfully proven. Since the device, shown in its final form in Figure 7.2 and requiring manual electrode selection using jumpers, was rather chaotic and unsurprisingly showed much noise and sensitivity to its environment, the creation of an entirely new circuit was scheduled.

V4.0 and above implemented my new measurement approach that supports integrity verification and differential capacitance measurements. Throughout both projects, the system was steadily improved. V4.0 denotes the initial COPYCAT design, V4.1 and V4.2 are the revisions developed for the COVER project. Regarding the method of measurement, all three revisions are identical, with only minor optimizations in their components.

7.4.2 Measurement Technique Selection

I examined several concepts for capacitive sensing [Bax97] in search for an adequate method for V3.0 and V4.0; however, the following ones were ruled out.

1. **Constant current charging:** This method is common, for example, in multimeters. A constant current is applied to the capacitor and the rise in voltage is

observed at the same terminal. The method is unsuited for C_m and ΔC measurement because the large parasitic shielding capacitance C_s is present. Thus, the measurement result will primarily show C_s instead of the desired value.

2. **RC or LC oscillator:** The capacitor in an RC or LC oscillator sets its frequency. By measuring the oscillation period and with knowledge of R or L, the capacitance can be derived. Despite the method was employed for the Coating PUF [TSŠ+06], it is not suited for B-TREPID. Compared to the constant current charging method, it suffers from a similar issue as it will primarily measure the parasitic capacitance. Additionally, this concept does not scale well as it can measure only a single TX-RX pair at a time and is not compatible with meshes. Furthermore, capacitance-dependent oscillations can leak information to the outside [MSSS11a], which contradicts the low-leakage requirement.

3. **TX pulse excitation, RX charge subtraction:** A pulse on a single TX electrode pushes charge onto all RX electrodes. The transferred charge, which is directly proportional to the absolute capacitance, is converted to a voltage and subtracted from another absolute capacitance measurement to acquire the differential capacitance ΔC. Despite this method being scalable and constant-time, it showed to be infeasible. V2.0 demonstrated the drawbacks of the approach. The minuscule ΔC becomes submerged in the system's unavoidable gain-mismatches, biases, and offsets as the system subtracts two almost equal and large mutual capacitances to obtain their small difference. Additional circuit-related issues such as unsuited operating points and missing component-matching in the circuitry before subtraction worsened the circuit's performance and could not be resolved sufficiently.

Since these common straight-forward solutions are infeasible, I developed the **DFT-based in-enclosure differential capacitance measurement concept** for my measurement circuit implementation. The basic idea is as follows: I move the measurement from the time domain into the frequency domain. Additionally, I shift the capacitance subtraction into the enclosure instead of struggling with subtraction afterward in an analog or digital system. Wherever possible, I replace analog circuitry with digital signal processing to counteract external influences, component imperfections, and aging.

During the design phase of the system, I developed circuit simulation models, running in LTspice [Ana18], for the analog section of the measurement system. This enabled me to verify the correct functionality of the circuit before ordering PCBs and components; thus, errors could be detected early, thereby reducing costs and development time. Creating a valid model has shown to be difficult for such a system as parasitic capacitances and component imperfections must not be neglected. If done incorrectly, the simulation model will not correspond to the real system, which could, cf. V2.0 and its RX charge subtraction method, lead to a non-functional system. However, the analog and digital sections of V4.0 can be entirely simulated and its results match the real-world performance.

7.4.3 Differential Capacitance Measurement Method

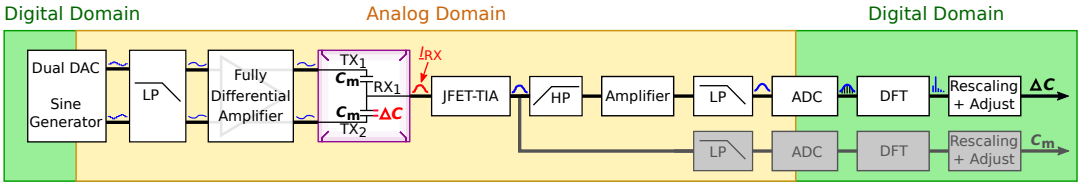


Figure 7.3: Block diagram of the differential capacitance measurement.

My approach is based on in-enclosure current subtraction to measure the differential capacitance by mutual cancellation of currents. The enclosure in Figure 7.3 is exemplarily reduced to two TX electrodes and a single RX electrode. The balanced electrode layout ensures that two neighboring TX electrodes have the same mutual capacitance C_m towards the RX electrode. This allows isolating the manufacturing variation ΔC which unavoidably differs for each TX and RX combination.

To excite two neighboring TX electrodes, the dual channel Digital-to-Analog Converter (DAC) and a subsequent fully-differential amplifier create a sinusoidal signal of amplitude V_TX and frequency f_TX, with $\omega = 2\pi \cdot f_\mathrm{TX}$. A phase shift of exactly 180° is applied to the signal at the even-numbered electrode, thereby inverting the signal. This generates two currents proportional to the mutual capacitances, flowing from two TX electrodes to one RX electrode. Also both currents have a 180° phase shift relative to each other, i.e., they are of inverse signs. They merge on the RX electrode resulting in the complex RX electrode current

$$\underline{I}_\mathrm{RX} = j\omega V_\mathrm{TX} \cdot C_\mathrm{m} + j\omega(-V_\mathrm{TX}) \cdot (C_\mathrm{m} + \Delta C) = -j\omega V_\mathrm{TX} \cdot \Delta C. \tag{7.1}$$

If both mutual capacitances were exactly matched, i.e., $\Delta C = 0$, the capacitances and thereby the currents would completely cancel each other out. However, due to manufacturing variations, i.e., $\Delta C \neq 0$, a tiny residual current remains. The resulting complex current $\underline{I}_\mathrm{RX}$ is directly proportional to the variation in capacitance ΔC as shown by Equation 7.1. Please note that the equation is independent from C_m and C_s, thus, neither the mutual capacitance nor the shielding capacitance influences $\underline{I}_\mathrm{RX}$ or the measured ΔC.

The RX current commonly ranges from several hundred picoamperes up to few nanoamperes. A Junction gate Field-Effect Transistor (JFET)-based Transimpedance Amplifier (TIA), which is a current-to-voltage converter, linearly translates the current $\underline{I}_\mathrm{RX}$ into a voltage. Subsequently, the voltage is processed by High-Pass (HP) and Low-Pass (LP) filtering and amplification stages to remove any offsets prior to further amplification. Finally, the RX signal is digitized by an Analog-to-Digital Converter (ADC).

After digitizing the signal, digital signal processing algorithms reconstruct ΔC by determining the RX signal's amplitude and phase. Instead of solving this issue

in the time domain, I transform the signal into the frequency domain via a digital dual-phase lock-in amplifier approach. This is implemented by applying a Discrete Fourier Transform (DFT) on the digitized RX time domain signal and achieving a complex result. This filter splits the signal into its frequency components, thereby separating most of the noise from the signal. The magnitude of the DFT bin at $f_{\mathrm{RX}} = f_{\mathrm{TX}}$ is proportional to $|\Delta C|$. Since this returns only the *absolute* value $|\Delta C|$, the sign is recovered from the phase information. The excitation signal is locked to the RX signal acquisition; thus, a negative ΔC reliably has a phase shift of 180° relative to a positive ΔC. Hence, two decision regions were defined, representing a negative and positive capacitance difference. This method requires a strict synchronization of the signal generation with the ADC data acquisition in order to preserve the phase information. As the final step, an algorithm rescales the data relative to the arbitrarily chosen full-scale value of $\pm 1 \cdot 10^4$. This completes the measurement system's data handling, as the raw PUF data is available in a signed integer format.

7.4.4 Absolute Capacitance Measurement Method

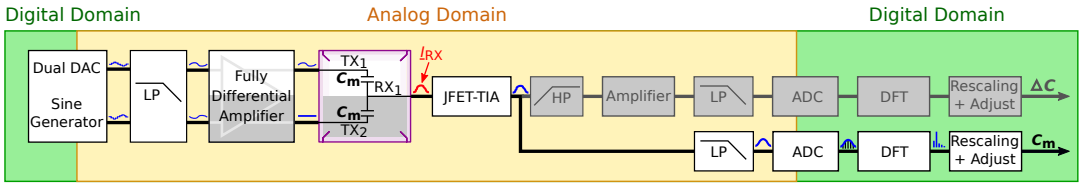


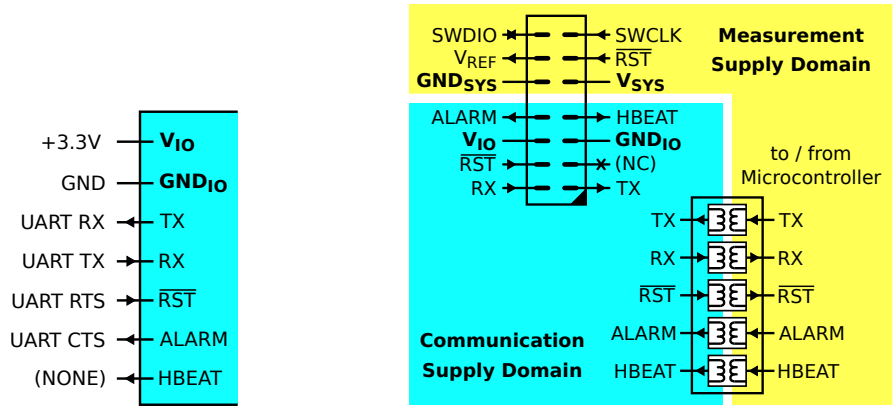
Figure 7.4: Block diagram of the absolute capacitance measurement.

Although not comprised by the initial system specification, the ability to perform absolute capacitance measurements yielding C_{m} was soon added to the system since it comes with almost no overhead in the analog circuitry. The corresponding signal path in Figure 7.4 is highlighted and the existing circuit is partially reused. In comparison with the differential measurement, the system applies the excitation signal to only one TX electrode by disabling one output. Furthermore, the excitation amplitude is attenuated by G_{att} to account for the larger measured capacitance that would otherwise overdrive the analog circuitry. At the RX electrode, this results in the current

$$\underline{I}_{\mathrm{RX}} \approx j\omega \cdot (V_{\mathrm{TX}} \cdot G_{\mathrm{att}}) \cdot C_{\mathrm{m}}. \tag{7.2}$$

Please note that the variation ΔC is negligible for this measurement since it is about three orders of magnitude smaller than the mutual capacitance.

The current $\underline{I}_{\mathrm{RX}}$ is amplified by the same JFET-TIA as before; however, it is directed into a different signal chain. The high-pass filter and the second amplification stage are skipped as the signal's quality and amplitude is sufficient for digitization. In the digital signal processing chain, it runs through the same algorithms. Computing the magnitude and without the phase would be sufficient to reconstruct C_{m}, as its sign



(a) Recommended connection betwen UART interface and measurement system.

(b) Galvanic isolation between communication interface and measurement system.

Figure 7.5: The communication interface of the measurement system.

is known to be positive. However, the sign is still computed as a sanity check of the enclosure and to make this helpful information available during development.

7.5 Measurement System Design

I designed the system to run on a single supply V_{DD} of 3.3 V, as this voltage is available in most systems. The decision was against a dual supply as negative voltages are comparably tricky to create; thus, the analog circuit is arranged around a center voltage of $V_{\mathrm{DD}}/2 = 1.65\,\mathrm{V}$.

The measurement system is controlled by an STM32F303 microcontroller, which has mixed-signal capabilities. It features an ARM Cortex-M4F core, including a Floating-Point Unit (FPU), which is useful for signal processing. Additionally, the microcontroller provides all required peripherals, such as Digital-to-Analog Converters (DACs), ADCs, DMA, and zero-waitstate Core-Coupled Memory (CCMRAM) for routine speedup. The system is well suited for the measurement, as DAC signal generation and ADC signal acquisition can be configured to run in hardware, freeing the processor from load.

7.5.1 System Communication Interfaces

A well-designed system communication interface is the cornerstone of a reliable system that also eases automatic measurements and testing. Thus, I implemented a UART-based interface, whose recommended connection to a UART adapter is shown in Figure 7.5a. It runs on UART TX and UART RX; thus, almost any UART adapter is compatible. Furthermore, the interface has three additional but optional signals, i.e., an RST input, the ALARM, and the Heartbeat (HBEAT) output signal. The first

one allows resetting the device, e.g., via the UART RTS signal. The ALARM signal displays the tamper status and is accessible via UART CTS, for example. Altogether, this interface provides sufficient means for remote-controlled and automated testing.

The communication interface is supplied by the UART adapter's own power supply but not by the measurement circuit itself. I chose this implementation since adapters have different UART output levels, e.g., 3.3 V or 5 V, hence, being compatible to both voltage levels and leaving the decision to the user was the optimum in this case. Additionally, the ground of the communication system is separated from the measurement system, as depicted in Figure 7.5b, to prevent ground loops and disturbances due to noise. Thus, the measurement system operates independently from the communication interface and signals are transported via a galvanically isolated coupler of the Analog Devices ADUM1400 series. This allows to power the communication interface separately from the measurement system, e.g., its supply input V_{SYS} is powered via a low-noise power supply while V_{IO} of the communication circuitry is connected to a usual laptop with a switching power supply. The high level of separation also contributes to the system's robustness, no mandatory power sequencing is required during startup and the power sources on both sides can be turned on and off independently at any time without risking damage.

7.5.2 The Analog Circuit

For the measurement, the system applies an excitation signal of 1 V RMS with the frequency $f_{\mathrm{TX}} = 33.\overline{3}\,\mathrm{kHz}$. The frequency has proven to be an adequate trade-off between several optimization goals. Nine periods of the excitation signal are generated, the first period is for analog circuit startup and eight periods are for measurement, resulting in 270 µs per differential capacitive PUF node. The signal acquisition operates with 12 bits resolution at $f_{\mathrm{sample}} = 5.14\,\mathrm{MS/s}$ to oversample the signal for noise reduction. This results in a total number of

$$\frac{f_{\mathrm{sample}}}{f_{\mathrm{TX}}} \cdot N_{\mathrm{periods}} = \frac{5.14 \cdot 10^6 \,\frac{1}{\mathrm{s}}}{33.\overline{3}\,\mathrm{kHz}} \cdot 9 = 1389 \tag{7.3}$$

acquired samples for each measured differential node. The first 155 samples, corresponding to one period, are discarded, the remaining 1234 samples are further processed.

7.5.2.1 TX Excitation

Both excitation signals must be matched in amplitude and show a relative phase of 180°. An amplitude imbalance or phase error would otherwise cause an offset in the extracted PUF data, as the enclosure-internal current cancellation would no longer work. A signal imbalance is amplified by C_{m}; thus, ΔC can easily become submerged as it is only a fraction of C_{m}. As a rule of thumb, the TX signal's imbalance should be at least an order of magnitude smaller than $\Delta C / C_{\mathrm{m}} \approx 1/1000 = -60\,\mathrm{dB}$.

The system comprises three stages for signal generation. First, two amplitude-matched and inversely phased $33.\overline{3}$ kHz signals are generated by the microcontroller's DAC. A second-order low-pass filter follows in the signal processing chain, which removes the 2 MHz DAC sampling frequency artifacts. As the DAC does not provide sufficient amplitude-matching itself, the signal is fed to a fully-differential amplifier. It enhances the signal quality by guaranteeing a 180° phase shift and precise amplitude matching. The THS4551, which is employed in the circuit prototype, features an output balance of 85 dB [Tex17], providing sufficient signal quality for the measurement.

7.5.2.2 RX Current Amplification

The RX current in the nanoampere range requires an amplifier circuit with high gain and low noise. Furthermore, the RX node has a high capacitive load, caused by the large shielding capacitance C_s, which is present at the amplifier's input and reduces its bandwidth [Ram09]. Additionally, the amplifier's input bias current must not exceed a certain level, otherwise, the TIA will be overdriven, rendering RX current measurement impossible. The system requires one RX amplifier per RX electrode; thus, the developed amplifier circuitry must be cost-efficient since common enclosures are expected to have around 16 RX electrodes or even more. A design showing low noise, high Gain Bandwidth Product (GBW), and low bias current is required, which was not achievable at once using only a single operational amplifier.

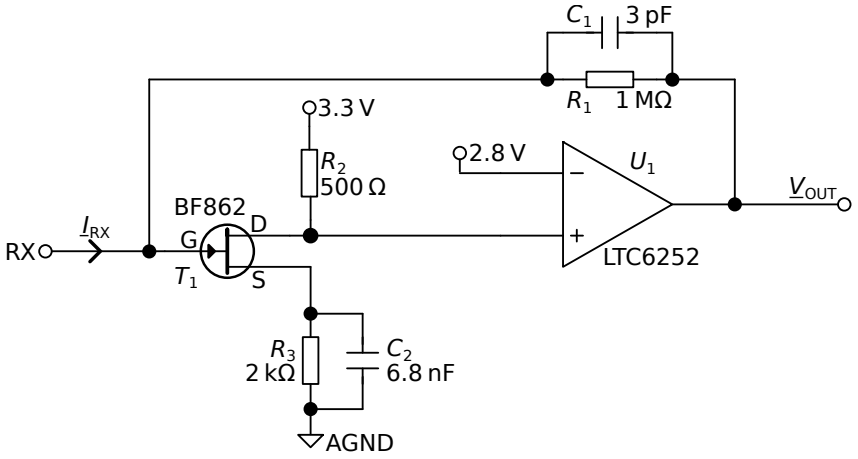


Figure 7.6: JFET-TIA for RX current to voltage conversion and amplification.

To overcome this problem, I developed a custom TIA implementation. I adapted the basic idea of a JFET-based TIA setup, which is common in optics to amplify a tiny photocurrent at a node of high capacitance, e.g., of a large-area photodiode [Cor12]. However, the circuit requires some adjustments for this application, e.g., in its operating point, as the input current will be positive and negative. The adjusted circuit is depicted in Figure 7.6. The circuit combines a high GBW operational amplifier (U_1) with a

low input bias JFET (T_1) as pre-amplifier and input stage to create an improved TIA. The resistors R_2 and R_3 and the voltage of 2.8 V at the inverting input set the DC operating point of the JFET and operational amplifier circuit. In this configuration, the current trough the JFET is $(3.3\,\mathrm{V} - 2.8\,\mathrm{V})/500\,\Omega = 1\,\mathrm{mA}$. Thereby, $\underline{V}_{\mathrm{OUT}}$ and the RX electrode voltage are set to a DC steady-state voltage of approximately 1.65 V which is the center voltage.

The circuit implements a control loop that maintains 1.65 V at the RX electrode by varying the amplifier's output voltage $\underline{V}_{\mathrm{OUT}}$. The RX electrode current $\underline{I}_{\mathrm{RX}}$ effectuates a slight increase in RX voltage, causing the JFET to open and carry more current via its drain-source channel. This decreases the voltage at the JFET's drain and thereby also at the non-inverting input of the operational amplifier. Next, this leads to a decrease in output voltage $\underline{V}_{\mathrm{OUT}}$ that pulls current through the feedback network $R_1 \parallel C_1$. This reduces the voltage at the RX electrode to its original level. The equilibrium is reached when the current through the feedback network is equal to the RX electrode's current since the JFET's gate current is negligible at the operating point. The impedance of the feedback network $R_1 \parallel C_1$ controls the relation between $\underline{I}_{\mathrm{RX}}$ and $\underline{V}_{\mathrm{OUT}}$, i.e., it sets the TIA's gain. With the presented configuration, the TIA's gain magnitude is $0.8 \cdot 10^6\,\mathrm{V/A}$ at an excitation frequency of $33.\overline{3}\,\mathrm{kHz}$ and $1.0 \cdot 10^6\,\mathrm{V/A}$ at DC.

After the JFET-TIA has converted the RX current into a voltage, a high-pass filter strips the signal's DC component. To further amplify the signal, it is fed to an inverting amplifier, resulting in a gain of $-100 \mathrel{\widehat{=}} 40\,\mathrm{dB}$ at DC and $-88 \mathrel{\widehat{=}} 38.8\,\mathrm{dB}$ at $33.\overline{3}\,\mathrm{kHz}$ which scales the signal to the ADC's dynamic range. The amplifier is implemented as an active first-order low-pass filter to remove high-frequency noise components and to prevent aliasing during digitization. The filter's low output impedance allows the signal to be directly connected to the ADC without a buffer since the ADC requires a source of low output impedance [STM16b].

The circuit is comparably robust against component imperfections and variations. JFETs are known to have a wide spread in their characteristics [NXP00], even inside the same batch of components. Thus, the operating point and thereby the output DC level varies between the TIAs. I assume that this might also be a PUF in the analog circuitry; however, I did not investigate this further.

Although a driven guard trace is sometimes recommended when measuring minuscule signals [MvRPG90, Jun05], my design does not require one. Guard traces are driven shields that surround the trace of the signal under protection in order to prevent external influences and especially leakage current across the PCB. However, any leakage current will only slightly shift the TIA's operating point, but this influence is well below the unavoidable JFET variation. Additionally, the effect is most visible for the signal's DC component, which will be removed in the signal processing chain; thus, small leakage currents are of no concern. Omitting the guard trace not only saves space for the corresponding trace, but this also removes the need for a shield driver that would add another operational amplifier for each RX electrode.

7.5.2.3 TX_R Integrity

The TX electrode integrity verification sequentially pulls each electrode to a higher voltage and checks whether this increase is seen at TX_R. No additional circuitry is added on the TX side since the DAC and TX amplifier are already able to perform this action. On the TX_R side, an LT1719 comparator senses whether the voltage exceeds the threshold of 1.87 V.

7.5.2.4 RX_R Integrity

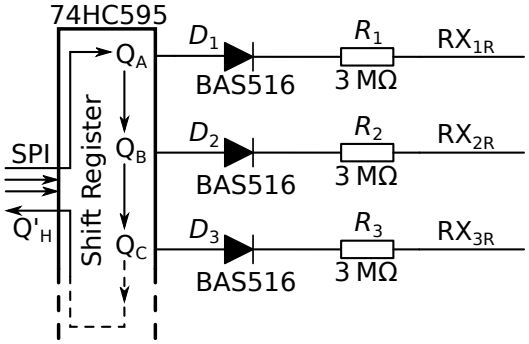


Figure 7.7: RX_R integrity verification current sources.

The RX electrode integrity verification injects a tiny current into RX_R and measures the resulting signal at RX. Since the integrity verification tests for open and short circuits only, the requirements on the accuracy of the current are relaxed.

