Analog Hardware Security and Trust

Julian Leonhard

▶ **To cite this version:**

Julian Leonhard. Analog Hardware Security and Trust. Electronics. Sorbonne Université, 2021. English. NNT : . tel-03681806v1

HAL Id: tel-03681806

https://hal.science/tel-03681806v1

Submitted on 21 Jun 2021 (v1), last revised 30 May 2022 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT
DE SORBONNE UNIVERSITÉ

ANALOG HARDWARE SECURITY AND TRUST

présentée par

JULIAN LEONHARD

École Doctorale Informatique, Télécommunications et Électronique

réalisée au

Laboratoire d'Informatique Paris 6

soutenue le 25 Mars 2021

devant le jury composé de :

M.	Giorgio DI NATALE	DR CNRS, TIMA, Grenoble	Rapporteur
M.	Ilia POLIAN	Prof., University of Stuttgart	Rapporteur
Mme.	Emmanuelle ENCRENAZ	MCF, Sorbonne Université	Examinatrice
M.	Helmut GRÄB	Prof., Technical University Munich	Examinateur
M.	Ozgur SINANOGLU	Prof., NYU Abu Dhabi	Examinateur
M.	Jean-Luc DANGER	Prof., Télécom ParisTech	Examinateur
Mme.	Marie-Minerve LOUËRAT	CR CNRS, Sorbonne Université	Invitée
M.	Hassan ABOUSHADY	MCF, Sorbonne Université	Invité
M.	Haralampos STRATIGOPOULOS	DR CNRS, Sorbonne Université	Directeur de Thèse

Julian Leonhard: *Analog Hardware Security and Trust,* © January 2021

Für meinen Doktoropa Werner Leonhard.
1926 – 2018

ABSTRACT

The ongoing globalization and specialization of the integrated circuit (IC) supply chain has led semiconductor companies to share their valuable intellectual property (IP) assets with numerous parties for means of manufacturing, testing, etc. As a consequence, sensitive IPs and ICs are being exposed to untrusted parties, resulting in serious piracy threats such as counterfeiting or reverse engineering. In this thesis we develop methods to secure analog and mixed signal IPs/ICs from piracy threats within the supply chain.

We propose an anti-piracy methodology for locking mixed-signal ICs via logic locking of their digital part. The capabilities of the technique are demonstrated on a $\Sigma\Delta$ ADC in a hardware experiment and in a real-world audio-application. Furthermore, we propose an obfuscation methodology towards analog IP protection against reverse engineering. Obfuscation is achieved by camouflaging the effective geometry of layout components via the use of fake contacts. We apply and evaluate the technique on an operational amplifier and an RF $\Sigma\Delta$ ADC. Finally, we propose an attack to break all analog circuit locking techniques that act upon the biasing of the circuit. The attack is based on re-synthesizing the biasing circuits and requires only the use of an optimization algorithm. It is put to test on a locked voltage regulator using a genetic algorithm.

This thesis demonstrates that establishing security and trust for analog and mixed signal IPs and ICs, while still in its infancy, is feasible. The presented techniques have the potential to protect analog and mixed-signal circuits against a large subset of all the possible risk scenarios while inflicting low overheads in terms of area, power and performance. The changes carried out in the ICs' analog sections are subtle, a key requirement for the adoption of our techniques by analog designers.

RÉSUMÉ

La mondialisation et la spécialisation actuelles de la chaîne d'approvisionnement des circuits intégrés (CI) ont conduit les entreprises de semi-conducteurs à partager leur précieuse propriété intellectuelle (PI) avec de nombreuses parties pour les faire fabriquer, tester, etc. En conséquence, les PI et les CI sensibles sont exposés à des parties potentiellement malveillantes, ce qui entraîne de graves menaces de piratage telles que la contrefaçon ou la retro ingénierie. Dans cette thèse, nous développons

des méthodes pour sécuriser les PI/CI analogiques et mixtes contre les menaces de piratage dans la chaîne d'approvisionnement.

Nous proposons une méthodologie anti-piratage pour verrouiller les circuits intégrés mixtes via l'application de logic locking à leur partie numérique. Les capacités de la technique sont démontrées sur un ADC $\Sigma\Delta$ dans le cadre d'une expérience matérielle et dans une application audio. En outre, nous proposons une méthodologie d'offuscation pour la protection de la propriété intellectuelle analogique contre la rétro-ingénierie. L'offuscation est réalisé en camouflant la géométrie effective des composants de layout par l'utilisation de faux contacts. Nous appliquons et évaluons la technique sur un amplificateur opérationnel et un RF $\Sigma\Delta$ ADC. Enfin, nous proposons une attaque pour contourner toutes les techniques de verrouillage des circuits analogiques qui agissent sur la polarisation du circuit. L'attaque est basée sur la ré-synthèse des circuits de polarisation et ne nécessite que l'utilisation d'un algorithme d'optimisation. Elle est démontrée sur un régulateur de tension verrouillé à l'aide d'un algorithme génétique.

Cette thèse démontre qu'il est possible d'établir la sécurité et la confiance pour les CI analogiques et mixtes, bien que ça soit qu'un début. Les techniques présentées ont le potentiel de protéger les circuits analogiques et mixtes contre une grande partie de tous les scénarios de risque possibles tout en infligeant de faibles coûts en termes de surface, de puissance et de performance. Les changements effectués dans les sections analogiques des CI sont subtils, ce qui est une condition essentielle pour l'adoption de nos techniques par les concepteurs de circuits analogiques.

PUBLICATIONS

As a result of this thesis the following publications have appeared:

[1] J. Leonhard, M. Yasin, S. Turk, M. Nabeel, M.-M. Louërat, R. Chotin-Avot, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos, "MixLock: securing mixed-signal circuits via logic locking," in *Proc. Design, Automation & Test in Europe Conference*, 2019.

[2] J. Leonhard, M.-M. Louërat, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos, "Mixed-signal hardware security using MixLock: demonstration in an audio application," in *International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design*, 2019.

[3] J. Leonhard, A. Sayed, M.-M. Louërat, H. Aboushady, and H.-G. Stratigopoulos, "Analog and mixed-signal ic security via sizing camouflaging," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020.

[4] J. Leonhard, M. Elshamy, M.-M. Louërat, and H.-G. Stratigopoulos, "Breaking analog biasing locking techniques via re-synthesis," in *26th Asia and South Pacific Design Automation Conference*, 2021.

To explain all nature is too difficult a task
for any one man or even for any one age.
'Tis much better to do a little with certainty,
and leave the rest for others that come after you,
than to explain all things

— Isaac Newton, 1704

ACKNOWLEDGMENTS

Three years passed in the blink on an eye, even though in the beginning the journey seemed infinite. Reflecting, I am humbled and filled with joy to have crossed paths with many wonderful people from all over the world. A bit like a particle moving through space and seeing its trajectory change in the attracting force field of other particles, only briefly crossing some, while traveling together with others for years.

One particle that had a profound impact on my trajectory is my supervisor Haralampos, or, as we say in German more fittingly, my doctor-father. Thank you Haralampos for bearing with me, for pushing me and for believing in me. You taught me to see the bigger picture, to be very precise with words and to focus on the goal and find a new way when suffering setbacks. Also, whenever in the future I should downplay my work, there will be a little Haralampos popping up on my shoulder saying: "Julian, for the last time, stop belittling your work!". I have gotten to know few PhD students who are as happy with their supervisor as I am. Thank you Haralampos, it truly was a pleasure to work with you, on a professional as well as on a human level.

My deep gratitude also goes to two very important people in the making of my PhD: Marie-Minerve and Hassan. Whenever an organizational problem arose, Marie-Minerve cared and helped us PhD students where she could, e.g., under her guidance we taught the university that climate action starts by taking the train for Paris-Florence. Whenever she proof-read one of our papers her criticism was excellent and I wish to one day be able to read a paper like she does. Hassan, the ruler over the $\Sigma\Delta$-universe provided us with all the system- and schematic-level models, layouts, hardware, the corresponding know-how and what else not we required on countless occasions. Hassan played a key role by providing excellent technical explanations as well as allowing our ideas to materialize in hardware.

Shadi and Alhassan both were irreplaceable and very reliable links in our (now secured) paper supply-chain. My deep gratitude goes to them for the many hours they spent measuring, supporting and simulating

with and for us. Shadi, please let me know when you find the time-multiplier so we can switch it off during vacation.

I'd like to salute my dearest and absolutely fantastic lab-particles & -mates: Antonios (Amp-tonios), Sarah, Alán, Theofilos, Gabriel, Ilias, Mohamed, Ning and Rodrigo. It makes me immensely happy to have shared these many Fredos, Nescafés and other Greek coffee specialties with you, to have sung and danced in the lab with you or to have feasted on countless sweets (Baklava, Künefe, or that fabulous Terkenlis chocolate Easter lamb from Thessaloniki) with you! I mean, we even were at a conference dinner at Montmartre with a full blown Drag show! When I'm gone, please never forget what Sarah the Wise once said: "a meeting without food should be an email."

Working together with the team at NYU Abu-Dhabi was a fantastic experience and I was always looking forward to our calls. Especially in the beginning when we, analog people, were trying hard to grasp digital logic locking, Ozgur would always find a way to make us understand. The NYU people also saved the day, when, just before the DATE deadline and after something had gone quite wrong, Yasin worked all night to send a fix, allowing a new batch of experiments to start at 4am. The night guard of Sorbonne later that night assured me I was definitely the last one to leave the campus that night. Thanks Yasin! A big thank you also goes to Nimisha for the live-support, the proof-reading and the good explanations on locking and synthesis!

I want to extend my thanks to some of the people in the LIP6 laboratory I had the pleasure to get to know a bit better: Amine, Roselyne, Manuel, Emmanuelle, Shahin, Sylvain, Farouk, Franck, Habib, Dimitri, Jean-Paul, Daniela, Mariam, Doaa, Tamer, Islam, Abdelkrim and Jacky. In the context of the university I also want to thank the EDITE de Paris, my doctoral school, for having awarded me their scholarship, allowing me to pursue my PhD studies in the best conditions.

After the lab you have to live somewhere and for me, living at Cité Universitaire de Paris was a privilege. I will keep so many memories of these great days here. Thank you to my friends in the Colégio de España for making me delicious food (I took photos of it, every day!) while I was quarantined in my room with COVID: Jan, Paula, Franzi, Claudia and Mercedes. Living in the Swedish House in 2019/20 was one of the greatest periods in Paris, especially during the first lockdown in spring 2020. I'll never forget these good times with you at the Cité U, my dear friends from near and far: Pedro, Myrto, Daniel, Micaela, Cecilia, Luna, Jeanne, Léna, Evelina, Sylvia, Carla, Hilda, Alex, Rebecca, Daria, Tangi, Pierre, Joséphine, Grégory, Yukiko, Yajna, Marie and so many, many more.

The Vélo Volant deserves a special place in my heart. Never would I have believed I would do a PhD and run a bike repair shop on the

weekends. I'm so very proud of you and of what you do for others: Marie, Geoffrey, Francesco, Matthew, Clément, Sara, Boris, Manon, Adrien, Leo and all the bénévoles who work and worked with us to help people learn about mechanics while never forgetting that at 5pm « c'est l'heure de goûter »!

A big thank you to Marlene, Marie, Alán, Sarah and Antonios for proof-reading the manuscript! I now know what an Oxford comma is!

Those particles that travel all your life with you, who are the most important ones there can be, are the family back home. I want to say thank you: to my sister Simone and to my Mama, for making sure things were going well and the right way in Paris, for their open ears and emotional support, for the small things, for the thirty post-cards with funny cat pictures; to my father for helping me do the right thing, when I was very, very stuck; to my beloved grandmother for watching over us and for teaching me gratitude and how to be happy.

And finally, this thesis is dedicated to my academic idol and my guiding light, my dearest grandfather. I hope you can see this.

CONTENTS

I MAIN PART

1 INTRODUCTION 3
1.1 Globalization and its Threats . . . 3
1.1.1 The Ever-Changing Supply Chain . . . 3
1.1.2 Actors in the Supply Chain . . . 4
1.1.3 Threats to an IP within the Supply Chain . . . 4
1.1.3.1 Hardware Trojan Injection . . . 5
1.1.3.2 Side-channel & Fault Injection Attacks . . . 5
1.1.3.3 Reverse Engineering . . . 6
1.1.3.4 Counterfeiting . . . 6
1.1.4 Impact of IP/IC Piracy . . . 7
1.1.5 Focus on Piracy . . . 8
1.2 The Need for Security and Trust . . . 8
1.2.1 From Software to Hardware Security . . . 8
1.2.2 Analog Hardware Security . . . 9
1.3 Thesis Structure . . . 11

2 PRIOR ART ON ANALOG HARDWARE SECURITY 13
2.1 Hardware Trojans in AMS/RF ICs . . . 13
2.1.1 Analog/RF Trojans . . . 13
2.1.2 Analog Triggers . . . 14
2.1.3 Trojans States . . . 14
2.1.4 Hardware Trojan Defenses . . . 15
2.1.4.1 Pre-Silicon Methods . . . 15
2.1.4.2 Post-Silicon Methods . . . 15
2.1.4.3 Design-for-Trust Methods . . . 16
2.2 Piracy Countermeasures for Analog ICs . . . 16
2.2.1 Analog Locking . . . 17
2.2.1.1 Analog Locking via Calibration Locking . . . 17
2.2.1.2 Analog Biasing Locking . . . 18
2.2.1.3 Compound Techniques . . . 21
2.2.2 Obfuscation . . . 21
2.2.3 Split Manufacturing . . . 21
2.2.4 Recycling Protection . . . 22

3 MIXLOCK: SECURING MIXED-SIGNAL CIRCUITS VIA LOGIC LOCKING 25
3.1 Introduction to MixLock . . . 25
3.2 The Proposed Technique: MixLock . . . 26
3.3 Threat Model . . . 27
3.4 Security Analysis . . . 28

3.5 Logic Locking . . . 29
3.5.1 Stripped Functionality Logic Locking . . . 30
3.5.2 Dishonest Oracle and Truly Random Logic Locking 31
3.6 Simulation Results with SFLL . . . 33
3.6.1 Case Study . . . 33
3.6.2 Setup . . . 34
3.6.3 Results & Security Analysis . . . 35
3.6.4 Implementation Cost . . . 37
3.7 Hardware Experiment Results with DisORC and TRLL . . 38
3.7.1 Case Study . . . 38
3.7.2 Setup . . . 39
3.7.3 Results & Security Analysis . . . 41
3.7.4 Implementation Cost . . . 43
3.8 MixLock Demonstration in an Audio Application . . . 43
3.8.1 Demonstrator Configuration and Setup . . . 43
3.8.2 Results . . . 45
3.9 Conclusion . . . 48
4 ANALOG SIZING CAMOUFLAGING 49
4.1 Introduction to Camouflaging . . . 49
4.2 Analog IC Camouflaging . . . 50
4.2.1 Threat Model . . . 50
4.2.2 Sizing Camouflaging . . . 51
4.2.3 The Defender Perspective: Design Flows with Camouflaging . . . 52
4.2.4 The Defender Perspective: Objectives . . . 54
4.2.5 The Attacker Perspective . . . 56
4.3 Library of Camouflaged Layout Components . . . 56
4.3.1 Transistors . . . 56
4.3.2 Capacitors . . . 58
4.3.3 Resistors . . . 58
4.4 Recommendations for Analog IC Camouflaging . . . 59
4.4.1 Number of Components to Resize and Degree of Resizing . . . 59
4.4.2 Degree of Performance Degradation . . . 60
4.4.3 Selection of Components to Resize . . . 60
4.5 Attacks against Analog IC Camouflaging . . . 62
4.6 Security Metrics . . . 65
4.7 Case Studies . . . 68
4.7.1 Miller Operational Amplifier . . . 68
4.7.2 RF ΣΔ ADC . . . 71
4.8 Conclusion . . . 74
5 BREAKING ANALOG BIASING LOCKING TECHNIQUES VIA RE-SYNTHESIS 77
5.1 Introduction and Context . . . 77

5.2 Prior Attack 78
5.3 Proposed Attack 80
5.4 Comparing the Attacks 82
5.5 Genetic Algorithm 83
5.6 Case-Study and Results 84
5.7 Conclusion 87
6 CONCLUSION & OUTLOOK 91
6.1 Conclusion 91
6.2 Contributions of the Thesis 91
6.3 Future Work & Outlook 92

II APPENDIX

BIBLIOGRAPHY 97

LIST OF FIGURES

Figure 2.1 Analog biasing locking approaches. 19
Figure 2.2 Locking a current mirror using the technique from [58]. 20
Figure 3.1 Mixed-signal IC locked with *MixLock*. 26
Figure 3.2 Simple application of logic locking. 29
Figure 3.3 SFLL architecture. 30
Figure 3.4 DisORC architecture. 32
Figure 3.5 The Bandpass ΣΔ ADC used as case study. 33
Figure 3.6 The analog 4th order LC bandpass ΣΔ modulator. 33
Figure 3.7 The digital DDC mixer and decimation filter. . . . 34
Figure 3.8 Trade-off between analog security level in terms of *error rate* and digital security level against SAT attack for different logic locking techniques. 36
Figure 3.9 SNR for 10^3 incorrect keys and the correct key. . . 37
Figure 3.10 Transient and frequency responses of the 1.5xSFLL locked ΣΔ ADC for an incorrect and the correct key. 38
Figure 3.11 The analog 2nd order LC bandpass ΣΔ modulator used for the hardware experiment. 39
Figure 3.12 Architecture of the digital section, including DDC mixer and decimation filter. 40
Figure 3.13 Implementation of the decimation filter of the ΣΔ ADC on a Xilinx Kintex-7 FPGA. 40
Figure 3.14 SNR for 10^3 incorrect keys in green and the correct key in orange applied to the DisORC & TRLL locked ΣΔ ADC hardware case-study. 41
Figure 3.15 Transient and frequency responses of the DisORC & TRLL locked ΣΔ ADC measured for an incorrect and the correct key. 42
Figure 3.16 *MixLock* demonstration in an audio application. . 44
Figure 3.17 Differences in the impact on the time domain signal of SFLL, 1.5xSFLL and DisORC & TRLL 47
Figure 4.1 Overview of analog IC camouflaging. 52
Figure 4.2 Attacker perspective. 53
Figure 4.3 Obfuscated multiple gate-finger transistor layout with its schematic. 57
Figure 4.4 Obfuscated common-centroid layout and schematic with AXXBBXXA pattern. 57
Figure 4.5 Side-view of obfuscated capacitor bank layout. . . 58
Figure 4.6 Obfuscated serpentine resistor layout 59

Figure 4.7 Schematic of Miller op-amp. The obfuscated components are highlighted. 68
Figure 4.8 Obfuscated layout of Miller op-amp highlighting the inactive instances that have been added. 69
Figure 4.9 Block-level diagram of the RF ΣΔ ADC. Obfuscated sub-blocks are highlighted. 71
Figure 4.10 Complete layout of the RF ΣΔ ADC. 71
Figure 4.11 Zoom in into the pre-amplifier and comparator layouts. The obfuscated blocks are highlighted. . . 72
Figure 4.12 Zoom in into the obfuscated areas of one amplification stage of the pre-amplifer. The obfuscated areas are highlighted. 73
Figure 4.13 SNR vs. input power amplitude for the nominal non-obfuscated, nominal obfuscated, and all-true contact designs. 74
Figure 5.1 Locking a current mirror using the technique from [58]. 78
Figure 5.2 Flowchart of the two versions of the proposed attack. 81
Figure 5.3 LDO regulator block-level schematic. 85
Figure 5.4 Error amplifier circuit inside the LDO regulator. . 85
Figure 5.5 Bandgap voltage reference circuit inside the LDO regulator. 86
Figure 5.6 Internal amplifier circuit inside the bandgap voltage reference circuit. 87
Figure 5.7 LDO regulator output voltage vs. temperature variations. 88
Figure 5.8 LDO regulator output voltage vs. supply voltage variations. 89
Figure 5.9 LDO regulator output voltage vs. time showing the response to a sudden variation of the load current. 89
Figure 5.10 Pareto front showing trade-off between two optimization objectives: ΔV_{out} @ 50 mA vs. V_{out} overshoot. 90

LIST OF TABLES

Table 1.1 Impacts of counterfeit ICs [15]. 7
Table 3.1 Overheads using different underlying logic locking techniques. 39

Table 3.2 Analog security metrics for 1000 random keys with respect to the SNR specification set at 30 dB for the hardware case-study. 42
Table 3.3 Overheads for locking the digital section with DisORC & TRLL when projected to the digital section (left) and when projected to the entire ADC (right). 43
Table 3.4 Audio samples processed with a ΣΔ ADC locked with *MixLock* 45
Table 3.5 Impact of locking with MixLock on audio quality of sample audios. 46
Table 4.1 Design specifications and performance of nominal obfuscated and all-true contact designs. 69
Table 4.2 Obfuscation of components in the Miller op-amp. 70
Table 4.3 Obfuscation metrics for the Miller op-amp. 70
Table 4.4 Obfuscation metrics for the RF ΣΔ ADC. 75
Table 5.1 Requirements for analog bias locking attacks. . . . 83
Table 5.2 Recovered LDO regulator performance trade-offs resulting from the attack. 88

ACRONYMS

ACE adaptive channel estimation
ADC Analog-to-Digital Converter
AFGT analog floating gate transistor
AI Artificial Intelligence
AMS analog, mixed-signal
BEOL back-end-of-line
BP bandpass
BER Bit Error Rate
BIST built-in self-test
DAC Digital-to-Analog Converter
DCSA degradation curve sensitivity analysis
DDC Digital Down-Conversion
DfTr Design-for-Trust
DisORC Dishonest Oracle
DNL Differential Non-Linearity

DNN Deep Neural Network
EVM Error Vector Magnitude
FEOL front-end-of-line
FLL Fault analysis-based Logic Locking
GA Genetic Algorithm
HD Hamming distance
HDL hardware description language
IC integrated circuit
IDM integrated device manufacturer
INL Integral Non-Linearity
IP Intellectual Property
LDO Low-Dropout
MSB Most Significant Bit
op-amp operational amplifier
OTA operational transconductance amplifier
PA power amplifier
PDK Process Design Kit
PLL Phase Locked Loop
PUF Physical Unclonable Function
RF Radio Frequency
RLL Random Logic Locking
RMSE root-mean-square error
RO ring oscillator
ROM read-only memory
SFLL Stripped Functionality Logic Locking
SLL Strong Logic Locking
SMT Satisfiability Modulo Theory
SNR Signal-to-Noise Ratio
SoC System-on-Chip
TRLL Truly Random Logic Locking
VGA variable gain amplifier

Part I

MAIN PART

1 INTRODUCTION

1.1 GLOBALIZATION AND ITS THREATS

1.1.1 *The Ever-Changing Supply Chain*

In the early days of the semiconductor industry, all the tooling, know-how and facilities required to build end-to-end a functional integrated circuit (IC) were to be found within single corporations. Such vertically integrated companies united all the different competences of the semiconductor supply chain, from silicon-wafer fabrication, over circuit design, manufacturing, packaging and testing, up to sales. Companies that operate a wafer foundry and have competences in the physical production of ICs are referred to as integrated device manufacturers (IDMs). Today, few IDMs in the possession of a wafer fab exist and mostly they are prominent and large corporations such as Intel, Samsung or Texas Instruments. IDMs for a long time dominated the IC market in terms of revenue and still do until today, however this is slowly changing [1], [2]. In the last decades, globalization changed the way the semiconductor industry functions by allowing IC companies to outsource large parts of their supply chain. For instance, many IC companies are founded or have transitioned to be 'fabless': they outsource the manufacturing step of their IC design to offshore foundries, many of which are located in separate continents [3]. This is an important advantage in terms of risk management and necessary capital that plays out advantageous for fabless companies. Thanks to globalization, they do not need to bear the enormous costs of building, maintaining and updating a manufacturing site, the costs of which additionally rise dramatically with each new technology node that is visited. While many large corporations such as Qualcomm, AMD or Apple have chosen this horizontally integrated business model, outsourcing has also allowed smaller companies with lower capitalization to enter the semiconductor market, oftentimes in niche-market segments, which has consequently led to a diversification of the market.