To implement the coarse current source, each RX_R input is connected in series to a 3 MΩ resistor, a fast diode, and a digital output, as shown in Figure 7.7. The digital outputs are provided by a 74HC595 shift register, which is controlled via SPI. For RX integrity verification, a single digital output is set to 1 (3.3 V) while all others remain at 0 (0 V), similar to a one-hot encoding. Since the RX electrode has a fixed voltage of $V_{DD}/2 = 1.65\,V$ and the diode's forward voltage is about $U_D = 0.4\,V$ this results in the current

$$I_{RXR} = (V_{DD} - V_{DD}/2 - U_D)/R_1 = (3.3\,V - 1.65\,V - 0.4\,V)/3\,M\Omega \approx 0.4\,\mu A \quad (7.4)$$

flowing into the RX electrode. All other outputs which are set to 0 do not carry current since the diodes at the RX_R electrode are reverse-biased. If the electrode is intact, the RX TIA senses the current and converts it to a voltage of

$$V_{DD}/2 - 1\,M\Omega \cdot 0.4\,\mu A = V_{DD}/2 - 0.4\,V = 1.25\,V. \quad (7.5)$$

The subsequent comparator, set to a threshold of 1.45 V, senses the voltage falling below and reports an intact electrode.

In the case that the electrode has an open circuit, no current will be sensed by the RX amplifier and no difference in output voltage will be generated. The 1.45 V threshold of the comparator will not be reached and thus, the electrode will be reported as being defective. In the case of a short circuit between two RX electrodes, the behavior is more complicated since the behavior is theoretically undefined. However, practical tests, as well as theoretical considerations, have shown that this will cause the TIAs of both affected RX electrodes to go into saturation. One TIA output will be saturated to the negative rail, i.e., near 0 V, while the other one will be in saturation at the positive rail, i.e., near 3.3 V. This happens due to the reason that both TIAs try to push the voltage of the shorted RX electrode to *their own* operating points which slightly differ, creating an unstable but not oscillating control system. Such a situation is handled by the integrity verification since neither the circuit's self-check at startup nor the following integrity verification of the enclosure will succeed.

The circuit can trivially be extended for an arbitrary number of RX electrodes if required. The current sources are replicated until the desired number of outputs has been reached. The 74HC595 shift registers feature a Q'_H output that allows a concatenation of a theoretically unlimited number of shift registers. For example, my circuit requires two shift register devices resulting in 16 outputs, i.e., the number of RX electrodes. The shift registers and thereby the current sources are controlled via SPI by the microcontroller.

7.5.3 Digital Signal Processing

Digital signal processing algorithms extract the amplitude and phase from the RX signal via a dual-phase lock-in amplifier. The ADC sampling is synchronized to the DAC's output and the DAC signal serves as the lock-in reference signal. The synchronization is achieved by interconnecting microcontroller peripherals such as timers, DMA channels, ADC, and DAC, providing a fixed phase relation for signal generation and acquisition.

Next, an adequate windowing function is applied to the data. Since the frequency of interest is known, I employ a rectangular window covering eight full periods of the signal. By choosing an integer number of periods for the window length, most information is preserved and spectral leakage of this frequency is not observed.

Computing a dual-phase lock-in amplifier in software is computationally expensive, as it requires two multiplications, two additions and two sine-table lookups per sampling point. The same result can be obtained by applying a DFT on the digitized signal and evaluating the bin, corresponding to the excitation frequency. The Goertzel Algorithm is a performance-optimized way to compute a *single bin* DFT [Goe58]. It requires only a single multiplication, addition, and subtraction per sample, but no table lookups. This is especially an advantage if memory access would be expensive in terms of performance, e.g., if the lookup table resides in flash memory and requires several cycles per access.

The algorithm runs on the FPU, which delivers single-cycle multiplication, addition, and subtraction, thereby surpassing the dynamic range of integer and fixed-point implementations without a decline in performance. All intermediate values are kept

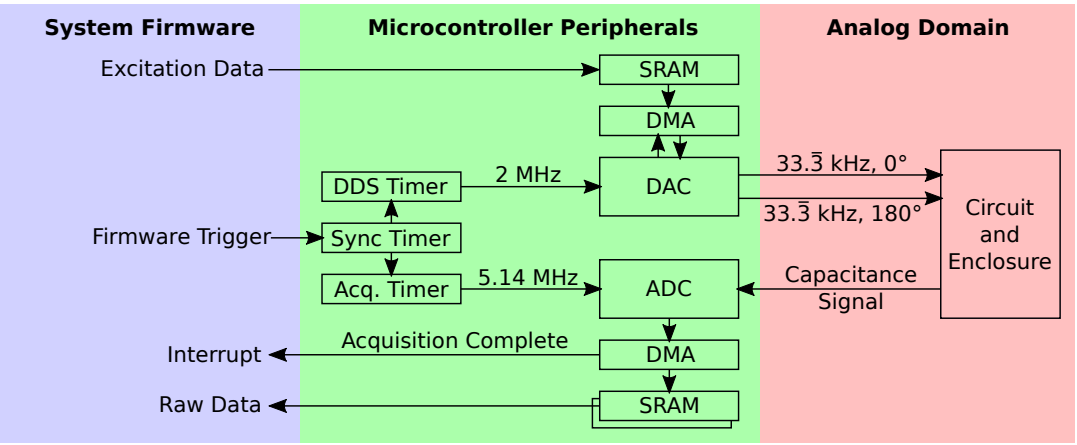


Figure 7.8: The microcontroller's peripherals are configured to autonomously perform the entire analog part of the measurement from signal generation until acquisition.

in FPU registers continuously, thereby speeding up computation. Performance is further increased by moving the algorithm from flash memory into CCMRAM since this memory is directly coupled to the microcontroller core and provides zero-waitstate memory. This reduces the computation time for one capacitance node by 25 % down to 258 µs without any drawbacks.

7.5.4 Measurement Synchronization

The microcontroller was configured such that a major part of the measurement runs in hardware, thereby freeing the Central Processing Unit (CPU) from load, as shown in Figure 7.8. Before a measurement is started, the microcontroller loads the excitation data, i.e., the lookup tables for the DAC's sine output signals, into the corresponding SRAM memory range. Usually, this step is only performed once at startup as the data remains there indefinitely as long as the system is powered on.

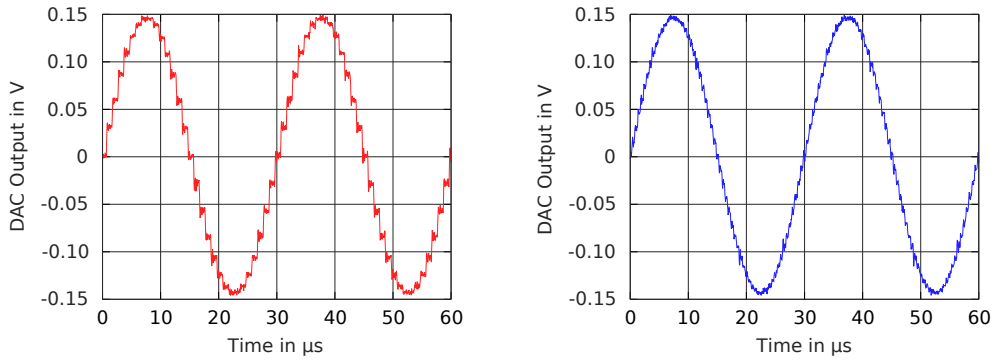
Initially, all timers and modules are idle; thus, when the firmware starts the measurement, it has to trigger the synchronization timer by setting a specific bit. Please note that this is the last interaction between the CPU and the measurement hardware until the raw measurement data is ready. The synchronization timer takes over control of the measurement as it is interconnected in hardware to the Direct Digital Synthesis (DDS) and acquisition timer. Both will be started exactly at the same moment as soon as they receive the synchronization timer's trigger signal. The DDS timer's output signal has a frequency of 2 MHz, which is the DAC's sample rate. The acquisition timer's output signal has a frequency of 5.14 MHz, corresponding to the ADC's maximum sample rate.

The output signals of both timers are the triggers for the DAC and ADC, respectively. When the DAC is being triggered, it outputs the next sample at its analog outputs,

representing the current value of the generated sine wave. At the same time, the DAC fetches the next sample from the SRAM via the DMA interface. This dual-phase sine signal leaves the microcontroller's DAC, is passed into the analog circuitry, and reaches the enclosure. When it returns to the microcontroller's ADC, it is still sine-shaped but contains information about the differential capacitance. This signal is being digitized by the ADC, each time the acquisition timer sends its trigger signal. As soon as the conversion has been finished, the value is automatically written into one of two SRAM acquisition buffers via another DMA channel. When the specified SRAM region has been entirely filled, i.e., the measurement has finished, the DMA triggers an interrupt. Thereby, the CPU is notified about the completed measurement cycle and it can read out the data.

Since two acquisition buffers are available for the ADC, the data of one buffer may be processed while the other one is still being acquired. This is possible as no interaction between the measurement hardware and the CPU is required except for starting the measurement and being notified about its completion. As explained in Section 7.6, the outsourcing of the measurement into the hardware and the double buffering mechanism provides a performance improvement of 40 % for an optimized implementation.

7.5.5 DAC Sample Rate



(a) AC-coupled DAC output at 1 MS/s. **(b)** AC-coupled DAC output at 2 MS/s.

Figure 7.9: Comparison between two DAC sample rates and their effect on signal quality.

The microcontroller's DAC generates the TX excitation signal. The DAC has two output channels that create the 0° and 180° signals with a resolution of 12 bits. The DAC features a common 32-bit configuration register that allows setting both outputs at once.

The frequency of the bus at the DAC is 36 MHz and the excitation signal is $33.\overline{3}$ kHz; thus, these numbers are divisible without remainder. This allows the system to load the output values from a simple lookup table instead of requiring computational expensive operations during DDS. The lookup table contains 32-bit entries that are directly

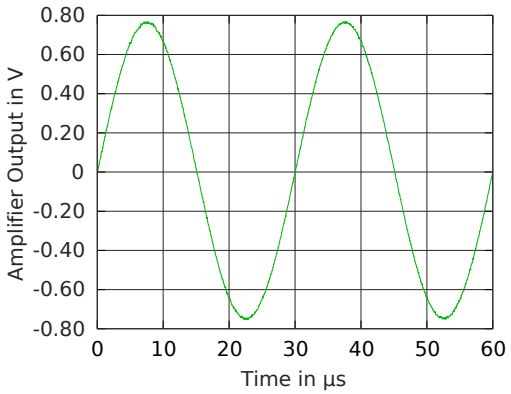


Figure 7.10: AC-coupled excitation signal, generated at 2 MS/s, after low-pass filtering and amplification.

loaded into the common DAC register, thereby updating both outputs synchronously and with a single transaction on the internal bus.

According to the microcontroller's datasheet [STM16b], the DAC's maximum sampling frequency is 1 MS/s. For $33.\overline{3}$ kHz excitation signals, this results in 30 samples per period. Hence, there are clearly visible and comparably large steps in the output signals, of which one channel is plotted in Figure 7.9a. Usually, these sampling rate artifacts are filtered out via a low-pass filter; however, the excitation and the sampling frequency are still rather close to each other, which would require higher-order filters and more effort.

However, an application note [STM15] explicitly allows DAC sampling rates of up to 4.5 MS/s for the STM32F3 microcontroller series, thereby contradicting the datasheet. Careful testing proved that increasing the sampling rate is technically possible and does not come with any drawback for my application. However, 2 MS/s instead of the maximum was chosen since the application note advises the developer to leave a safety margin for other transactions on the bus. I expected to improve the signal's quality significantly and to shift the sampling artifacts to higher frequencies.

The resulting signals confirm the expectations; the same output channel is depicted in Figure 7.9b. The step size is strongly reduced and the deviation from a perfect sine became smaller.

Please note that each output signal has a peak-to-peak amplitude of only 0.3 V, which is later increased by the fully-differential amplifier. Experiments have shown that this seems to improve the output signals' quality. One channel of the final signal, after two-stage low-pass filtering and amplification, is shown in Figure 7.10. The resulting signal does not have visible steps anymore and resembles a high-quality sine signal.

To verify my claims regarding signal quality, I computed the magnitude of the spectrum for each time-domain signal. As expected, the spectrum of the 1 MS/s signal in Figure 7.11 displays peaks at $33.\overline{3}$ kHz, at the sampling frequency, and its multiples,

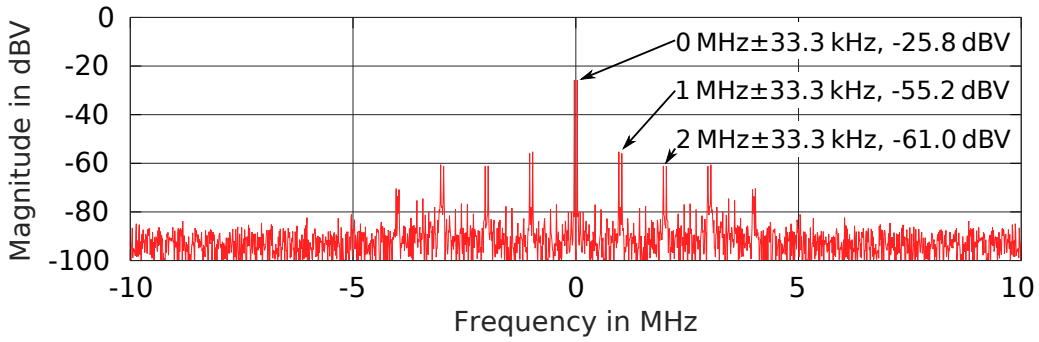


Figure 7.11: Spectrum of the 1 MS/s excitation signal of Figure 7.9a.

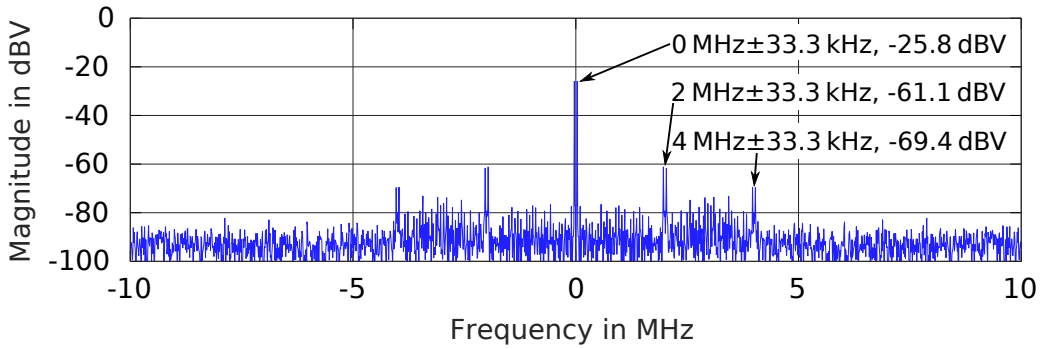


Figure 7.12: Spectrum of the 2 MS/s excitation signal of Figure 7.9b.

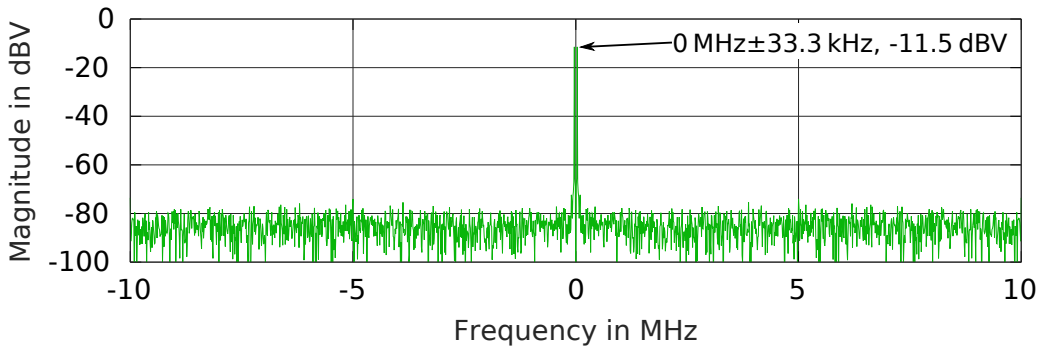


Figure 7.13: Spectrum of the filtered and amplified 2 MS/s excitation signal of Figure 7.10.

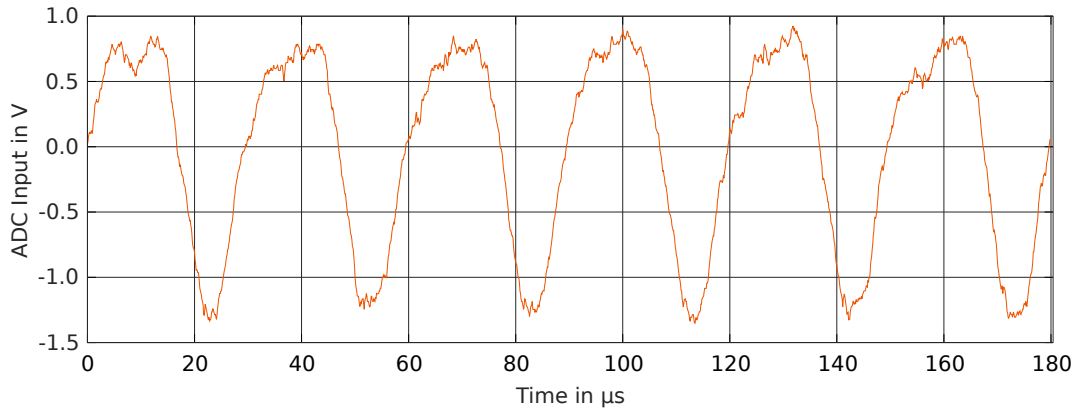


Figure 7.14: Converted and amplified RX electrode AC current at the ADC's input.

i.e., around 1, MHz, 2 MHz, 3 MHz and above. Please note that the sampling frequency peaks are double peaks, which are shifted upwards and downwards by $33.\overline{3}$ kHz.

In contrast to it, the spectrum of the 2 MS/s signal in Figure 7.12 has less peaks and the first sampling artefact is found at around 2 MHz. The remaining peaks are at approximately the same height; however, half of them are not present anymore due to the increased sampling rate. This reduces the noise power significantly and increases the Signal-to-Noise Ratio (SNR). Thus, increasing the sampling rate was an adequate choice for increasing the signal's quality.

When the spectrum of the resulting excitation signal is analyzed, Figure 7.13 shows no relevant peaks above the noise floor; thus, all sampling artifacts were successfully removed by the low-pass filters. Altogether, the excitation signal is of sufficient quality and does not need to be optimized currently.

To verifiy the acceptable signal quality, the resulting capacitance difference signal at the ADC was analyzed in a similar manner. For an exemplary node, the signal is plotted in Figure 7.14. The signal does not exhibit any frequency components originating from DAC sampling artifacts. It shows a sine wave of $33.\overline{3}$ kHz frequency but also some noise; however, this is unsurprising as the minuscule RX current signal has been strongly amplified.

In order to analyze the noise component further, I computed an FFT of the signal, shown in Figure 7.15. As expected, no sampling artifacts are present; however, several new noise peaks have emerged and the noise shows an overall $1/f$ shape. They are present at multiples of the excitation frequency, e.g., at $66.\overline{6}$ kHz and above. While the magnitude of the peak at $33.\overline{3}$ kHz depends on the differential capacitance, the amplitude of the aforementioned noise components seems to be independent of it. I suspect that these noise components emerge from non-linear behavior of components such as the amplifiers or power supplies. However, since the amplitude is comparably small and the peak is very narrow, such a signal can be filtered out, as I will show in the next subsection. Nevertheless, one remaining drawback is the reduction of the

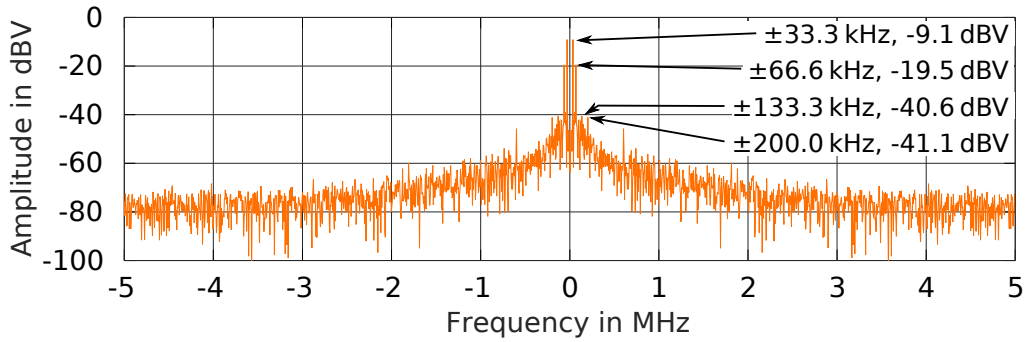


Figure 7.15: Spectrum of the converted and amplified RX electrode current whose time domain signal is shown in Figure 7.14.

dynamic range by a few percents, which are occupied by the noise signal. To counteract this issue, one could add additional analog filters after the TIA stage that reduces the noise's amplitude. Nonetheless, this was not found to be necessary for a conceptual evaluation of the system since I classify this rather as an optimization.

7.5.6 Analog and Digital Filter Performance

The RX electrode current is processed by analog and digital filters. To analyze their correct implementation and functionality, I externally applied an input signal to the RX electrode connector. The signal was generated by an *EXACT Model 119 Function Generator* and the frequency was swept in steps of 1 kHz from 1 kHz to 75 kHz while maintaining constant amplitude. At the same time, the system performs measurements to acquire the signal.

The resulting amplitude of the signal at the ADC's input is shown in Figure 7.16 together with a reference plot. Both curves have been normalized independently from each other such that they have a gain of exactly 1.0 at $33.\overline{3}$ kHz to map the signal generator's amplitude to a digitized value. The reference curve takes into account the analog circuitry, including its frequency-dependent gain, and the digital signal processing algorithms. The resulting plot clearly shows that the measurement matches the reference curve very well; thus, the system was correctly implemented. Although there is some minor deviation at low amplitudes, presumably due to signal generator imperfections, the minima are all at the correct frequencies. Please note that the filter gain at $66.\overline{6}$ kHz equals zero; thus, the undesired signal components described in Subsection 7.5.5 are entirely filtered out. The same is true also for other multiples of $33.\overline{3}$ kHz; thus, the entire filter chain is well-adjusted to the measurement characteristics.

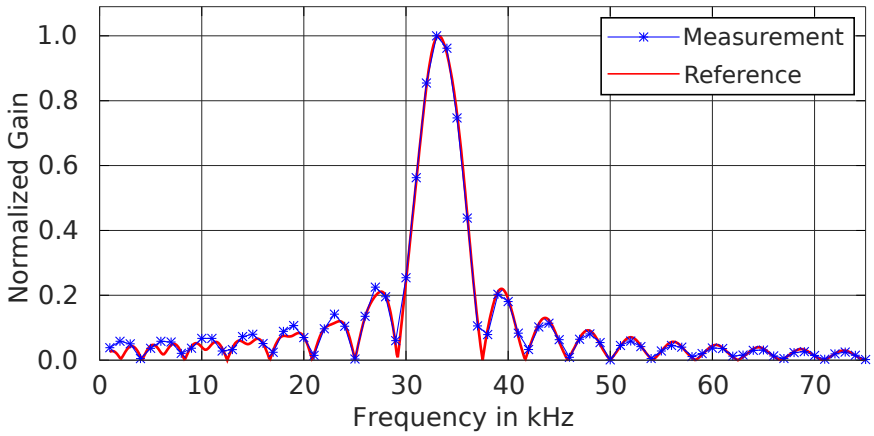


Figure 7.16: Comparison of the expected and measured normalized filter gains.

7.5.7 Computation Approximations

To determine the sign of the differential capacitance, the arctangent of the complex result is computed. However, such an operation is computationally expensive, especially on a microcontroller. Hence, an approximation formula was employed that provides a good trade-off between performance and accuracy. Approximation number 10 in the corresponding publication [RWIJ06] was chosen which computes the arctangent via

$$\operatorname{atan}(x) \approx \frac{x}{1.0 + (0.28086 \cdot x^2)}, \quad -1 \leq x \leq 1. \tag{7.6}$$

The formula is run on the FPU and requires only three multiplications, one addition, and one division. On the Cortex-M4 FPU, multiplications, as well as additions, take a single clock cycle while divisions are more expensive and take 14 clock cycles [STM16a]. Since this approximation only needs a single division, this reduces the overall computation time compared to other approximations. Furthermore, only a single constant, i.e., 0.28086, is loaded from memory, thereby saving time. Please note that loading 1.0 into an FPU register does not trigger an additional memory access as the constant 1.0 is coded directly into the `vmov` instruction.

Continuation of the approximation for values less than -1 and above 1 is possible by extending the formula [Lyo04]. This comes at a nearly negligible performance overhead as the formula is extended by only one addition or subtraction.

The resulting error due to the approximation is plotted in Figure 7.17. Although discontinuities are clearly visible near the restrictions of the approximation formula's domain, this has no relevant influence on the measurement results. The formula returns a valid output for any input. The approximation has shown to be sufficiently precise; however, it could be replaced by an even faster but less accurate formula since the result will be converted into the capacitive measurement's sign only.

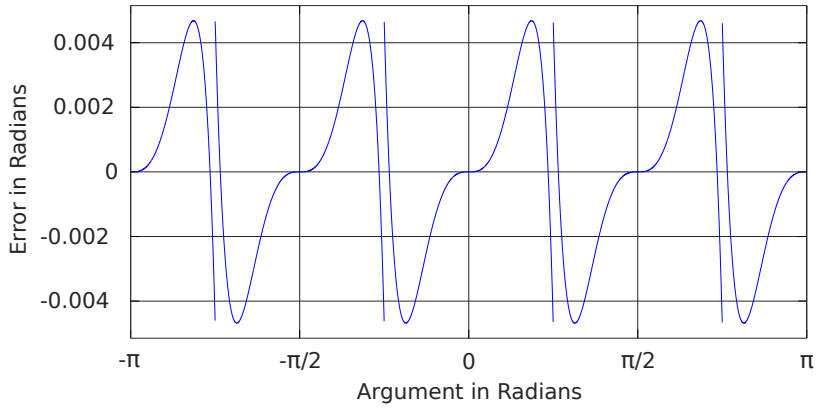


Figure 7.17: Error of the atan2 approximation, which is less than 0.005 radians.

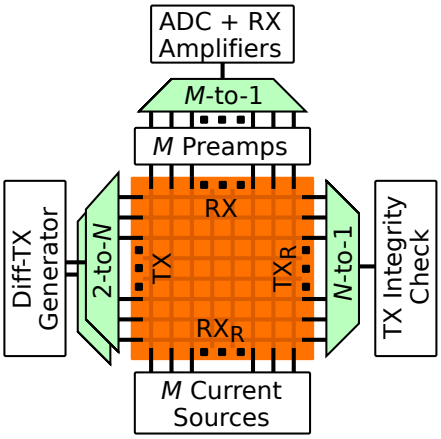


Figure 7.18: Multiplexers scale the concept to enclosures of numerous electrodes.

7.5.8 Scaling the System to $N \times M$ Enclosures

The concept is adaptable to enclosures of different electrode counts. As a general rule, following from *differential* measurement, the number of TX electrodes N must be even; however, no such restriction is implied on the number of RX electrodes M. Although the prototypes were developed with $N = M$, the concept is compatible also to $N \neq M$ as long as the above constraint on an even N is retained. Although no such enclosure was created, yet, this was successfully proven by instructing the measurement firmware to measure only a subset of the available electrodes, such that the number of measured TX electrodes did no longer match the number of RX electrodes. As the nodes are measured one after another, this action was completed successfully without any issues.

Analog multiplexers are introduced to scale the system to other enclosure sizes, as shown in Figure 7.18. Since these are designed towards analog signals they do not have a direction restriction, thus, a 1-to-8 multiplexer can also be used as 8-to-1 multiplexer, for example.

The TX side requires an N-to-2 multiplexer. Because the 0° TX signal is only connected to odd-numbered electrodes and the 180° TX signal is exclusively for even-numbered electrodes, an optimization can be taken. Instead of one large N-to-2 multiplexer, two 1-to-$N/2$ multiplexers suffice, which simplifies the practical realization. The first one handles the 0° excitation signal for odd-numbered electrodes while the second multiplexer handles the other half. Please note that the multiplexers require an enable input which can disable the entire multiplexer in order to test a single electrode during integrity verification or when measuring the absolute capacitance. For example, two ADG708 1-to-8 multiplexers switch the TX signals in my prototype. They have $3\,\Omega$ on-resistance with $0.75\,\Omega$ flatness [Ana14], thus, the influence of the multiplexer on the measurement is expected to be small.

On the $\mathrm{TX_R}$ side, a single N-to-1 multiplexer is connected between the $\mathrm{TX_R}$ outputs and the TX integrity verification comparator. My prototype uses an ADG706 16-to-1 multiplexer, which has comparable characteristics [Ana16] to the aforementioned multiplexer.

On the RX side, an M-to-1 multiplexer is connected after the first stage of RX amplifiers. The multiplexer may not be placed directly at the RX electrode since the currents are roughly in the range of possible leakage currents through the multiplexer; thus, undesired cross-coupling and interference may occur. The current solution prevents such potential issues by amplifying the signals independently and by switching the larger and buffered first-stage output signals. My implementation is also based on the ADG706 16-to-1 multiplexer.

The $\mathrm{RX_R}$ side is not scaled by multiplexers but by adding more current sources. The chosen concept is extendable to any size by replicating the current source, which is described in detail in Subsubsection 7.5.2.4.

Depending on the desired speed-to-area trade-off, the system can be parallelized by evaluating several RX signals in parallel. This requires additional multiplexer devices, signal chains, and ADCs at the RX side. Please note that also the required computational power must be available to acquire and process the incoming signals.

Thus, such an implementation may be better suited when the concept is adapted to run on an FPGA or is integrated into an ASIC. However, the presented system is area-optimized and does not use parallelization.

7.5.9 Overall System Summary

Figure 7.19 shows the entire measurement system, including a general description independent from the enclosure size but including the number of required signals. Each N-to-1 multiplexer requires $\log_2(N)$ output selection signals and one additional signal for output enable control. The same is true for the M-to-1 and $N/2$-to-1 multiplexers. For the current sources, the number of required signals is always 4 independent from

the number of RX electrodes since the current sources are based on shift registers, i.e., they have an SPI-compatible interface. For example, only 28 analog and digital Input/Outputs (IOs) of the microcontroller are required for a 16×16 enclosure. I implemented the measurement system based on an STM32F303RCT7, which comes in an LQFP64 package having 64 pins, of which 52 are IOs. Thus, the implementation only requires moderate resources.

7.6 Optimization: Pipelined Analog Measurement and Signal Processing

The measurement of differential capacitances runs sequentially and the next measurement is only started when the current one has been fully processed. This is depicted in Figure 7.20, which shows the measurement timing method A. Initially, the multiplexers are configured appropriately to address the selected node, then the analog measurement is started and afterward, the digital processing takes place. Although this approach works, there is a large amount of standby-time of the analog and digital circuitry, marked in orange.

The measurement is characterized by its conversion time and latency. The first metric describes the average time for one differential node to be measured, which is related to the system's throughput. The latter one indicates the timespan between starting a measurement and receiving its result. In this case, the measurement's conversion time is the same as its latency, i.e.,

$$T_{\text{Conversion,A}} = T_{\text{Mux}} + T_{\text{Measure}} + T_{\text{Process}}, \tag{7.7}$$

$$T_{\text{Latency,A}} = T_{\text{Mux}} + T_{\text{Measure}} + T_{\text{Process}}. \tag{7.8}$$

The total differential capacitance measurement time can be approximated via

$$T_{\text{Total,A}} \approx N/2 \cdot M \cdot T_{\text{Conversion,A}}. \tag{7.9}$$

For the implemented prototype I measured $T_{\text{Total,A}}$ to be 84 ms.

However, the total measurement time is one crucial performance metric that should be optimized. In some cases, decreasing the average conversion time is possible at the cost of increasing latency — an optimization that appears to be reasonable in this application.

7.6.1 Concept

To optimize the total measurement time, I chose a pipelining approach that interleaves the analog measurement of the next node with the digital processing of the current one. Despite the measurement runs on a microcontroller and not on an FPGA, this sort of optimization is feasible since the analog measurement is handled by hardware, as explained in Subsection 7.5.4, and only the digital signal processing

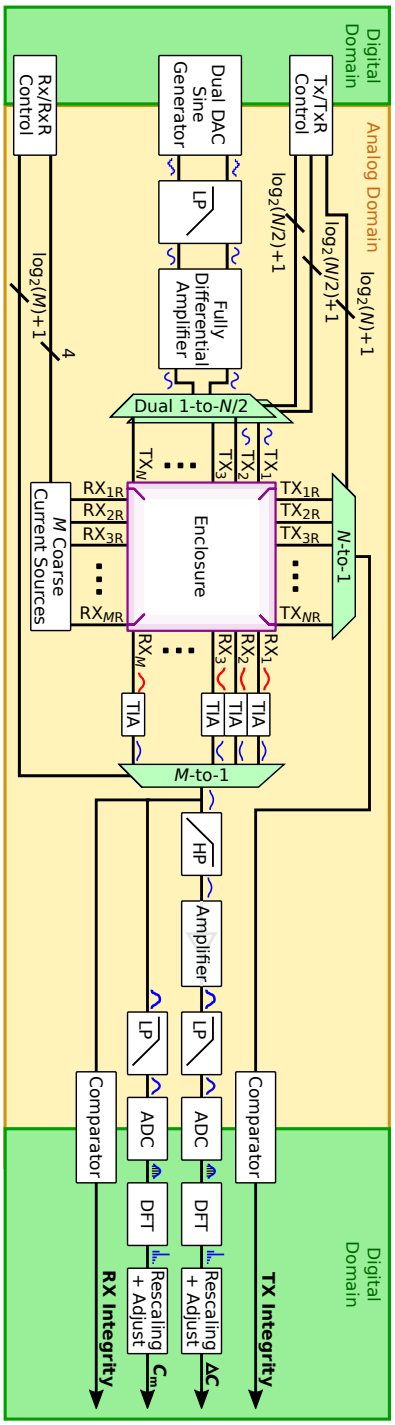


Figure 7.19: Block diagram of the entire measurement system.

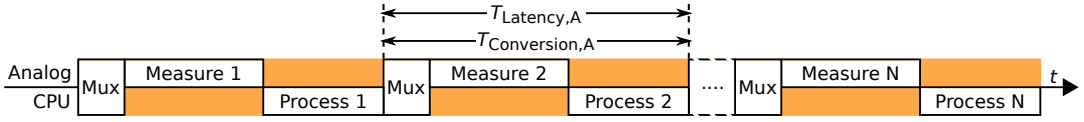


Figure 7.20: Measurement timing in the prototype without optimization (Method A).

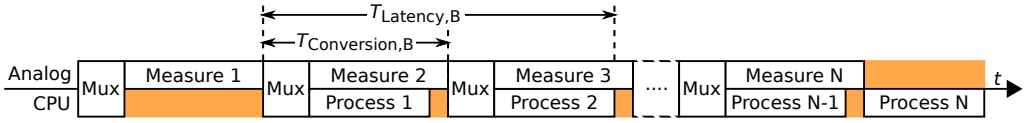


Figure 7.21: Measurement timing in the *optimized* implementation (Method B).

requires the CPU. Furthermore, sufficient RAM is available to implement a double buffering mechanism for the raw analog data.

In this improved method, the multiplexers are configured as before and the first measurement is started. As soon as the measurement has been finished, the analog circuitry is immediately reconfigured to acquire the second measurement. When the second measurement has been started, the CPU processes the data of the first measurement in the meantime while waiting for the acquisition to finish. Thus, these two steps are running in parallel. Due to an optimized implementation, the required time for both operations is very similar; thus, there is almost no waiting time involved.

The measurement is conducted this way except for the first and last measurement, where acquisition and processing have to run alone, respectively. Since these are only two out of more than hundreds of measurements in usual implementations, the following approximations are adequate. The conversion time has decreased and is only the time the multiplexers need to switch and the maximum of the measurement and processing time, i.e.,

$$T_{\text{Conversion,B}} = T_{\text{Mux}} + \max(T_{\text{Measure}}, T_{\text{Process}}). \tag{7.10}$$

For the latency, the time has increased approximately by T_{Mux}, i.e.,

$$T_{\text{Latency,B}} = 2 \cdot T_{\text{Mux}} + \max(T_{\text{Measure}}, T_{\text{Process}}) + T_{\text{Process}}. \tag{7.11}$$

7.6.2 Verification

To verify the method's correct functionality, its implementation was analyzed regarding two properties: The measurement must yield an equal result and the majority of the signal acquisition and data processing must run in parallel. The first requirement was verified by taking measurements and comparing the newly generated measurements to previous ones. For the second requirement, the internal state of the measurement system was made accessible at IO pins, thereby enabling measurements via oscilloscope or logic analyzer.

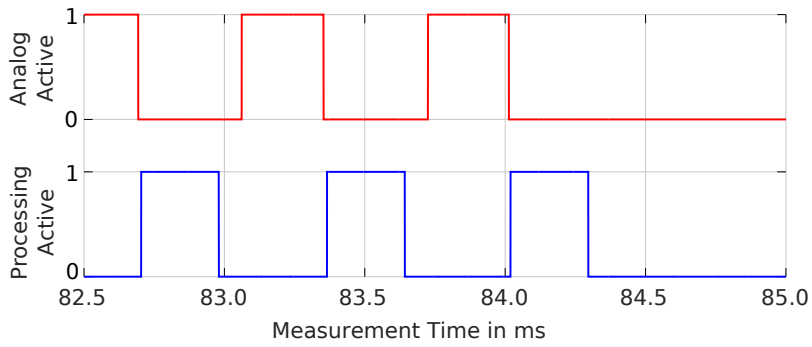


Figure 7.22: Performance/State plot of the end of a measurement cycle of method A.

Initially, the measurement sequence of method A was analyzed, shown in Figure 7.22. In the upper region, the analog IO activity is plotted, which comprises the timespan between the start of the excitation signal generation until the end of the ADC signal acquisition. CPU activity for signal processing is shown in the lower region of the plot; it includes the actions from reading the data from the ADC's DMA buffer until the differential capacitance computation is finished. Since this method is not optimized yet, there is no overlap between both curves resulting in a relatively high total measurement time.

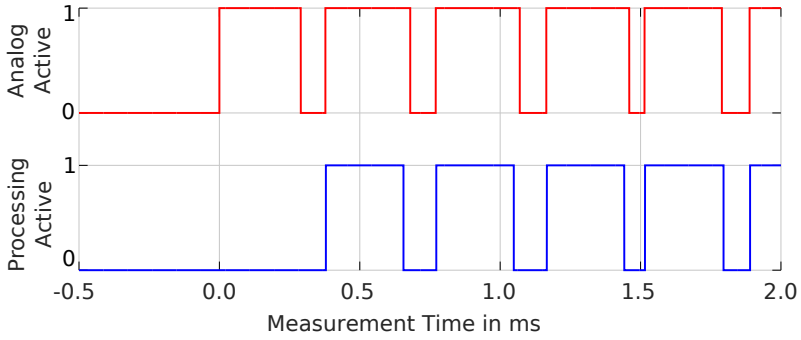


Figure 7.23: Performance/State plot of the beginning of a measurement cycle of method B.

Figure 7.23 and 7.24 show the beginning and end of a 128-node differential measurement cycle based on method B. Altogether, there is a significant overlap between the red and blue activity times; thus, the concept works as designed and both actions are running in parallel. The gap in both traces comes from the time required by the internal state machine and for reconfiguring the multiplexers for the next measurement that takes CPU resources and blocks the analog circuitry as well. A closer look

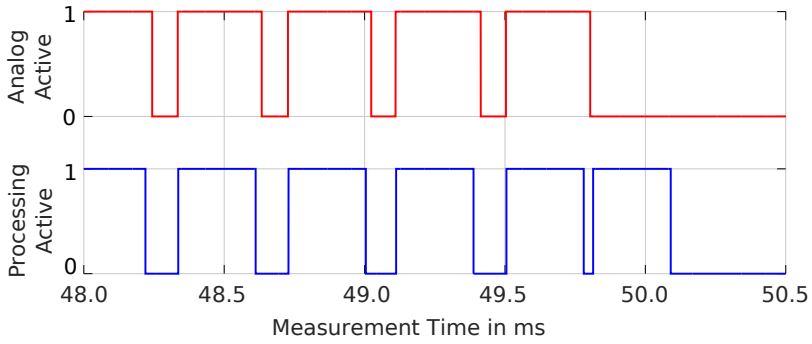


Figure 7.24: Performance/State plot of the end of a measurement cycle of method B.

reveals that the signal processing finishes several microseconds earlier than the data acquisition; thus, spare CPU time is available for additional data processing during the measurement. Altogether, the total conversion time for 128 differential nodes is 50 ms, which is a 40 % improvement relative to the 84 ms measurement time of the initial method.

7.7 Practical Verification

I verified my concept of a measurement system on two different enclosure technologies: One solution is based on an envelope, while the other solution is a cover-based one. For the first one, I built up a single measurement system and I created two prototypes for the latter one. Both prototype designs are mostly based on Surface-Mounted Device (SMD) components as they provide a small footprint that allows moving the components closer, thereby reducing parasitic effects and the susceptibility to external influences. The PCBs were designed using KiCad, an open source Electronic Design Automation (EDA) suite [KiC18].

7.7.1 COPYCAT Prototype System

I implemented the proposed measurement system for the 16×16 envelope of the COPYCAT project. The prototype is shown in Figure 7.25 which resembles the measurement circuit V4.0. The system features a four-layer board, providing power planes and shields, separating analog and digital signals for a low-noise design. The components are placed on both sides of the PCB; however, the majority is located on the front side. The system is divided into three main regions: the analog measurement circuit, the digital circuit, and the debug interfaces. An external board was added, which is shown in the upper right corner of Figure 7.25. It partially replaces the original TX circuitry of V4.0 with the fully-differential amplifier design of V4.1 that provides better amplitude and phase matching.

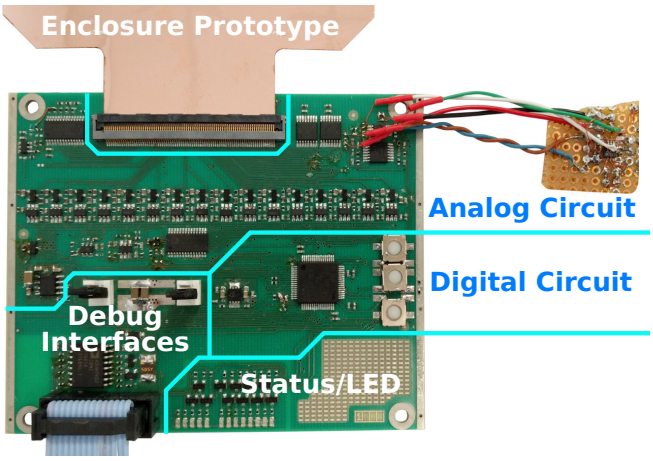


Figure 7.25: The measurement system prototype.

Altogether, V4.0 has a full-scale range of ±73 fF represented by ±10 000 points at a theoretical digital resolution of 7.3 aF. However, for the full setup of the measurement system and enclosure, precision is limited by a noise standard deviation of 0.3 fF corresponding to approximately 41 points. The measurement time per node is 390 µs on average whereas the total differential measurement time for the entire enclosure is 50 ms.

7.7.2 COVER Project Prototype System

Additionally to the COPYCAT project, I implemented the measurement system also for the 16 × 16 topside cover of the COVER project. The prototype is shown in Figure 7.26 to 7.28 which resembles the measurement circuit V4.2. Version 4.1 is not shown as it is very similar to V4.2 except for some minor improvements. The presented measurement system features a six-layer board; thus, two additional layers are present, which, among other functions, shield the cover's traces on the bottom from the mixed-signal signals on the top. Additionally, power and ground planes are present that provide a low-impedance power source for low-noise measurements. For this prototype, all measurement circuit components are located on the top side of the PCB while the cover is placed on the bottom side. This comes with the considerable advantage that all signals are accessible even if the cover is in place, thereby easing debugging and testing.