The rise of the System-on-Chip (SoC) has led to further embodiments of fabless companies. A SoC is a single silicon die on which a multitude of general and specialized functions are monolithically integrated, e.g., a microcontroller with a number of Analog-to-Digital Converters (ADCs) on the same chip. Formerly, for building such a complex system a number of standalone ICs and possibly discrete components were required. Backed

by the appearance of SoCs new companies with distinct business models have seen the light of day: (a) Intellectual Property (IP)-providers: design houses building specific functions and assembling them in IP blocks; (b) 3rd party IP-vendors: platforms for finding and selling said IP blocks; (c) SoC-integrators: companies specialized in integrating numerous IP blocks in a single SoC design, thus building a complex system from them; (d) other business models combining flavors of the above-mentioned fabless companies.

In conclusion, IDMs with their self-owned production facilities still dominate the semiconductor market, although globalization has led to the appearance of numerous companies following business models that largely rely on outsourced services as well as on external IP blocks. Growth figures of fabless companies persistently trump those of IDMs, suggesting that the trend to alternative business models is continuing, gaining market share to the detriment of established IDMs [1], [2].

1.1.2 *Actors in the Supply Chain*

The supply chain of an IC is divided into the steps of design, fabrication, testing and deployment. In the design phase the circuit is designed as to perform within its specifications, either through the creation and use of in-house IP, integration of externally sourced IPs or a combination of both. The actors involved in the design phase are thus the IP providers and the entity integrating the IPs to a final design, e.g., a SoC integrator. The design phase ends when the circuit layout is taped-out and sent to the next actor in the supply chain, the fabrication plant, aka the foundry. In the fabrication phase, the foundry produces bare die ICs according to the layout. After production, the dies are packaged and tested. Packaging and testing steps may be carried out by further actors, namely test and packaging facilities. Finally, the chips that successfully passed production testing are deployed in the field in the end-user's application. By their end-of-life, the ICs are discarded, thereby ending their life-cycle.

1.1.3 *Threats to an IP within the Supply Chain*

Fundamentally, fabless companies, just as any company that outsources parts of its supply chain, expose and share their most valuable assets, in the form of their IPs, with potentially malicious parties for means of manufacturing, testing, etc. Depending on where in the supply chain the malicious party is located, the threats to an IP are different. In a first effort, these threats can be divided into four main classes:

1. Hardware Trojan injection
2. Side-channel & fault injection attacks

3. Reverse engineering
4. Counterfeiting

Below these threats will be defined in detail.

1.1.3.1 *Hardware Trojan Injection*

Hardware Trojans are malicious design modifications carried out by an attacker within a hardware IP. Trojans may be used to reveal secret information, such as cipher keys, or influence the functionality of the IC, leading for instance to its denial of service [4], [5]. A hardware Trojan implemented in an IP will lead it to present some malicious, undocumented functionality that only is activated under certain, oftentimes rare conditions allowing the Trojan to avoid detection and allowing for a well-timed activation. The activation mechanism is commonly referred to as trigger, while the Trojan's effects on a system's functionality are referred to as payload [6].

Hardware Trojans received major attention in the scientific community throughout the last two decades. This is caused by the fear that a well placed Trojan within hardware responsible for driving, e.g., critical military, medical or electricity infrastructure poses severe risks with possibly disastrous outcomes. The actual threat of hardware Trojans on the other hand is indeed hard to assess. So far, this risk vector seems rather academic with only very few actual implementations having been identified [7]. For an attacker, it appears much more efficient in terms of cost and effort to attack a system on software rather than hardware level. The threat model for hardware Trojans therefore assumes a technologically powerful opponent for whom economical efficiency is not an issue. Such a threat model would therefore exclude private, profit-driven companies as attackers and would in turn lead to the suspicion that state controlled instances are possible adversaries: an arguably challenging threat that may only apply to the minority of ICs that are critical enough to be the target of a state-run attack.

1.1.3.2 *Side-channel & Fault Injection Attacks*

Side-channel & fault injection attacks impair an IC's integrity at runtime and apply when an IC is already deployed. By observing physical characteristics of the running IC or by transitioning it into a state where it malfunctions, possible goals of these attacks are to gather secret information, cipher keys, etc. Conceptually, side-channel and fault injection attacks target the data processed and stored by an IC, while not explicitly targeting the proprietary information needed to produce the design itself.

Side-channel attacks are *passive* attacks that record and analyze information leaked by the attacked IC. The information is gathered non-invasively,

by recording side-channels or by exploiting, e.g., an IC's design-for-test infrastructure. Side channels can include power consumption, electromagnetic radiation, timing information, etc. and are a result of the physical implementation of mathematical functions.

In contrast, fault injection attacks are *active* attacks that try to override and wrongfully set an internal state in the attacked IC, thus tampering with its functionality. For instance, making an IC incorrectly execute an encryption algorithm may be the goal: an otherwise mathematically strong encryption method such as AES can be weakened when injecting a fault that leads to a reduction in the number of rounds the algorithm executes [8]–[12].

1.1.3.3 *Reverse Engineering*

Reverse engineering refers to the extraction of an IP's or an IC's proprietary information. Depending on the intents of the attacker, he may derive different levels of abstractions of an IP/IC, such as architecture, netlist, layout, etc. The attacker's goal can be to level out his technological shortcomings vis-à-vis a competitor by gathering the information necessary to produce an identical or similar IP/IC or to locate the IC's root-of-trust to extract secret information such as cipher keys. Reverse engineering can under certain jurisdictions be legal. At the same time, reverse engineering is oftentimes the first step towards illegal practices, in particular if used to extract the required proprietary information needed to produce a counterfeit IC, which is elaborated in Sec. 1.1.3.4

The procedure of reverse engineering consists of a physical and a logical layer. In the physical layer the attacker prepares the sample IP/IC by depackaging it and proceeding by delayering and carefully imaging the chip's layers. In the logical layer he uses software to stitch together the prepared images, thereby recovering a layout. Through image processing algorithms, such as pattern recognition, circuit elements can be recognized, allowing the attacker to generate a netlist [13], [14].

1.1.3.4 *Counterfeiting*

A counterfeit is a pirated IP/IC that is similar or identical to the original IP/IC. Different types of counterfeits are the result of a multitude of potential security breaches within the supply chain. To understand the different variants of counterfeits we provide a finer taxonomy of possible counterfeits and their origins below:

- *Cloned* IPs/ICs are partial or complete copies of a design obtained through reverse engineering or theft of a design's blueprints such as layouts or schematics.

GOVERNMENT	INDUSTRY	CONSUMER
• National security and civilian safety risks • Law enforcement costs • Loss in tax revenue	• Costs of risk mitigation • Replacement of failing parts and products • Sales losses • Loss in brand values	• Replacement of failing products • Reliability issues • Safety risks

Table 1.1: Impacts of counterfeit ICs [15].

- *Overproduced* chips are ICs that are manufactured over-quota and thereafter sold on the gray market, taking advantage of the original mask set the rogue foundry has access to.
- *Remarked* ICs have their inscriptions altered so they appear of, e.g., higher quality so they can be sold at higher prices on the gray market.
- *Out-of-spec* ICs are units that were rejected during post-manufacturing test due to faulty or insufficient performances and are sold on the gray market by the rogue test facility.
- *Recycled* ICs are original, possibly used ICs that instead of being disposed and destroyed at their end-of-life are sold as new, leading to high failure rates.

1.1.4 *Impact of IP/IC Piracy*

Illegitimate reverse engineering and counterfeiting constitute IP/IC piracy. The injection of counterfeits into the market has effects that range from revenue and tax losses up to safety risks when low-quality ICs are built into products. The impact of such practices are listed in Table 1.1.

Counterfeiting concerns all types of electronic circuits, although according to Guin *et al.* [15], numbers from 2011 suggest that analog ICs, digital microprocessors and memory ICs are among the most concerned types of circuits. Why analog ICs are such an interesting target is discussed in section 1.2.2.

1.1.5 *Focus on Piracy*

This thesis sets its focus on proposing countermeasures against IP/IC piracy. We argue that while it is absolutely justified to research Hardware Trojans, fighting piracy is of an even larger interest to governments, industry and societies, as outlined in Section 1.1.4. This has to do with the attacker's motivation to pirate and counterfeit a design, which is in fact simple in its nature. At the core of such malicious activities lies the interest of the attacker to generate revenue from illegal sales or alternatively to reach the capacity to compete in the market with the help of the pirated material. Both are common and widespread threats that concern all branches of industry working with intellectual property.

The immense importance industry gives to the protection of intellectual property against piracy can also be observed when looking at the ever-increasing numbers of submitted patents. Intellectual property rights including copyrights or patents play an important role in protecting IPs, although they are slow to enforce and only helpful in countries where intellectual property laws are taken seriously. Copyright and patent infringements also need to be detected and followed up on, which can prove itself difficult for certain threats such as reverse engineering. After all, patents are an important lever for IP authors, although they should ideally be complemented by other layers of protection, such as hardware security.

The threat model with regard to the competences of a possible attacker as well as the effort in time and resources he is willing to invest to reach his goals may therefore also be considered to be less severe than for, e.g., hardware Trojans or fault injection attacks. Even a simple protection method may thus already be sufficient to deter an attack. This is of course no reason to underestimate the attacker. The pace at which reverse engineering and thereby piracy capabilities are growing is remarkable and especially a higher level in automation allows for speedy reverse engineering cycles even for complex ICs [13], [14], [16].

1.2 THE NEED FOR SECURITY AND TRUST

1.2.1 *From Software to Hardware Security*

In recent years, security breaches have reached from electronics counterfeiting rings in Guangdong, China [17], over infiltrating and scanning industrial networks for infrastructure equipment in the US [18], forcing power outages on entire regions in Ukraine [19] up to the destruction of Iranian uranium enrichment centrifuges [20]. While all of the above cases have attracted mayor attention from researchers and the broader public they have something substantial in common: these security breaches were

not only data breaches, but much more targeted - and sometimes even destroyed - physical electronic systems [21], [22]. While the most spectacular threats undermine the attacked systems via software [18]–[20], the hardware on which those systems operate is more and more being recognized as a potential playground for security breaches [23]–[26].

Hardware security refers to understanding the threats to an IP/IC during its life-cycle and presenting countermeasures to secure the hardware against the threats classified in Section 1.1.3. Addressing these security breaches is considered of utmost importance, especially for critical ICs deployed in sensitive sectors, such as infrastructure, military, health or telecommunications.

1.2.2 *Analog Hardware Security*

In the last decade, extensive research has been carried out to understand trust and security threat scenarios in digital ICs. While the search for solutions that are easy to implement and resist attacks is still ongoing, the body of work proposing solutions for digital circuits is extensive.

Analog, mixed-signal (AMS) and RF ICs on the other hand have so far received far less attention and as a consequence the number of proposed solutions is little. There are multiple reasons why AMS and RF ICs have so far been less looked into. Therefore, in the following it is outlined what makes proposing solutions for AMS/RF hardware difficult, what differentiates analog from digital hardware and why analog ICs are susceptible to piracy.

- *Wide-spread application.* While digital circuits exploit discretized electrical signals for the means of logic operations, signals in analog circuits are continuous in time, voltage and current, such as is nature around us. At the very least, analog circuits are therefore required at any interface a digital circuit may have with the real world. This includes all wired and wireless communication, sensors, actuators, etc. Analog circuits are therefore very common and serve as a bridge between the digital and analog world. Moreover, analog circuits can be enhanced by digital assistance circuitry, e.g., to compensate the effects of process variations. Thereby, the possibility arises to secure a mixed-signal IC via the protection of its digital part, where a larger variety of security techniques is available. This approach is elucidated in Chapter 3.

- *Valuable IP.* The design of analog circuits is to this day very challenging. Even with advancements in CAD tools, analog IC designs are mostly handcrafted, require experienced engineers and long design-cycles and tend to be more error-prone than their digital counterparts. All these factors make analog IPs very valuable and

give reason to carefully protect the know-how implemented in such an IP.

- *Difficulty to automate.* The design of a digital circuit is largely assisted by automated CAD tools, e.g., allowing a designer to automatically synthesize millions of transistors from a high-level description of a circuit. In analog design, no such standardized hardware description language (HDL) exists and furthermore no automated synthesis tools for generating complex analog circuits have reached market maturity. Implementing security primitives in an automated fashion into analog hardware may thus be hard to achieve.

- *Few components.* While on the one hand in analog circuits the component count is generally quite low, on the other hand the footprints of said components, especially passive components such as resistors, inductors and capacitors, are big. Analog transistors are oftentimes sized larger than the minimum feature size of a given technology would allow. Larger feature sizes are required to achieve good matching between components and performances with regard to, e.g., noise, linearity, intrinsic gain, etc. [27]. Having only few components to identify that are at the same time large is beneficial to an attacker carrying out a reverse engineering attack. Compared to last-generation digital ICs with very low feature sizes, this enables attacks from less well-equipped and less automated reverse-engineering laboratories.

- *Operating point.* The functionality and performance of an analog circuit is largely determined by its operating point, which is set via biasing currents and voltages. Setting the operating point of the analog circuit allows it to perform according to its specifications, whereas setting another operating point will result in a non-foreseen performances trade-off. For instance, the same amplifier circuit, depending on how it is biased, may be a class-A or class-B amplifier and thus perform entirely different in terms of, e.g., linearity and current consumption. For an attacker, this makes reverse engineering analog ICs harder, because in order to derive a meaningful and functional design he also must recover said biases. Security techniques for analog that leverage the biasing of analog circuits will be examined in Section 2.2.1.

- *Failing gracefully.* Analog circuits are designed to be robust so they function under the influence of process variations and effects such as noise, temperature, radiation, etc. If the magnitude of said effects exceeds the circuit's operating specifications, in the case of a digital circuit this would lead to catastrophic failure. However an analog circuit will, at least to a certain extent, continue to produce

meaningful results and it is thus said that analog, contrarily to digital, «fails gracefully». To illustrate this with an example: an analog amplifier operated outside its temperature specification will see its performances such as linearity or gain reduced, but will still accomplish its basic task, namely to amplify the input signal, thus not entirely breaking the signal processing chain it is a part of. A digital processor operated outside its temperature specification will suffer from effects such as timing violations, inducing incorrect computations that in turn lead to errors that propagate along the processing chain. Hence, the robustness of analog designs must be taken into account when proposing security mechanisms that function by generating errors within the analog circuit; offsetting the analog circuit a little may not be enough to fully break the circuit's or the system's functionality.

- *Sensitive.* If adding security features requires additional components within the analog circuit's core, this will find the circuit designers reluctant. Many parts of analog circuits are sensitive to added parasitics, such as capacitance, and would consequently suffer from performance degradation. Security methods must take into account the sensitivity of analog circuits and be as non-invasive as possible. If security techniques penalize performance they will find the disapproval of designers and, as stated above, analog design being a very manual task, such a technique would be unsuccessful.

- *Supplying a key.* Related to the sensitivity is the issue of how a strong key, possibly in the form of a digital bit-sequence, can be supplied to the analog circuit. Given the sensitivity of the circuit it would be prohibitive to add an important amount of components within the analog core circuit. Any sort of key mechanism must therefore be small in terms of area to allow for a sufficiently strong key.

1.3 THESIS STRUCTURE

This thesis proposes a variety of countermeasures for the protection of analog IPs/ICs against piracy. Furthermore, it proposes an attack to remove and, consequently, break biasing locking techniques. The thesis is structured as follows: in Chapter 2, we investigate the previous work in analog hardware security with regard to piracy. In Chapter 3, we present *MixLock*, a technique to secure mixed-signal ICs against piracy. In Chapter 4, we present an obfuscation technique for analog circuits to fight against reverse engineering. In Chapter 5, we present an attack against analog biasing locking techniques and in Chapter 6, we conclude and provide an outlook to further developments in the field of analog hardware security.

2

PRIOR ART ON ANALOG HARDWARE SECURITY

2.1 HARDWARE TROJANS IN AMS/RF ICS

2.1.1 *Analog/RF Trojans*

With most hardware Trojans implemented in the digital domain [5], [6], for an attacker to integrate a hardware Trojan in an AMS/RF IC can pay off in a unique way. As outlined in Sec. 1.2.2, AMS and RF ICs oftentimes operate at interfaces, such as in communication systems for wired and wireless data transmission. A cleverly implemented Trojan in a communication system has the potential to leak secret information by exploiting its host's infrastructure, i.e., the attacker can leverage the data transmission capability of the Trojan-infected device to establish a side-channel, without the need to gain physical access to the infected-device.

Elaborating [28] in [29] and [30], Liu *et al.* demonstrate a hardware Trojan implemented within the RF section of a wireless cryptographic IC. The circuit's digital section is modified in a way, so it forwards bit-by-bit the content of the secret key register of the crypto-core to the Trojan, which is located in the ultra-wideband transmitter. Two types of Trojans are implemented, where the first one leaks the key-bits by modulating the amplitude of the regular transmissions of the cryptographic IC, whereas the second one modulates their frequency. The authors also show that the modulated transmissions stay well within the transmission specifications, thus making the modifications invisible during production testing.

In [31], Subramani *et al.* present a hardware Trojan that is inserted in a transceiver circuit to leak sensitive data, derived from the baseband processor. The hardware Trojan consists of a pair of resistors, one with a higher, the other with a lower ohmic resistance that are connected to the input of the transmitter's power amplifier (PA) via a switch. The switch, actuated by the sensitive data, electrically connects one or the other resistor to the PA's input, impacting the impedance matching and thereby the transmitted signal power. [32] extends this work by proposing a second Trojan in the transceiver that operates by manipulating the gain of the variable gain amplifiers (VGAs) which precede the RF PA. As function of the sensitive data-bits the transmitted signal is amplitude modulated by varying the gain of the VGAs.

In [33], Lin *et al.* establish multi-bit sensitive data leakage through a power side-channel of a digital crypto-processor while remaining stealthily below the noise floor. In [34] and [35], the authors present

how to leverage spread spectrum techniques to hide illegitimate wireless data transmissions in the ambient noise while not impairing legitimate transmissions.

In [36], another Trojan is presented that resides in a digital IP within a SoC. It is triggered in the digital IP, while infecting an analog IP via the common test bus and the built-in self-test (BIST) interface of the analog IP. The payload consists in transitioning the analog IP to test mode, thereby activating BIST procedures, which are normally used to increase test coverage for, e.g., detecting defects that disrupt the analog IP's functionality. In the case study the Trojan exploits a Low-Dropout (LDO) voltage regulator's BIST infrastructure to impair the entire SoC, the LDO is integrated on.

2.1.2 *Analog Triggers*

For a hardware Trojan to avoid detection during manufacturing test, its triggering condition should rarely occur throughout normal operation of the infected IC. In the digital domain, for example, rare input combinations or counters can constitute trigger mechanisms, while in the analog domain, triggering may be achieved through means of charge accumulation on capacitors. Analog trigger circuits are attractive for the attacker since with their small footprint and their incompatibility and thus invisibility to digital functional testing and verification techniques, they can avoid detection.

To give an example, the authors in [37] propose an analog trigger circuit based on a switched capacitor circuit. Charge is siphoned from nearby toggling wires to load a capacitor while leakage currents discharge it. If the toggling frequency is high enough and the capacitor is sufficiently charged to pass a certain threshold voltage, a comparator triggers the payload. In [38], Bidmeshki *et al.* propose new circuit configurations for capacitor-based analog triggers.

2.1.3 *Trojans States*

Furthermore, Trojans in analog ICs can be present in a very inconspicuous way, i.e., without the requirement for additional circuit components and thus undetectable. Fundamentally, in the presence of positive feedback loops, analog circuits present multiple DC operating points, which an attacker can exploit to alter the output characteristics of a multitude of basic analog circuits, such as current mirrors, filters, oscillators, bandgap reference sources and operational amplifiers. [39]–[43].

2.1.4 *Hardware Trojan Defenses*

Defenses to hardware Trojans can be divided into pre-silicon and post-silicon hardware Trojan detection methods, as well as Design-for-Trust (DfTr) techniques. Detection methods try to reveal the presence of a potential Trojan, whereas DfTr techniques are mostly methods that try to either facilitate Trojan detection or hinder Trojan insertion [10], [44], [45]. While many of the presented detection methods are set in the digital domain, they may still be relevant for the detection of analog Trojans, e.g., in the case of an infected mixed-signal circuit, where the Trojan trigger is located in the digital and the payload in the analog domain. Given that the prior art for hardware Trojan detection is especially large, we report a representative, albeit not exhaustive, state of the art below.

2.1.4.1 *Pre-Silicon Methods*

Pre-silicion methods for Trojan detection include functional validation [46], as well as formal verification [47]. In [46], Waksman *et al.* propose a method using Boolean functional analysis to identify suspicious nets at design time within soft digital IPs to expose possible Trojan triggers. In [47], Bidmeshki *et al.* propose a formal verification method, applying information flow tracking, to discover information leakage, applicable to analog, mixed signal and digital circuits.

2.1.4.2 *Post-Silicon Methods*

Post-silicon, i.e., post-production, methods include, (a) optical inspection [48], closely related to reverse-engineering; (b) functional testing that tries to expose the hardware Trojan through the application of suitable test vectors [49]; (c) examining compliance of an IC's transmission profile with the spectral mask specifications; (d) analyzing I/Q constellation diagrams; (e) examining performances such as Signal-to-Noise Ratio (SNR), Error Vector Magnitude (EVM) or Bit Error Rate (BER); (f) channel estimation [31]; (g) statistical side-channel fingerprinting that aims at exposing a hardware Trojan through its systematic distortion in parametric measurements, such as power, delay, temperature, etc. [30], [35], [50].

In [31] the authors leverage adaptive channel estimation (ACE) to identify Trojans in a WLAN transmitter. By removing channel fading and Gaussian noise from the channel estimates, which can be achieved due to the slow-fading characteristics of indoor communications, ACE can distinguish between channel impairments and hardware Trojan activity. In [48] the authors present a low-effort method to inspect a digital circuit for hardware Trojan infection via optical inspection of only the uppermost metal layer. In [49] the authors propose a methodology to identify hard-to-excite, and thus suspect, signals in a digital circuit, aiming at identifying

a Trojan's triggering condition and observing the Trojan's payload. In [30] the authors propose statistical side-channel fingerprinting, consisting in training a one-class classifier in a feature space, composed of parametric measurements, i.e., transmitted power, with golden Trojan-free devices. The Trojan-infected devices have a feature vector that lies outside the classification boundary and, thereby, can be distinguished from Trojan-free devices. In [35] the authors analyze the frequency spectrum of the transmitted signals to identify a Trojan, whereas in [50] golden ICs' power traces are used for fingerprinting to subsequently statistically identify Trojans.

2.1.4.3 *Design-for-Trust Methods*

Methods leveraging DfTr can be based on runtime monitoring or sensors, such as [51] or [52] and also on prevention methods that make Trojan insertion difficult for an attacker, i.e., techniques such as locking, obfuscation, split manufacturing, etc. that will be treated in Sections 2.2.1, 2.2.2 and 2.2.3, respectively.