The board is divided into several sections, marked on Figure 7.26. Region A contains the system's interfaces, such as the UART and debugging interface, as well as several LEDs for status indication. Region B comprises the power supply, i.e., two Low-Dropout Regulators (LDOs) at the upper and current measurement taps at the lower region. The digital and mixed-signal part of the circuit where the microcontroller is present is marked with C. The primarily analog measurement components of the

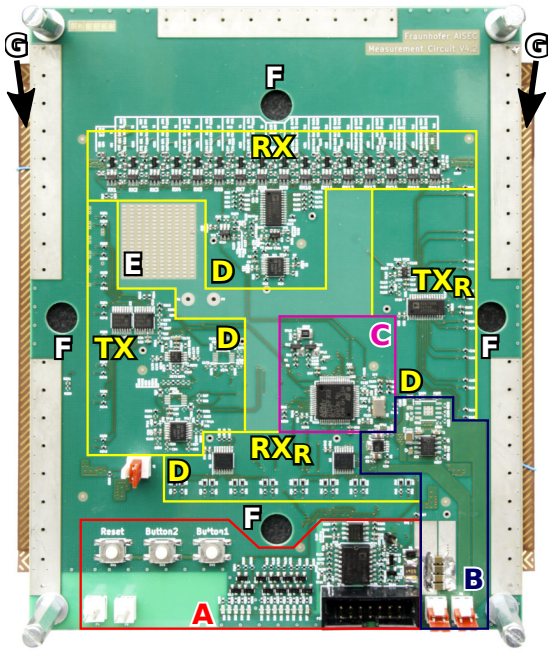


Figure 7.26: Front side of the COVER project measurement system prototype.

system are in Region D. The markings TX, TX_R, RX, and RX_R indicate the purpose of each region. Region E solely contains the unused region of the board, including a test and prototyping area. Four holes on the PCB are present, which are marked with F. Their purpose is to provide access to the stiffener frame to ease the detaching of the cover during testing. Next to the PCB, the edge of the bottom side of the cover is visible, marked by G.

The back side of the system, shown in Figure 7.27 and 7.28, contains only the cover and its connectors. Each edge of the cover's connectors is dedicated to one trace functionality. Metal feet, marked with J, have been attached to the system to prevent unwanted mechanical force on the cover. This side of the PCB is shielded from the top side by an internal copper plane, denoted by K. Additionally, the galvanically isolated communication supply area and ground are also visible, marked with L.

The measurement range of the system was increased by adjusting gain-setting resistors to account for the larger variation in the FlexPCB-based cover. The COVER project's measurement system covers a full-scale range of ±134 fF which is mapped onto ±10 000 points at a theoretical digital resolution of 13.4 aF. The achieved resolution due to noise, caused by the cover and measurement system, limits the precision by a noise standard deviation of 1.7 fF, which corresponds to approximately 129 points. Since this system runs a similar firmware, the same performance figures apply, the measurement time per node is 390 µs on average whereas the total differential measurement time for the entire enclosure is 50 ms.

Figure 7.27: Back side of the measurement system including cover.

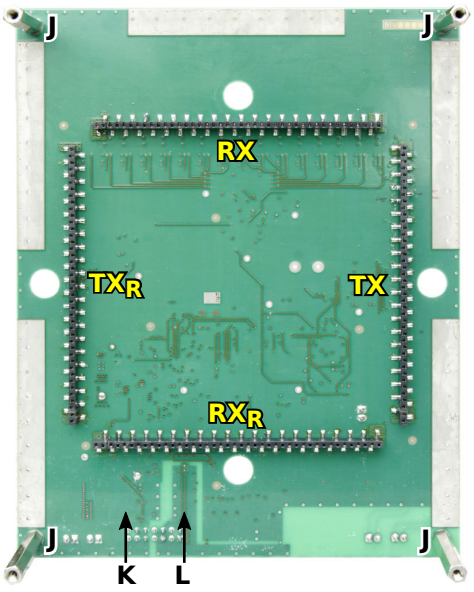


Figure 7.28: Back side of the board after the cover has been detached.

7.7.3 Measurement System Practical Tests

I tested all three revisions of my measurement system with several prototypes. For the COPYCAT project, the Fraunhofer EMFT manufactured enclosure prototypes on their proprietary four-layer thin-film technology. For the COVER project, the covers were manufactured in a commercial FlexPCB technology according to a design created by the DSO National Laboratories. Both batches of enclosures and covers enabled thorough testing of the system.

When starting the measurement, the integrity verification correctly detects shorted and interrupted electrodes. This was tested by means of one COPYCAT enclosure that had an interrupted electrode. I tested the system also via a manually configurable switch matrix that allows inserting short and open circuits. The erroneous electrodes were correctly identified. During development, an error in the integrity verification will not trigger zeroization and the enclosure will be measured; nevertheless, however, defective electrodes will be omitted to protect the measurement system.

The differential capacitance measurement was verified via the enclosure and cover prototypes. Each revision of the measurement system works correctly. For instance, the extracted ΔC of the COPYCAT enclosure is depicted in Figure 7.29. Each data point represents one differential capacitive node. The first 16 data points indicate RX_1 to RX_{16} excited by the first TX pair TX_1-TX_2. The plot continues with RX_1 to RX_{16} excited by TX_3-TX_4 until RX_1 to RX_{16} excited by the last pair TX_{15}-TX_{16}. The data

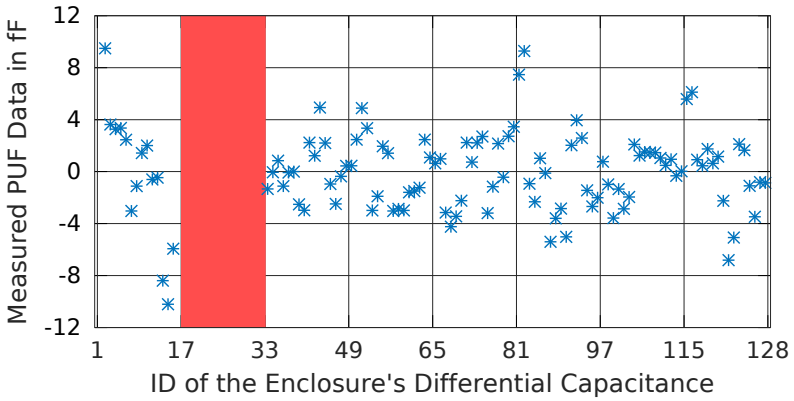


Figure 7.29: Data of 112 differential PUF nodes (blue) and a successfully detected defect (red) in the TX_3-TX_4 excitation pair.

of each TX group has a mean value of zero since TX group offsets are removed by the preprocessing algorithms. A TX group is defined as all nodes that have been measured via exciting the same TX electrodes, thus, each TX group consists of 16 nodes. This procedure removes several sources of offsets and the background is further explained in the next subsection. The highlighted column in the plot marks the successful detection of a defect in the TX_3-TX_4 electrode pair. The results demonstrate that both features, the integrity verification and the differential capacitance measurement, work as designed.

7.7.4 Independence of PUF Data from the Measurement System

The measurement system was designed such that the PUF should be extracted only from the enclosure or cover but not from the circuit; however, this has not been proven in practice, yet. To investigate this further, the same cover was evaluated using two *distinct* measurement systems, i.e., device revision V4.1 and V4.2, and then the results were compared.

Regarding the revisions, the second PCB and its components were ordered several months after the first one; thus, they were not manufactured in the same batch. I expected at least a small amount of variation in the characteristics of both PCBs; however, this must not affect the measurement results. In order to estimate the variation in PCB trace manufacturing, I added a resistive test structure on an internal PCB layer. The structure consists of meander traces with a width of 0.15 mm, a spacing of 0.25 mm, and a total length of roughly 1.95 m.

The test structure's resistance was measured after receiving the manufactured PCBs. The resistance is 7.84 Ω for the PCB of V4.1 and 10.29 Ω for the PCB of V4.2. Thus, there is a surprisingly large variation in the trace resistance of V4.1 compared

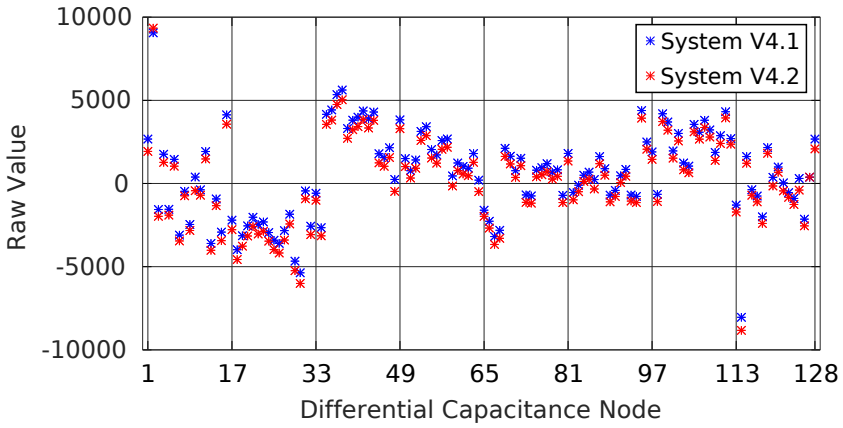


Figure 7.30: The same cover evaluated via two measurement systems.

to V4.2 that amounts to more than 30 %; however, this must not affect the extracted PUF data.

For my test, I chose an entirely intact cover with a typical set of differential PUF nodes. The raw measurement results for both measurement systems before any preprocessing are shown in Figure 7.30. Although the values differ by 471 points, corresponding to less than 3 % of the full-scale range, on average, they have a very similar structure. The correlation coefficient between both measurements is 0.998, which strongly supports their similarity.

A closer visual analysis of the plot reveals that the offset for every node is almost equal. As a solution, I developed a compensation method that computes the average of each TX group and subtracts it from the measurement. Thereby, global and TX group local offsets vanish, but at the cost of a small loss of information. The information being lost is the sum of all nodes inside each group, i.e., $\sum_{c=1}^{M} C_{a,b,c}$ for $\mathrm{TX_a}$ and $\mathrm{TX_b}$, which is set to zero by this method. Applying this algorithm is computationally cheap as there are only $M - 1$ additions, one division, and M subtractions per TX group. Since M is a constant, the division could also be replaced by a multiplication with the constant $1/M$ as this is faster by a factor of 14 on the microcontroller's FPU.

The resulting data after applying the compensation to both measurements is depicted in Figure 7.31. Obviously, the deviation between the results of both measurement systems has been strongly reduced and is barely recognizable anymore. The source of this error might be a mismatch in the amplitude matching of both excitation signals, which is known to cause an offset in the measurement data. It can either be introduced by the fully-differential amplifier or by an imperfection in the multiplexer's channel resistance matching. Since the error is small, it can be compensated, and it does not limit the dynamic range significantly, no further action is required.

These results verify that the PUF of the preprocessed data solely lies in the enclosure but not in the measurement system. The circuit-intrinsic error can be easily

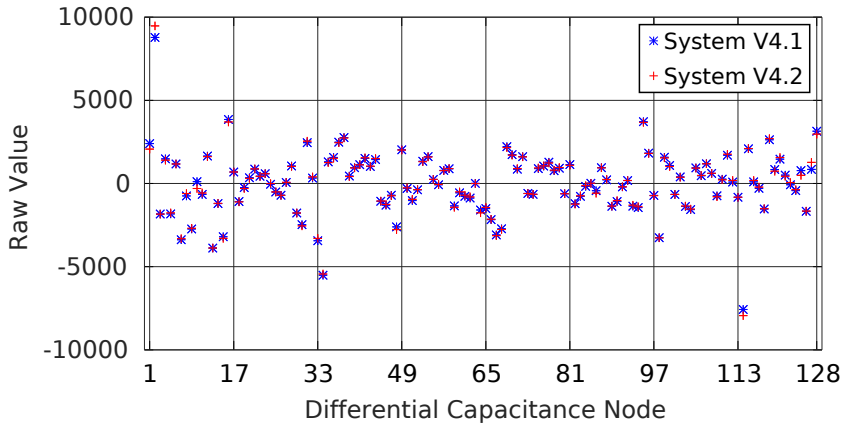


Figure 7.31: The offset-compensated differential PUF data of one cover evaluated by two measurement systems.

compensated so that its influence is entirely negligible and no unintended contributions to the PUF exist.

Applying a TX group-based compensation has the additional advantage to enhance the PUF's statistical distribution and to compensate for environmental effects. When analyzing the second and third TX group of the differential PUF nodes before pre-processing, the first 16 nodes all show a negative bias while the next 16 nodes show a strong positive bias. After applying the compensation algorithm, all nodes are then conveniently arranged around zero; thus, the mean value of the distribution has been optimized. When a sufficiently large number of covers is being analyzed, a Gaussian distribution becomes visible as desired by the concept [ION+18]. Additionally, early temperature and climate tests indicate that this compensation algorithm also accounts partially for humidity and temperature deviations. Thus, even without the deviation caused by the circuitry, such a compensation mechanism would be required to obtain well-usable PUF data.

I performed the same measurement on the absolute capacitance of the cover to verify the result's independence from the measurement system. Since the absolute measurement is rather coarse and does not rely on perfect amplitude matching, it is not subjected to the same error mechanism. As shown in Figure 7.32, the values are already matching precisely and no compensation mechanism is required so far. Please note that the number of trace overlaps and thereby the absolute capacitance may vary between nodes.

Altogether, this concludes the PUF-source evaluation of the measurement system. The results clearly show that the PUF solely emerges from the enclosure and that there is, despite a trivially compensatable offset, no influence from the measurement system. Thus, the system provides an enclosure-PUF but not a circuit-PUF.

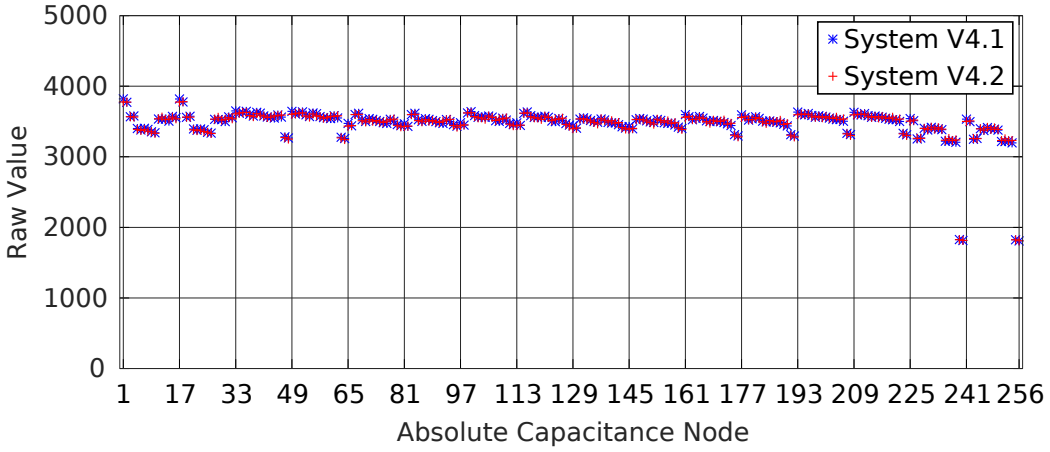


Figure 7.32: Absolute capacitance data of one cover evaluated by two measurement systems.

7.7.5 Thermal Analysis

On average, the measurement system draws 132 mA on the analog rail and 43 mA on the digital rail at 3.3 V. This results in a power dissipation of 0.6 W, which is within the specified limit of 1 W. However, heat distribution on the PCB shall be analyzed to identify potential problems, especially when components are integrated more densely.

The COVER project measurement system is well suited for such an analysis since all components reside exclusively on the top side of the PCB, exposing them for thermal analysis. Figure 7.33 displays the board with a thermal overlay that has been acquired via a *FLIR TG167* thermal camera. The temperature difference between the coldest (blue) and hottest (red) spot is only roughly 4 K. Such temperatures are of no concern; however, the four locations with the highest temperatures are marked. Label A denotes the microcontroller, which has a weak thermal connection to the PCB only through its pins but without a power pad. The analog and digital 3.3 V LDOs, marked with B and C, respectively, are thermally enhanced packages, designed to spread the heat into the PCB, resulting in lower temperatures. Label D marks the LEDs, which also have a slightly increased temperature against their environment but which is also without any concern. Please note, that the entire analog circuitry shows no visible heating at all, minimizing thermal offsets thereby.

The results show that the heat is distributed well across the PCB's surface and no noteworthy hot spots exist. Additionally, self-heating of the system was minimized by design as its overall power dissipation, including LDOs and LEDs, is less than 1 W. Furthermore, the PCB comprises three power and GND planes in total, which help to spread the heat across the entire device, preventing hot spots. Hence, no further actions or optimizations for thermal management are required.

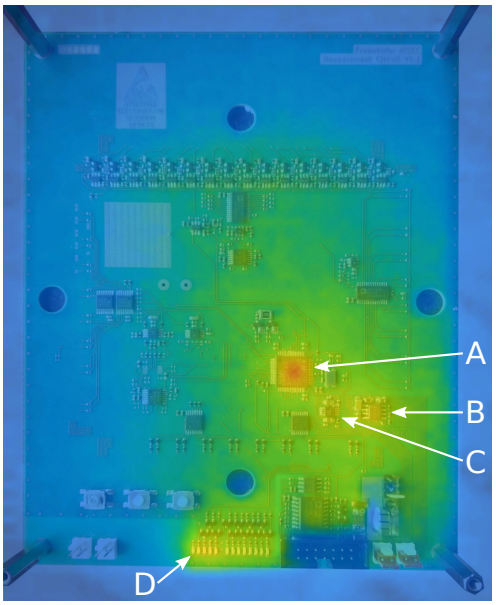


Figure 7.33: COVER project measurement system with thermal image overlay.

7.8 Attack Experiments

Due to the novelty of PUF-based capacitive security enclosures, no attack results or methods are available so far. However, the attacker profile is similar to that of previous resistance-based enclosures, i.e., an attacker wants to gain access to the internal embedded system without being detected. The subsequent attacks could range from simple probing of traces up to decapsulation of integrated circuits and the extraction of information via an electromagnetic field probe. An example for such an attack is depicted in Figure 7.34 which shows a cover that has been opened and partially repaired to pass the integrity verification although the PUF was destroyed.

The aforementioned attack is the most apparent type of intrusion that creates holes in the cover to perform further manipulations. Without any repair, a drilled hole as small as 0.3 mm has always been successfully detected in attack experiments. This has been analyzed in detail and the results showed the destruction of several PUF nodes at the same time. Thus, this class of attacks has been sufficiently covered [ION+18].

However, since designing, building, and optimizing the measurement system provides a deep insight into the overall system and potential weak spots, I decided to investigate attacks specifically directed onto PUF-based capacitive solutions. Instead of trying to create a hole in the protective cover undetectedly, I aim at extracting the PUF's secrets and systematically disabling portions of the cover. This experiment will show whether additional countermeasures against such attacks are required, e.g., improved and more sensitive cover materials.

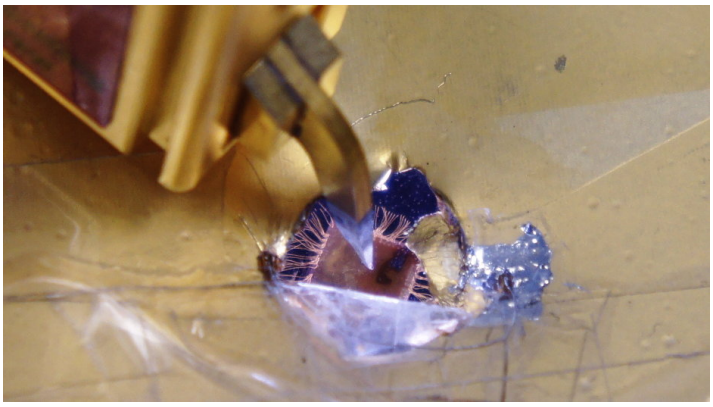


Figure 7.34: Example for an EM probe attack on a decapsulated IC after removal of the security cover.

7.8.1 Concept

The attack concept is based on the idea that information may be extractable via probing the electrodes. The excitation signal of TX electrodes is very easy to sense by merely probing the trace *voltage* with an oscilloscope. However, no new information can be gained therefrom since neither the amplitude nor the frequency carries any secret information because both parameters are independent of the PUF. All raw PUF data is solely transported on the RX traces in the form of minuscule sub-nanoampere *currents*. Thus, probing the voltage of an RX trace will not yield information, but current probing is difficult.

My search for commercial off-the-shelf solutions for non-invasive sub-nanoampere current sensing in the kilohertz range was unsuccessful. Hall-sensor based probes are not suited for such small currents as their magnetic fields are too small to be sensed well. The noise of such probes is in the hundreds of microampere range [Key18] and thereby much larger than the current that shall be sensed. Even interrupting an RX trace and measuring with standard current measurement tools such as multimeters is not successful, since the measurement burst has a duration of only 240 µs which cannot be acquired quickly enough. Additionally, the phase information and thereby the sign of the measured data is lost. Altogether, this prevents any quick and straightforward solutions — as intended by the measurement concept.

To overcome this, I developed an attack method that leaves the enclosure mostly intact but enables access to the currents on the RX traces. Initially, the shielding and its substrate are removed for a tiny region, and one RX trace is being interrupted, as shown in Figure 7.35. Next, two fine and shielded wires are soldered onto the RX trace that both lead to an external attack tool, which is named *RX probe*. This circuit makes an amplified version of the RX current at RX IN available as a voltage at the monitoring output (MON OUT) and reproduces the original RX current at another output (RX OUT). This replicated current is sent back into the enclosure to trick the

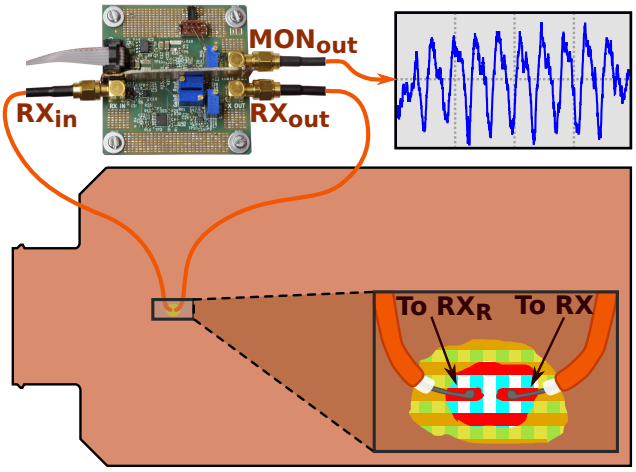


Figure 7.35: Concept for attacking the security enclosure via the RX probe tool.

system into the assumption that the enclosure is intact. The voltage at the monitoring output can be used to extract the PUF data. This attack concept allows witnessing the enclosure-internal current without impeding the measurement or integrity verification.

7.8.2 RX Probe Implementation

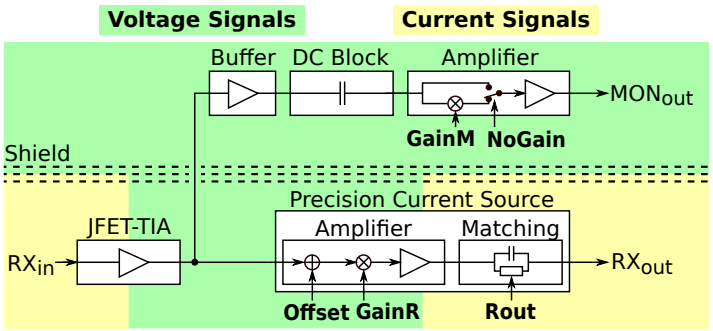


Figure 7.36: Attack circuit concept for the RX probe tool.