In [51] the authors propose carrying out minor changes in a digital circuit to obtain ring oscillators (ROs), while changes in the ROs' frequencies can be traced back to Trojan insertions. In [52] the authors propose inserting an array of current sensors in a digital design, allowing for precise monitoring of current consumption, which, in turn, can be used to analytically identify Trojans.

2.2 PIRACY COUNTERMEASURES FOR ANALOG ICS

The previous work on hardware security to harden specifically analog circuits against piracy threats can be classified in the following manner:

- Analog locking
  - Analog locking via calibration locking
  - Analog biasing locking
  - Compound techniques
- Obfuscation
- Split manufacturing
- Recycling protection

Relative to the whole body of work, the largest number of publications is based on analog locking, an attractive approach which, depending on the implementation, allows the IP/IC owner to fight a majority of threats from within the supply chain. By inserting a lock mechanism in the

circuit, the original design is transformed into a design that is functionally equivalent but requires a secret key to unlock the correct functionality. The fashion in which the mechanism and key are implemented can be classified as proposed above. Obfuscation techniques specifically aim at reducing an attacker's capabilities to reverse engineer a protected design. The approach of split manufacturing tries to hinder a rogue foundry from recovering the blueprint of a circuit. Using this classification, an in-depth analysis of the prior art in piracy countermeasures for analog and mixed-signal ICs is given below.

2.2.1 *Analog Locking*

2.2.1.1 *Analog Locking via Calibration Locking*

Calibration locking mechanisms insert locks in programmable or configurable parts of the circuit. Compensation or calibration structures that are already available in the IC are modified and leveraged for lock insertion.

In [53] the authors propose exploiting the naturally available programmability fabric of a highly-digitized analog IC in order to lock it. It is argued that inserting additional circuitry is not strictly necessary, since the configuration settings can serve as secret key-bits. Furthermore, the calibration algorithm to determine the circuit's correct configuration is kept secret. For the approach to be successful said calibration algorithm must be sufficiently complex, so it cannot be reverse engineered. Moreover, the circuit's performances must be highly sensitive to an incorrect configuration, since otherwise multiple keys will unlock the circuit's functionality.

In [54], Nimmalapudi *et al.* propose hindering the use of illegitimately acquired analog ICs by leveraging analog floating gate transistors (AFGTs) which serve to calibrate the IC. When correctly programmed, the AFGTs in the operational transconductance amplifier (OTA) case-study eliminate the offset of a differential pair that arises due to process variations. When starting the unlocking procedure, the tuning range of the AFGTs is reduced and their full tuning range is only unlocked, when a number of secret waypoints is followed. To follow these waypoints, the AFGTs must be programmed in a certain order and with certain voltages, verified via a comparator, a finite state machine and internally generated reference voltages. The waypoints, i.e., their secret order and voltage level, therefore constitute a secret key in the analog domain. As soon as all waypoints have been tracked, the complete programming range of the AFGTs is unlocked, thus allowing the differential pair's offset to be eliminated. For the approach to be feasible, AFGTs must be available in the manufacturing process and the possibly high voltages applied for Fowler-Nordheim

tunneling for programming the floating gates must be taken into account during design.

In [55], Jayasankaran *et al.* propose logic locking the digital optimizer block as part of a BIST calibration feedback loop. The feedback loop maps an analog circuit's performances to selected tuning knobs, such as resistors or capacitors, which can be set to reduce the impact of process variations on the performances of the analog circuit. Stripped Functionality Logic Locking (SFLL) is used as logic locking method that was presented in [56]. Given that SFLL only protects a subset of possible input patterns, the protected patterns and the tuning knobs must be carefully selected so that no approximate functionality can be established with an incorrect key or that the performance degradation due to an incorrect key is acceptable for the attacker. Furthermore the technique suffers from the threat posed by a removal attack, i.e., the removal of the protected optimizer block. This problem arises since (a) the protected block is not in the signal processing chain, thus removing the calibration loop altogether renders the circuit functional with reduced performances, since the tuning knob values will remain in their default settings [55] and (b) given the simplicity of the locked optimizer block, the design of a replacement circuit is easy to achieve.

2.2.1.2 *Analog Biasing Locking*

The majority of analog locking techniques propose the insertion of a lock into the biasing circuitry of the analog IC. Shown in Fig. 2.1 (a) is the standard configuration of an analog core circuit being biased with currents or voltages, all of which are generated in the biasing block. Providing the correct biases to the analog core is essential, given that the core will only function as foreseen if operated in the regime, i.e., the DC operating point, it was originally designed for.

The underlying principle of biasing locking, on the other hand, is to redesign the biasing circuits so that they present a non-well-behaved programmability via a key, normally in the form of a k-bit digital key. An incorrect key offsets the biasing, leading to functionality corruption of the analog circuit in that one or more performances are pushed outside their specified range. For biasing locking to be effective, the locking mechanism must (a) allow the key size k to be large enough to make brute-force attacks fail and (b) assure no incorrect keys can transition the analog circuit in a sufficiently well-performing operating state. Two types of approaches can be found in the literature, namely expanding the biasing circuit, as shown in Fig. 2.1 (b) [57], [58], and the design of a standalone block that generates the desired bias, as shown in Fig. 2.1 (c) [59], [60].

In [57], the authors apply the principle of *expanding the biasing* on current biasing transistors. The authors suggest replacing a current source

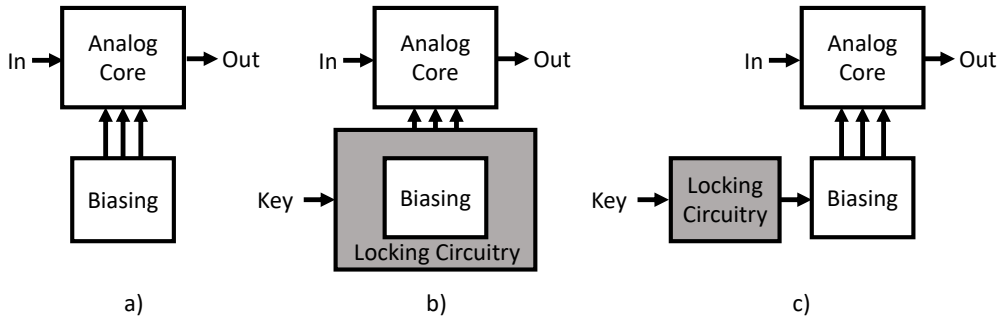


Figure 2.1: An analog core circuit having its operating point set by a biasing circuit shown in (a). Analog biasing locking via the expansion of the biasing circuit [57], [58] shown in (b) and via the design of a standalone block [59], [60] shown in (c).

transistor, such as Y in Fig. 2.2 (a), with an array of parallel-connected transistors. The number of conducting transistors is controlled by a key, which controls the gate voltage of each transistor. Thereby, in the case of a correct key, the aggregate width of the active transistors equals that of the replaced transistor Y, whereas a wrong key can lead to a different aggregate width, generating an incorrect current. The approach in [57] is problematic since (a) no care is taken to assure that only one key leads to the correct biasing current and (b) it is not assured that all incorrect keys result in sufficient performance degradation. The authors try to eliminate both drawbacks in [61] using a Satisfiability Modulo Theory (SMT) solver, shown before in [58] and explained below.

In [62], extending [57], a transistor is replaced with a number of parallel and series-connected transistors, thereby creating a mesh. The authors propose sizing each transistor in the mesh differently and controlling each transistor with a key-bit. The aggregate width and length of such a mesh of transistors is then a function of the key.

In [58], Wang *et al.* propose a locking method based on controllable current mirrors that is exemplified in Fig. 2.2 a) and b). A non-locked current mirror transistor Y is locked with 4 key-bits q_i, $i = 1, \cdots, 4$ by replacing Y with an array of 4 branches. Each branch consists of an (analog) NMOS current mirror transistor X_j with an individual current mirroring ratio α_i and a pass device Q_j, controlled by key-bit q_j. The current I_B the array drives depends on which pass devices are "on" and conductive and thereby on the key. The authors suggest introducing a number of pass devices per branch and, additionally, the use of PMOS current mirror devices to ensure a non-monotonic relationship between key and I_B. An SMT solver is leveraged to find a configuration of mirroring ratios α_i and pass-device connectivity so that I_B lies outside a designer-defined range for all but the correct key.

In [59], Hoe *et al.* propose securing a sense amplifier by using memristor-based circuitry to program a matched transistor pair's body-biasing volt-

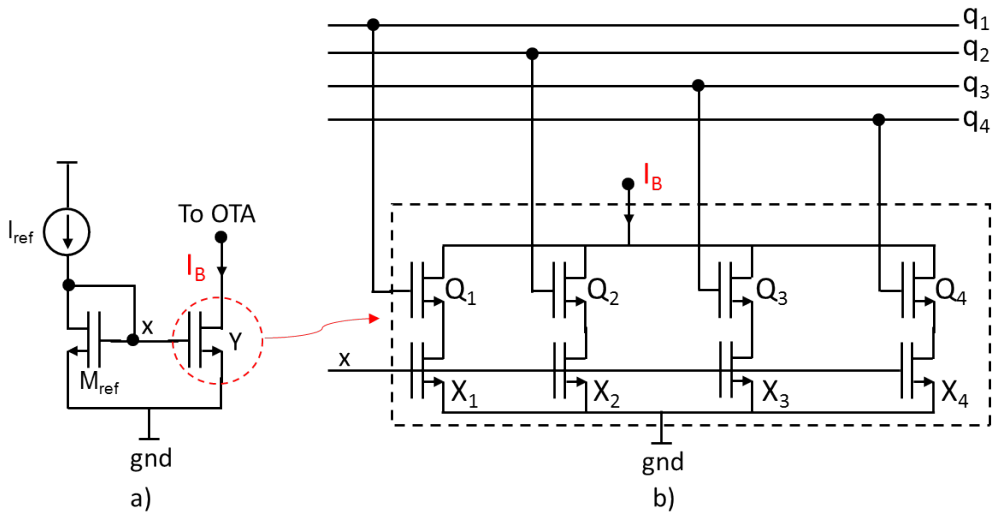


Figure 2.2: Locking a current mirror using the technique from [58].

age, used to eliminate the offset of the amplifier arising due to process variations. A memristor crossbar is programmed via a secret key allowing in turn the programming of a voltage divider that generates the body biasing. When applying an incorrect key the breakdown voltage of the emerging technology device can be reached, thus limiting the number of trials an attacker has for an attack. Other than the use of an eccentric, hard-to-integrate technology, the technique's requirement to program every single memristor in a multi-step process prior to activation will result in prohibitive efforts during activation and calibration.

In [60], the authors propose the implementation of an on-chip neural network. Receiving as input an analog key in the form of DC voltages, the neural network in turn generates output biases. In the case-study, the authors lock an RF amplifier requiring three bias voltages using this approach. The neural network is therefore trained to present an impulse function at the correct key, i.e., only when the correct input voltages are provided at the key inputs will the neural network generate the correct biases at its output, while otherwise remaining constantly at an incorrect level. Storing precise analog voltages for key and the model's weights is a challenge and the authors propose storing the model's weights in (non-volatile) AFGTs and providing the key directly, via pins, as digital key in combination with a Digital-to-Analog Converter (DAC) or AFGTs. This work presents the first application of an analog key to lock an analog/RF circuit in the literature.

Overall, analog biasing locking can be considered an attractive defense for analog circuits, since (a) it can be applied to all analog circuits, because they all require biasing; (b) analog circuits are sensitive to incorrect biasing, providing a leverage for their protection; (c) the methods in Fig. 2.1 (b) and (c) are applied outside the sensitive analog core circuit, thereby

avoiding performance penalties if a method were to operate within the core circuit. Still, these defenses do not provide holistic security, given that they can easily be identified and removed by an attacker, a weakness that will be elaborated in Chapter 5.

2.2.1.3 *Compound Techniques*

The authors in [63] propose a compound technique where a mixed-signal circuit is locked via logic locking of its digital part and analog biasing locking of the analog section via the technique proposed in [57]. The authors argue that making analog and digital section receive their key from the same key-bits, and thus sharing an identical key, stops an attacker from breaking the analog and digital block's locking mechanism independently.

2.2.2 *Obfuscation*

In [64], Ash-Saki *et al.* take a different approach and apply methods of physical design obfuscation to hinder an attacker from reverse engineering an analog design. In analog IC processes, multiple types of transistors are available in Process Design Kits (PDKs), defined by different threshold voltages V_{th}, such as high, normal and low V_{th}. The authors suggest replacing a number of normal-V_{th} transistors in a design with low or high V_{th} transistors and redesign the concerned circuit blocks to meet their specifications. In a reverse engineered design the analog transistors do not reveal their V_{th}, since the threshold voltage is a function of channel doping, which is not visible through classic optical imaging. The reverse engineering process thus reveals a partially ambiguous design to the attacker. The search space for the attacker for, e.g., 3 V_{th} flavors, is 3^n, while n is the number of transistors that are potentially obfuscated. n is obtained after deducting from the total transistor count those transistors that must, for instance, be of the same type or require a certain V_{th}, such as matched transistors or certain switches. Overall, this defense mechanism is expensive for the IC owner since he will have to invest additional design time and possibly accept performance penalties due to changes in the analog core.

2.2.3 *Split Manufacturing*

In [65], Bi *et al.* suggest the application of split manufacturing to analog and RF designs. The principle of split manufacturing is to divide the manufacturing of ICs between two foundries: an untrusted front-end-of-line (FEOL) foundry fabricating the lowest layers of the IC, including transistors, and a trusted back-end-of-line (BEOL) foundry fabricating

the IC's upper layers, such as metal layers and interconnects. The FEOL foundry only receives a reduced layout, containing information merely up to a certain metal layer, up to which it processes the IC. The trusted BEOL foundry thereafter completes the manufacturing step by processing the upper metal layers of the IC. Thereby, a rogue FEOL foundry is stripped of the necessary information to pirate or reverse engineer a design. In RF designs, contrarily to digital design, metal layers not only serve for establishing inter-device connectivity, but also for forming electrically active devices such as capacitors or coils. The authors therefore argue that an untrusted foundry not knowing coil and capacitor sizings, grants a higher level of security of split manufacturing for RF, than for digital designs. Process design rules, such as the exclusion of metal layers below coils and capacitors, on the other hand, may provide hints about size and location of said devices, which leads the authors to create empty zones in the original design, thereby obfuscating the reduced layout meant for the rogue FEOL foundry.

2.2.4 *Recycling Protection*

In [66] the authors propose two types of sensors for deducing the age of an IC. The first sensor is based on the implementation of two ROs on an IC, the first RO being always-on, thus aging and reducing its frequency, whereas the second RO is only turned on when serving as a reference to compute the frequency difference between the two oscillators. The greater the frequency difference, the longer the IC has been in service. The second class of aging sensors is based on anti-fuse, one-time programmable memory, where in certain intervals, such as a number of clock cycles, an anti-fuse is programmed, allowing for deduction of the chip's operating time at a later point.

In [67], Huang *et al.* propose detecting recycled ICs with the help of statistical tools in the form of one-class classifiers and degradation curve sensitivity analysis (DCSA). Both methods are based on silicon aging effects that make parametric measurements of an IC, such as quiescent current and maximum oscillation frequency [44], drift with the time the IC is in use. Using a reference point in time, the methods try to detect recycled ICs. The DCSA approach is more robust to the effects of process variation, since instead of using absolute parametric measurements, it uses their sensitivity. This is advantageous for detecting recycled chips, as the performance degradation of a brand-new chip is distinct from, and normally higher than, that of an already used chip. At the same time the degradation itself is mostly independent of process variations, thereby allowing to compare different chips, even in the presence of these variations.

The authors of [68] suggest the implementation of hardware sensors that allow the deduction of the age of an IC based on aging effects that appear due to electromigration. Leveraging an aging model for interconnect wires under DC current stress enables the authors to design wire structures that will fail, i.e., increase their resistance, after a predictable amount of time.

3

MIXLOCK: SECURING MIXED-SIGNAL CIRCUITS VIA LOGIC LOCKING

This chapter deals with hardware trust and security aspects specifically for mixed-signal ICs, which is a large subclass of analog ICs, including data converters, Phase Locked Loops (PLLs), Radio Frequency (RF) transceivers, etc. In particular, we develop a locking methodology for mixed-signal ICs, called *MixLock* [69], that prevents IC piracy. Locking aims at transforming the original design into one that is functionally equivalent, but requiring a secret key to unlock the correct functionality. Applying an invalid key will result in dramatically degraded mixed-signal performance.

3.1 INTRODUCTION TO MIXLOCK

In *MixLock*, locking is achieved via logic locking (aka logic encryption) of the digital section of the mixed-signal IC. For this purpose, we employ two recent logic locking techniques, namely Stripped Functionality Logic Locking (SFLL) [56] and Dishonest Oracle (DisORC) in conjunction with Truly Random Logic Locking (TRLL) [70]. Metrics are proposed to quantify the analog security level, i.e., the mixed-signal functionality corruption for invalid keys. The digital security level is expressed in terms of resilience to all known logic locking attacks. We show that *MixLock* is capable of co-optimizing security in the analog and digital domains. In addition, *MixLock* presents several appealing properties. It is non-intrusive for the analog section, it incurs reasonable area and power overhead, it can be fully automated, and it is virtually applicable to a wide range of mixed-signal ICs. We demonstrate *MixLock* on a $\Sigma\Delta$ ADC via simulations and in a hardware experiment. Finally an audio demonstrator using *MixLock* in the signal processing chain of an audio application is shown.

The chapter is structured as follows. In Section 3.2, we provide an overview of *MixLock*. In Sections 3.3 and 3.4, we discuss the threat model and the attack resilience analysis. In Section 3.5, we discuss logic locking. In Section 3.6, we present our simulational case study, while in Section 3.7 we present our hardware experiment. In Section 3.8 we discuss our demonstrator for *MixLock* in an audio application. Finally, Section 3.9 concludes the chapter.

3.2 THE PROPOSED TECHNIQUE: MIXLOCK

MixLock aims at locking a mixed-signal IC via a logic locking mechanism embedded into its digital section, as illustrated in Fig. 3.1. Only when the valid secret key is provided, referred to as common key K_C, the correct functionality is unlocked, that is, the digital section implements the correct function for any input. Logic locking will be discussed in more detail in Section 3.5.

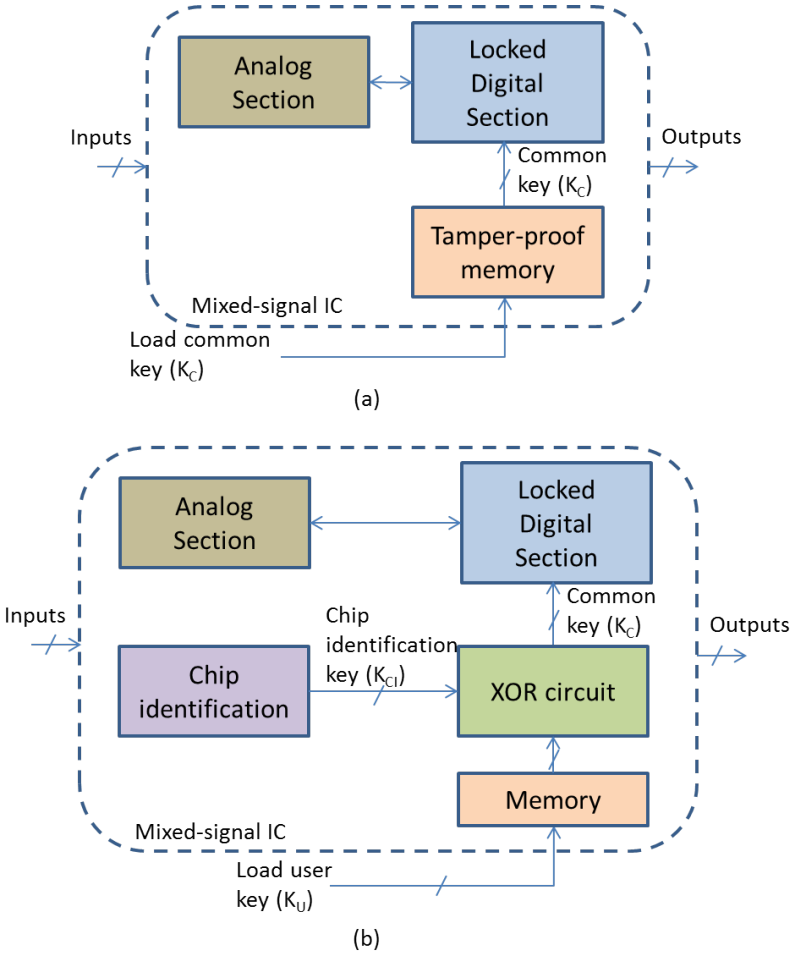


Figure 3.1: Mixed-signal IC locked with *MixLock*.

The common key K_C can be stored directly in a tamper-proof memory, as shown in Fig. 3.1(a) [71]. Alternatively, the locking system in Fig. 3.1(b) can be used [72], which employs a chip identification key K_{CI} that is unique for every IC and can be generated, for example, by a Physical Unclonable Function (PUF) [73]. The unique user key K_U for the IC that is finally given to the user is the one that, when XORed with the chip identification key K_{CI}, generates the common key K_C. The common key K_C and chip identification key K_{CI} are kept secret and should not be shared with an untrusted party.

The digital section is typically part of a signal-processing chain or part of a feedback loop and, according to the mixed-signal IC type, can perform different whole functions or subfunctions. The underlying idea is that logic locking of the digital section becomes a means for corrupting the mixed-signal IC performance trade-off. The objective is that unless the valid key is provided, the performance trade-off is locked, that is, one or more performances lie outside their acceptable specification range.

MixLock presents several appealing properties:

- NON-INTRUSIVE. It is non-intrusive since it does not alter the analog section and since any performance degradation in the digital section can be easily absorbed with no degradation in the mixed-signal performance. This is key for its wide adoption by analog designers.

- LOW-OVERHEAD. Typically die area and power consumption in a mixed-signal IC is largely dominated by the analog section. The area and power overhead in the digital section introduced by logic locking is already affordable considering the digital section alone; this overhead, when projected to the entire mixed-signal IC, will be even easier to justify.

- FULLY AUTOMATED. Typically, design-for-X (DfX) techniques for mixed-signal ICs, where "X" can be test, reliability, calibration, diagnosis, etc., require significant extra design effort. DfX also needs to be revisited for every new product or new technology node. In contrast, the proposed Design-for-Trust (DfTr) *MixLock* technique is fully automated since it is based on logic locking of the digital section, which is fully automated.

- WIDE APPLICABILITY. It can be virtually applied to a wide range of mixed-signal ICs that have a large digital section, including data converters, PLLs, RF transceivers, etc. It also fits well the general trend towards digitally-assisted analog designs and digital centric mixed-signal architectures, where the goal is to make a thoughtful shift of functionality from the analog into the digital domain, in order to alleviate analog design complexity and enable post-manufacturing tuning, self-calibration, and reconfigurability.

3.3 THREAT MODEL

MixLock is intended to serve as a countermeasure against mixed-signal IC piracy, which can be broken down into several distinct threats. These threats, the assumptions on the capabilities of the adversary, and the conditions under which *MixLock* delivers resilience are described next.

- REVERSE ENGINEERING. We assume that the adversary has full capabilities to extract the architecture, netlist, layout, etc. Even in this case, the adversary will not be able to reveal the exact functionality as the common key is unknown. Of course, a smart adversary can quickly realize by tracing the key bits structurally that the digital section is locked. In this case, the digital section can be removed and replaced with a "fresh" one with no locking mechanism. But this requires that the adversary has the required design expertise and is willing to spend some significant design effort, given also that the design of the digital section is closely tied to that of the analog section. The requirement for significant redesign effort clearly goes against the original incentive of the adversary; thus, *MixLock* provides good resilience against this threat.