I implemented the proposed attack concept in a hardware circuit; the chosen design is shown in Figure 7.36. The RX probe was created similarly to the measurement system; thus, it comprises a JFET-TIA as the first input state. This amplifier converts the current into a voltage and the resulting signal is split for two sub-circuits. The first one generates the signal of the monitoring output. The very small voltage signal is buffered at first to prevent influence from the subsequent circuitry. Afterward, a DC block removes the DC offset component and the signal is then amplified further to reach a suitable amplitude. The gain is fine-adjustable via a potentiometer and the

gain-setting can optionally be bypassed. The bypass has to be selected for absolute capacitance measurement to prevent overdriving the amplifier since the input signal is larger in this measurement mode.

The second sub-circuit is a precision current source that replicates the original input signal. Since this part has to be able also to reproduce the integrity verification signal, no DC block is present. The source has an adjustable offset to set the output voltage for the idle state. The adjustable gain helps to fine-tune the current source so that the amplitudes of the input and output signals match. The third adjustable parameter is the output resistance of the source's output matching circuit, which can also be tuned to improve signal quality. All those parameters are set via potentiometers on the PCB.

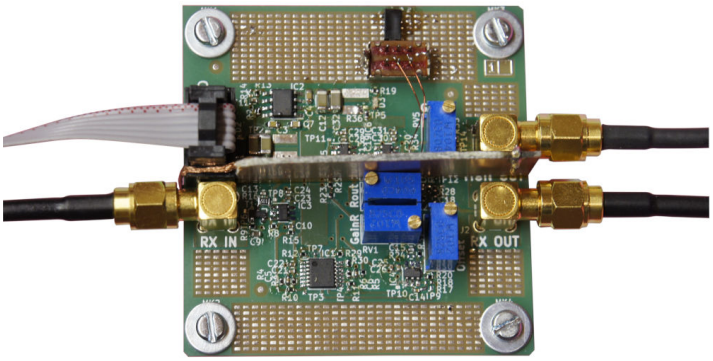


Figure 7.37: Implementation of the RX probe tool used for RX electrode current tapping.

I created the circuit, shown in Figure 7.37, according to the aforementioned design. The circuit is powered from an external 5 V linear power source. The RX input, RX output, and the monitoring output all have an SMA connector to provide a well-shielded interface. The potentiometers, appearing in blue, on the PCB allow fine-tuning of circuit parameters. However, one underestimated aspect during development was the coupling between traces and components on top of the PCB that introduced an unwanted offset in the replicated RX signal. The source of this error was the comparably strong monitoring signal; thus, I retrofitted a shielding. I soldered a grounded copper plate near the center atop the PCB between the monitoring and the RX replicating circuitry. This fully resolved the issue and the circuit showed the intended performance. The switch, which allows bypassing the monitoring amplifier gain, was also a late-addition to the circuit and is located near its upper edge. Both additions were quickly implemented as the circuit was designed to ease such extensions.

7.8.3 Experimentation and Testing

The entire measurement setup in my lab is shown in Figure 7.38. The test comprises a cover of the COVER project but could conceptionally also be performed on the COPYCAT enclosures. Instead of soldering the RX probe wires to the cover's traces, the attack was conducted directly at the connectors of the measurement circuit. My

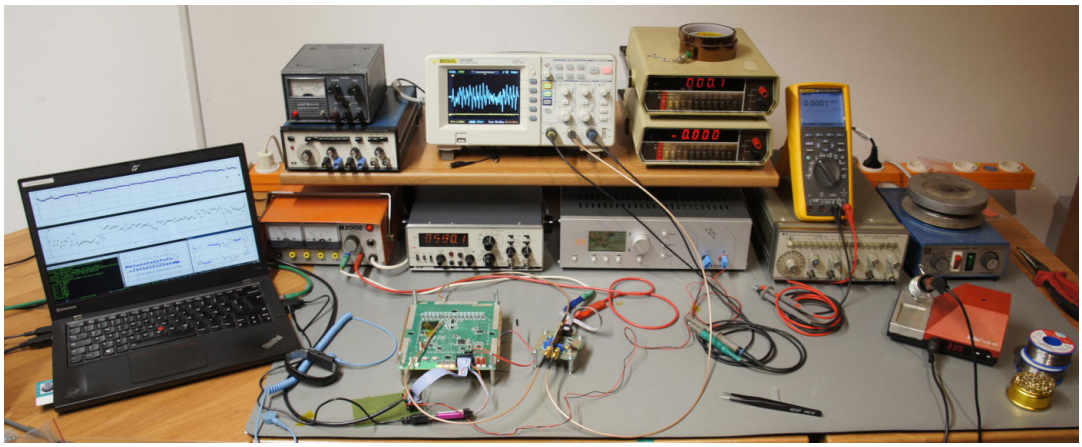


Figure 7.38: The RX probe attack setup in my lab during experimentation.

experiments have shown that soldering wires to the 100 µm thin traces is indeed possible; thus, this is already proven. To estimate the worst-case effects of the attack, the most critical location was chosen for tapping the RX signal: As close as possible to the RX connector of the cover since this is the point where the entire information-carrying RX current passes through. The other extreme is the RX_R end of the cover; however, no current is sensible at this location. Please note that the concept only works correctly if the current is tapped near the cover's connector; otherwise capacitors and the information contained therein are partially invisible. Thus, interrupting the trace and inserting the attack circuitry directly at the RX connector represents the perfect attack position.

All experiments were conducted under three situations and data was obtained via the measurement circuit in order to check for noticeable differences. For the attack, the RX_5 electrode was chosen since it represents a typical electrode in the center region of the cover. First, the system is in the state prior to tampering and a reference measurement is generated. Secondly, the RX probe is connected in between the cover and the RX connector of the measurement circuit. Thirdly, the RX probe is connected in between the cover and the RX_R connector of the measurement circuit. If the attack tool works as designed, no difference in the integrity verification results, differential measurement data, and absolute measurement data must be observable.

In the first test, the integrity verification was started. Since the attack circuit replicates the integrity verification signal correctly, this attack was not detected independently from the position of the RX probe.

Next, a differential measurement was started and three raw results without TX group compensation were obtained. They are depicted in Figure 7.39 and are represented by different symbols that allow a visual assessment of their conformity. As clearly visible in the figure, the RX probe circuit nearly perfectly reproduces the RX current signals when it is connected at the RX end of the cover. Even if it is connected at the RX_R

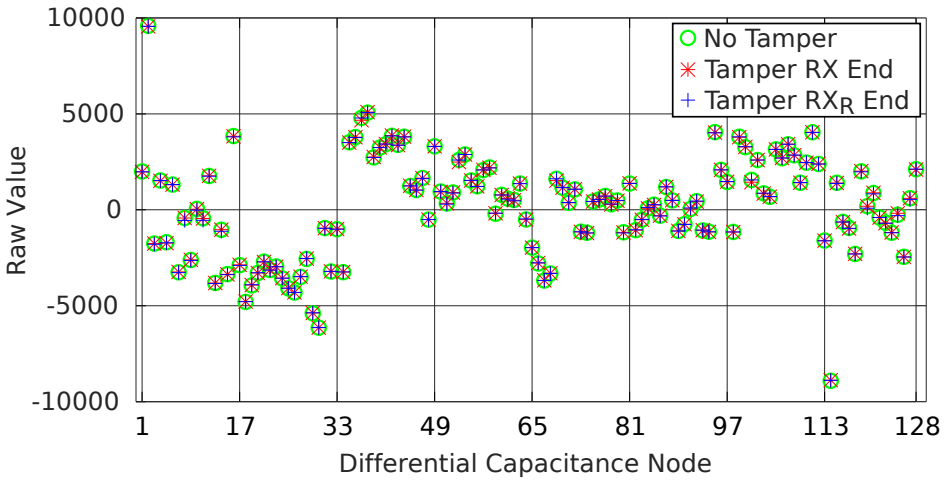


Figure 7.39: Comparison of a differential capacitance reference measurement with the tampered system using the RX probe tool.

end, it does not prevent a successful measurement as the RX probe does not consume any RX current at its *output*. This means that independently from the position of the RX probe, the measurement is not affected.

The same experiment was conducted for the absolute capacitance measurement, shown in Figure 7.40. Also, in this case, there is no clearly visible difference between the tampered and the non-tampered case. Only a small deviation for RX_5 is visible where the value diverges by a few percents towards a lower value. However, this is hardly detectable and could most likely be compensated by adjusting the potentiometers more precisely.

Finally, the resulting signal at the measurement system's ADC, plotted in Figure 7.41, is analyzed. A practically identical signal is also available at the monitoring output of the RX probe; thus, the analog current signal has been copied precisely. The depicted signal corresponds to the measurement for $C_{3,4,5}$, which has been replicated by the RX probe. The previous and next measurements are original measurement traces from the cover and were not modified. The signal appears similar to that of the other capacitances, thus, there is no clearly visible deviation and detection of tampering in such a condition is difficult. However, at a closer look, some minor deviations could be used for attack detection, which will be discussed in the next subsection.

7.8.4 Detection of Tampering and Countermeasures

As shown in the previous subsection, the detection of an attack may be difficult. However, a close analysis of the raw analog data revealed that the RX probe circuit manipulates some noise components. In Figure 7.41, the peak of the center trace appears to be shifted slightly to the left while the peak of the two other traces is

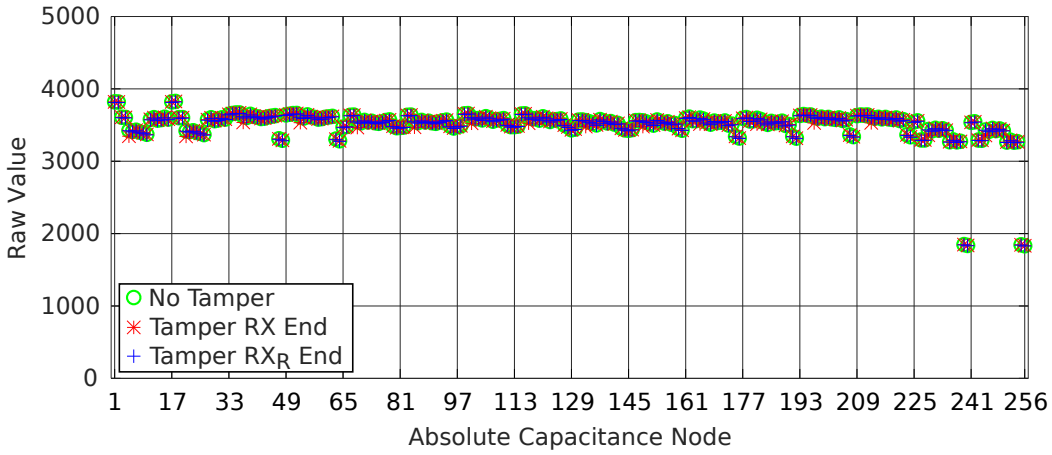


Figure 7.40: Comparison of an absolute capacitance reference measurement with the tampered system using the RX probe tool.

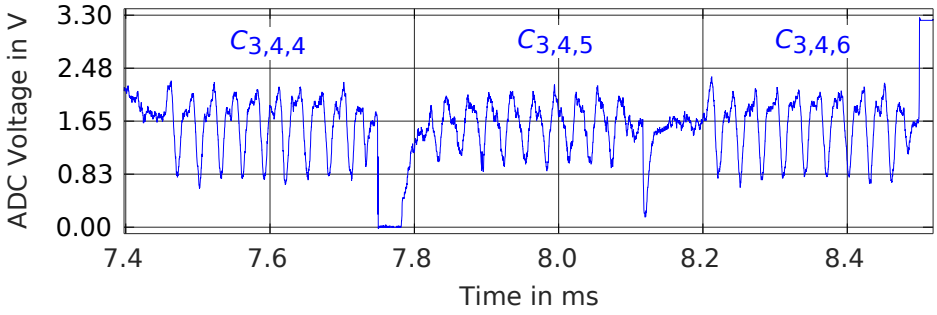


Figure 7.41: The measurement circuits raw ADC input during a differential capacitance measurement.

slightly at their right side. An FFT of the tampered electrodes' signals reveals that the phase of its $66.\overline{6}$ kHz noise component has an offset of 135° compared to the non-tampered electrodes. This deviation is depicted in Figure 7.42, which clearly indicates that RX_5 was manipulated. From a technical point of view, such a check can be implemented comparably easily into the firmware by performing an additional Goertzel Filter computation.

However, such an indicator is difficult to identify and justify technically since the source of this deviation is unclear. Additionally, the RX probe circuit can most likely be adapted to take this into account and such a countermeasure would be futile. Thus, the countermeasure to such an attack might not lie within the measurement circuit, but the system must be protected by other means.

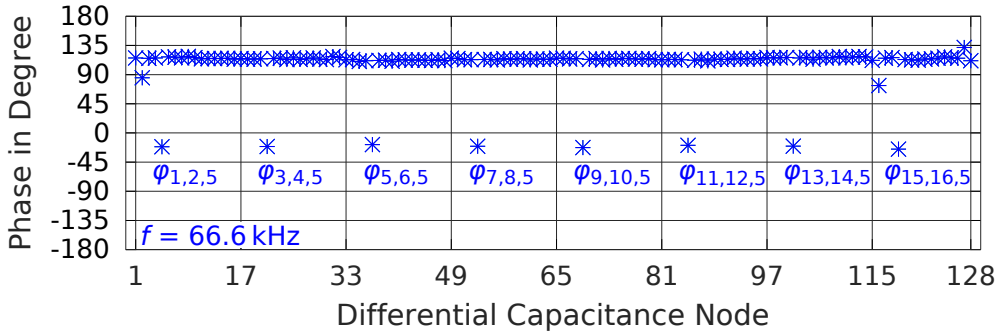


Figure 7.42: The RX-Probe circuit at RX_5 distorts the second harmonic at $66.\overline{6}$ kHz and causes a 135° phase shift for $\varphi_{a,b,5}^{f=66.6\,\text{kHz}}$.

7.8.5 Consequences and Possible Countermeasures

These results indicate a potential security issue, but it does not nullify security immediately. In practice, the attack might not be as simple as described since the attack experiment made several assumptions regarding a nearly-perfect attack. For example, the enclosure, which wraps the device entirely, has an overlapping region where the enclosure partially covers itself. This overlapping region is close to the connector; hence, the described attack is more difficult as the attacker would either have to penetrate multiple layers or could not access the enclosure near the connector. Additionally, significant reverse-engineering would be required to get a sufficiently deep understanding of the system to exploit this and to interpret the extracted raw PUF data correctly.

For best security, a countermeasure to this type of attack should be present in the enclosure or cover itself as the weakness cannot be fixed solely in the measurement system. Security enclosures and covers should be made from materials that do not allow soldering or are mechanically sensitive against tampering. Currently, the thin TX layer of the COPYCAT enclosure cannot be soldered directly as it is destroyed immediately. Switching TX and RX layers, i.e., choosing the thicker layer for TX and the thinner one for RX, would already improve security significantly. Also, with respect to the most recent developments of the new layer stack, its effect on tamper resistance has to be evaluated. In addition to it, future materials could implement some kind of avalanche effect that shatters the entire enclosure or cover upon puncturing it.

7.9 Future Developments

During development and testing, the system performed as designed, thus, no major redesign is required. Since the concept has matured in the last few years, some optional components, such as 0 Ω resistors for testing purposes, are not required anymore and can be removed to preserve space. In contrast to this, a larger microcontroller which

provides more IOs and memory might be necessary in the future since the current one is close to its limits. I implemented the system's firmware with portability in mind; thus, upgrading the system's microcontroller to a larger device of the same device family comes at minimal effort. Furthermore, the firmware has successfully been built and tested with the most recent GCC compiler version in mid-2018, i.e., the GCC 8.2.0; thus, the code base is well-prepared for any future (re)use.

During his research internship, Martin Achtner showed that the measurement concept, especially the signal generation, acquisition, and processing, can be implemented on an FPGA. Thus, the concept is not bound to a microcontroller-based system but may also be integrated into an FPGA or even an ASIC. This opens up new possibilities, especially for more time-critical systems that require measurement parallelization, which could easily be implemented on an FPGA.

However, some measurement-related tasks have to be resolved for a higher technology readiness level. For example, the influences of environmental conditions such as humidity and temperature on the enclosure have to be compensated to achieve sufficient reliability. The first experiments have shown that the enclosure is able to absorb water, which changes the measurement results; thus, future work has to be performed in this direction.

7.10 Conclusion

In this chapter, I presented a scalable system that combines both integrity measurement and PUF evaluation for tamper protection enclosures. Combining both properties has the advantage of protecting a system during operation by integrity measurements and during non-operating phases, such as storage or transport, by a key derived from a PUF on-demand at runtime. This has a significant impact on the usability of tamper protected products: It relaxes storage and transport requirements and also extends their lifetime. Unfortunately, these advantages are accompanied by the challenge to measure capacitance variations in the femtofarad range that is several orders of magnitude smaller than the nominal values while fulfilling strict timing constraints defined by the security requirements.

My measurement solution solves all these challenges in one system. I demonstrated the concept's feasibility by implementing and testing prototypes that successfully evaluate a 128 differential node capacitive PUF enclosure manufactured in two very different technologies. My system extracts the PUF properties within the specified time and precision despite a large parasitic capacitance being present. As my concept can be scaled by two parameters, i.e., electrode count and speed vs. area, it is a flexible building block for PUF-based security enclosures. Furthermore, the most critical components of the measurement concept were successfully implemented on an FPGA, showing that the presented solution is basically compatible with FPGAs and might even be implemented on an ASIC.

Additionally, I performed an evaluation of one physical attack on the system that chooses a signal replication approach. I was able to show that sophisticated attack tools

are required to impede system security; however, the attack is difficult to perform and a solution is out of the scope of this dissertation as physical attacks must be prevented by the enclosure itself. Nevertheless, it gives an insight into future work that has to be completed before the entire protection concept is ready for use.

This chapter presented a solution for the measurement of capacitive PUF-based security enclosures. However, the next step in the development process is the creation of a concept to integrate this measurement system into an HSM. Thus, I developed the Embedded Key Management System (EKMS) firmware, which serves a basis for combining the measurement with a security-enhanced operating system [OHHS18].

Chapter 8

An Embedded Key Management System for PUF-Based Security Enclosures

Protected devices, such as HSMs, comprise the host system and a monitoring circuit. In resistance-based solutions, the monitoring circuit has only a few and simple tasks such as the verification of the traces' integrity, supervision of supply voltages, and finally, zeroization of memory in case of a tamper event. Compared to resistance-based solutions, a PUF-based solution requires a more sophisticated monitoring circuit not only for integrity verification but also for PUF usage and management of the key material derived therefrom. Thus, key handling becomes part of the monitoring circuit, which is therefore called Embedded Key Management System (EKMS). However, this is accompanied by a greater integration effort since it requires a corresponding system architecture, interface, and firmware to be well-integrated into the overall system. Altogether, this architecture completes my contributions to B-TREPID is it paves the way to its practical deployment in applications and thereby increases the system's overall technical readiness level.

In this chapter, I address this issue by discussing an adversary model, deriving design requirements, and presenting a hardened firmware architecture for PUF-based security enclosures. I present the complementing security extensions for FreeRTOS that enhance the operating system's security. To verify the concept's feasibility, the proposed system was implemented and its performance was evaluated. My results show that this security architecture for an EKMS can serve the basis for PUF-based HSMs.

I presented these results previously at the 7^{th} *Mediterranean Conference on Embedded Computing (MECO)* in 2018 and the corresponding paper is available in the conference proceedings [OHHS18]. This chapter is primarily based on the publication. The work was done in cooperation with Vincent Immler, who supported creating the technical concept and Matthias Hiller, who assisted in writing the paper. The development of the FreeRTOS extensions and the implementation of the architecture was done by Florian Hauschild in his bachelor's thesis.

8.1 Overview

While there are security architectures for HSMs and related secure devices, no such concept exists for PUF-based capacitive security enclosures. Hence, I investigated existing architectures at first to derive a starting point for my concept.

8.1.1 Related Work

Suh et al. presented the AEGIS concept for a secure processor architecture, which provides tamper-protected processing on a single-chip platform [SCG+14] that can also interface a PUF [SOSD05]. While this system is powerful and versatile, it is accompanied by high complexity and integration effort. The architecture requires an FPGA or ASIC and cannot be realized on a low-cost Commercial-Off-The-Shelf (COTS) microcontroller system. Furthermore, as the system does not use a PUF-based enclosure but an on-chip silicon PUF as key source, evaluation and integrity verification is less complex to realize technically. Compared to this architecture, my concept aims at less integration effort, a more modular concept by supporting multi-chip systems, but requires a more complex PUF evaluation and verification.

A related architecture based on a large-scale PUF-based security cover was proposed by Vai et al. who presented an optical waveguide PUF and a basic description of its corresponding security architecture [VNK+15, VWNU16]. Their concept includes a cryptographic coprocessor, operating on PUF data extracted from its security cover. Although the papers give an overview of their system, it spares firmware aspects, such as run time tamper detection, data isolation, and zeroization. All in all, firmware architectures for recently developed *PUF-based* security enclosures have not been discussed sufficiently in detail and no ready-to-use solutions exist.

8.1.2 Contributions

To overcome these limitations, I developed the EKMS concept for B-TREPID. This chapter presents the following contributions:

- A concept for generation, usage, and deletion of CSPs in applications using PUF-based security enclosures.
- A security-centered communication interface between the enclosure's host system and the EKMS.
- Hardening extensions for FreeRTOS to enhance operating system security, i.e., enhanced memory protection, isolation, task switching, and system calls.
- An implementation and evaluation of the EKMS on the measurement system's STM32F303 microcontroller, demonstrating the concept's feasibility.

8.1.3 Structure

Initially, the chapter discusses related work and presents my contributions in Section 8.1. Next, I focus on the adversary model in Section 8.2 and derive a corresponding security architecture. In Section 8.3, I present security extensions to FreeRTOS. Finally, I demonstrate the practical feasibility of the concept in Section 8.4 and summarize the results in Section 8.5.