- CLONED COUNTERFEITS. The common key is unknown; thus, with *MixLock* in place, a cloned counterfeit is practically unusable.

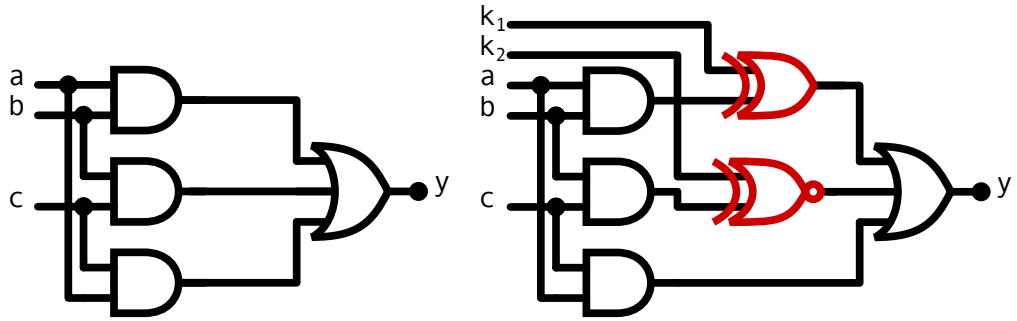
- RECYCLED COUNTERFEITS. *MixLock* does not provide any protection, unless the scheme in Fig. 3.1(b) is used and the user key is reloaded every time the IC is powered up. But arguably there are simple techniques to detect recycled ICs, for example, through the use of on-chip lightweight sensors [74].

- OVERPRODUCED COUNTERFEITS. *MixLock* provides protection as long as one of the following approaches is used: (a) The test is performed in a trusted test facility; (b) The ICs after fabrication are sent from the fab to the trusted party, i.e., the design house, that loads the common key using the scheme in Fig. 3.1(a) and sends them back to the untrusted facility for testing; (c) The trusted party remotely activates the chip for testing using asymmetric cryptography [75].

3.4 SECURITY ANALYSIS

The security level of *MixLock* is defined in terms of security level of the underlying logic locking, called *digital security level*, and the security level of performance trade-off locking, called *analog security level*. An attacker can try to unlock explicitly the digital section without caring about the analog section, or can try to unlock directly the performance trade-off, i.e., achieve a satisfactory performance trade-off, which perhaps can be achieved with an incorrect common key.

The digital security level is measured by the effort that the designer must spend for identifying the common key. It is dictated by the resilience against known attacks, as will be explained in more detail in Section 3.5.



(a) Original circuit.

(b) Locked circuit modified to contain two key-gates. The correct key restoring functionality is $k_1 = 0$, $k_2 = 1$

Figure 3.2: Simple application of logic locking.

We propose to measure the analog security level using different metrics in the analog domain, namely, *error rate*, *mean absolute error*, and *minimum error*. *Error rate* is the percentage of incorrect keys resulting in violation of one or more performance specifications. The *error rate* may be misleading if for incorrect keys the violated performance(s) is (are) slightly outside the(ir) specification(s). To account for this scenario, we also use the *mean absolute error* metric defined as the average absolute performance difference between the unlocked mixed-signal IC and locked versions. *Minimum error* is the minimum observed performance difference between the unlocked mixed-signal IC and locked versions, indicating the *worst-case* locking. These metrics can be quantified by putting to a test a large set of random incorrect keys.

Attacks to unlock directly the performance trade-off are not known at this point. A possible scenario is that the attacker uses optimization algorithms, such as gradient descent, simulated annealing, etc., to search for a common key that brings the performances within the acceptable specification range. Such an attack is very unlikely to succeed since the mixed-signal performances are not related to the key through a well-behaved and smooth function. The optimization is likely to “zigzag" endlessly. Especially if the key width is large and if the *minimum error* defined above is large, this attack is doomed to fail.

3.5 LOGIC LOCKING

Logic locking protects the digital circuit by modifying it and adding new logic into it, such that its functionality is controlled by a key with Fig. 3.2 showing a simple example of the modification. The earliest traditional logic locking techniques, e.g., Random Logic Locking (RLL) [75], Fault analysis-based Logic Locking (FLL) [76], and Strong Logic Locking (SLL) [77], aimed at inserting key gates into the design that are

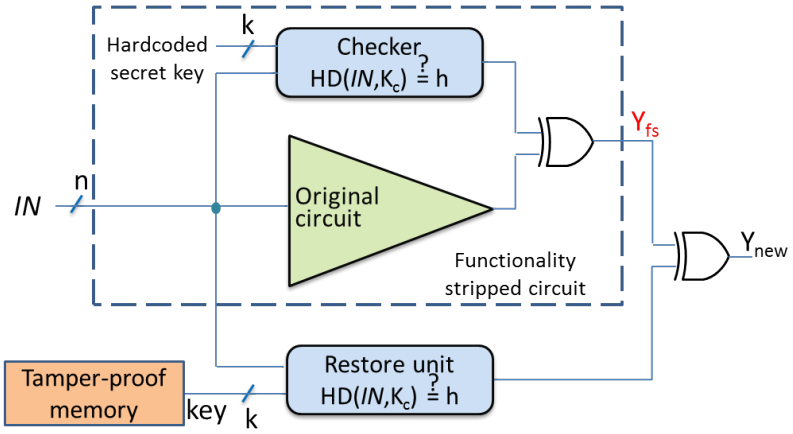


Figure 3.3: SFLL architecture.

controlled by key-bits, which compose the key. The best key gate locations are determined while balancing the security objectives and the implementation overhead. However, the SAT attack [78], which is based on a Boolean satisfiability solver, was able to break all these techniques and recover the secret key with very reasonable effort. Techniques to thwart SAT attack, e.g., SARLock [79] and Anti-SAT [80], were shown to be susceptible to removal attacks [81], which aim to identify and isolate the protection logic. These SAT-resilient techniques can be combined with traditional logic locking for improving the output corruptibility [79], yet such integration can be circumvented using the approximate attacks, e.g., AppSAT [82] and Double-DIP [83], which reduce the security level down to the one provided by the SAT-resilient technique and extract a key that establishes an incorrect but approximate functionality.

3.5.1 *Stripped Functionality Logic Locking*

A powerful logic locking technique is SFLL [56] that achieves holistic security against SAT, removal, and approximate attacks in a quantifiable manner. As illustrated in Fig. 3.3, in the SFLL architecture, a part of the original circuit functionality is stripped away using a *checker*. In particular, for all input patterns that are Hamming distance (HD) h away from the secret key, the output of the original circuit is flipped. Only upon supplying the valid secret key, the *restore unit* cancels the errors introduced by the *functionality-stripped circuit*, recovering the original output.

Let k be the number of key-bits composing the key and let n be the number of inputs of the digital circuit. It can be shown that SFLL is $slSAT$-secure against SAT attack, where $slSAT = k - \lceil \log_2 \binom{k}{h} \rceil$, meaning that the SAT attack effort required to extract the secret key is equivalent to breaking a $k - \lceil \log_2 \binom{k}{h} \rceil$-bit key in a brute-force way [56]. It can also

be shown that SFLL is $slREM$-resilient against removal attacks, where $slREM = \binom{k}{h} \cdot 2^{n-k}$ is the number of protected input patterns, meaning that the larger the number of protected input patterns, the more intricate the changes to the original logic are, and, thereby, the harder it is for the removal attack to succeed [56]. Finally, it can be shown that SFLL is $slAPX$-resilient against approximate attacks, where $slAPX = \frac{\binom{k}{h}}{2^k}$ is the error rate [56], meaning that the higher the error rate, the more difficult it is to find a key that establishes approximate functionality. Therefore, SFLL allows a designer to trade-off the desired security level against different attacks by choosing appropriately k and h [56].

3.5.2 *Dishonest Oracle and Truly Random Logic Locking*

The most powerful attacks on logic locking are oracle-guided [70], where an oracle is a fully functional and unlocked chip. In an oracle-guided attack on a logic locked digital circuit it is generally assumed that an attacker has (i) access to the locked circuit's netlist; (ii) access to the primary inputs and outputs of the oracle; (iii) access to the oracle's test infrastructure, i.e., the scan chains.[1] As a consequence of (iii) the attacker can apply test patterns to the scan chains, wich grants him deep access to the design's internal flip-flops. Oracle-guided attacks, such as SAT-based attacks, make massive use and thus require access to a design's scan chains.

On the other hand, mixed-signal ICs do not always include scan chains into their digital section. When the amount of logic is large enough, using scan chains is a recommended test practice so as to increase defect coverage and diagnosability. However, even then, scan chains are not always used for various reasons. For example, the mixed-signal IC will be, after all, tested as a whole; inserting scan chains affects speed; many analog designers are not familiar with scan chains, etc. If scan chains are absent, only oracle-less attacks apply, easing the requirements on the logic locking technique and thus allowing the implementation of traditional logic locking techniques to achieve strong analog security with small overheads. If scan chains are present on the other hand, then achieving strong digital security becomes another dimension of the problem.

A new logic locking method named Dishonest Oracle (DisORC) was recently proposed in [70] and its concept is shown in Fig. 3.4. DisORC protects digital circuits against oracle-guided attacks, even in the presence of scan chains, through the addition of circuitry around the logic locked design. The technique introduces a test mode that revokes an attacker's access to the oracle by withdrawing the secret key from the locked circuit

1 A scan chain is a design-for-test methodology which allows reading and writing all of a digital circuit's internal registers.

upon activation of the circuit's test mode, i.e., when access to the scan chains by setting the *Scan-enable* bit is detected. As soon as *Scan-enable* is high, the *Corrupt* signal will follow, making the key-multiplexer select another, user-supplied key that is applied to the locked digital circuit. While scan chain access is now allowed, the incorrect, user-supplied key leads the digital circuit to present a modified functionality. The user or attacker may thus probe and structurally test the digital circuit, but has lost access to the oracle. Once *Scan-enable* is set, the only way to recover the circuit's correct functionality is by resetting the chip via the *Chip Reset* signal or by restarting the chip.

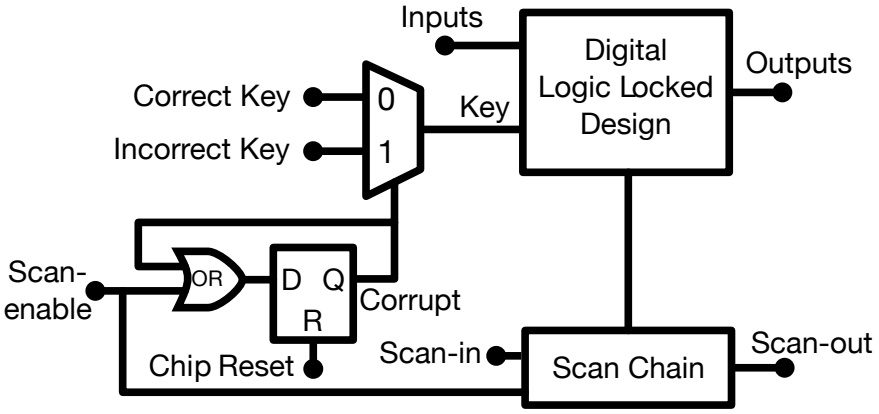


Figure 3.4: DisORC architecture.

While DisORC "cripples" the oracle, it does not modify the digital core circuit itself, but is rather placed around the circuit. Therefore a logic locking technique is required to lock the core circuit itself and thereby ensure functionality corruption for any incorrect key. Traditional logic locking techniques such as RLL [75], FLL [76], SLL [77] ensure good output corruption, but recently machine learning-based, oracle-less attacks were proposed on these schemes which may recover the key-value only from a locked netlist [84], [85]. A naive integration of these locking schemes together with DisORC may thus leave the circuit susceptible to these attacks. For that reason we complement DisORC with a logic locking technique called Truly Random Logic Locking (TRLL) [70] that allows thwarting oracle-less attacks. TRLL's leitmotif is that it is truly random, i.e., there is no methodology determining where key-gates are inserted in the digital circuit and the type of key-gate inserted (e.g., XOR followed or not by an inverter) does not allow any conclusion to be drawn about the corresponding key-value. Thereby, in a TRLL-locked netlist no patterns or structures are left behind that an attacker could learn or infer information about the key from, and thus, contrarily to other schemes such as RLL, decouples the key-bit values from the key-gate types.

3.6 SIMULATION RESULTS WITH SFLL

3.6.1 *Case Study*

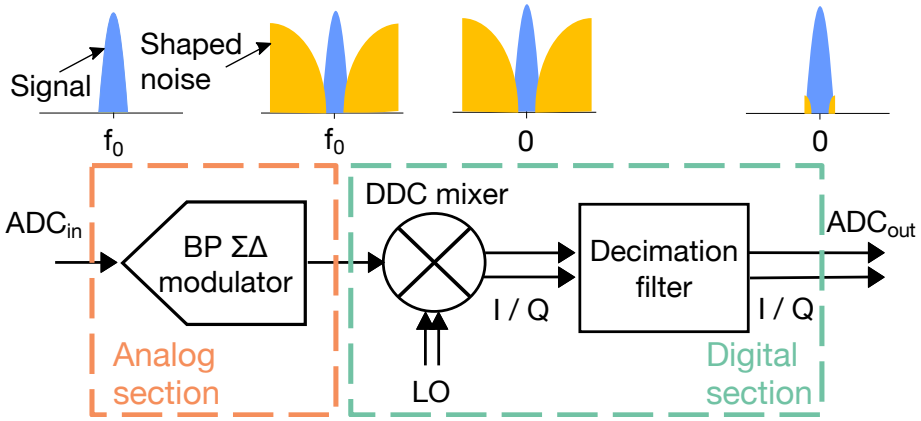


Figure 3.5: The Bandpass ΣΔ ADC used as case study.

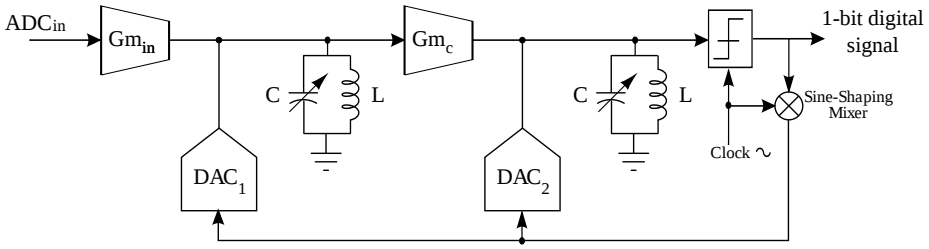


Figure 3.6: The analog 4th order LC bandpass ΣΔ modulator.

To demonstrate *MixLock*, we used as case study a bandpass (BP) ΣΔ ADC which converts a band $B = 25\,\text{MHz}$ centered at $f_0 = 2.4\,\text{GHz}$ with a sampling frequency $f_s = 3.2\,\text{GHz}$ [86]. The block-level schematic is shown in Fig. 3.5.

The analog section of the ΣΔ ADC is a 4th order LC BP ΣΔ modulator, shown in Fig. 3.6. The ΣΔ modulator converts the analog input signal to an oversampled low-resolution 1 bit digital signal with frequency f_s.

The digital section of the ΣΔ ADC, shown in Fig. 3.7, is composed of a Digital Down-Conversion (DDC) mixer and a multi-stage multi-rate decimation filter. The decimation filter removes the out-of-band noise from the ΣΔ modulator output and down-samples it, with a factor $\text{OSR}=\frac{f_s}{2B} = 64$, to convert it to a high-resolution 28 bit digital signal sampled at the Nyquist rate. The decimation filter is composed of a comb filter COMB, a first half-band filter HBF1, and a second half-band filter HBF2, with down-sampling factors of 16, 2, and 2, respectively. The DDC mixer offers an additional down-sampling factor of 2, thus the total down-sampling factor is 128.

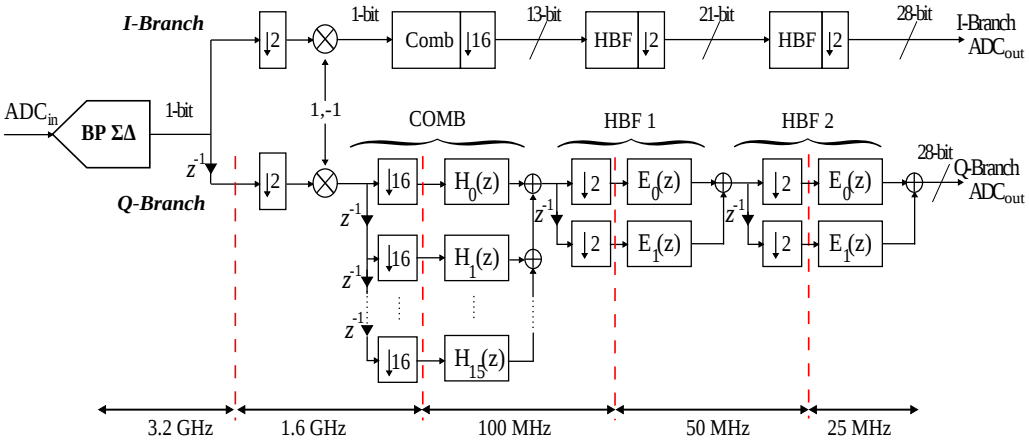


Figure 3.7: The digital DDC mixer and decimation filter.

The aim is to lock the SNR performance of the ΣΔ ADC via locking its digital section. Any incorrect key should result in an SNR performance that violates the specification, rendering the ΣΔ ADC unusable. The ΣΔ ADC has a nominal SNR of 70 dB with a specification set at 65 dB.

3.6.2 *Setup*

A pure sinusoidal signal with $f_{in} = f_0 + \Delta f$, where $\Delta f = 20\,\text{kHz}$, is applied at the input of the ΣΔ ADC and 2^{20} samples are recorded at the output of the ΣΔ modulator, which is simulated via a system-level model in Simulink, corresponding to 13 input signal periods. This recorded output bitstream is used as input to the digital section.

We study several logic locking techniques in terms of their impact on digital and analog security. The digital section is transformed into the locked digital section at RT-level VHDL. The locked digital section is synthesized using the Encounter RTL Compiler with a 65 nm CMOS low-threshold voltage library and appropriate timing constraints, in order to compute estimated area, power consumption, and performance overhead compared to the original version.

Evaluating the SNR for a given key involves loading the key into the digital section, simulating the digital section at RT-level VHDL using the recorded ΣΔ modulator output bitstream as input, and performing a Fast Fourier Transform (FFT) at the output of the decimation filter to calculate the SNR.

The analog security level metrics defined in Section 3.4 are calculated based on 10^3 randomly generated incorrect keys. This calculation is fully automated and evaluating a locked decimation filter takes around 40 minutes on a Intel(R) Core(TM) i5-4690 CPU @ 3.5 GHz with 8 GB of RAM.

3.6.3 *Results & Security Analysis*

We implement the SFLL technique [56], using a secret key of $k = 128$ bits. With SFLL, we lock the Most Significant Bit (MSB) of the COMB filter's output. In this context, locking a bit line means that we strip the functionality of the sub-circuit that drives the bit line. We selected this bit line aiming to introduce high functionality corruption early in the digital signal processing chain. We chose $h = 15$ since it leads to a strong 64 bit resilience against the SAT attack, and also strong resilience against removal and approximate attacks, according to the formulas in Section 3.5.1.

For this choice of SFLL parameters, we observed that not all secret keys result in functionality corruption that is high enough to achieve 100% error rate. This can be explained by the fact that SFLL protects a subset of input patterns, as discussed in Section 3.5. For a sinusoidal input signal, such as the one specified in Section 3.6.2, the number of input patterns generated at the input of the functionality-stripped sub-circuit is limited and, thereby, for a random key it is likely that not enough of these input patterns are protected to achieve 100% error rate. To this end, we crafted a secret key to achieve 100% error rate for the selected sinusoidal input signal. The number of keys that meet this objective is very high and, in any case, the selected sinusoidal input signal based on which the key is crafted is unknown to the attacker. Note also that in a real application, inputs are not well-structured and well-behaved like a sinusoidal. For example, $\Sigma\Delta$ ADCs are the most popular choice for a variety of precision measurement applications and for voiceband and audio applications. Real-application signals are time-varying in nature, their spectral contents vary with time, they are rich in frequencies, etc. For these high-activity signals, the number of input patterns generated at the input of the functionality-stripped sub-circuit will be large, and, thereby, any key will result in recurrent functionality corruption, a property we will exploit in our audio demonstrator in Section 3.8. Note that in [56], SFLL with $h = 0$ was used to lock a microcontroller designed using the ARM Cortex-Mo microprocessor. Choosing $h = 0$ implies only one protected input pattern, but still this was enough to break functionality.

We also implement the basic RLL technique [75] using the same secret key and locking the same sub-circuit. We observed that RLL achieves 100% error rate regardless of the secret key that is chosen.

Fig. 3.8 shows the trade-off between the analog security level, defined using the *error rate* metric, and the digital security level against the most powerful and lethal SAT attack. As it can be seen, RLL results in *error rate* of 100%, but it offers no resilience against the SAT attack in the case where scan chains are present. For the SFLL technique, the different points on the curve are produced by varying h. For $h = 15$, the sweet

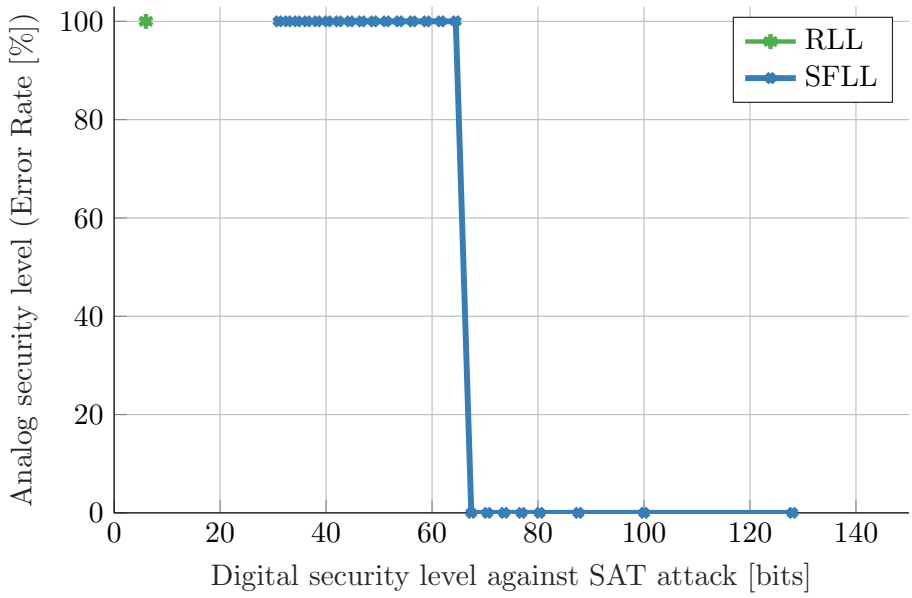


Figure 3.8: Trade-off between analog security level in terms of *error rate* and digital security level against SAT attack for different logic locking techniques.

trade-off point 100% error rate and 64 bit resilience against the SAT attack is obtained. Regarding the other metrics for analog security, we obtain 54.5 dB *mean absolute error* and 54.5 dB *minimum error*. Choosing $h > 15$ will increase functionality corruption and, thereby, will improve further the *mean absolute error* and *minimum error* metrics, at the expense of decreased resilience against the SAT attack.

To increase functionality corruption, we can alternatively implement a compound technique. For example, a first SFLL mechanism with $k = 128$ and $h = 15$ can be intertwined with a second SFLL mechanism with $k = 32$ and $h = 16$, which locks the MSB-1 bit of the COMB filter's output. We refer to this SFLL version as 1.5xSFLL. In fact, SFLL can be combined with other techniques too; for example, we can intertwine SFLL with RLL with $k = 32$. In theory, these compound techniques can be reduced to the first SFLL mechanism by applying AppSAT [82] or Double-DIP [83], as mentioned in Section 3.5, so they are appropriate only for the naive attacker.

Fig. 3.9 plots the SNR for 10^3 incorrect keys and the correct key using the 1.5xSFLL technique. The unlocked $\Sigma\Delta$ ADC stands out with a correct SNR of 70 dB. Locked versions have an SNR below the specification of 65 dB. Besides the 100% *error rate*, locking results in 70.6 dB *mean absolute error* and 65 dB *minimum error*. In fact, unless the correct key is provided, the input signal gets completely buried under the noise floor. Fig. 3.10 considers an arbitrarily selected incorrect key and compares the

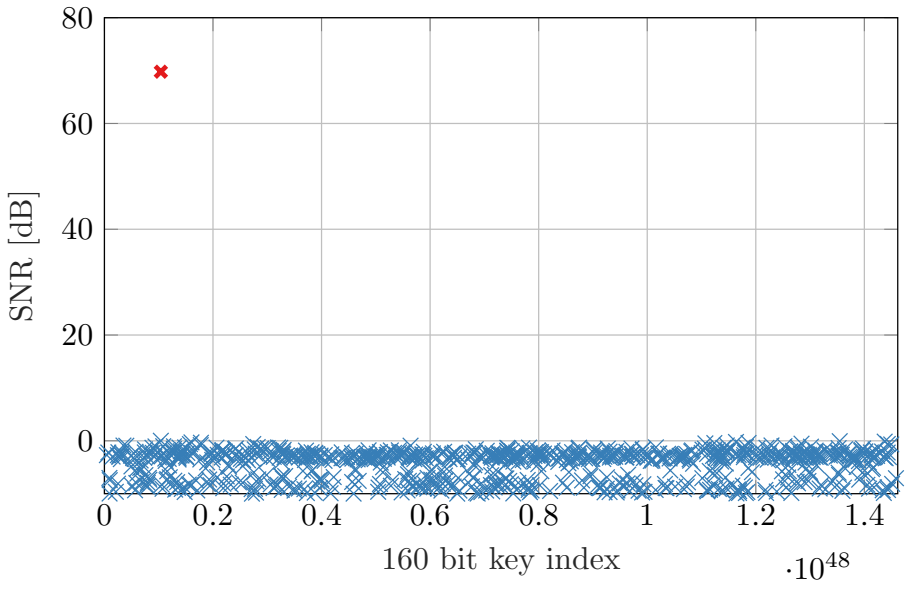


Figure 3.9: SNR for 10^3 incorrect keys and the correct key.

transient and frequency responses of the unlocked and a locked ΣΔ ADC. The locked ΣΔ ADC presents a large amount of glitches in its transient response, which translate to a high noise floor in the frequency response, resulting in corrupted SNR.

3.6.4 *Implementation Cost*

Table 3.1 shows the overhead using the different underlying logic locking techniques SFLL, RLL, 1.5xSFLL, and SFLL+RLL. *MixLock* incurs overhead only for the digital section. This overhead is projected to the entire ΣΔ ADC considering that the digital section occupies about 30% of the die area and is responsible for about 30% of the total power consumption. The slack reserve in the critical path of the digital section is large enough to accommodate the additional gates; thus, the delay penalty gets easily absorbed and does not translate to an SNR performance penalty. Regardless of the employed logic locking technique, the unlocked ΣΔ ADC has an SNR of 70 dB, that is, there is no performance degradation due to locking. If scan chains are absent, then the basic RLL technique can be used since it provides lower overhead compared to SFLL. If scan chains are present, then SFLL achieves optimal all-around analog and digital security levels with an area and power overhead of 6.7% and 9.8%, respectively, which are very reasonable. For higher functionality corruption, one can use 1.5xSFLL or SFLL+RLL at the expense of slightly higher area and power overhead.

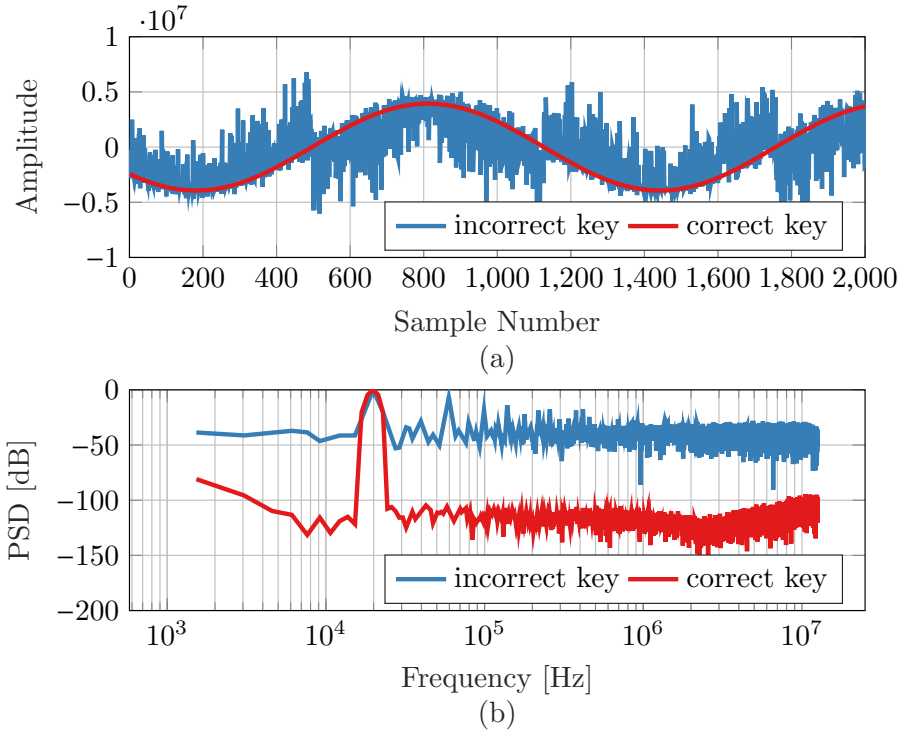


Figure 3.10: Transient and frequency responses of the 1.5xSFLL locked ΣΔ ADC for an incorrect and the correct key.

3.7 HARDWARE EXPERIMENT RESULTS WITH DISORC AND TRLL

3.7.1 *Case Study*

To prove the case for *MixLock* we implemented the technique in a hardware experiment where the silicon implementation of a state-of-the-art BP RF ΣΔ ADC, recently published in [87], served as case-study. The architecture of the ADC is the same as in Section 3.6.1 and is depicted in Fig. 3.5, except that the ΣΔ modulator in this study is a 2nd order LC BP ΣΔ modulator, shown in Fig. 3.11. Said modulator is implemented as packaged silicon IC, manufactured in a 65 nm CMOS process.

The ADC's digital section, depicted in Fig. 3.12, is identical to that of Section 3.6.1, although the processed signal frequencies are different. The decimation filter is implemented in hardware on a Xilinx Kintex-7 FPGA operating at a frequency of 187.5 MHz. Logic synthesis is carried out using the Nangate 45 nm open source cell library. The Nangate library's Verilog behavioral description is modified in a way so it is compatible with the following physical synthesis step by removing any non-synthesizable constructs such as timing statements and adapting flip-flop functionality definition. The final hardware implementation on

TECHNIQUE	SFLL	RLL	1.5XSFLL	SFLL+RLL
Digital Section				
Area [%]	20.1	5.6	24.4	21.1
Power [%]	29.5	9.3	35.3	30.9
Delay [%]	19.5	3.76	22.2	21.8
ΣΔ ADC				
Area [%]	6.7	1.9	8.1	7.0
Power [%]	9.8	3.1	11.8	10.3
Performance [%]	0	0	0	0

Table 3.1: Overheads using different underlying logic locking techniques.

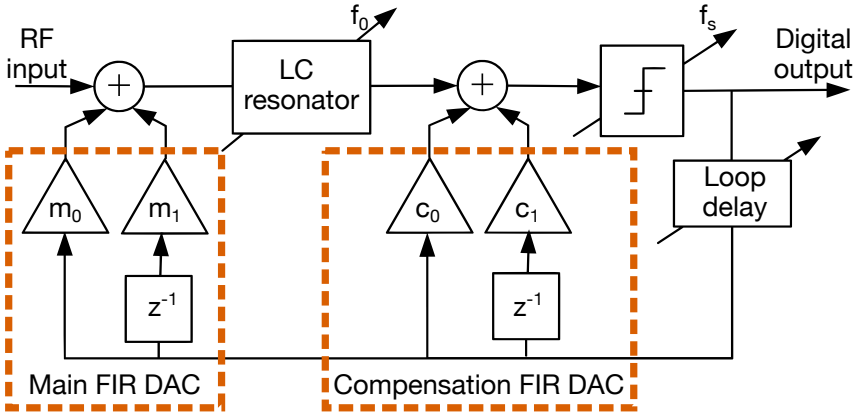


Figure 3.11: The analog 2nd order LC bandpass ΣΔ modulator used for the hardware experiment.

the Kintex-7 FPGA is carried out with Xilinx's Vivado Design Suite, Fig. 3.13 showing the FPGA implementation's architecture.

The complete ADC has a tunable center frequency f_0 from 1.5 GHz to 3.0 GHz and a corresponding sampling frequency f_s from 6.0 GHz to 12.0 GHz, converting a band of 47 MHz and 93 MHz centered around f_0, respectively.

The goal of this second study is to prove that locking the ΣΔ ADC's SNR is also feasible in hardware. This is achieved via locking of the ADC's digital section, thus any incorrect key should break the converter's SNR performance.

3.7.2 *Setup*

To allow for the generation of comparable results from hundreds of repetitions of the same experiment in which different keys are applied, we record the ΣΔ modulator's high-frequency output for a defined input

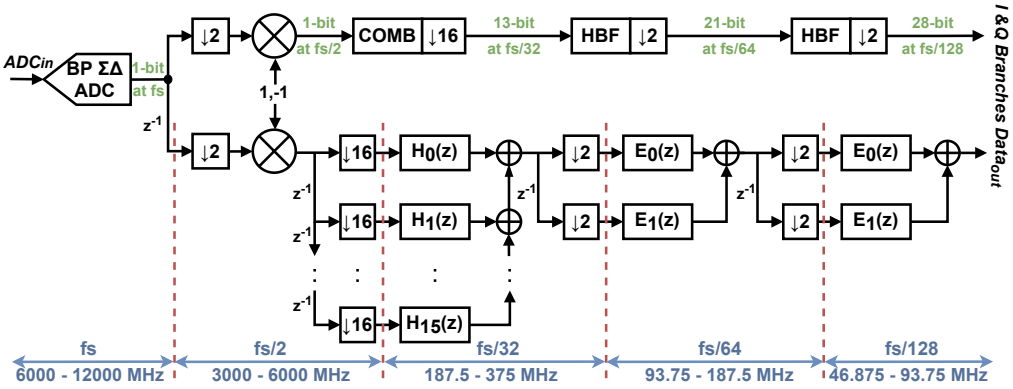


Figure 3.12: Architecture of the digital section, including DDC mixer and decimation filter.

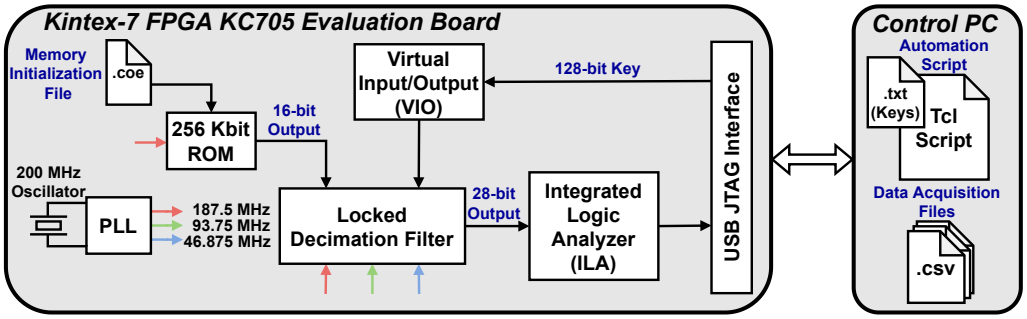


Figure 3.13: Implementation of the decimation filter of the ΣΔ ADC on a Xilinx Kintex-7 FPGA.

waveform. We excite the input of the ΣΔ ADC with a sinusoidal of frequency $f_{in} = f_0 + \Delta f = 1.5\,\text{GHz} + 300\,\text{kHz}$, corresponding to an f_s of 6 GHz. In a first step we then record 524 288 samples of the output bit-stream of the ΣΔ modulator, corresponding to 12 full input signal periods. Subsequently we down-convert the measured bit-stream to the baseband with the help of a system level DDC mixer.

This down-converted bit-stream is transformed to a memory initialization file and committed to the read-only memory (ROM) of the FPGA, as shown in Fig. 3.13. In each repetition of the experiment the ROM provides said bit-stream to the locked decimation filter which then processes it. A PLL generates the three different clock frequencies required for the decimation filter's sub-stages. Fetching the key, as well as the acquisition of the 28 bit decimation filter output, is achieved via a JTAG-USB interface with the control PC. After each experiment 4096 data samples are sent to the control PC for SNR evaluation.

A single experiment thus consists of the following steps:

1. The FPGA loads a key via the USB interface and applies it to the locked decimation filter.

2. The decimation filter processes the pre-recorded bit-stream which is retrieved from the ROM memory.

3. The integrated logic analyzer acquires the decimation filter's output and allows exporting it as a comma-separated values file to the control PC via the USB connection.

4. On the control PC a fast Fourier transform is performed on the captured data and its SNR is derived.

3.7.3 *Results & Security Analysis*

For our study we leverage the locking techniques DisORC in conjunction with TRLL, as presented in Section 3.5.2, to lock the digital section of the ΣΔ ADC. In particular DisORC and TRLL are applied at gate-level to lock the first sub-filter stage of the decimation filter, i.e., the comb filter. We implement a secret key k with 128 bit, which is, contrarily to the approach in Section 3.6.3, not crafted to present a sufficient amount of errors but instead chosen at random.

Given that the circuit is locked with DisORC it presents a very strong 128 bit security level against SAT attack while, thanks to TRLL and DisORC, it is not susceptible to known removal attacks.

With the circuit's SNR specification set to 30 dB we start the hardware experiment. The FPGA implementation of the digital section processes the same input signal for 1000 randomly generated 128 bit keys as well as the secret key. Running the hardware experiment for all keys on the Kintex-7 FPGA takes approximately 20 minutes with the entire experiment, except for the SNR computation, being automated. Fig. 3.14 shows the converter's performance for the above hardware experiment.

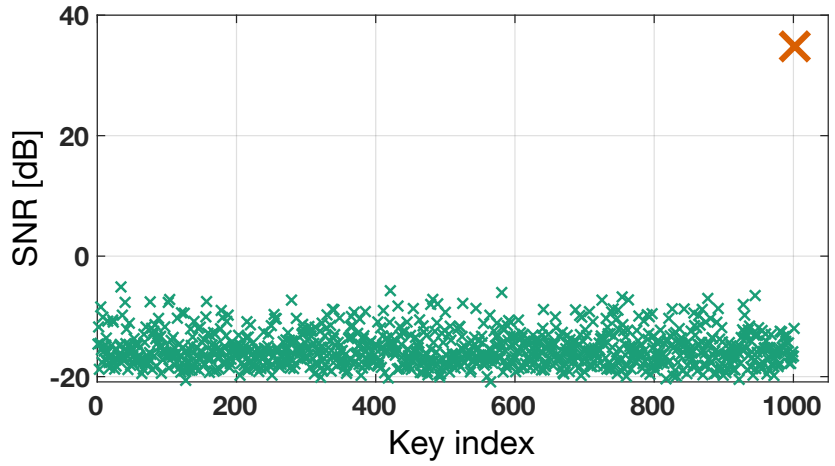


Figure 3.14: SNR for 10^3 incorrect keys in green and the correct key in orange applied to the DisORC & TRLL locked ΣΔ ADC hardware case-study.

METRIC	VALUE
Error rate	100 %
Mean absolute error	45.4 dB
Mean SNR of failing keys	−15.4 dB
Minimum error	35.1 dB

Table 3.2: Analog security metrics for 1000 random keys with respect to the SNR specification set at 30 dB for the hardware case-study.

We achieve the strong analog security metrics shown in Table 3.2, where notably a perfect analog security level of 100 % is achieved, i.e., any random key breaks the circuit's functionality. The SNR of the filter unlocked with the correct key is identical to that of the nominal, non-locked filter with 34.8 dB, thus locking the circuit does not induce any performance penalty.

Fig. 3.15 depicts the transient and frequency responses recorded at the ADC's output, where an incorrect key leads to important levels of distortion in the transient domain while in the frequency domain this distortion is visible in the form of an important increase in the noise level.

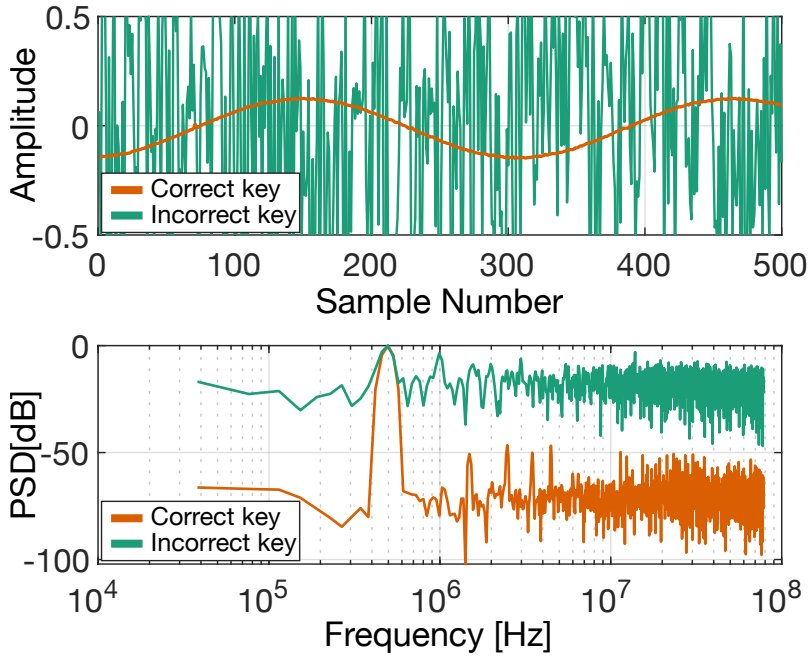


Figure 3.15: Transient and frequency responses of the DisORC & TRLL locked ΣΔ ADC measured for an incorrect and the correct key.

DIGITAL SECTION	OVERHEAD
Area [%]	10.7
Power [%]	21.3
Delay [%]	13.4

ΣΔ ADC	OVERHEAD
Area [%]	3.6
Power [%]	7.1
Performance [%]	0

Table 3.3: Overheads for locking the digital section with DisORC & TRLL when projected to the digital section (left) and when projected to the entire ADC (right).

3.7.4 *Implementation Cost*

Table 3.3 presents the overheads with regard to the nominal design that locking the ΣΔ ADC induces. On the left hand side the overheads with respect to only the ADC's digital section are given, on the right hand side the overheads are projected to the entire ADC. While we cannot compare the absolute numbers from the SFLL case-study with above's numbers, because the circuits are not identical, they still allow us to see that the DisORC technique in combination with TRLL does not cause any performance penalty in terms of SNR and results in overheads that are relatively lower compared to SFLL's overheads from Table 3.1.

3.8 MIXLOCK DEMONSTRATION IN AN AUDIO APPLICATION

3.8.1 *Demonstrator Configuration and Setup*

Hereafter we demonstrate *MixLock* in an audio application [88], in order to evaluate the impact of locking on a real-world application. We hereby aim to complement the prior results that had used ideal sine-wave inputs. In essence this demonstrator allows us to *listen to* the effect of locking. The methodology is illustrated in Fig. 3.16. An audio sample is read from the microphone of the PC and thereafter it is captured and sampled using the Matlab function `audioread`. Upsampling based on linear interpolation is used to artificially smoothen the signal so that it can be presented to the oversampling ΣΔ ADC for a second digitization. The output of the ΣΔ ADC can be heard directly from the speaker of the PC using the Matlab function `audiowrite`. The ΣΔ modulator in this demonstrator is a system-level model of a second-order low-pass continuous-time ΣΔ modulator. The modulator is modeled and simulated in Simulink, while the decimation filter is modeled with VHDL and is simulated in Modelsim.

For locking the decimation filter we use two methods presented in Section 3.5, namely SFLL and DisORC in combination with TRLL:

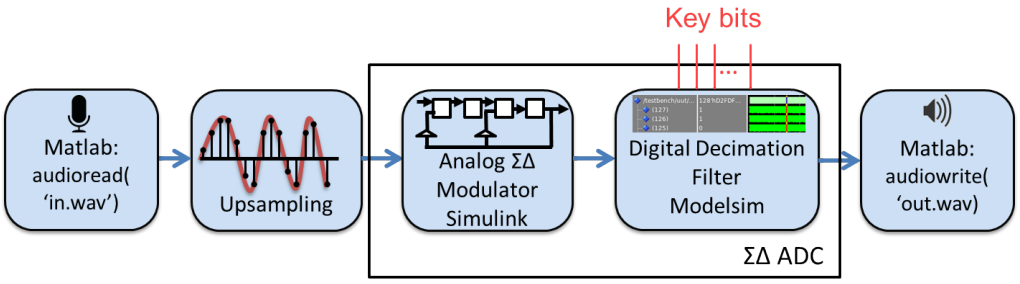


Figure 3.16: *MixLock* demonstration in an audio application.

- In the first application, we employ the same setup as was presented in Section 3.6.3. First, a single SFLL mechanism with $k = 128$ and $h = 15$ is used to lock the MSB of the comb filter's output within the decimation filter. This approach provides a 64 bit resilience against the SAT attack and sufficient resilience against removal and approximate attacks, as dictated by the formulas in Section 3.5.1. As another flavor of SFLL we also implement 1.5xSFLL, i.e., above's SFLL mechanism combined with a second SFLL mechanism with $k = 32$ and $h = 16$ that locks the MSB-1 bit of the comb filter's output. 1.5xSFLL increases significantly the number of protected patterns and the error rate, i.e., it increases functionality corruption. In theory, however, 1.5xSFLL can be reduced to the single SFLL mechanism, as put forward in Section 3.6.3, so it is appropriate only for the naive attacker.

- In the second application the mechanism used to lock the decimation filter is DisORC & TRLL, as elaborated in Section 3.5.2. It is applied to the comb filter so as to make the ADC require a 128 bit key to unlock it. We assume the attacker to have loaded a key and to have set the circuit in operating mode. The audio samples put to test are shown in Table 3.5. All samples are evaluated with the correct and two incorrect keys with Hamming distances of 12 and 63 with respect to the 128 bit correct key. While the former is quite favorably chosen for the attacker, the latter key with a Hamming distance of 63 is chosen at random.

In this demonstrator, the SNR metric cannot be used for quantifying analog security as it was used in Sections 3.6 and 3.7. The reason is that SNR requires a sinusoidal input, while audio signals are time-varying in nature; their spectral contents vary with time, they are rich in frequencies, etc. For this purpose, we use the root-mean-square error (RMSE) as metric

AUDIO SAMPLE	DURATION [S]	SAMPLE RATE [Hz]
English voice recording	4	8192
Bob Marley - No Woman No Cry	15	16384
Benny Goodman - Bugle Call Rag	15	16384
Kenny Ball - I Wanna Be Like You	15	16384
John Coltrane - Nature Boy	15	16384
Beethoven - Symphony No. 9	15	16384

Table 3.4: Audio samples processed with a ΣΔ ADC locked with *MixLock*

to evaluate the error between two waveforms processed by an unlocked and a locked converter for the same input:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} |x_i - y_i|^2}, \tag{3.1}$$

where N is the total number of samples in the processed waveforms, x_i is the i-th sample of the output waveform processed by the unlocked converter having the correct secret key applied and y_i is the i-th sample of the output waveform processed by a locked converter having an incorrect key applied.