8.2 Architecture of the Embedded Key Management System

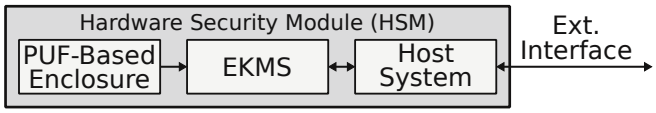


Figure 8.1: The basic architecture of the PUF-based HSM comprising the EKMS.

My modular HSM concept, shown in Figure 8.1, comprises four main components.

1. The **PUF-Based Enclosure** is the entropy source for key generation and serves as a tamper sensor.
2. The **EKMS** supervises the enclosure, generates the Key Encryption Key (KEK) from the PUF, and provides a cryptographic Application Programming Interface (API) to the host system.
3. The **Host System** is a high-performance device, such as an FPGA, which implements the high-level task of the HSM, e.g., high-performance decryption of large data sets.
4. The **External Interface** is the only data connection of the system to the outside and is used for communication.

8.2.1 System Overview and Comparison

Compared to a classical HSM, my concept comprises a PUF. Replacing the usual key storage with a PUF as root of trust usually requires a system redesign — a step, which may be infeasible for existing high-security applications. To overcome this issue, the EKMS provides a transparent interface to the host system by implementing a PUF abstraction layer. The host system does not require any knowledge about the PUF itself; every PUF-specific task is solely handled by the EKMS. Thereby, the underlying application will require only a little down to no modification and remains PUF-agnostic.

However, this is accompanied by non-negligible cost in terms of complexity. While the KEK is permanently available in current HSMs, PUF-based solutions need to

measure the PUF and derive the KEK first. This is a computationally expensive action, especially for a resource-limited embedded system. Additionally, the periodically executed enclosure monitoring generates a high system load continuously during runtime. The system has to react immediately to capacitance anomalies and similar integrity violations by triggering zeroization and alarm signals, implying a hard real-time requirement. The EKMS addresses all these increased demands.

8.2.2 Adversary Model

The HSM and especially the EKMS store various critical information, which must never fall into the hands of an adversary. This includes the raw measurement data, the extracted PUF data, the PUF key derived therefrom, and the CSPs. Adapted from the attack potential rating on smart cards [SOG13], I define the following two adversaries.

The **physical adversary** aims at extracting critical information from the system by penetrating the enclosure and by subsequently accessing internal components and signals. Attacks on the enclosure are carried out by drilling a hole of at least 300 µm in diameter, according to common certification requirements. The enclosure's trace layout was designed to be physically destroyed by such attacks; thus, such physical tampering causes an interruption of traces and a measurable change in capacitance. The EKMS must detect this alteration, it must raise alarm signals, and zeroization is to be performed within the predefined time limit.

The **interface adversary** tries to manipulate the system to extract critical information by exploiting weaknesses in the HSM's external communication interface. The attack aims at vulnerabilities, such as buffer overflows, which are then exploited to take over the host system and, in the next step, the EKMS. The adversary must be addressed by the EKMS's firmware, which has to implement multiple layers of data protection mechanisms.

8.2.3 EKMS Concept Overview

I designed the EKMS concept with respect to the adversary model. The system overview is shown in Figure 8.2.

The EKMS is connected to the Analog Front-End (AFE), which is the electrical interface to the PUF. The AFE is a circuit, either built from discrete components or an ASIC, that converts the enclosure's minuscule capacitance variations into a voltage signal, i.e., the analog part of the measurement system described in Chapter 7. This analog signal is digitized and processed by the EKMS's PUF signal processing module. It applies filtering algorithms, preprocessing steps, and prepares the raw data for the next processing step. Then, the KEK is derived from the PUF data, error correction is applied, and the KEK is written to the SRAM.

Deriving a KEK from the PUF secret allows the system to provide several different keys. An encrypted set of CSPs is stored in non-volatile memory, which is decrypted using the KEK. These keys can be of any number, type, and length, customized for the application, thereby providing high flexibility. They will be decrypted on-demand

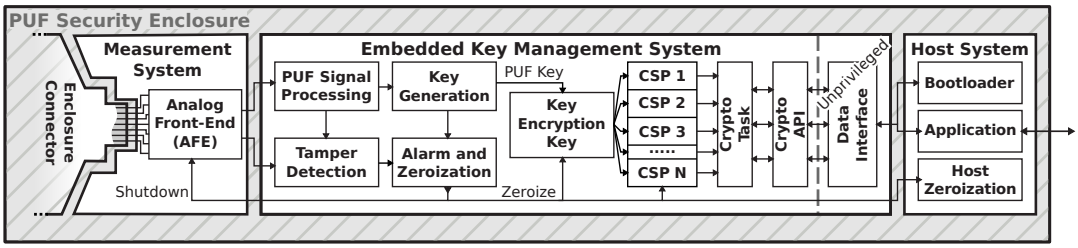


Figure 8.2: Modules of the PUF-based HSM and its EKMS.

when a cryptographic operation depending on this key is requested. All CSPs which are currently not in use remain encrypted and protected.

Attacks by the physical adversary are detected by the Tamper Detection module of the EKMS. For this purpose, the AFE provides additional data about the enclosure's integrity. The integrity and the raw PUF data are both continuously acquired and then processed by the Tamper Detection module. Anomalies in the measurement data, e.g., sudden changes in capacitance or an interrupted trace, will trigger the tamper detection. A processing failure in the PUF signal processing or key generation will also be redirected to the tamper detection module to take action. Upon a detected tamper event, an alarm is raised and the zeroization mechanism is triggered. The KEK and all decrypted CSPs will immediately be wiped from the EKMS. Additionally, the alarm signal is forwarded to the host system, which will carry out host-specific zeroization steps that depend on the actual application. These protection mechanisms provide effective means against physical adversaries.

Additionally, the EKMS is especially hardened against the interface adversary by three security barriers. The host system can only request operations depending on the CSPs, such as encrypt, decrypt, or signing. The EKMS will execute the operation and will return the result, but the system will never expose any key to the host system. Thus, taking over the host system will not give an adversary access to the EKMS key or PUF data.

8.2.4 Security Barriers

As the EKMS is not directly reachable from the outside, the host system serves as the first security barrier. However, the security of the host system depends on the application and its implementation is outside the control of the EKMS design. Thus, an attacker has to take over the host system first to gain access to the next security barrier.

Next, the interface adversary faces the second barrier as the EKMS's data interface must be exploited to gain deeper access. Depending on the application, such an interface has to perform complex data parsing and an adversary might be able to find vulnerabilities due to implementation weaknesses. To minimize the impact of such attacks, the EKMS's data interface runs in unprivileged mode and has only restricted

access to the system, i.e., it does not have the permission to access any decrypted CSPs or PUF data directly. Thus, the EKMS's data interface, running in unprivileged/user mode, introduces the second barrier for an adversary.

The system call (syscall/SVC) interface is the third barrier to the internal cryptographic API. The syscall mechanism enforces isolation between the data interface and CSPs. To execute a cryptographic operation depending on a CSP, the unprivileged task has to execute a system call. All key material can solely be accessed by the privileged worker task of the syscall API, which will then perform the operation. Thus, if adversaries want to gain access to key material, they need to find a third vulnerability in the system to exploit the syscall interface. Altogether, these three security boundaries aim at setting a high bar for the interface adversary.

8.2.5 Operating System and Tasks

In order to realize PUF-based key management with concurrent tamper detection, the EKMS requires a Real-Time Operating System (RTOS) supporting software and hardware security features. There are several systems available, e.g., F9, Zephyr, Mynewt, and FreeRTOS, which all have different capabilities and complexity. I decided for FreeRTOS since it is rather simple but flexible and has sufficient kernel and scheduling features. It is available for a wide range of processors, such as the Cortex-M4F and supports its MPU. Additionally, FreeRTOS targets high reliability, its code complies with MISRA-C, and it is actively maintained.

For the EKMS, FreeRTOS is configured with preemptive scheduling; thus, the operating system always executes the highest prioritized task. This is especially important for PUF measurement and tamper detection, as delaying this task impacts the reaction latency to tamper events and thereby affects security. To provide well-assessable scheduling, the priority of all tasks is strictly ordered and no tasks of equal priority exist.

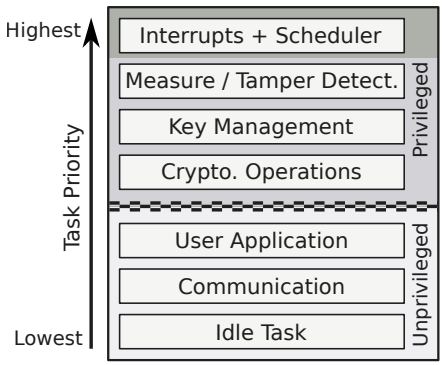


Figure 8.3: The EKMS task architecture divides the system into real-time tasks handling critical data and unprivileged user tasks.

The EKMS's functionality is implemented by the tasks listed in Figure 8.3. They are prioritized with respect to their time and security criticality. The interrupt handlers, including the periodic scheduler, have the highest priority as they orchestrate the system and guarantee the enforcement of real-time constraints. The next high priority item is the PUF measurement and tamper detection task, which performs the PUF readout, verifies the enclosure's integrity, and raises tamper alarms when required. The two other privileged tasks provide key management and perform operations for the cryptographic API.

Three unprivileged tasks run on the system. The first one implements the actual user application, e.g., collecting data, which is later on decrypted via the crypto API. The second one, which is the communication task, reads data from the data interface, parses the data packets, and forwards them to the user application. When no task is currently active, the idle task is run by default.

8.2.6 Alarm and Data Zeroization

When a tamper event is detected, the EKMS's alarm output signals to the host system is activated. Additionally, an EKMS internal zeroization mechanism is triggered, deleting all data in the system's SRAM. This operation can be entirely executed using solely the energy stored in several buffer capacitors, which serve as an independent but very limited power source during zeroization.

At first, all interrupts and the scheduler are disabled so that the zeroization mechanism runs uninterruptible at the highest priority. Next, the procedure switches off all unneeded internal and external modules, such as the AFE, to save power. Then, zeroization of the SRAM begins and the privileged data sections, containing the KEK and decrypted CSPs, are prioritized for deletion. Afterward, the CPU and FPU registers, potentially containing critical data, are cleared and the code continues to zeroize all remaining userspace data in the SRAM. Finally, no critical data is present in memory anymore. For performance and security reasons, zeroization is implemented in assembly code in order to gain low-level access to the system, i.e., to explicitly clear every single FPU register.

8.3 FreeRTOS Hardening

The EKMS firmware is based on FreeRTOS 10.0.1, which was the latest release in early 2018. Some extensions were added to obtain the security level defined in Section 8.2.

8.3.1 Data Isolation Enforcement

The Cortex-M4F features an MPU which controls the CPU's access privileges to memories, such as the flash, SRAM, and peripherals [ARM10]. Access permissions can be defined independently for up to eight address regions. Each region can be configured to be inaccessible, read-only, or read-writable. This can be set for privileged

and unprivileged tasks individually, e.g., memory regions containing critical data are restricted to privileged-only access. Furthermore, each region can be configured to be either executable or non-executable.

FreeRTOS supports the MPU, but its default configuration is suboptimal in terms of security. FreeRTOS requires the developer to explicitly tag functions and data which shall be privileged-only accessible. Unfortunately, everything else, not marked intentionally or by error, is accessible by unprivileged tasks by default. This creates a high risk for configuration errors since accidentally leaving critical data unprivileged-accessible will expose it. Such an error is not visible immediately since the system will still operate correctly.

Hence, I turned the FreeRTOS MPU configuration upside down. Instead of tagging critical data and functions, the entire address space is set to privileged-only accessible by default. If functions or data shall be accessed by unprivileged tasks, they must be tagged explicitly. Forgetting to enable unprivileged access to data will *not* expose this data but will crash the system immediately in most cases due to a memory management fault. Such an error is comparably quick to spot during testing and otherwise results only in limited functionality instead of exposing critical data.

Beside that, I follow the `W^X` concept which states that memory regions must *either* be writable *or* executable. Thus, I configure the SRAM as writable but set its Execute-Never bit. For the firmware memories, i.e., Flash and CCMRAM, I chose the inverse setting, which allows execution but prevents write accesses. This impedes adversaries, as code that has been injected into the SRAM is not executable. Together with strict MPU settings, the attack surface for alternative attacks, such as Return Oriented Programming (ROP), is also limited because a smaller accessible address space makes finding gadgets more difficult. Altogether, this concept configures the MPU more strictly and provides a higher level of security at practically no overhead during runtime as access control is solely handled in hardware.

8.3.2 Syscall Interface Hardening

Due to the aforementioned configuration, the MPU inhibits access from unprivileged functions to kernel functions and data. However, an unprivileged task must still be able to call *selected* kernel functions. Such functionality is implemented via system calls, a feature that is already present in FreeRTOS. Unfortunately, the already existing interface does not provide strong security as it breaks the strict separation between unprivileged and privileged data and tasks.

The original syscall concept of FreeRTOS is depicted in the upper region of Figure 8.4. Prior to accessing a kernel function, the task performs a `portSVCRAISE_PRIVILEGE` syscall. The kernel raises the task's privilege and it becomes a privileged task. Then, the calling task performs the privileged operation and accesses kernel functions and data as required. After the operations have been finished, *the task* has to explicitly drop its privileges to return to its initial unprivileged level. The original concept assumes that every unprivileged task is fully trustworthy; however, I drop this assumption to limit the power of a compromised unprivileged task. If adversaries gain access to an

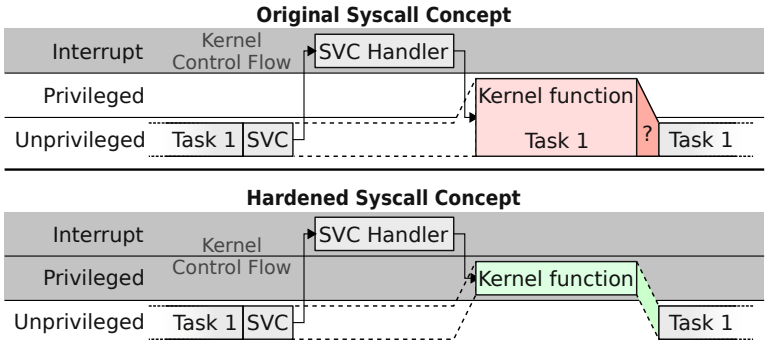


Figure 8.4: Comparison between the original and the hardened syscall interface.

unprivileged task, they could obtain privileges easily and all critical data is immediately exposed; thus, security is broken.

To counteract this issue, I redesigned a major part of the syscall control flow, shown in the lower region of Figure 8.4. When an unprivileged task executes a system call to a kernel function, the privileges of the task are still raised. But instead of handing the control back to the task, the syscall handler performs a direct return into the requested kernel function. After the kernel function has finished execution, privileges are *automatically* dropped and control is given back to the calling task. Thus, only the requested system call is executed and the calling task never gains privileged control over the system. This prevents privilege escalation without limiting functionality.

The hardened concept does not impact the real-time behavior of the system. The syscall interrupt handler calls the kernel function by return, thus, the high prioritized interrupt is left immediately and the kernel function is executed with the calling task's priority, i.e., preemption is possible. The *call by return* mechanism is implemented by modifying the SVC Handler's return address to point directly to the requested kernel function. Thereby, system call performance is increased by reducing the number of required function calls. Altogether, the hardened system call concept provides secure access to kernel functions without impeding real-time behavior.

8.3.3 Task Switch FPU Isolation

Each time a task switch is performed, the scheduler saves the current task's state and loads the state of the next task. However, the FPU context, consisting of all FPU registers, is not saved by default to speed up task switching. It will only be saved and restored for tasks that are using the FPU.

This behavior causes critical data leakage between tasks when switching from an FPU to a non-FPU task, as explained in the following example. The PUF measurement task heavily operates on the FPU during signal processing. After the task has finished, the scheduler switches to a task not using the FPU, e.g., the highly exposed communication task. Since no FPU context is restored for the communication

task, secret measurement data remains present in the FPU registers when this task is being executed. While this is acceptable from a functional point of view, leaking data through FPU registers is unacceptable for a security critical system. This issue could be solved by saving and restoring the FPU context on *every* task switch precautionarily. However, system performance is strongly impacted as $(32 + 1) \cdot 4 = 132$ additional bytes, corresponding to 32 FPU registers and 1 status register, are written to and loaded from SRAM during each task switch.

Our FreeRTOS extension addresses this issue by enforcing FPU isolation during task switching with very little overhead. The task scheduler checks whether the previously executed task *has* an active FPU context and the next task *has no* FPU context — which is the only case that may cause data leakage. In this special case, the scheduler resets all FPU registers to zero during the task switch. Otherwise, which is the common case, the resetting operation will be skipped at negligible overhead. Thus, the extension enforces FPU data isolation at minimal cost and without impeding real-time requirements.

8.4 Practical Implementation

The concept was implemented on the measurement system, which serves as a development platform for the EKMS. The platform features an AFE, built from discrete components, and an STM32F303 microcontroller for the EKMS. The host system is not present but is emulated by a computer system that is connected to the EKMS's communication interface.

I tested the system's security through test cases validating the FreeRTOS security enhancements. Data isolation via the MPU works as expected. An access of an unprivileged function to privileged data or device peripherals fails with a memory management fault and zeroization of the system is successfully triggered. Next, the new system call interface was implemented and tested. Trying to bypass the interface, e.g., by directly calling the kernel function, triggers a memory management fault and zeroization, as designed. Further tests showed that the FPU no longer leaks data between tasks; thus, all tasks can securely use the FPU.

I verified the concept's claims in practice by testing the system in an HSM-like use case. When the EKMS is powered up, the firmware verifies the enclosure's integrity and extracts its PUF. Next, the host system requests the derivation of the KEK, followed by the decryption of CSP 1, which contains another user-defined key. Then, the host system transmits one data packet which is successfully decrypted via CSP 1. The data is then transmitted back to the host, who verifies that the data was correctly decrypted. Tampering with the security enclosure causes the system to detect an intrusion and zeroization is triggered successfully. Thereby I show that my EKMS concept is a viable basis to offer PUF-based cryptographic functions to a host system through a PUF-agnostic interface while maintaining security.

The modifications of FreeRTOS did not impact the real-time constraints. I measured the overhead on the real system, running at its designated maximum CPU clock

frequency of 72 MHz. The data isolation enforcement does not impact the system's performance at all, as only the MPU configuration has been updated, which is transparent for the running software and handled in hardware. The overhead of the hardened system call mechanism is less than 2.8 µs, corresponding to about 200 clock cycles, per system call. The FPU data isolation increases the task switch time by 0.042 µs, which equals 3 clock cycles, in the standard case when no FPU clearing is required. If the FPU has to be cleared, the task switch time increases by 0.542 µs, which is still only 39 clock cycles. All in all, the overhead in the range of a few microseconds is negligible with respect to the gain in security and does not impact real-time constraints.

8.5 Conclusion

In this chapter, I completed the B-TREPID concept with the software architecture of the EKMS. First, I presented an adversary model and created a corresponding system concept. Next, I explained the EKMS's role in handling PUF data, performing cryptographic operations, and providing a PUF abstraction layer to the host system. Afterward, security extensions were developed for FreeRTOS since a real-time operating system providing high security is required. Finally, I verified the concept on the measurement system prototype platform successfully, demonstrating the EKMS's feasibility. Thus, the concept was not only theoretically proposed but verified in practice. Altogether, the EKMS is another cornerstone in raising B-TREPID's technology readiness level.

Chapter 9

Conclusion and Outlook

In this dissertation, I presented the way from an analysis of embedded software and hardware security to a novel concept for PUF-based intrinsic embedded system security. The insight into embedded security was given from the attacker's and from the system developer's perspective. I started with an analysis of embedded system security ranging from IoT device network security over clock glitch attacks and continued the topic up to semi-invasive hardware attacks and vulnerabilities in debug interfaces. Following the attacks, physical mitigation techniques on a system-level were discussed. This part of the dissertation presented different solutions for physical tamper protection by security enclosures. Afterward, my dissertation introduced B-TREPID, a PUF-based security enclosure that exploits variation in capacitive coupling between traces comprised in the enclosure. My following work focused on simulating the enclosure's electrical properties via a FEA model and especially on measuring the capacitance variation in the range of only a few femtofarads. I completed my work by creating the EKMS, which allows integrating B-TREPID into embedded systems by providing a PUF abstraction layer implemented via a cryptographic API.

9.1 Contributions of my Thesis

Initially, in Chapter 2, I discovered severe conceptual security design flaws in several selected IoT devices by the example of cloud-based camera systems. Apart from this theoretical analysis, I demonstrated their exploitability in practice, thereby underlining the possible impact of such issues. My contributions primarily include:

- An exemplary analysis of IoT security on the basis of cloud-based video surveillance systems.
- An attacker model tailored to cloud-based video surveillance systems, in particular.
- An analysis of the network security of these devices, discovering several weaknesses.
- A practical verification proving the discovered vulnerabilities.

- A discussion of common design issues and anti-patterns discovered across different manufacturers.

But even if software vulnerabilities are resolved, devices may still be subjected to hardware attacks that target the underlying hardware. Thus, in Chapter 3, I presented the fuzzy clock glitch attack, which has shown to be able to inject faults into the tested embedded system leading to an early exit from loops and computation errors. My contributions include in particular:

- A novel concept to create a clock signal glitch via two length-adjustable ring oscillators and an XOR gate.
- Use of the random and non-deterministic fuzzy clock glitch signal instead of an exactly timed glitch for security testing.
- An implementation of the concept on a Spartan-3E FPGA platform.
- A demonstration of a successful fuzzy clock glitch attack on the state-of-the-art ARM Cortex-M0 based microcontroller STM32F030.

However, even if sufficient security in software and resilience against hardware attacks is assumed, an embedded system might still be affected by further vulnerabilities in its physical interfaces. To demonstrate this, I performed an extensive security analysis on the programming and debug interface of the STM32F0 microcontroller series. This analysis including PoCs is presented in Chapter 4 and comprises:

- A conceptual analysis of three STM32F0 security core components.
- The discovery of three weaknesses therein and concepts for their exploitation, i.e.,
  - **Cold-Boot Stepping:** Enforcement of single-stepping under limited debugging capabilities allowing firmware extraction under special conditions.
  - **Security Downgrade:** A semi-invasive attack leveraging a lock-level design issue to downgrade the firmware protection setting.
  - **Debug Interface Exploit:** Non-invasive exploitation of a race condition in the debug interface allowing reading out the microcontroller's protected firmware.
- A discussion of the impact of each vulnerability on overall system security and possible countermeasures.