While RMSE allows to quantify the error in the processed audio samples, it does not allow us to understand in what kind of way an audio sample is impacted, i.e., two RMSE values with similar magnitude may be derived from erroneous audio samples that *sound* differently. Even so, unless the valid secret key is applied to the converter, locking introduces errors that get translated into audible glitches or noise that deteriorate the output signal, which can be measured via the RMSE metric.

3.8.2 *Results*

Our experiment involves processing the audio samples listed in Table 3.4 through the system in Fig. 3.16 and examining the effect of locking on the audio quality. Audio samples include a speech recording in English and professional music recordings of various genres with different duration and sampling rates. Table 3.5 shows the results of the experiment. The second to fifth columns show the RMSE of the audio samples for a locked ΣΔ ADC using the locking approaches discussed in Section 3.8, namely SFLL, 1.5xSFLL and DisORC & TRLL. The fourth and fifth column show the RMSE for DisORC & TRLL when an incorrect key of Hamming distance 12 and 63 from the correct key is applied. RMSE is computed

AUDIO SAMPLE	SFLL	1.5XSFLL	DISORC & TRLL	
			HD=12	HD=63
English voice recording	0.000	0.209	0.341	0.379
Bob Marley - No Woman No Cry	0.004	0.286	0.184	0.354
Benny Goodman - Bugle Call Rag	0.002	0.241	0.156	0.333
Kenny Ball - I Wanna Be Like You	0.008	0.245	0.167	0.335
John Coltrane - Nature Boy	0.005	0.252	0.161	0.349
Beethoven - Symphony No. 9	0.041	0.231	0.177	0.298

Table 3.5: Impact of locking with MixLock on audio quality measured in RMSE between the nominal waveform and the respective waveform processed by converters locked with different techniques.

according to Eq. (3.1) by using every available sample point, i.e., the metric is calculated over the complete duration of the respective audio file.

The interested reader can also download and listen to the output audio samples via this link: https://nuage.lip6.fr/s/QNHoQWmcCbRyR24. The downloadable archive includes the output audio samples for an unlocked design, where the valid key is applied, as well as for locked designs using SFLL, 1.5xSFLL and for DisORC & TRLL as listed in Table 3.5.

DisORC & TRLL lead to a corruption of the audio output in a way that is comparable to white noise. The impact of corruption becomes greater, the higher the provided key's Hamming distance from the correct key is, or, to put it in simpler words, the more the provided key is incorrect. The sound is comparable to that of a miss-tuned car radio for low Hamming distances and for higher Hamming distances that of the same radio for when the car goes through a tunnel. Quite similarly to the latter technique, 1.5xSFLL corrupts the audio quality dramatically, audible in very frequent glitches. In fact, the recording gets buried under the noise level and is hardly recognizable. SFLL on the other hand results in a number of glitches for the music recordings that can be heard as single, loud "cracks". However, for the home-made voice recording, no glitches are noticed.

These results can be explained as follows:

- *SFLL* by default corrupts the output of the targeted digital circuit for some and not all input patterns. In our case, we have $n = k = 128$ and $h = 15$, thus the number of protected patterns is $\binom{k}{h} \cdot 2^{n-k} \approx 1.32 \cdot 10^{19}$, which is a very small subset of all possible $2^{128} \approx 3.4 \cdot 10^{38}$ input combinations. An analog input, i.e., an audio signal in our case, gets translated into a sequence of patterns at the input

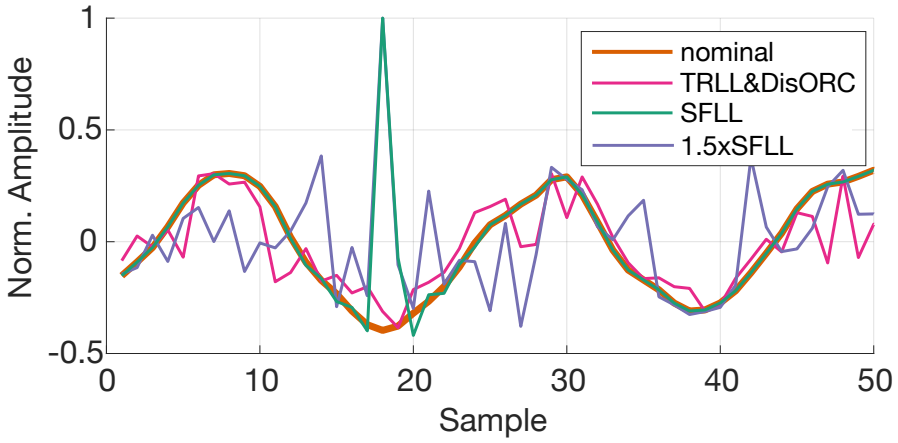


Figure 3.17: Differences in the impact on the time domain signal of SFLL, 1.5xSFLL and DisORC & TRLL with HD = 12.

of the protected digital block within the decimation filter. Since music recordings have higher signal activity compared to voice recordings, it turns out that they get translated to a larger number of distinct patterns at the input of the protected digital block. Thus, the probability of hitting protected input patterns is higher, resulting in a higher probability of audio quality corruption. Note that short duration samples were recorded for practical purposes and that for longer duration samples SFLL is expected to also result in glitches when voice is processed.

- *1.5xSFLL* is not sensitive to the the nature of the input. Most of the errors it induces are generated by the second SFLL mechanism that is set up in a way that it maximizes error-rate.

- *DisORC & TRLL* is also not sensitive to the recording's input type. Given that the locations where TRLL modifies the comb filter's gate-level netlist are chosen randomly, the errors that are induced due to an incorrect key are not restricted to the MSB or MSB-1 such as was the case for both SFLL variants. Errors in fact appear anywhere in the comb filter circuit, thereby inciting faults of reduced scale, albeit in very elevated numbers. While both SFLL and 1.5xSFLL are independent of the user provided key used to unlock a locked circuit, for DisORC & TRLL we observe a dependence of the RMSE metric on the key. The closer a key is to being correct, the fewer errors are induced. E.g., for HD = 12 where 116 out of 128 key-bits are set correctly, the RMSE and the audible noise power is lower than for a random key where HD = 63 and only 65 out of 128 key-bits are set correctly.

Fig. 3.17 shows a short excerpt of the sampled output of the ADC for Beethoven's Symphony No. 9 processed by the nominal converter

and converters locked with SFLL, 1.5xSFLL and DisORC & TRLL. While the SFLL waveform closely follows the nominal waveform, at sample 18 a single glitch occurs. Both, 1.5xSFLL and DisORC & TRLL behave similarly in that they result in the corresponding waveforms being in error at nearly every sample.

At this point it is important to recall the purpose of hardware locking and hardware obfuscation in general. The aim is not necessarily to encrypt the data that is processed by the hardware, i.e., corrupt audio quality to bare random noise. The aim is to render the hardware low-quality and unusable unless the valid secret key is known; i.e., glitches occurring at regular and frequent intervals are sufficient.

3.9 CONCLUSION

Hardware security vulnerabilities have been addressed through various methods in the digital domain while similar solutions are largely missing in the analog domain. We proposed *MixLock* which protects mixed-signal ICs via locking their digital part. We developed security metrics to connect IC locking notion to intentional disruption of mixed-signal performance. We adapt and use SFLL as well as DisORC & TRLL as part of *MixLock* to enable effective trade-offs between analog and digital security, delivering a holistic protection on a given mixed-signal IC. We illustrate the application of *MixLock* on a ΣΔ ADC in simulation. Furthermore we conducted a hardware experiment using a silicon ΣΔ modulator and a locked, FPGA-implemented decimation filter to show *MixLock*'s capabilities. We show that *MixLock* thwarts all known attacks in the digital domain while delivering perfect analog security levels. This is achieved without degrading the mixed-signal performance and at very reasonable area and power overheads. Finally we demonstrated the effect of locking a mixed-signal circuit that is part of the signal processing chain in an audio application leveraging the *MixLock* locking technique. The effect of locking is measured via the RMSE, but it can be also clearly heard in audio samples that are provided. We demonstrate that locking results in disturbing glitches and noise, rendering the device low-quality and unusable unless the valid secret key that unlocks the design is known. To the best of our knowledge, this the first demonstrator showing the effect of locking on a circuit in a way that can be perceived by humans via a sense.

4

ANALOG SIZING CAMOUFLAGING

In this chapter we treat the problem of analog IC obfuscation towards analog IP protection against reverse engineering, thereby providing protection against subsequent piracy threats [89]. Obfuscation is achieved by camouflaging the effective geometry of analog layout components via the use of fake contacts, which originally were proposed for gate camouflaging in digital ICs. We present a library of obfuscated layout components, we give recommendations for effective camouflaging, we discuss foreseen attacks and the achieved resiliency, and we propose security metrics for assessing the hardness of reverse engineering. The proposed methodology is demonstrated on an operational amplifier and an RF ΣΔ ADC.

4.1 INTRODUCTION TO CAMOUFLAGING

A well-known physical obfuscation mechanism is based on fake contacts[1] between metal layers and polysilicon, diffusion or metal layers [90]. True contacts span the entire dielectric to connect the two layers, whereas fake contacts have a thin gap creating an open-circuit. Fake contacts are 100% CMOS compatible requiring no foundry process changes [90]. An attacker cannot differentiate between true and fake contacts as they appear identical under a microscope and by slicing the die it will be unlikely to pass through the thin gap. Besides, fake contacts are distributed at different heights and this would require slicing the die in several pieces which is infeasible. Another approach is to make true contacts with magnesium (Mg), which displays very good electrical conductivity, and fake contacts with magnesium oxide (MgO), which is a perfect insulator [91]. When delayering a protected IC the Mg contacts oxidize within minutes to MgO, thereby destroying the information where real and where fake contacts are placed in the layout. A remedy for the attacker could be to delayer in an oxygen-free environment, an approach that would make costs and efforts soar prohibitively. Generally, fake contacts can be leveraged to inconspicuously blend extra circuitry into the IC which, however, is inactive and completely irrelevant for the functionality of the IC. In [92], fake contacts are used to design a camouflaged cell that can perform either as an XOR, NAND, or NOR gate according to which contacts are

1 For the sake of simplicity but without loss of generality we will refer to all types of interconnects, including vias, as contacts.

true and fake. The designer can replace some standard gate cells with this camouflaged cell to obfuscate the functionality.

In this Chapter, we propose an analog IC camouflaging technique based on the use of fake contacts. Compared to gate camouflaging, the proposed analog IC camouflaging works differently. Gate camouflaging hides the gate functionality, whereas analog IC camouflaging hides the correct sizing of the components, such that the extracted netlist from the reverse-engineered circuit has deceiving sizing and, thereby, unacceptable performance trade-off. Gate camouflaging requires camouflaging a large percentage of gates so as to increase the reverse engineering hardness, which inevitably results in large area, delay, and power overheads [93]. In contrast, in analog IC camouflaging it suffices to obfuscate a small number of components, thus the overheads can be well-controlled and can be practically negligible. In gate camouflaging the attacker can recognize the camouflaged gates, which can be informative for launching attacks, whereas in analog IC camouflaging the attacker will have to consider every component as potentially obfuscated, which increases dramatically the hardness of reverse engineering. We present a library of obfuscated analog layout components that is sufficient for camouflaging virtually any analog IC and we provide recommendations to designers for best camouflaging practices. We also discuss foreseen attacks and the resiliency offered by the proposed technique. Finally, we propose security metrics specific to analog ICs to quantify the hardness of reverse engineering. The technique is demonstrated on two case studies, namely a Miller operational amplifier (op-amp) and an RF ΣΔ ADC.

The rest of the Chapter is structured as follows. In Section 4.2, we provide an overview of the analog IC camouflaging technique. In Section 4.3, we present the library of obfuscated analog layout components. In Section 4.4, we provide recommendations for best camouflaging practices. In Section 4.5, we discuss foreseen attacks and the achieved resiliency. In Section 4.6, we develop security metrics for quantifying the hardness of reverse engineering. In Section 4.7, we present our experimental results on the chosen two case studies. Section 4.8 concludes the Chapter.

4.2 ANALOG IC CAMOUFLAGING

4.2.1 *Threat Model*

The proposed analog IC camouflaging is a defense against reverse-engineering attempted by a malicious end-user. In our threat model, the attacker legally purchases a functional chip from the market. We assume that the attacker has access to the technology PDK and has full capabilities to reverse-engineer the chip and resolve geometries down to sub-gate-level sizes, thus recovering an exact schematic and layout. The

attacker can also purchase a second chip that can be used as an oracle, i.e., for applying inputs and observing the outputs.

The proposed analog IC camouflaging does not protect an IP block from a malicious SoC integrator or a malicious foundry that fabricates the IC, since the IP/IC owner inevitably shares with these potentially untrusted parties the blueprint of the IP/IC, e.g., GDS-II file, whereby fake contacts are directly revealed.

4.2.2 *Sizing Camouflaging*

The proposed analog IC camouflaging consists in inconspicuously hiding by means of fake contacts the active geometry of layout components and, thereby, the actual sizing of schematic components extracted from reverse-engineering. The methodology takes advantage of the special handcrafted layout techniques used in analog designs for improving component matching, tolerating process variations, and achieving compact layouts [94].

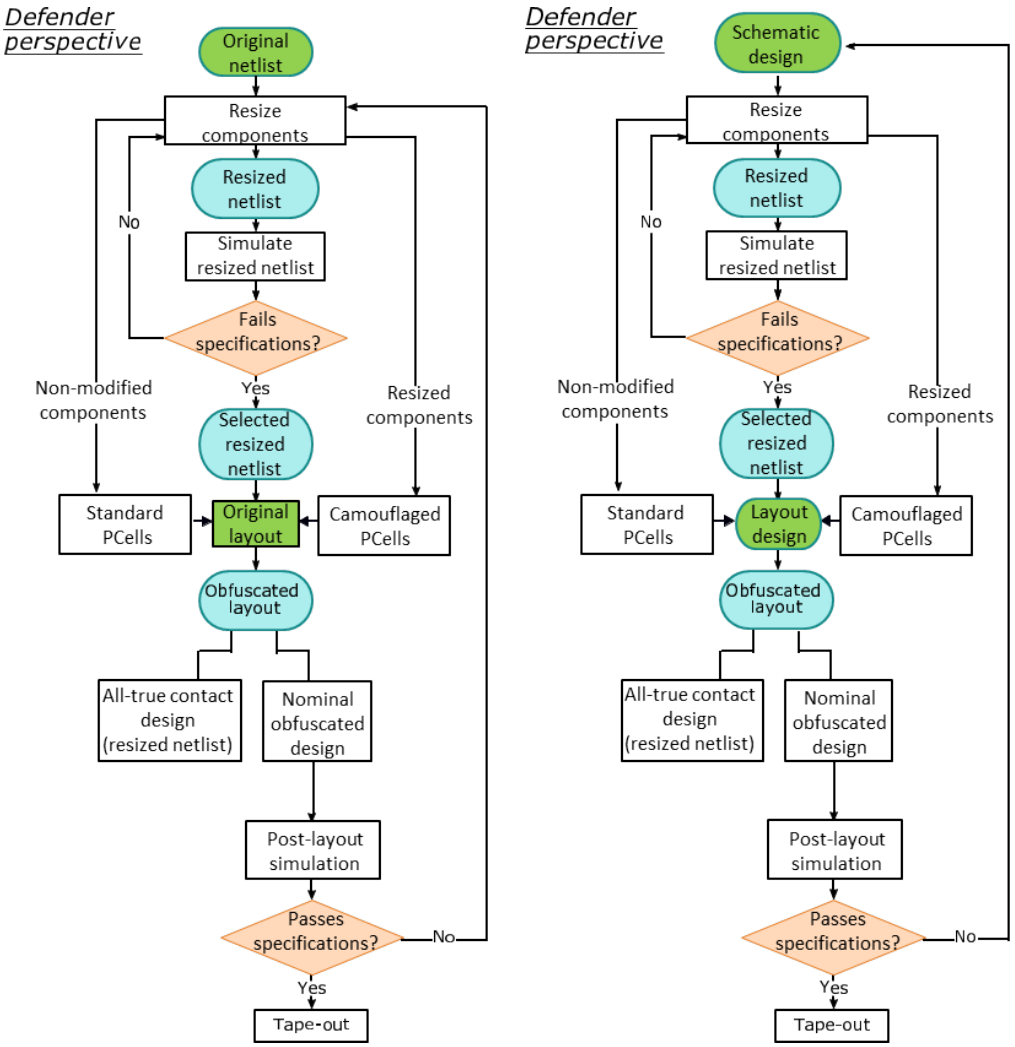
In particular, non-minimum size transistors are most often laid out as several sub-transistors connected in parallel and sharing diffusion strips, known as gate fingers. Common-centroid layouts are also preferred for transistor pairs that are required to be well-matched. Similarly, resistors are laid out in a serpentine serial connection of unit resistors and capacitors are laid out as capacitor banks consisting of several unit capacitors.

The underlying idea is to use fake contacts so as to add seemingly connected yet in reality inactive and electrically disabled gate fingers, unit resistors, and unit capacitors. In this way, the nominal sizing of components, i.e., the effective width of transistors and the values of resistors and capacitors, is camouflaged.

In Section 4.3, we will present in detail camouflaged layout versions of analog components using fake contacts. These camouflaged layout versions can be parametrized into PCells to compose a library of camouflaged PCells that is combined with the library of standard PCells and is seamlessly integrated into the design flow. A camouflaged PCell combines the functionality of the standard PCell while also adding extra electrically disabled instances. Building the library of camouflaged PCell is a one-time effort for each technology node and thereafter can be reused for readily obfuscating any design. Moreover, the same design principle can be reused for every technology node. For a target component to be resized, the designer will simply have to replace the standard PCell with the camouflaged PCell and set the parameters of the camouflaged PCell. This set of parameters includes the active sizing, as well as the number of extra inactive instances and their locations, i.e., the arrangement of active and inactive instances.

4.2.3 *The Defender Perspective: Design Flows with Camouflaging*

We can distinguish two design flows, namely camouflaging of an existing design, shown in Fig. 4.1a, and involving camouflaging already from the design phase, shown in Fig. 4.1b.



(a) Defender perspective when camouflaging an existing design.

(b) Defender perspective when involving camouflaging in the design phase.

Figure 4.1: Overview of analog IC camouflaging.

1) *Camouflaging an existing design:* The defender has the *original* design, including the original netlist and layout, which we refer to as the *nominal non-obfuscated* design. Beginning with the original netlist, the defender will perform re-design iterations, shown with the inner loop in Fig. 4.1a, where in each step a set of components is resized and the resized netlist is simulated to obtain the performances. This inner loop stops when a suitable resized netlist is found that has one or more performances failing their specifications.

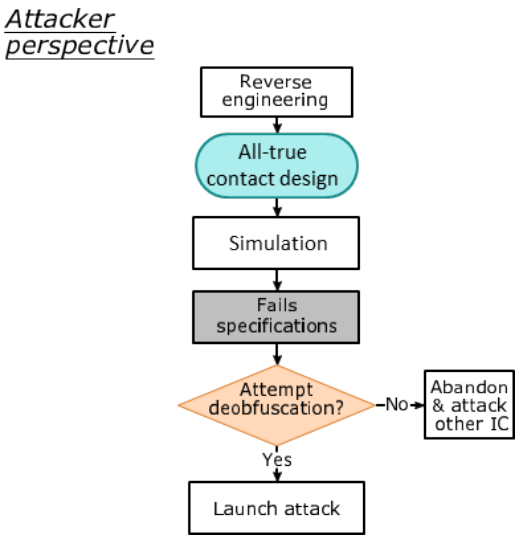


Figure 4.2: Attacker perspective.

With this selected resized netlist, the defender will next obfuscate the original layout. The layout of non-modified components remains unchanged, while the layout of resized components is replaced with an obfuscated layout version using the library of camouflaged PCells, as explained in Section 4.2.2. This replacement possibly will require changes in the floor-planning and routing, in order to fit into the original layout the camouflaged layout versions of the resized components. The resulting layout is an obfuscated layout that is electrically equivalent to the original layout since the resizing is cancelled out by the use of fake contacts. Therefore, the obfuscated layout embeds the nominal design which we refer to as the *nominal obfuscated* design. However, if fake contacts cannot be distinguished from true contacts and are all reckoned as true, then the obfuscated layout can be deceivingly thought to embed the resized netlist which we refer to as the *all-true contact* design.

Compared to an original component layout, a camouflaged component layout will add extra parasitics which, albeit small, may perturb the intent performance trade-off of the original design. Perturbation may result also from changes in the floor-planning and routing. To ensure that the nominal obfuscated design does not incur any performance penalty with respect to the nominal non-obfuscated design, as a final step, the defender will perform post-layout simulation to evaluate the performances of the nominal obfuscated design. If unacceptable performance degradation is noticed, then the defender will have to repeat the camouflaging procedure, as illustrated by the outer loop in Fig. 4.1a. The defender can identify the modified components that are the root-cause of this degradation and will target resizing another set of components

that results in no degradation. With this outer loop, obfuscation via sizing camouflaging can be viewed as an additional step in the design flow that can be performed on top of the original design.

2) *Involving camouflaging in the design phase:* The designer knows in advance before actually starting the design that the design should be protected against reverse-engineering. In this scenario, camouflaging is fully integrated into the design flow. The designer will proceed as normal and will first design the circuit at schematic-level with no camouflaging in mind. Once the intent design specifications are met at schematic-level and before moving to layout design, the designer will perform the resizing operation for camouflaging, shown with the inner loop in Fig. 4.1b, similarly to the design flow in Fig. 4.1a. Thereafter, the layout will be designed as normal using camouflaged layout versions for the resized components. Thus, in this case, floor-planning and routing naturally takes into consideration the camouflaged layout versions of components. Once the layout is completed, post-layout simulations will be performed as normal. Typically, several design iterations take place until post-layout performances are satisfactory, as shown with the outer loop in Fig. 4.1b. During this design optimization, the designer will change the nominal component values, i.e., transistor dimensions, etc., will perform changes in the layout, floor-planning, and routing, and may also perform topology modifications. This outer loop is not related to the obfuscation objective. However, for every iteration of the outer loop, we may have to repeat the inner loop which is related to the obfuscation objective.

4.2.4 *The Defender Perspective: Objectives*

The defender has the following main two objectives:

1. For the design flow in Fig. 4.1a, maximize the performance penalty of the all-true contact design with respect to the nominal non-obfuscated design. For the design flow in Fig. 4.1b, maximize the performance penalty of the all-true contact design with respect to the specified performance trade-off.

2. For the design flow in Fig. 4.1a, minimize any performance penalty of the nominal obfuscated design with respect to the nominal non-obfuscated design. For the design flow in 4.1b, the nominal obfuscated design should meet the specified performance trade-off.

We can define two additional objectives:

3. Minimize the obfuscation area overhead.

4. Minimize the obfuscation design effort towards satisfying faster the above objectives 1 and 2.

For the design flow in Fig. 4.1a, minimizing the obfuscation design effort implies: (a) reducing the number of iterations of the inner loop and (b) reducing the number of iterations of the outer loop which, in turn, will reduce the number of the repetitions of the inner loop. As mentioned in Section 4.2.3, the outer loop aims at correcting any performance penalty of the nominal obfuscated design with respect to the nominal non-obfuscated design. This performance penalty is due to camouflaged layout-induced parasitics and changes in the floor-planning and routing. Reducing this performance penalty will reduce the number of iterations of the outer loop and possibly may eliminate completely the need to enter into this loop, thus iterating over the inner loop only once.

For the design flow in Fig. 4.1b, the outer loop aims at design optimization such that post-layout performances meet the intent specifications. As mentioned in Section 4.2.3, this outer loop is not related to the obfuscation objective, yet the inner loop which is related to this objective is revisited at every iteration of the outer loop. Therefore, for the design flow in Fig. 4.1b, minimizing the obfuscation design effort implies: (a) reducing the number of iterations of the inner loop and (b) avoiding repeating the inner loop during outer loop iterations. The latter can be achieved by aiming at minimizing the effect of camouflaged layout-induced parasitics on post-layout performances. In this way, camouflaged layout-induced parasitics will not be among the root-causes of unsatisfactory post-layout performances which is what enables the outer loop. The set of resized components for obfuscation as well as their resizing values can be kept fixed during outer loop iterations. As long as the resized netlist, i.e., the all-true contact design, fails the specifications, it will not be necessary to repeat the inner loop and find another set of components to resize.

Therefore, minimizing the obfuscation design effort boils down to the following objectives:

4 a) For both design flows, reduce the number of iterations of the inner loops in Figs. 4.1a and 4.1b towards satisfying faster objective 1.

4 b) For both design flows, minimize camouflaged layout-induced parasitics towards satisfying faster objective 2.

4 c) For the design flow in Fig. 4.1a, additionally minimize changes in the floor-planning and routing towards satisfying faster objective 2.

Recommendations for best camouflaging practices will be given in Section 4.4.

4.2.5 *The Attacker Perspective*

Fig. 4.2 illustrates the attacker perspective. The attacker will initially perceive all contacts as true and only after running simulations will realize that the performances of the all-true contact design are not in agreement with those promised in the datasheet having a degraded performance trade-off with one or more specifications lying outside their specification range. At that point the attacker will understand that the design is obfuscated, but cannot tell which are the fake contacts and for that reason cannot tell which are the obfuscated components either. Every component is potentially an obfuscated one. As a result, the attacker will have extracted the architecture and netlist, but will not recover the sized netlist nor a correct layout and is hindered from replicating the functionality and performances promised in the datasheet. The attacker may choose to attack another unprotected IC promising similar functionality, or may decide to attempt an attack to de-obfuscate. Foreseen attacks will be detailed in Section 4.5 and security metrics to assess the hardness of de-obfuscation will be given in Section 4.6.

4.3 LIBRARY OF CAMOUFLAGED LAYOUT COMPONENTS

Herein, we provide a library of obfuscated layout versions of components that are most commonly met in analog layouts, including multiple gate-finger transistors, common-centroid layout of transistors, interdigitized transistors, serpentine resistors, and capacitor banks. Of course, this is a non-exhaustive list of possible obfuscated layout versions of such components, and a non-exhaustive list of components that can be obfuscated, i.e., it excludes inductors and diodes, but it largely suffices to camouflage the sizing of virtually any analog IC.

4.3.1 *Transistors*

Multiple gate-finger transistors are parallel transistors of equal gate width where each transistor shares its inner diffusion regions for drain or source with its two neighbouring transistors. Fig. 4.3 shows an example of a compact transistor layout with 3 gate fingers. The inner diffusion regions control the state of 2 gate fingers at once, while the outer regions control a single finger. A transistor can be obfuscated by connecting extra gate fingers and deactivating them by using fake contacts in the drain or source terminals such that these nodes are floating. In Fig. 4.3, the fake contacts are shown with orange color, whereas true contacts are shown with black color. Two fake contacts are used to deactivate two gate fingers. Two fake contacts are also used to disconnect completely the two adjacent gates. This is preferred so as to reduce the parasitic load, but is only

possible if no shared poly-silicon gate is drawn. The equivalent schematic with the open-circuits resulting from fake contacts is also shown on top of Fig. 4.3. In this example, the transistor has 1 active gate finger, but the attacker observes a transistor with a gate width 3 times larger.

Certain transistor arrangements, i.e., differential transistor pairs or current mirrors, require special layout techniques to ensure matching. Common-centroid layouts are typically used for differential transistor pairs ensuring that gradients across the die will impact both transistors equally. Fig. 4.4 shows a layout of a common-centroid differential transistor pair A and B showing an AXXBBXXA pattern, with X representing deactivated transistors due to the inserted fake contacts. The equivalent schematic is shown on top of Fig. 4.4. With the inserted fake contacts the attacker observes that A and B consist of 4 active transistors each while in reality they consist of 2. By changing the gate connections in Fig. 4.4 we can turn the circuit into an interdigitized current mirror with obfuscated current ratio between A and B according to where the fake contacts are placed. The actual current ratio will be invisible to the attacker.

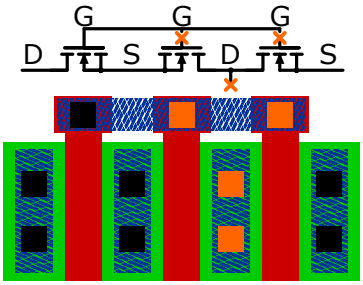


Figure 4.3: Obfuscated multiple gate-finger transistor layout with its schematic. Diffusion, poly-silicon, and metal are drawn respectively in green, red, and blue. True contacts are drawn in black and fake contacts in orange.

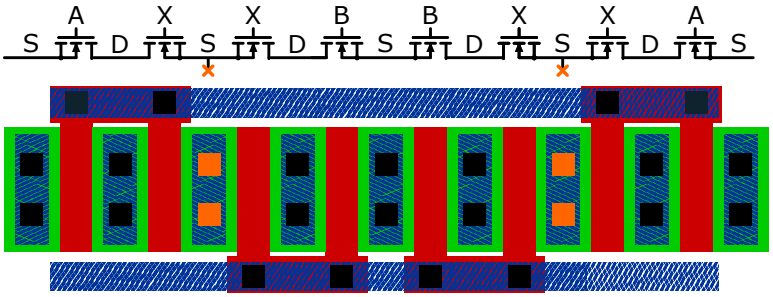


Figure 4.4: Obfuscated common-centroid layout and schematic with AXXBBXXA pattern, where the letters A,B and X over the gates mark to which transistor structure the transistor layout below belongs to. X marks deactivated instances due to fake contacts. To not impair visibility the connections between respective sources and drains of A and B are not drawn.

4.3.2 *Capacitors*

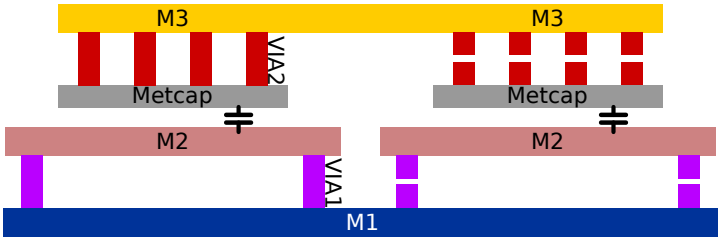


Figure 4.5: Side-view of obfuscated capacitor bank layout. The obfuscated capacitor on the right has fake contacts seemingly connecting both its capacitor plates.

The capacitor value of a capacitor bank can be obfuscated by adding extra capacitor units and disconnecting them through the use of fake contacts. Fig. 4.5 shows the side-view of an exemplary layout of a metal-insulator-metal (MIM) capacitor bank consisting of 2 parallel-connected unit capacitors. Metcap[2] and metal 2 (M2) are the respective plates of a capacitor. Through the use of fake contacts, shown with a thin gap, the right-hand capacitor is disconnected from the capacitor bank. Both plates are disconnected so as to reduce parasitic capacitance to a minimum. In this example, the attacker observes an incorrect, two times bigger capacitor value.

4.3.3 *Resistors*

The value of a serpentine resistor can be obfuscated by adding extra unit resistors. As an example, Fig. 4.6 shows a serpentine resistor composed of 5 unit resistors. The idea is to use wiring across each unit resistor to create short-circuits and place fake contacts to cut the short-circuits for those unit resistors that will be active. Interestingly, in contrast to transistors and capacitors, fake contacts here are used to activate instances. In this example, the nominal resistance is 3R, whereas the attacker observes a resistor value $k \cdot R$, but does not know k which could take any value in $\{0, \cdots, 5\}$.

A camouflaged PCell is readily built from the standard PCell and can be instantiated to implement any degree of resizing and any arrangement of active and inactive instances. It can be viewed as a standard PCell with a subset of contacts replaced with fake contacts, in order to deactivate the corresponding instances. The camouflaged PCell takes as parameters the standard PCell parameters, as well as the number and location of inactive instances. For example, for a camouflaged PCell of a multiple

2 Metcap is an additional layer used to realize MIM capacitors. Across different technologies this layer may be called differently.

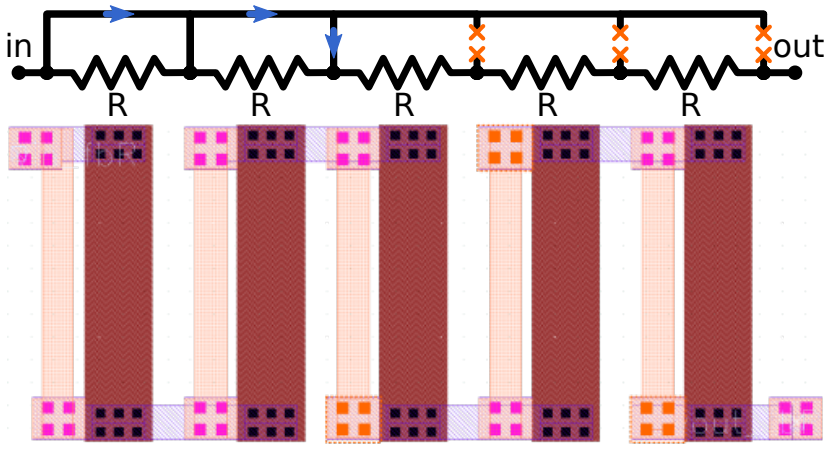


Figure 4.6: Obfuscated serpentine resistor layout. Resistive poly, metal 1, metal 2, true contacts, fake contacts, and metal1-metal2 vias are shown respectively in dark red, blue, light orange, black squares, orange squares, and pink squares.

gate-finger transistor, the designer will have to set the nominal transistor dimensions, i.e., length, width, and number of gate fingers, the number of inactive extra gate fingers, as well as their arrangement with respect to the active gate fingers.

4.4 RECOMMENDATIONS FOR ANALOG IC CAMOUFLAGING

The number of components to resize, the degree of resizing per obfuscated component, and the selection of components to resize are driven by the objectives defined in Section 4.2.4.

4.4.1 *Number of Components to Resize and Degree of Resizing*

With the proposed camouflaging approach, in the reverse-engineered netlist all components are potentially obfuscated in the eye of the attacker. Therefore, the hardness of reverse-engineering does not depend on the number of resized components. We can turn this fact to our advantage and target resizing only a small number of components that is sufficient for achieving an all-true contact design that has a degraded performance trade-off (objective 1).

Achieving objective 1 is an easy task since analog ICs are very sensitive to component sizing. Although analog IC design optimization and centering can be a very time-consuming and tedious task requiring high expertise, here the defender aims at the “inverse" task, i.e., untuning the circuit and destroying the performance trade-off, which arguably can be achieved in an effortless way. It is not surprising if objective 1 is achieved by resizing a single component. In general, the first inner loops in the

design flows in Figs. 4.1a and 4.1b should take only a few iterations to achieve objective 1 (objective 4a).

By only resizing a small number of components, we can meet additional objectives defined in Section 4.2.4. Specifically:

(a) obfuscation area overhead is kept at a minimum (objective 3);

(b) total camouflaged layout-induced parasitics will be effectively minimized (objective 4b);

(c) for the design flow in Fig. 4.1a, minor changes in the floor-planing and routing will be required (objective 4c).

Note that objective 1 can also be met by distributing the resizing across many components and applying a smaller degree of resizing for each component. However, this strategy intuitively will be more time-consuming for meeting objective 1, requiring more iterations of the inner loops in Fig. 4.1a and 4.1b. Besides, in this way, the camouflaged layout-induced parasitics get distributed too and it will be more difficult controlling them. Moreover, it is not guaranteed that this strategy will overall reduce the obfuscation area overhead, and for the design flow in Fig. 4.1a it is likely that changes in the floor-planing and routing would be more significant. For these reasons, we recommend obfuscating a small number of components with the resizing required to satisfy objectives 1 and 2, and only when the resizing turns out to be very large try to distribute the resizing across more components. This last recommendation aims at avoiding having unnaturally large layout components that from the attacker perspective will look suspicious and likely obfuscated.

4.4.2 *Degree of Performance Degradation*

One question that arises is to what degree to degrade the performance trade-off of the all-true contact design. If the all-true contact design is functional, showing small performance deviation outside the allowable specification range, then the cloned design can still be used in applications where the performance requirements are less stringent. Therefore, the defender goal should be to introduce a performance penalty in the all-true contact design at least to a point where it becomes of low-quality and unusable and, thereby, not appealing any more for cloning.

4.4.3 *Selection of Components to Resize*

The following guidelines can be used:

1) Selecting to resize components that largely influence the performance trade-off will result in a smaller number of resized components. This helps meeting several objectives as explained in Section

4.4.1 (i.e., objectives 1, 3, and 4). However, we recommend that the selection process should not follow any formal or established methodology, such as a sensitivity analysis, which ranks the components according to their influence. The reason is that the attacker may think of employing the exact same methodology to trace back the resized components. We argue that the best approach towards increasing the reverse engineering hardness is to randomly select components to obfuscate based on intuition about their influence.

2) On top of resizing a small number of components, avoiding resizing components that are connected to sensitive or high-frequency nodes will further minimize the effect of camouflaged layout-induced parasitics on the performance trade-off (objective 4b).

3) In Section 4.3, we presented obfuscation layout versions of transistors, resistors, and capacitors. This library can be extended to include other components, i.e., inductors. Clearly, adding extra inactive fingers to transistors results in much lower area overhead compared to adding extra inactive unit capacitors, unit resistors, or extending the coil of a wire inductor. Thus, priority should be given to obfuscating transistors rather than passive components towards low obfuscation area overhead (objective 3). In addition, for the design flow in Fig. 4.1a, this will reduce the required changes in the floor-planing and routing (objective 4c).

4) Regarding the design flow in Fig. 4.1a, selecting to resize components that have enough empty space in their periphery on the layout, i.e., they are located in layout areas that are not compact, will reduce the obfuscation area overhead (objective 3) and will avoid introducing changes in the layout that may require reexamining the floor-planning and routing (objective 4c). If components can be resized without changing the placement of surrounding components in the layout, then obfuscation area overhead will be zero. In general, in analog layouts many areas are left unoccupied, in order to leave sufficient space between sensitive blocks with the goal to mitigate electromagnetic interference, crosstalk, thermal-related issues, etc. This gives us large flexibility for inserting the camouflaged layout versions in the existing floor-planning. Since the resized portion of the component is seemingly connected with fake contacts making it inactive and electrically disabled, it should not change the profile of the circuit. In any case, minimum distances between adjacent objects as defined in the PDK should be respected and electromagnetic compatibility compliance should not be compromised.

5) If the to-be-protected circuit is a complex system consisting of a number of sub-blocks, then the straightforward approach would be to obfuscate every sub-block, i.e., resize components in every sub-block. However, this is not strictly necessary as the aim of obfuscation is to act on the global system-level performances. In other words, for complex systems it suffices to resize a small number of components in a few sub-blocks to obtain an all-true contact design with degraded performance (objective 1), thus also minimizing the obfuscation design effort (objective 4). We will discuss this case also in relation to foreseen attacks in Section 4.5.

6) A common layout practice found in analog layouts is the placement of dummy components for better matching properties and compensation of process variations. Existing dummy components can be seemingly connected to their neighboring active components if they have the same geometry via the use of fake contacts, thus naturally extending the resizing. In this way, we can naturally degrade further the performance trade-off of the all-true contact design (objective 1), reduce the obfuscation area overhead (objective 3), and iterate less over the inner loops in Figs. 4.1a and 4.1b (objective 4a). For the design flow in Fig. 4.1a, this additionally helps minimizing changes in the floor-planning and routing (objective 4c). However, this strategy should be followed conservatively and cautiously so as to maintain low camouflaged layout-induced parasitics.

4.5 ATTACKS AGAINST ANALOG IC CAMOUFLAGING

1. *Attacks on gate camouflaging:* SAT-based attacks [95], [96] that have compromised the security of gate camouflaging techniques for digital ICs do not apply to analog ICs. The reason is that SAT solvers rely on Boolean algebra while analog circuits carry continuous-time signals.

2. *Brute-force attack:* The attacker will massively try different combinations of component sizing in the hope of eventually guessing a sizing that results in a satisfactory performance trade-off. Our defense is that the attacker is obliged to consider every component in the circuit as potentially obfuscated. The search space size is $\prod_{i=1}^{D} N_i$, where D is the number of components and N_i denotes the number of instances in the i-th component. This search space can be reduced if the attacker makes some informed assumptions, as it will be explained in more detail in Section 4.6. A second defense is the fact that analog simulation can be very time-consuming. Thus, in practice a very small fraction of the search space can be explored.

3. *SMT-based attack:* The SMT-based attack proposed in [97] can be used to speed up de-obfuscation as long as circuit equations can be written. In particular, for component i we can write an equation $y_i = \phi(\mathbf{q}^i)$, where $\mathbf{q}^i = [q_1^i, \cdots, q_{N_i}^i]$ is a string of key-bits of size N_i, N_i is the number of instances, and $q_j^i = 1$ if the j-th instance is active and 0 if it is inactive. For example, for transistors $y_i = \sum_j q_j^i * W/L$, where W is the gate finger width and L is the length. For D components we can write $\mathbf{y} = [y_1, \cdots, y_D]$ and combine keys in a single key $\mathbf{q} = [\mathbf{q}^1, \cdots, \mathbf{q}^D]$. Then, based on the m performances $\mathbf{p} = [p_1, \cdots, p_m]$ found in the datasheet, we can write m equations $p_j = \theta_j(\mathbf{y})$ linking each performance p_j to several y_i. An SMT-solver is used to find a key that satisfies all equations $p_j = \theta_j([\phi(\mathbf{q}^1), \cdots, \phi(\mathbf{q}^D)])$. The search space size is the same as in the brute-force attack, but with this approach we circumvent circuit simulations and we speed up the search. The difficulty with this approach is deriving the functions θ_j.

4. *Hierarchical decomposition attack:* For a complex system, to reduce the computational effort, the attacker may try to transform the extracted low-level netlist into a hierarchical, block-level representation and subsequently attack the circuit's sub-blocks individually. As mentioned in Section 4.4, the simple defense is to obfuscate every single sub-block, but this is not strictly necessary. The reason is that sub-blocks are connected in feedback loops and only the global specifications are given in the datasheet, whereas many of the specifications of the sub-blocks are not released as they are not relevant for the end-user. We can imagine the scenario where obfuscation results in a circuit that has part of its sub-blocks obfuscated to a small degree such that the global specifications fail. We will see this obfuscation approach in the RF ΣΔ ADC case study in Section 4.7. This scenario is confusing for the attacker as all sub-blocks are functioning correctly with an apparently decent performance trade-off, but the global performances are not met. Thus, the attacker cannot tell which sub-blocks have been obfuscated and all sub-blocks, even those that are left untouched by obfuscation, become candidates for de-obfuscation.

5. *Automatic analog circuit sizing attack:* We make the additional assumption that the attacker has access to a CAD tool for automatic analog circuit sizing. Such a tool starts with a given topology and aims at producing a sized topology that conforms to the performance objectives. An attacker may employ this tool to re-size the topology extracted from reverse engineering.

   There exist several commercial CAD tools for automatic analog circuit sizing, for example the Optimizer in Eldo tool by Mentor

Graphics, A Siemens Business, the WiCkeD tool by MunEDA, and the ID-Xplore by Intento Design. There are also several tools proposed in the literature (for example, see [98]–[106]).

To perform the sizing the attacker will need to define design variables and an objective function that measures the performance goal. To evaluate the objective function, the attacker will have to develop test benches for simulating the performances.

All these CAD tools require simulating the circuit at transistor-level a very large number of times. While this is possible for smaller circuit blocks, larger and complex circuits and systems, which are composed of several sub-blocks and have very long simulation times, cannot be handled as a single circuit. In this case, first a hierarchical decomposition of the circuit is needed. More specifically, the attacker will have to develop an abstract behavioral-level system model of the circuit that interconnects the sub-blocks and operates at data processing level, i.e., Simulink, VHDL-AMS, VerilogA, SystemC-AMS, etc. Having developed this system model, the attacker will need to guess sub-block performances to reach the global system-level performances since this information is lacking from the datasheet, as mentioned also in the hierarchical decomposition attack. With the guessed specifications, the attacker will launch the sizing tool to automatically size each sub-block at transistor level separately. Typically, the attacker will have to go through several iterations to meet the global performances using mixed-level simulations, where some sub-blocks are at transistor-level and some at behavioral-level.

Then, the next step is designing the layout. The attacker already has an extracted layout, but the automatically sized netlist will be different from the “deceivingly" sized reverse-engineered netlist. This is because many component sizing combinations achieve the same objective. Typically, the CAD tool will produce a Pareto front with several feasible solutions achieving different performance trade-offs. Therefore, the attacker will have to re-design large portions of the layout and change the floor-planing and routing. Typically, the attacker will have to do several design iterations going back and forth between schematic and layout, in order to meet post-layout performances.

In this regard, the attacker may rely on automated analog layout synthesis tools (for example, see [107]–[109]). However, these tools are not yet mature enough to produce first-time-right layout designs and require subsequent manual optimization to handle correctly symmetries, current flows, net parasitics, layout-dependent effects,

etc. This is an active research area and there are no commercialized tools yet.

In short, most of the effort spent by an analog designer is not bypassed with this attack, with the exception that sub-blocks at transistor-level can be automatically sized at every design iteration. This attack requires a very high analog design expertise that goes far beyond the assumptions typically made on the capabilities of the attacker. In particular, the attacker will need to: (a) have knowledge on the use of automatic analog circuit sizing; (b) specify optimization objectives; (c) develop test benches for simulating performances; (d) develop an hierarchical behavioral-level model which is a challenging task on its own; (e) assign sub-block performances from target system-level performances; (f) have knowledge on analog layout design; (g) perform several design iterations that are driven by tough design decisions.

6. *Physical attacks:* These include: (a) optical imaging, i.e., using Scanning Electron Microscopy (SEM); (b) heat maps; (c) Focused Ion Beam (FIB)-assisted probing; and (d) electromagnetic (EM) side-channel analysis. As pointed out in [110], optical imaging would require first to narrow the search to the target obfuscated area so as to be able to extract such fine detail. However, the attacker has no means to pinpoint the obfuscated areas since every component is potentially an obfuscated one. Heat maps would not work either as they lack the necessary resolution to resolve the sub-gate-level inactive instances of an obfuscated component. With FIB-assisted probing the attacker will sequentially get access to all individual components to measure them and extract their sizing since every component is potentially an obfuscated one. This will be a very tedious and costly approach for large circuits, requiring several chips since FIB is destructive to the chip. Regarding EM side-channel analysis, it is very unlikely to be able to resolve analog component sizings from the collected electromagnetic signals.