Hence, my results show that embedded security is threatened on multiple levels and several of the discovered vulnerabilities might be outside the control of the system developer. For this reason, I focused on security enclosures and covers for physical security, which counteract such attacks by preventing physical access to the device, and presented an analysis of various solutions for device-level physical embedded system security in Chapter 5. This is presented via:

- An analysis of **past** trace resistance-based physical security solutions, including a detailed dismantling and evaluation of their monitoring circuits.
- An overview of **present** PUF-based solutions that are available or in development, including a discussion of basic concepts, advantages, and potential weaknesses.
- A motivation for **future** developments towards PUF-based security enclosures.

PUF-based physical security enclosures make use of a promising technology. My work focused especially on B-TREPID which is a capacitive PUF-based solution that is introduced in Chapter 6 and includes the following contributions:

- An explanation of the concept from the measurement point of view.
- The general concept with a focus on capacitive properties.
- A FEA of the proposed enclosure focusing on optimizations and enclosure material properties.

The proposed enclosure is based on capacitance variations in the femtofarad range and comprises an additional integrity verification mechanism. However, the measurement task has proven to be very demanding; thus, it is one of the main topics of my dissertation and is explained and discussed in Chapter 7. Therein, I presented the following contributions including:

- A differential measurement concept for capacitive PUF-based security enclosures whose parasitic capacitances are orders of magnitude larger than the PUF's variation.
- A scalable measurement concept enabling enclosure integrity verification and capacitive PUF measurement in the range of milliseconds.
- A proof of concept implementation of the proposed system and a practical verification by means of enclosure prototypes.
- A practical evaluation of an electrode probing attack on the security enclosure and a discussion of countermeasures.

Since the integration of a PUF into embedded systems can be effortful, I developed and presented the EKMS in Chapter 8. This system eases system integration of B-TREPID and completes the concept. I presented the following contributions to the system integration aspect, namely:

- A concept for generation, usage, and deletion of CSPs in applications based on PUF-based security enclosures.
- A security-centered communication interface between the enclosure's host system and the EKMS.
- Hardening extensions for FreeRTOS to enhance operating system security, i.e., enhanced memory protection, isolation, task switching, and system calls.
- An implementation and evaluation of the EKMS on the measurement system's STM32F303 microcontroller, demonstrating the concept's feasibility.

Altogether, my results present an overview of embedded security from the attacker's and the developer's perspective. They comprise several novel attack approaches and their practical demonstration on state-of-the-art systems as of 2018. My attacks severely affect the security of these devices and, in some cases, entirely nullify it. My research results concerning B-TREPID present one way to prevent most of the previously presented attacks. The newly developed concept creates a cornerstone for novel capacitive PUF-based security enclosures providing intrinsic security. Since my work takes system integration into account, the concept takes a leap forward closely to a ready-to-use solution.

Thus, PUF-based security enclosures are a promising candidate for the protection of future HSMs and related high-assurance solutions that rely on system-level protection. This especially includes automotive embedded systems, military applications, up to datacenter usage.

9.2 Future Work

Despite all progress, the B-TREPID's Technology Readiness Level (TRL) is currently insufficient for commercial use. As a consequence, future work focuses on raising the TRL to 6, corresponding to a functional prototype in its designated environment. This work includes especially topics around PUF processing, such as key derivation and error correction. Hardware topics, e.g., improvement of the measurement noise and circuit size, are further aspects that are under consideration. Additionally, the enclosure is scheduled to undergo further improvements that shall increase its mechanical flexibility and lower its production cost.

Regarding PUF processing, the evaluation algorithms have to face several challenges. First, environmental compensation is to be further analyzed as fluctuations in temperature and humidity affect the PUF and introduce offsets. Algorithms must take this into account to differentiate between the effects of an attack and unperilous environmental influences. Additionally, the algorithms must not exhibit tamper-compensating properties such as an overly strong error-correction that would conceal the effect of tampering.

The current measurement system's primary purpose was to prove the basic functionality of the overall PUF-based system. Despite this goal was more than reached and the system is able to provide a powerful measurement and development platform, some further steps are necessary for commercial use. A primary concern is the circuit's size; thus, the analog circuitry has to be analyzed for possible optimizations for the next revision. Furthermore, optimizations could also include noise minimization, which could reduce the currently required measurement time even further. In the long term, the integration of this measurement system into an ASIC might be a viable way to improve all these aspects.

On the level of the physical enclosure, optimizations are ongoing and require further effort. The presented layer stack was sufficient to prove the concept; however, its mechanical flexibility requires improvement. Enclosure samples with more sophisticated

materials such as flexible carbon or other conductive pasts have to be created and evaluated. This is especially important concerning my results that showed that probing of the enclosure might be technically possible. However, there are more open questions, for example, the influence of potting materials on the enclosure is yet to be investigated.

My work only considered *capacitive* PUF-based security enclosures. For other applications, alternative technologies such an inductive, magnetic, or optical PUF might be adequate. Nevertheless, the solutions presented in this thesis might still apply to these systems, e.g., the integration concept provided by the EKMS.

In the following years, attack tools, as well as security enclosures, will advance. These developments will set the agenda for future work regarding security enclosures and further PUF-based solutions.

Bibliography

[AAB+17] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. Understanding the mirai botnet. In *26th USENIX Security Symposium (USENIX Security 17)*, Vancouver, BC. USENIX Association, 2017.

[AB17] Lillian Ablon and Andy Bogart. *Zero Days, Thousands of Nights: The Life and Times of Zero-Day Vulnerabilities and Their Exploits*. Rand Corporation, 2017.

[ABC+12] Todd W. Arnold, Carl Buscaglia, Felix Chan, Vincenzo Condorelli, John Dayka, William Santiago-Fernandez, Nihad Hadzic, Michael D. Hocker, Michael Jordan, Thomas E. Morris, and Klaus Werner. IBM 4765 cryptographic coprocessor. *IBM Journal of Research and Development*, 2012.

[ABF+03] Christian Aumüller, Peter Bier, Wieland Fischer, Peter Hofreiter, and Jean-Pierre Seifert. Fault attacks on RSA with CRT: Concrete results and practical countermeasures. In *Cryptographic Hardware and Embedded Systems (CHES)*. Springer Berlin Heidelberg, 2003.

[Abi06] Asad A. Abidi. Phase noise and jitter in CMOS ring oscillators. *IEEE Journal of Solid-State Circuits*, 2006.

[ABP+13] Nadhem AlFardan, Daniel J. Bernstein, Kenneth G. Paterson, Bertram Poettering, and Jacob C. N. Schuldt. On the security of RC4 in TLS. In *22nd USENIX Security Symposium*, Washington, D.C. USENIX Assosciation, 2013.

[ADN+10] Michel Agoyan, Jean-Max Dutertre, David Naccache, Bruno Robisson, and Assia Tria. When clocks fail: On critical paths and clock faults. In *International Conference on Smart Card Research and Advanced Applications*, 2010.

[Ana14] Analog Devices. *ADG708/ADG709 datasheet*, 2014.

[Ana16] Analog Devices. *ADG706/ADG707 datasheet*, 2016.

[Ana18] Analog Devices. LTspice. `https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html`, 2018.

[And10] Ross J. Anderson. *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons, 2010.

[ARM06] ARM Limited. *AMBA 3 AHB-Lite Protocol Specification v1.0*, 2006.

[ARM09a] ARM Limited. *CoreSight Components Technical Reference Manual*, 2009.

[ARM09b] ARM Limited. *Cortex-M0 Technical Reference Manual*, 2009.

[ARM10] ARM Limited. *Cortex-M4 Technical Reference Manual*, 2010.

[Bax97] Larry K. Baxter. *Capacitive Sensors: Design and Applications*. Wiley-IEEE Press, 1997.

[BCMV03] Roberto Bez, Emilio Camerlenghi, Alberto Modelli, and Angelo Visconti. Introduction to flash memory. *Proceedings of the IEEE*, 2003.

[Bea17] Antoine Beaupré. A comparison of cryptographic keycards. `https://lwn.net/Articles/736231/`, 2017.

[BECN+06] Hagai Bar-El, Hamid Choukri, David Naccache, Michael Tunstall, and Claire Whelan. The sorcerer's apprentice guide to fault attacks. *Proceedings of the IEEE*, 2006.

[BGV11] Josep Balasch, Benedikt Gierlichs, and Ingrid Verbauwhede. An in-depth and black-box characterization of the effects of clock glitches on 8-bit MCUs. In *2011 Workshop on Fault Diagnosis and Tolerance in Cryptography*, 2011.

[BOU07] BOURNS INC. Application note – security housing, 2007. `http://application-notes.digchip.com/176/176-48205.pdf`.

[BQ04] Ray Burke and Carl Queen. Patent EP1462907 – A security enclosure for a circuit, 2004.

[Bri04] Gary A. Brist. Design optimization of single-ended and differential impedance PCB transmission lines. In *PCB West Conference Proceedings*, 2004.

[Cor12] Linear Technology Corporation. *LTC6252 / LTC6253 / LTC6254 datasheet*, 2012.

[CT05] Hamid Choukri and Michael Tunstall. Round reduction using faults. *FDTC*, 2005.

[CZF+14] Andrei Costin, Jonas Zaddach, Aurélien Francillon, Davide Balzarotti, and Sophia Antipolis. A large-scale analysis of the security of embedded firmwares. In *USENIX Security Symposium*. USENIX Association, 2014.

[CZF16] Andrei Costin, Apostolis Zarras, and Aurélien Francillon. Automated dynamic firmware analysis at scale: A case study on embedded web interfaces. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '16, Xi'an, China. ACM, 2016.

[Del15] Deloitte & Technische Universität München. Ready for takeoff? Smart Home aus Konsumentensicht, `http://www.connected-living.org/content/4-information/5-downloads/4-studien/8-ready-for-takeoff/deloitte-smart-home-consumer-survey-20150701.pdf`, 2015.

[DMA08] Saar Drimer, Steven J. Murdoch, and Ross Anderson. Thinking inside the box: system-level failures of tamper proofing. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*. IEEE, 2008.

[DTB+15] Rémy Druyer, Lionel Torres, Pascal Benoit, Paul-Vincent Bonzom, and Patrick Le-Quéré. A survey on security features in modern FPGAs. In *2015 10th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC)*, 2015.

[EFK+12] Thomas Esbach, Walter Fumy, Olga Kulikovska, Dominik Merli, Dieter Schuster, and Frederic Stumpf. A new security architecture for smartcards utilizing PUFs. In *ISSE Conference*, 2012.

[EHA+15] Chris Evans, Ben Hawkes, Heather Adkins, Matt Moore, Michal Zalewski, and Gerhard Eschelbeck. Feedback and data-driven updates to google's disclosure policy. `https://googleprojectzero.blogspot.com/2015/02/feedback-and-data-driven-updates-to.html`, 2015.

[Eil90] Edwards S. Eilley. Patent US4891609 – Ring oscillator, 1990.

[ES05] Halit Eren and Lucas D. Sandor. Fringe-effect capacitive proximity sensors for tamper proof enclosures. In *Sensors for Industry Conference*, 2005.

[ESH+11] Sho Endo, Takeshi Sugawara, Naofumi Homma, Takafumi Aoki, and Akashi Satoh. An on-chip glitchy-clock generator for testing fault injection attacks. *Journal of Cryptographic Engineering*, 2011.

[Eur16] European Union. Regulation (EU) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/EC (general data protection regulation). https://eur-lex.europa.eu/eli/reg/2016/679/oj, 2016.

[FBBV08] Viktor Fischer, Florent Bernard, Nathalie Bochard, and Michal Varchola. Enhancing security of ring oscillator-based TRNG implemented in FPGA. In *2008 International Conference on Field Programmable Logic and Applications*, 2008.

[FCC12] FCCID.io. FCC ID WHLIP229. https://fccid.io/WHLIP229/, 2012.

[FWKM15] Muhammad Umar Farooq, Muhammad Waseem, Anjum Khairi, and Sadia Mazhar. A critical analysis on the security concerns of internet of things (IoT). *International Journal of Computer Applications*, 2015.

[GC15] Ana Gainaru and Franck Cappello. *Errors and Faults*. Springer International Publishing, 2015.

[GCDD02] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Silicon physical random functions. In *ACM CCS*, 2002.

[GfK15] GfK. Smart home beats wearables for impact on lives, say consumers, http://www.gfk.com/fileadmin/user_upload/dyna_content_import/2015-11-24_press_releases/data/Documents/Press-Releases/2015/2015-11-11_smart-home_press-release_global.pdf, 2015.

[Goe58] Gerald Goertzel. An algorithm for the evaluation of finite trigonometric series. In *The American Mathematical Monthly*, Volume 65. JSTOR, 1958.

[Goo19] Google. Google trends for IoT, https://trends.google.com/trends/explore?date=2010-01-01%202018-05-01&geo=US&q=iot, 2019.

[GOR07] W. L. GORE & Associates Inc. GORE tamper respondent surface enclosure (commercial brochure), 2007.

[GPdM15] Christina Garman, Kenneth G. Paterson, and Thyla Van der Merwe. Attacks only get better: Password recovery attacks against RC4 in TLS. In *24th USENIX Security Symposium (USENIX Security 15)*, Washington, D.C. USENIX Association, 2015.

[Gra11] Grain Media. *Grain Media GM8128/8126/8125*, 2011.

[Gre15] Matthew Green. Attack of the week: FREAK (or 'factoring the NSA for fun and profit'), `http://blog.cryptographyengineering.com/2015/03/attack-of-week-freak-or-factoring-nsa.html`, 2015.

[Hil16] Matthias Hiller. Key derivation with physical unclonable functions. Dissertation, Technische Universität München, München, 2016.

[HMH+12] Johann Heyszl, Stefan Mangard, Benedikt Heinz, Frederic Stumpf, and Georg Sigl. Localized electromagnetic analysis of cryptographic implementations. In *Topics in Cryptology – CT-RSA 2012*. Springer Berlin Heidelberg, 2012.

[HNT+13] Clemens Helfmeier, Dmitry Nedospasov, Christopher Tarnovsky, Jan Starbug Krissler, Christian Boit, and Jean-Pierre Seifert. Breaking and entering through the silicon. In *ACM Conference on Computer and Communications Security (CCS)*, 2013.

[HSH+09] J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten. Lest we remember: Cold-boot attacks on encryption keys. *Communications of the ACM*, 2009.

[HSZF13] Maxim Hennig, Oliver Schimmel, Philipp Zieris, and Bartol Filipovic. Patent DE102013205729 – Vorrichtung und verfahren mit einem träger mit schaltungsstrukturen, 2013.

[HSZS13] Maxim Hennig, Oliver Schimmel, Philipp Zieris, and Georg Sigl. Manipulationssensible Kopierschutzfolie. *D A CH Security*, 2013.

[HYKD14] Charles Herder, Mandel Yu, Farinaz Koushanfar, and Srinivas Devadas. Physical unclonable functions and applications: A tutorial. *Proceedings of the IEEE*, 2014.

[HYKS10] Yohei Hori, Takahiro Yoshida, Toshihiro Katashita, and Akashi Satoh. Quantitative and statistical performance evaluation of arbiter physical unclonable functions on fpgas. In *2010 International Conference on Reconfigurable Computing and FPGAs*. IEEE, 2010.

[IBM12] IBM. IBM 4765 cryptographic coprocessor security module security policy (compliant to FIPS 140-2 level 4), 2012. `https://csrc.nist.gov/CSRC/media/projects/cryptographic-module-validation-program/documents/security-policies/140sp1505.pdf`.

[IBM16] IBM. *4767 PCIe Cryptographic Coprocessor Installation Manual*, `https://www-03.ibm.com/security/cryptocards/pciecc2/pdf/4767install.pdf`, 2016.

[Ign16] Valentina Ignat. Patent and product piracy. *IOP Conference Series: Materials Science and Engineering*, 2016.

[IHKS16] Vincent Immler, Maxim Hennig, Ludwig Kürzinger, and Georg Sigl. Practical aspects of quantization and tamper-sensitivity for physically obfuscated keys. In *Workshop on Cryptography and Security in Computing Systems (CS2)*. ACM, 2016.

[IHL+17] Vincent Immler, Matthias Hiller, Qinzhi Liu, Andreas Lenz, and Antonia Wachter-Zeh. Variable-length bit mapping and error-correcting codes for higher-order alphabet PUFs. In *Security, Privacy, and Applied Cryptography Engineering (SPACE)*, 2017.

[IHOS17] Vincent Immler, Matthias Hiller, Johannes Obermaier, and Georg Sigl. Take a moment and have some t: Hypothesis testing on raw PUF data. In *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2017.

[IMJFC13] Phil Isaacs, Thomas Morris Jr, Michael J. Fisher, and Keith Cuthbert. Tamper proof, tamper evident encryption technology. In *Pan Pacific Symposium*, 2013.

[Imm19] Vincent Immler. Higher order alphabet physical unclonable functions: Constructions, properties, and applications. Dissertation, Technische Universität München, München, 2019.

[IO19] Vincent Immler and Johannes Obermaier. Patent EP355046 – PUF-film and method for producing the same, 2019.

[IOK+18a] Vincent Immler, Johannes Obermaier, Martin König, Matthias Hiller, and Georg Sigl. B-TREPID: batteryless tamper-resistant envelope with a PUF and integrity detection. In *2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2018.

[IOK+18b] Vincent Immler, Johannes Obermaier, Martin König, Matthias Hiller, and Georg Sigl. Next-generation anti-tamper envelopes for cyber physical defense systems. In *STO-MP-SCI-300*, 2018.

[ION+18] Vincent Immler, Johannes Obermaier, Kuan Kuan NG, Fei Xiang KE, Jin Yu LEE, Yak Peng LIM, Wei Koon OH, Keng Hoong WEE, and Georg Sigl. Secure physical enclosures from covers with tamper-resistance. *IACR Transactions on Cryptographic Hardware and Embedded Systems (CHES)*, 2018.

[Jer11] Jerzmacow. Verifone Vx570 payment terminal teardown. `https://www.instructables.com/id/Verifone-Vx570-Payment-terminal-teardown/`, 2011.

[Jer17] James A. Jerkins. Motivating a market or regulatory solution to IoT insecurity with the mirai botnet code. In *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, 2017.

[Jon14] David L. Jones. EEVblog #687 – EFTPOS PIN pad terminal teardown. `https://www.youtube.com/watch?v=tCgtTPwlDSo`, 2014.

[Jon16] David L. Jones. EEVblog #942 – mystery monday teardown. `https://www.youtube.com/watch?v=IpdJEo9r-HQ`, 2016.

[Jun05] Walter G. Jung. *Op Amp applications handbook*. Newnes, 2005.

[Key18] Keysight Technologies. *Infiniium Oscilloscope Probes and Accessories*, 2018.

[KH14] Thomas Korak and Michael Hoefler. On the effects of clock and power supply tampering on two microcontroller platforms. In *2014 Workshop on Fault Diagnosis and Tolerance in Cryptography*, 2014.

[KHEB14] Thomas Korak, Michael Hutter, Baris Ege, and Lejla Batina. Clock glitch attacks in the presence of heating. In *2014 Workshop on Fault Diagnosis and Tolerance in Cryptography*, 2014.

[KiC18] KiCad EDA. A cross platform and open source electronics design automation suite, `https://www.kicad-pcb.org/`, 2018.

[KII02] Chandrasekharan Kothandaraman, S. Sundar Kumar Iyer, and Subramanian S. Iyer. Electrically programmable fuse (eFUSE) using electromigration in silicides. *IEEE Electron Device Letters*, 2002.

[KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Advances in Cryptology — CRYPTO' 99*. Springer Berlin Heidelberg, 1999.

[KKSV17] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. DDoS in the IoT: Mirai and other botnets. *Computer*, 2017.

[KLMR04] Paul Kocher, Ruby Lee, Gary McGraw, and Anand Raghunathan. Security as a new dimension in embedded system design. In *Proceedings of the 41st Annual Design Automation Conference*, DAC '04, San Diego, CA, USA. ACM, 2004.

[KLR08] Wolfgang Killmann and Kerstin Lemke-Rust. Common criteria protection profile – cryptographic modules, security level "enhanced", 2008.

[Kre16] Brian Krebs. New mirai worm knocks 900k germans offline. `https://krebsonsecurity.com/2016/11/new-mirai-worm-knocks-900k-germans-offline/`, 2016.

[KS67] D. Kahng and S. M. Sze. A floating gate and its application to memory devices. *The Bell System Technical Journal*, 1967.

[kw718] kw71. Shedding too much light on a microcontroller's firmware protection (2017) (fraunhofer.de). `https://news.ycombinator.com/item?id=17587673`, 2018.

[Lai11] William Lai. Characteristics of dielectric elastomers and fabrication of dielectric elastomer actuators for artificial muscle applications. *Digital Repository @ Iowa State University*, 2011.

[LLX+17] Zhen Ling, Junzhou Luo, Yiling Xu, Chao Gao, Kui Wu, and Xinwen Fu. Security vulnerabilities of internet of things: A case study of the smart plug system. *IEEE Internet of Things Journal*, 2017.

[LLX+18] Zhen Ling, Kaizheng Liu, Yiling Xu, Chao Gao, Yier Jin, Cliff Zou, Xinwen Fu, and Wei Zhao. IoT security: An end-to-end view and case study. *arXiv preprint arXiv:1805.05853*, 2018.

[Lyo04] Richards Lyons. Another contender in the arctangent race. *IEEE Signal Processing Magazine*, 2004.

[Man18] Art Manion. Vulnerability disclosure policy. `https://vuls.cert.org/confluence/display/Wiki/Vulnerability+Disclosure+Policy`, 2018.

[Mas13] Mastermod. Merkur datenbank beschreibung. `https://www.youtube.com/watch?v=JdgPxkUl99c`, 2013.

[Max81] James Clerk Maxwell. *A treatise on electricity and magnetism*, Volume 1. Clarendon press, 1881.

[MCMS10] Abhranil Maiti, Jeff Casarona, Luke McHale, and Patrick Schaumont. A large scale characterization of RO-PUF. In *2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2010.

[MDH+13] N. Moro, A. Dehbaoui, K. Heydemann, B. Robisson, and E. Encrenaz. Electromagnetic fault injection: Towards a fault model on a 32-bit microcontroller. In *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography*, 2013.

[Med12] MediaTek. MT7620 – 2T2R 802.11n platform for N300/AC750/AC1200 router and repeater. `https://www.mediatek.com/products/homeNetworking/mt7620n-a`, 2012.

[Med16] Aref Meddeb. Internet of things standards: who stands out from the crowd? *IEEE Communications Magazine*, 2016.

[MGS13] Abhranil Maiti, Vikash Gunreddy, and Patrick Schaumont. A systematic method to evaluate and compare the performance of physical unclonable functions. In *Embedded systems design with FPGAs*. Springer, 2013.

[Mil08] George Milad. Surface finishes in a lead-free world. *Circuit World*, 2008.

[MIT18] MITRE. CVE-2017-18347, `https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-18347`, 2018.

[MR04] Jelena Mirkovic and Peter Reiher. A taxonomy of DDoS attack and DDoS defense mechanisms. *SIGCOMM Comput. Commun. Rev.*, 2004.

[MSI16] Masato Matsubayashi, Akashi Satoh, and Jun Ishii. Clock glitch generator on SAKURA-G for fault injection attack against a cryptographic circuit. In *2016 IEEE 5th Global Conference on Consumer Electronics*, 2016.

[MSSS11a] Dominik Merli, Dieter Schuster, Frederic Stumpf, and Georg Sigl. Semi-invasive EM attack on FPGA RO PUFs and countermeasures. In *Proceedings of the Workshop on Embedded Systems Security*. ACM, 2011.

[MSSS11b] Dominik Merli, Dieter Schuster, Frederic Stumpf, and Georg Sigl. Side-channel analysis of PUFs and fuzzy extractors. In *Trust and Trustworthy Computing*. Springer Berlin Heidelberg, 2011.

[MV10] Roel Maes and Ingrid Verbauwhede. *Physically Unclonable Functions: A Study on the State of the Art and Future Research Directions*. Springer Berlin Heidelberg, 2010.

[MvRPG90] A. C. Metting van Rijn, A. Peper, and C. A. Grimbergen. High-quality recording of bioelectric events. *Medical and Biological Engineering and Computing*, 1990.

[MYAZ15] Rwan Mahmoud, Tasneem Yousuf, Fadi Aloul, and Imran Zualkernan. Internet of things (IoT) security: Current status, challenges and prospective measures. In *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, 2015.

[NIS01a] NIST. FIPS PUB 140-2, security requirements for cryptographic modules, 2001.

[NIS01b] NIST. FIPS PUB 197, specification for the advanced encryption standard (AES), 2001.

[NIS15] NIST. FIPS PUB 180-4, secure hash standard (SHS), 2015.

[NXP00] NXP. *BF862 datasheet*, 2000.

[Obe17] Johannes Obermaier. OpenOCD patch: flash/nor/stm32f1x: Added RDP level 2 support, `http://openocd.zylin.com/4111`, 2017.

[OH16] Johannes Obermaier and Martin Hutle. Analyzing the security and privacy of cloud-based video surveillance systems. In *Proceedings of the 2nd ACM International Workshop on IoT Privacy, Trust, and Security*, IoTPTS '16, Xi'an, China. ACM, 2016.

[OHHS18] Johannes Obermaier, Florian Hauschild, Matthias Hiller, and Georg Sigl. An embedded key management system for PUF-based security enclosures. In *2018 7th Mediterranean Conference on Embedded Computing (MECO)*, 2018.

[OI18] Johannes Obermaier and Vincent Immler. The past, present, and future of physical security enclosures: From battery-backed monitoring to PUF-based inherent security and beyond. *Journal of Hardware and Systems Security*, 2018.

[OIH19] Johannes Obermaier, Vincent Immler, and Robert Hesselbarth. Patent EP3550475 – PUF-film and method for producing the same, 2019.

[OIHS18] Johannes Obermaier, Vincent Immler, Matthias Hiller, and Georg Sigl. A measurement system for capacitive PUF-based security enclosures. In *Proceedings of the 55th Annual Design Automation Conference*, DAC '18, San Francisco, California. ACM, 2018.

[OLR14] Johannes Obermaier, Tobias Laas, and Markus Roner. Timing attack on a modified dynamic S-box implementation of the AES invsubbytes operation. In *Informatik 2014*, Stuttgart. Gesellschaft für Informatik e.V., 2014.

[OSS17] Johannes Obermaier, Robert Specht, and Georg Sigl. Fuzzy-glitch: A practical ring oscillator based clock glitch attack. In *2017 International Conference on Applied Electronics (AE)*, 2017.

[OT17] Johannes Obermaier and Stefan Tatschner. Shedding too much light on a microcontroller's firmware protection. In *11th USENIX Workshop on Offensive Technologies (WOOT 17)*, Vancouver, BC. USENIX Association, 2017.

[Ozm05] Andy Ozment. The likelihood of vulnerability rediscovery and the social utility of vulnerability hunting. *The Workshop on Economics and Information Security*, 2005.

[Pay13] Payment Card Industry Security Standards Council. *Payment Card Industry PIN Transaction Security (PTS) v4.0.* PCI, Wakefield, MA, USA, 2013.

[PB61] William Wesley Peterson and Daniel T Brown. Cyclic codes for error detection. *Proceedings of the IRE*, 1961.

[PBBJ15] Stjepan Picek, Lejla Batina, Pieter Buzing, and Domagoj Jakobovic. *Fault Injection with a New Flavor: Memetic Algorithms Make a Difference*. Springer International Publishing, Cham, 2015.

[PCI09] PCI Security Standards Council. Payment card industry (PCI) – hardware security module (HSM), `https://www.pcisecuritystandards.org/documents/PCI%20HSM%20Security%20Requirements%20v1.0%20final.pdf`, 2009.

[Pop15] Andrey Popov. RFC 7465 – prohibiting RC4 cipher suites, 2015.

[PRTG02] Ravikanth Pappu, Ben Recht, Jason Taylor, and Neil Gershenfeld. Physical one-way functions. *Science*, 2002.

[Ral12] Ralink Technology Corp. MT7620 datasheet. `https://www.electrodragon.com/w/images/3/34/MT7620_Datasheet.pdf`, 2012.

[Ram09] Xavier Ramus. Transimpedance considerations for high-speed amplifiers, 2009.

[Rea18] Realtek. RTL8197D – 5-port 10/100M ethernet router network processor. `http://www.realtek.com/products/productsView.aspx?Langid=1&PNid=12&PFid=58&Level=5&Conn=4&ProdID=402`, 2018.

[RND18] Thomas Roth, Dmitry Nedospasov, and Josh Datko. wallet.fail – hacking the most popular cryptocurrency hardware wallets. `https://media.ccc.de/v/35c3-9563-wallet_fail`, 2018.

[RWIJ06] Sreeraman Rajan, Sichun Wang, Robert Inkol, and Alain Joyal. Efficient approximations for the arctangent function. *IEEE Signal Processing Magazine*, 2006.

[SA03] Sergei P. Skorobogatov and Ross J. Anderson. Optical fault induction attacks. In *Cryptographic Hardware and Embedded Systems (CHES)*. Springer Berlin Heidelberg, 2003.

[SCG+14] Edward Suh, Dwaine Clarke, Blaise Gassend, Marten van Dijk, and Srinivas Devadas. AEGIS: architecture for tamper-evident and tamper-resistant processing. In *ACM International Conference on Supercomputing 25th Anniversary Volume*, Munich, Germany. ACM, 2014.

[Sch07] Bruce Schneier. Full disclosure of security vulnerabilities a 'damned good idea'. `https://www.schneier.com/essays/archives/2007/01/schneier_full_disclo.html`, 2007.

[SDK+13] Daehyun Strobel, Benedikt Driessen, Timo Kasper, Gregor Leander, David Oswald, Falk Schellenberg, and Christof Paar. *Fuming Acid and Cryptanalysis: Handy Tools for Overcoming a Digital Locking and Access Control System*. Springer Berlin Heidelberg, 2013.

[SFIC14] Merrielle Spain, Benjamin Fuller, Kyle Ingols, and Robert Cunningham. Robust keys from physical unclonable functions. In *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2014.

[She03] Stephen Shepherd. Vulnerability disclosure: How do we define responsible disclosure? *GIAC SEC Practical Repository, SANS Inst*, 2003.

[SHL+13] Y. W. Siah, Y. J. Hong, Q. Liu, H. B. Kor, and C. L. Gan. Uniform delayering of copper metallization. In *Proceedings of the 20th IEEE International Symposium on the Physical and Failure Analysis of Integrated Circuits (IPFA)*, 2013.

[SHO19] Bodo Selmke, Florian Hauschild, and Johannes Obermaier. Peak clock: Fault injection into PLL-based systems via clock manipulation. In *Proceedings of the 2019 Workshop on Attacks and Solutions in Hardware Security (ASHES)*. ACM, 2019.

[SHS16] Bodo Selmke, Johann Heyszl, and Georg Sigl. Attack on a DFA protected AES by simultaneous laser fault injections. In *2016 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, Volume 00, 2016.

[Sko00] Sergei Skorobogatov. Copy protection in modern microcontrollers. `https://www.cl.cam.ac.uk/~sps32/mcu_lock.html`, 2000.

[Sko02] Sergei Skorobogatov. Low temperature data remanence in static RAM. Technical Report UCAM-CL-TR-536, University of Cambridge, Computer Laboratory, 2002.

[Sko05] Sergei Skorobogatov. Semi-invasive attacks – a new approach to hardware security analysis. Technical Report UCAM-CL-TR-630, University of Cambridge, Computer Laboratory, 2005.

[Sko10] Sergei Skorobogatov. Flash memory 'bumping' attacks. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, 2010.

[SMC09] Dhiman Saha, Debdeep Mukhopadhyay, and Dipanwita Roy Chowdhury. A diagonal fault attack on the advanced encryption standard. *IACR Cryptology ePrint Archive*, 2009.

[SO19] Marc Schink and Johannes Obermaier. Taking a look into execute-only memory. In *13th USENIX Workshop on Offensive Technologies (WOOT 19)*, Santa Clara, CA. USENIX Association, 2019.

[SOG13] SOG-IS. Application of attack potential to smartcards. `https://www.sogis.org/documents/cc/domains/sc/JIL-Application-of-Attack-Potential-to-Smartcards-v2-9.pdf`, 2013.

[SOSD05] Edward Suh, Charles W. O'Donnell, Ishan Sachdev, and Srinivas Devadas. Design and implementation of the AEGIS single-chip secure processor using physical random functions. In *Proceedings of the 32Nd Annual International Symposium on Computer Architecture*, ISCA '05, Washington, DC, USA. IEEE Computer Society, 2005.

[SP08] Dimitrios N. Serpanos and Andreas Papalambrou. Security and privacy in distributed smart cameras. *Proceedings of the IEEE*, 2008.

[STM12] STMicroelectronics. STM32F0xxx Cortex-M0 programming manual, 2012.

[STM15] STMicroelectronics. *AN4566 Application Note, Extending the DAC performance of STM32 microcontrollers*, Rev. 2, 2015.

[STM16a] STMicroelectronics. *AN4044 Application Note, Floating point unit demonstration on STM32 microcontrollers*, Rev. 2, 2016.

[STM16b] STMicroelectronics. *Datasheet STM32F303xB STM32F303xC*, Rev. 13, 2016.

[STM17] STMicroelectronics. *RM0091 Reference manual, STM32F0x1/STM32F0x2/STM32F0x8 advanced ARM®-based 32-bit MCUs*, Rev. 9, 2017.

[SW12] Sergei Skorobogatov and Christopher Woods. *Breakthrough Silicon Scanning Discovers Backdoor in Military Chip*. Springer Berlin Heidelberg, 2012.

[SYY+13] Zhengguo Sheng, Shusen Yang, Yifan Yu, Athanasios V. Vasilakos, Julie A. Mccann, and Kin K. Leung. A survey on the IETF protocol suite for the internet of things: standards, challenges, and opportunities. *IEEE Wireless Communications*, 2013.

[Tex17] Texas Instruments. *THS4551 datasheet*, 2017.

[Tie18] Andrew Tierney. Bypassing CRP on microcontrollers. `https://www.youtube.com/watch?v=DTuzuaiQL_Q&t=1885s`, 2018.

[TJ09] Randy Torrance and Dick James. The state-of-the-art in IC reverse engineering. In *Cryptographic Hardware and Embedded Systems (CHES)*. Springer Berlin Heidelberg, 2009.

[TMA11] Michael Tunstall, Debdeep Mukhopadhyay, and Subidh Ali. Differential fault analysis of the advanced encryption standard using a single fault. In *IFIP International Workshop on Information Security Theory and Practices*, 2011.

[TO15] Shahar Tal and Lior Oppenheim. The internet of TR-069 things: One exploit to rule them all. https://www.rsaconference.com/events/us15/agenda/sessions/1574/the-internet-of-tr-069-things-one-exploit-to-rule, 2015.

[TSŠ+06] Pim Tuyls, Geert-Jan Schrijen, Boris Škorić, Jan Van Geloven, Nynke Verhaegh, and Rob Wolters. Read-proof hardware from protective coatings. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*. Springer, 2006.

[UBE18] UBE Industries Ltd. UPILEX – ultra heat-resistant films (grade details) – UBE heat-resistant polyimide materials, http://www.upilex.jp/en/upilex_grade.html#01, 2018.

[Var11] Various authors. Mikrocontroller.net: Meteotime crypt, https://www.mikrocontroller.net/topic/220947, 2011.

[VB09] Hauke Vagts and Jürgen Beyerer. Security and privacy challenges in modern surveillance systems. In *Proceedings of the Future Security Research Conference*, 2009.

[Ver08] Verheugen. Parlamentary questions – answer to: China export (CE) mark feeding off the reputation of the european conformité européenne (CE) mark. http://www.europarl.europa.eu/sides/getAllAnswers.do?reference=P-2007-5938&language=EN, 2008.

[VNK+15] Michael Vai, Ben Nahill, Josh Kramer, Michael Geis, Dan Utin, David Whelihan, and Roger Khazan. Secure architecture for embedded systems. In *IEEE High Performance Extreme Computing Conference (HPEC)*, 2015.

[VWNU16] Michael Vai, David Whelihan, Benjamin Nahill, and Daniil Utin. Secure embedded systems. In *MIT Lincoln Laboratory Journal Volume 22 November 1*, 2016.

[War13] Henry S. Warren. *Hacker's delight*. Pearson Education, 2013.

[Was16] Garret Wassermann. What is vulnerability coordination? https://vuls.cert.org/confluence/pages/viewpage.action?pageId=4718642, 2016.

[Wei00] Steve H. Weingart. Physical security devices for computer subsystems: A survey of attacks and defenses. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*. Springer, 2000.

[Wie15] Norbert Wiedermann. Absicherungskonzepte für industrie 4.0. *Datenschutz und Datensicherheit – DuD*, 2015.

[Win16] Winbond. *W25Q64FW – 1.8V 64M-BIT SERIAL FLASH MEMORY WITH DUAL/QUAD SPI & QPI datasheet*, 2016.

[WK16] Michael Waidner and Michael Kasper. Security in industrie 4.0 – challenges and solutions for the fourth industrial revolution. In *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2016.

[WR14] Thomas Winkler and Bernhard Rinner. Security and privacy protection in visual sensor networks: A survey. *ACM Computing Surveys (CSUR)*, 2014.

[WR15] Thomas Winkler and Bernhard Rinner. Secure embedded visual sensing in end-user applications with TrustEYE.M4. In *2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2015.

[WSL+15] Lingxiao Wei, Chaosheng Song, Yannan Liu, Jie Zhang, Feng Yuan, and Qiang Xu. BoardPUF: Physical unclonable functions for printed circuit board authentication. In *IEEE/ACM International Conference on Computer-Aided Design*, 2015.

[WSNS18] Mark Walker, Kelton Shockey, Andrew Neiman, and Ashtyn Stephan. Awakening global governments: An international survey of internet of things regulation. *TPRC 46: The 46th Research Conference on Communication, Information and Internet Policy*, 2018.

[Xil18] Xilinx. *7 Series FPGAs Configuration – User Guide*, `https://www.xilinx.com/support/documentation/user_guides/ug470_7Series_Config.pdf`, 2018.

[ZCW+14] Zhi-Kai Zhang, Michael Cheng Yi Cho, Chia-Wei Wang, Chia-Wei Hsu, Chong-Kuan Chen, and Shiuhpyng Shieh. IoT security: Ongoing challenges and research opportunities. In *2014 IEEE 7th International Conference on Service-Oriented Computing and Applications*, 2014.

List of Prior Publications

[OIH19] **Johannes Obermaier**, Vincent Immler, and Robert Hesselbarth. Patent EP3550475 – PUF-film and method for producing the same, 2019.

[IO19] Vincent Immler and **Johannes Obermaier**. Patent EP355046 – PUF-film and method for producing the same, 2019.

[SO19] Marc Schink and **Johannes Obermaier**. Taking a look into execute-only memory. In *13th USENIX Workshop on Offensive Technologies (WOOT 19)*, Santa Clara, CA. USENIX Association, 2019.

[SHO19] Bodo Selmke, Florian Hauschild, and **Johannes Obermaier**. Peak clock: Fault injection into PLL-based systems via clock manipulation. In *Proceedings of the 2019 Workshop on Attacks and Solutions in Hardware Security (ASHES)*. ACM, 2019.

[OHHS18] **Johannes Obermaier**, Florian Hauschild, Matthias Hiller, and Georg Sigl. An embedded key management system for PUF-based security enclosures. In *2018 7th Mediterranean Conference on Embedded Computing (MECO)*, 2018.

[OIHS18] **Johannes Obermaier**, Vincent Immler, Matthias Hiller, and Georg Sigl. A measurement system for capacitive PUF-based security enclosures. In *Proceedings of the 55th Annual Design Automation Conference*, DAC '18, San Francisco, California. ACM, 2018.

[OI18] **Johannes Obermaier** and Vincent Immler. The past, present, and future of physical security enclosures: From battery-backed monitoring to PUF-based inherent security and beyond. *Journal of Hardware and Systems Security*, 2018.

[IOK18a] Vincent Immler, **Johannes Obermaier**, Martin König, Matthias Hiller, and Georg Sigl. B-TREPID: batteryless tamper-resistant envelope with a PUF and integrity detection. In *2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2018.

[IOK18b] Vincent Immler, **Johannes Obermaier**, Martin König, Matthias Hiller, and Georg Sigl. Next-generation anti-tamper envelopes for cyber physical defense systems. In *STO-MP-SCI-300*, 2018.

[ION18] Vincent Immler, **Johannes Obermaier**, Kuan Kuan NG, Fei Xiang KE, Jin Yu LEE, Yak Peng LIM, Wei Koon OH, Keng Hoong WEE, and Georg Sigl. Secure physical enclosures from covers with tamper-resistance. *IACR Transactions on Cryptographic Hardware and Embedded Systems (CHES)*, 2018.

[OSS17] **Johannes Obermaier**, Robert Specht, and Georg Sigl. Fuzzy-glitch: A practical ring oscillator based clock glitch attack. In *2017 International Conference on Applied Electronics (AE)*, 2017.

[OT17] **Johannes Obermaier** and Stefan Tatschner. Shedding too much light on a microcontroller's firmware protection. In *11th USENIX Workshop on Offensive Technologies (WOOT 17)*, Vancouver, BC. USENIX Association, 2017.

[Ob17] **Johannes Obermaier**. OpenOCD patch: flash/nor/stm32f1x: Added RDP level 2 support, `http://openocd.zylin.com/4111`, 2017.

[IHOS17] Vincent Immler, Matthias Hiller, **Johannes Obermaier**, and Georg Sigl. Take a moment and have some t: Hypothesis testing on raw PUF data. In *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2017.

[OH16] **Johannes Obermaier** and Martin Hutle. Analyzing the security and privacy of cloud-based video surveillance systems. In *Proceedings of the 2nd ACM International Workshop on IoT Privacy, Trust, and Security*, IoTPTS '16, Xi'an, China. ACM, 2016.

[OLR14] **Johannes Obermaier**, Tobias Laas, and Markus Roner. Timing attack on a modified dynamic S-box implementation of the AES invsubbytes operation. In *Informatik 2014*, Stuttgart. Gesellschaft für Informatik e.V., 2014.

List of Figures

List of Tables

List of Acronyms

ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
AFE	Analog Front-End
AHB	Advanced High-performance Bus
API	Application Programming Interface
ASIC	Application-Specific Integrated Circuit
BBRAM	Battery-Backed Random-Access Memory
CA	Certificate Authority
CBS	Cold-Boot Stepping
CCMRAM	Core-Coupled Memory
CE	Communauté Européenne
CFB	Cipher Feedback
COTS	Commercial-Off-The-Shelf
CPLD	Complex Programmable Logic Device
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CSP	Critical Security Parameter
DAC	Digital-to-Analog Converter
DCM	Digital Clock Manager
DDS	Direct Digital Synthesis
DES	Data Encryption Standard
DFT	Discrete Fourier Transform
DMA	Direct Memory Access
DoS	Denial-of-Service

EDA	Electronic Design Automation
EFP	Environmental Failure Protection
EKMS	Embedded Key Management System
ENIG	Electroless Nickel Immersion Gold
ENISA	EU Agency for Network and Information Security
FEA	Finite Element Analysis
FPGA	Field-Programmable Gate Array
FPU	Floating-Point Unit
GBW	Gain Bandwidth Product
GDPR	General Data Protection Regulation
HF	High Frequency
HP	High-Pass
HSM	Hardware Security Module
HTTP	Hyper Text Transfer Protocol
IC	Integrated Circuit
IO	Input/Output
IoT	Internet of Things
IP	Intellectual Property
JFET	Junction gate Field-Effect Transistor
JIL	Joint Interpretation Library
KEK	Key Encryption Key
LDO	Low-Dropout Regulator
LED	Light Emitting Diode
LP	Low-Pass
MAC	Media Access Control
MitM	Man in the Middle
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
MPU	Memory Protection Unit
MUP	Module Under Protection
NAT	Network Address Translation
NOP	No Operation
PCB	Printed Circuit Board
PCI	Payment Card Industry

PLL	Phase-Locked Loop
PoC	Proof of Concept
PoS	Point of Sale
PUF	Physical Unclonable Function
RDP	Readout Protection
RNG	Random Number Generator
RO	Ring Oscillator
RTC	Real-Time Clock
RTOS	Real-Time Operating System
SDRAM	Synchronous Dynamic Random-Access Memory
SHA-1	Secure Hash Algorithm 1
SMB	Server Message Block
SMD	Surface-Mounted Device
SNR	Signal-to-Noise Ratio
SoC	System On Chip
SRAM	Static Random-Access Memory
SSL	Secure Sockets Layer
SWD	Serial Wire Debug
TIA	Transimpedance Amplifier
TLS	Transport Layer Security
TRL	Technology Readiness Level
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Bus
UV-C	Short-wave Ultraviolet
VCO	Voltage-Controlled Oscillator
VPN	Virtual Private Network
WLAN	Wireless Local Area Network
ZIF	Zero Insertion Force