4.6 SECURITY METRICS

Let us assume that the circuit has D components and that the i-th component has N_i instances out of which N_i^{obf} are inactive resulting from obfuscation. The *search space* for an attacker is defined as the number of all possible variants of the circuit:

$$S = \prod_{i=1}^{D} N_i. \tag{4.1}$$

However, the search space is in fact smaller for the following reasons: (a) certain components should be clearly matched and identical, for example the input transistor pair of an op-amp; (b) certain basic building blocks in the design are clearly replicated, i.e., switches, buffers, etc.; (c) the sizing of certain components may not be critical for setting the desired performance trade-off, i.e., this may be the case for digital control sub-blocks. Given these considerations, let O denote the set of components that are potentially obfuscated and let the cardinality of O be $|O| = D' \leqslant D$. This reduces the initial search space to:

$$S' = \prod_{i \in O} N_i. \tag{4.2}$$

This reduced search space S' is a metric of the hardness of reverse engineering. The attacker will try to reduce further the effective search space by making informed assumptions. In particular, the attacker knows that most likely the majority of components have not been obfuscated since otherwise this would have increased the obfuscation area overhead. In general, increasing the number of obfuscated components would make it more difficult to meet the intent design specifications. Specifically for the design flow in Fig. 4.1a, this would additionally require significant changes in the floor-planning and routing and, thereby, it would have been difficult to maintain a low performance penalty of the nominal obfuscated design with respect to the nominal non-obfuscated design. For this reason, the attacker would rather search using instance numbers close to the maximum value N_i. Let us assume that the attacker will try out the β% higher instance numbers for each potentially obfuscated component. This reduces the effective search space to:

$$S'' = \prod_{i \in O} \left\lceil \frac{\beta}{100} N_i \right\rceil. \tag{4.3}$$

The attacker can perform a brute-force analysis in this reduced search space in the hope of eventually guessing the correct sizing of the circuit.

Let us now define the parameters:

$$\alpha_i = \frac{N_i^{obf}}{N_i}, \tag{4.4}$$

$$\alpha_{max} = \max_i \alpha_i. \tag{4.5}$$

For the i-th component, the true number of active instances is $N_i - N_i^{obf}$, whereas the attacker will try out numbers of instances from $N_i - \left\lceil \frac{\beta}{100} N_i \right\rceil$ to N_i. Therefore, the attacker will "hit" the nominal sizing of the

component during the search if $N_i - \left\lceil \frac{\beta}{100} N_i \right\rceil \leqslant N_i - N_i^{obf}$, which can be re-written as $\alpha_i \leqslant \left\lceil \frac{\beta}{100} \right\rceil$. Considering all components, the attacker will "hit" the nominal sizing of the circuit if $\alpha_{max} \leqslant \left\lceil \frac{\beta}{100} \right\rceil$. The parameter α_{max} is unknown to the attacker. The most favorable condition for the attacker is that he chooses exactly $\frac{\beta}{100} = \alpha_{max}$. Based on this most favorable condition, we define the following security metric λ_1 that *pessimistically for the defender* approximates the search space:

$$\lambda_1 = \log_2 \left(\prod_{i \in O} \lceil \alpha_{max} N_i \rceil \right). \tag{4.6}$$

The value of λ_1 is computed in bits to make it comparable to security levels from the digital domain.

We can define also a security metric λ_2 to express the total simulation time for an exhaustive search in the above reduced search space:

$$\lambda_2 = 2^{\lambda_1} \cdot T, \tag{4.7}$$

where T is the total simulation time for computing all performances using appropriate test benches.

We also acknowledge the possibility that circuit instances within the search space, other than the nominal circuit, may satisfy all specifications. For this reason, we define a security metric λ_3 to express their percentage:

$$\lambda_3 = 100 \cdot \frac{\sum_{j=1}^{n} I(j)}{n}, \tag{4.8}$$

where $n \leqslant 2^{\lambda_1}$ is the number of simulations that we afford to run and $I(j)$ is an indicator function with $I(j) = 1$ if the j-th circuit instance fails and $I(j) = 0$ otherwise.

Let now $\mathbf{p}_j = (p_{j1}, \cdots, p_{jk})$ denote the performance vector for the j-th circuit instance, where k is the number of performances, and let $\mathbf{s} = (s_1, \cdots, s_k)$ denote the specification vector. Other useful security metrics express in % the average deviation of failing circuits from specifications:

$$\lambda_4 = \frac{100}{n} \cdot \sum_{j=1}^{n} \|\mathbf{u} - \hat{\mathbf{p}}_j\|_2 \tag{4.9}$$

and the deviation of the "best" failing circuit that is closest to the specification boundary:

$$\lambda_5 = 100 \cdot \min_j \|\mathbf{u} - \hat{\mathbf{p}}_j\|_2, \tag{4.10}$$

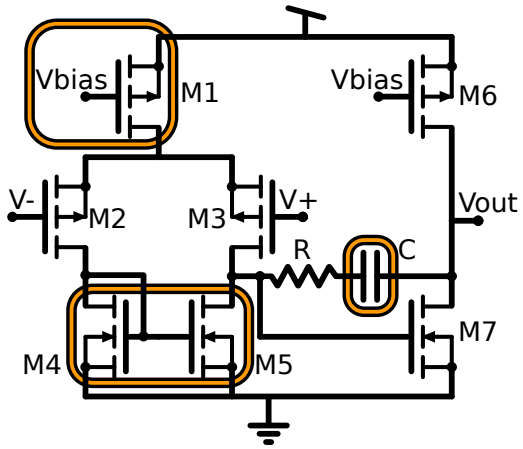


Figure 4.7: Schematic of Miller op-amp. The obfuscated components are highlighted.

where $\hat{\mathbf{p}}_j = (\hat{p}_{j1}, \cdots, \hat{p}_{jk})$, $\hat{p}_{ji} = \frac{p_{ji}}{s_i}$ if the j-th circuit fails the i-th specification and $\hat{p}_{ji} = 1$ if the j-th circuit passes the i-th specification, $\mathbf{u}$ is the $k \times 1$ vector with ones, and $\|\cdot\|_2$ is the L_2 norm. Note that $\|\mathbf{u} - \hat{\mathbf{p}}_j\|_2 = 0$ for passing circuits and $u_i - \hat{p}_{ji} = 0$ for passing performances.

4.7 CASE STUDIES

The proposed camouflaging methodology is demonstrated on two case studies, namely a Miller op-amp and an RF ΣΔ ADC. The Miller op-amp is a small basic building block and design guidelines can be found in textbooks. While not interesting for obfuscation as a stand-alone block, we provide this case study as a detailed and instructive example to illustrate also obfuscation metrics at block-level. The RF ΣΔ ADC is a large and complex circuit and demonstrates the true capabilities of the camouflaging methodology.

The simulation experiments were performed on an Intel(R) Xeon E5-2640 @ 2.5 GHz with 128 GB of RAM.

4.7.1 *Miller Operational Amplifier*

The Miller op-amp is designed in a 0.35 µm CMOS technology following the design flow in Fig. 4.1b. Fig. 4.7 shows the schematic and the first two columns of Table 4.1 show the main performances and the target specifications.

We randomly obfuscated components to the point where we largely satisfied objective 1 while meeting objective 2. As shown in Table 4.1, the performances of the nominal obfuscated design meet the target specifications, whereas the all-true contact design violates the Gain, Phase Margin (PM), power consumption (I_{dc}), and Total Harmonic Distortion

PERFORMANCE	SPECS	NOMINAL OBFUSCATED	ALL-TRUE CONTACT
Gain	⩾67 dB	70.1 dB	51.8 dB
GBW	⩾60 MHz	60.6 MHz	64.3 MHz
PM	⩾70°	71.2°	72.3°
THD	⩽0.1 %	0.04 %	5.1 %
I_{dc}	⩽400 µA	391 µA	416 µA

Table 4.1: Design specifications and performance of nominal obfuscated and all-true contact designs.

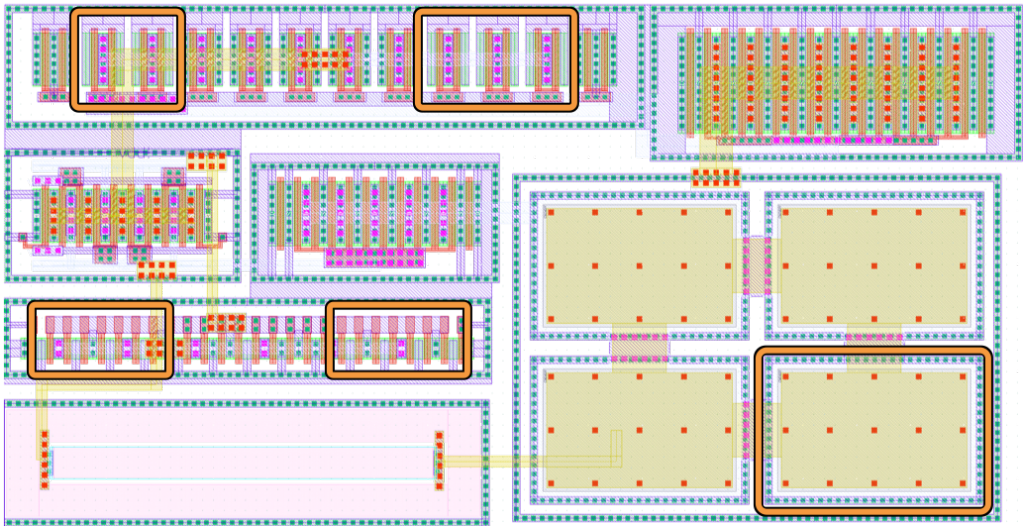


Figure 4.8: Obfuscated layout of Miller op-amp highlighting the inactive instances that have been added.

(THD) specifications. In total, we iterated three times over the inner loop of Fig. 4.1b, and we did not have to repeat the inner loop during outer loop iterations for design optimization.

The obfuscated components include the biasing transistor M1, the current mirror transistors M4 and M5, and the feedback capacitor C, and are highlighted in the schematic in Fig. 4.7. M1 is laid out as a multi gate-finger transistor with 20 gate fingers out of which 10 are inactive. M4 and M5 are laid out in an interdigitized pattern and each has 12 gate fingers out of which 8 are inactive. Capacitor C is laid out as a capacitor bank with 4 unit capacitors out of which 1 is inactive. The obfuscated layout with this camouflaged sizing is illustrated in Fig. 4.8 highlighting the added inactive instances. The resultant obfuscation area overhead is 15%.

	M1	M2	M3	M4	M5	M6	M7	R	C
i	1	2	3	4	5	6	7	8	9
N_i	20	4	4	12	12	10	16	1	4
N_i^{obf}	10	0	0	8	8	0	0	0	1
α_i	$\frac{1}{2}$	0	0	$\frac{8}{12}$	$\frac{8}{12}$	0	0	0	$\frac{1}{4}$

Table 4.2: Obfuscation of components in the Miller op-amp.

METRIC	EVALUATION
S'	$2^{19.2}$
λ_1	16.4 bits
λ_2	120 h
λ_3	100 %
λ_4	12 600 %
λ_5	2 %
Nominal obfuscated performance penalty	no, see Table 4.1
Obfuscation area overhead	15 %
All-true contact performance penalty	significant, see Table 4.1

Table 4.3: Obfuscation metrics for the Miller op-amp.

Table 4.2 shows for each of the $D = 9$ components the total number of instances N_i, the number of obfuscated instances N_i^{obf}, and the parameter α_i. Out of these components, transistors M2 and M3 in the input differential pair are matched and transistors M4 and M5 in the current mirror are matched, thus $O = \{M1, M2 \text{ or } M3, M4 \text{ or } M5, M6, M7, R, C\}$ and $D' = 7$. The search space is computed from Eq. (4.2) to be $S' = 614400 \approx 2^{19.2}$. α_{max} is given by the current mirror transistors M4 and M5 and is computed to be $\alpha_{max} = 8/12 \approx 0.67$. Using these values Eq. (4.6) gives $\lambda_1 = 16.4$ bits. The simulation time to compute all performances is $T = 5$ seconds, thus $\lambda_2 = 120$ hours. Finally, we simulated a set of $n = 1000$ random variants of the circuit. None of them passed all the specifications, thus $\lambda_3 = 100\%$. The other two metrics evaluate to $\lambda_4 = 12600\%$ and $\lambda_5 = 2\%$. λ_4 turns out to be very high as for many circuit variants the THD is over 20% while it has an upper specification of 0.1%.

Table 4.3 summarizes the obfuscation metrics. In conclusion, involving camouflaging during the design flow did not increase design iterations and the nominal obfuscated design met the target specifications. Camouflaging with 15% area overhead resulted in a relatively high search space of 16.4 bits for such a small-size circuit, yet the brute force attack on an ideally reduced search space can be successfully completed in less than 120 hours.

4.7.2 *RF ΣΔ ADC*

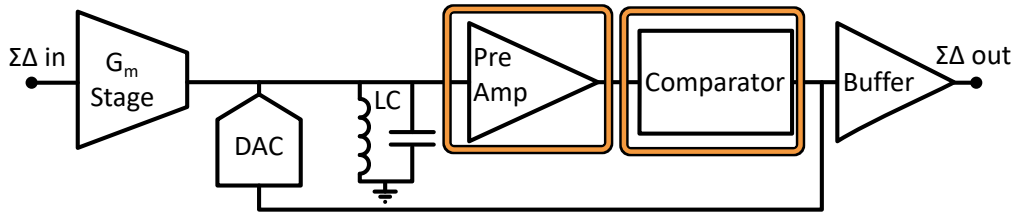


Figure 4.9: Block-level diagram of the RF ΣΔ ADC. Obfuscated sub-blocks are highlighted.

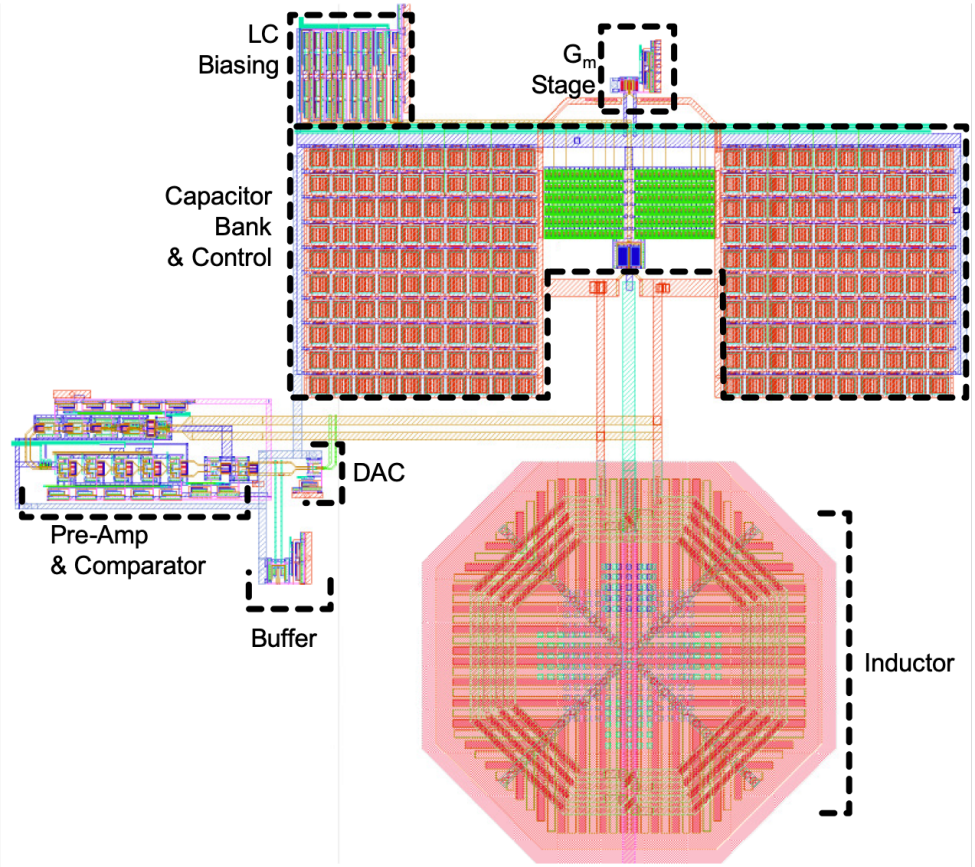


Figure 4.10: Complete layout of the RF ΣΔ ADC.

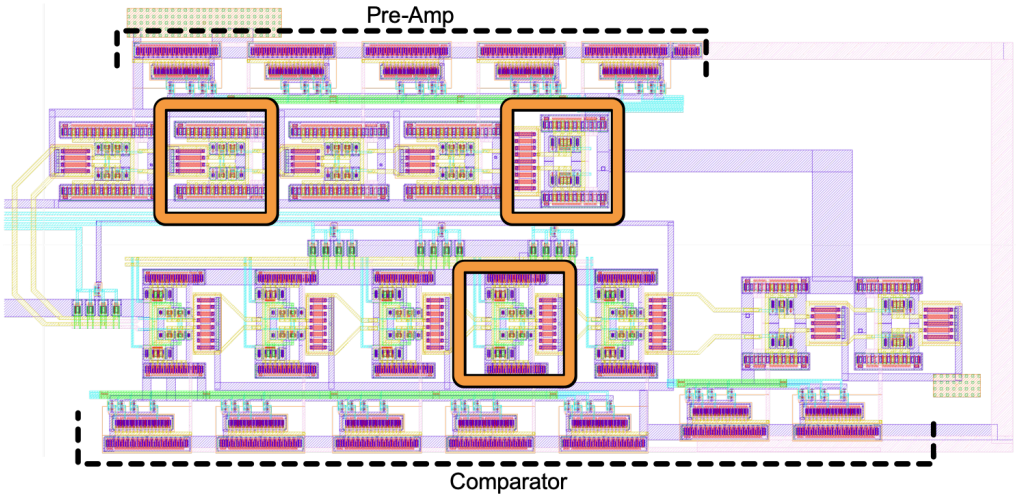


Figure 4.11: Zoom in into the pre-amplifier and comparator layouts. The obfuscated blocks are highlighted.

We obfuscated an existing bandpass RF ΣΔ ADC design in a 65 nm CMOS technology [87] following the design flow (a), shown in Fig. 4.1a. Its block-level schematic is shown in Fig. 4.9. It is a large-size circuit with $D = 1100$ components and is composed of several sub-blocks. It is part of an RF receiver and is re-configurable such that the RF receiver can be programmed to serve for establishing communication using several standards within the frequency range from 1.5 GHz to 3 GHz, including Bluetooth, ZigBee, WiFi 802.11b, LTE1800, LTE2100, LTE2600, etc. Herein, we consider a fixed configuration setting where the center frequency of the bandpass ΣΔ ADC is set at $f_0 = 3$ GHz and the sampling frequency is set at $f_s = 12$ GHz.

A careful look at the circuit netlist shows that $D' = 75$ components are candidates for obfuscation. To satisfy objective 1 we iterated three times over the inner loop of Fig. 4.1a, and then to satisfy objective 2 we had to iterate once over the outer loop of Fig. 4.1a. The two objectives were met by obfuscating a few components in only two of the sub-blocks, namely the pre-amplifier and the comparator, as illustrated in Fig. 4.9. In particular, within the pre-amplifier we obfuscated two differential transistor pairs and a resistor in two different amplification stages, and within the comparator we obfuscated a latch through its input differential transistor pair. The differential transistor pairs are laid out in common-centroid pattern and the resistor in a serpentine pattern. Fig. 4.10 shows the obfuscated layout. Fig. 4.11 zooms in into the obfuscated pre-amplifier and comparator. Fig. 4.12 shows a further zoom in into the obfuscated areas of one amplification stage of the pre-amplifer. The obfuscation area overhead is practically zero.

We consider the main performance which is the SNR. Fig. 4.13 shows the SNR as a function of input power amplitude for the nominal non-

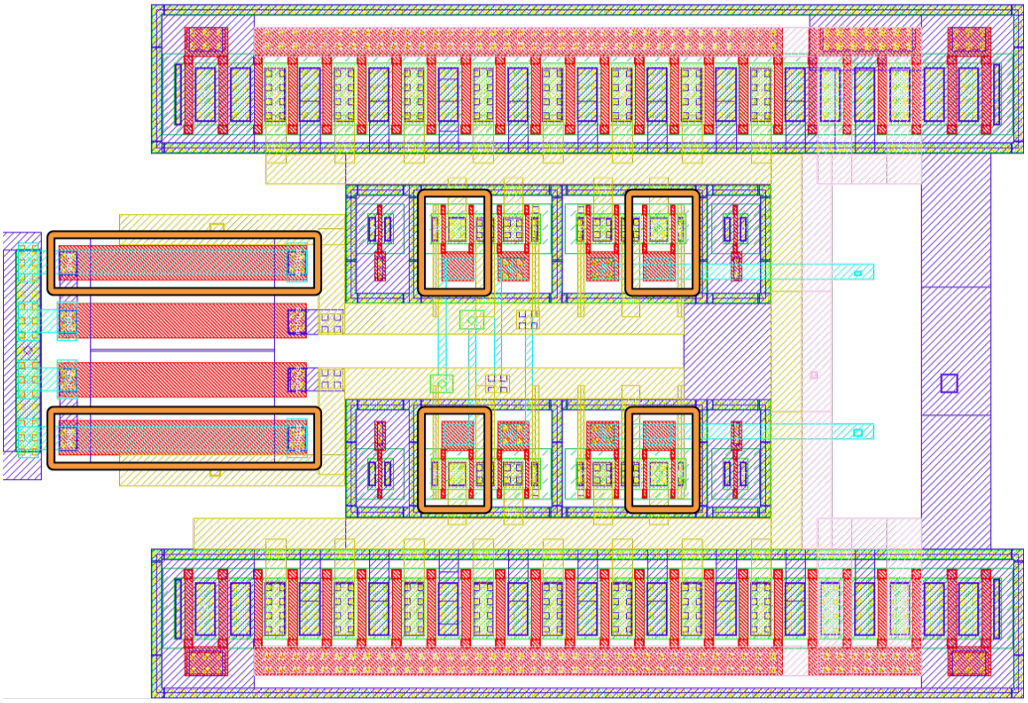


Figure 4.12: Zoom in into the obfuscated areas of one amplification stage of the pre-amplifer. The obfuscated areas are highlighted.

obfuscated, nominal obfuscated, and all-true contact designs. The SNR is computed on the layout extracted netlist with parasitics. One approximate SNR simulation for a given input power amplitude took up roughly 5 hours. As it can be seen, the nominal obfuscated design shows no performance penalty, whereas the all-true contact design shows a degraded SNR that even falls below 0 dB for smaller power amplitudes, which means that the signal is completely buried under noise. In fact, the small obfuscation within the pre-amplifier and comparator resulted in slight performance deviation for these two sub-blocks. Since the specifications of the sub-blocks are unknown to the attacker, the attacker cannot identify which are the obfuscated sub-blocks.

In total, the search space is computed from Eq. (4.2) to be $S' = 8.9 \times 10^{41} \approx 2^{139}$. α_{max} is given by an obfuscated differential transistor pair in a pre-amplifier stage and is computed to be $\alpha_{max} = 8/12 \approx 0.67$. Using these values Eq. (4.6) gives $\lambda_1 = 110$ bits. Since simulating the extracted layout netlist is very time-consuming, we will assume that the attacker will perform a first step analysis at schematic-level where one approximate SNR simulation for a given input power amplitude takes up far less time, about 20 minutes. We consider that the attacker will measure SNR at 5 input power amplitudes $P_{in} = \{-60, -50, -40, -37.5, -35\}$ dBm spanning the input dynamic range, thus total simulation time will be 100 minutes. In fact, the attacker will have to verify additional performances, e.g., Spurious Free Dynamic Range (SFDR). Assuming $T \geqslant 100$ minutes

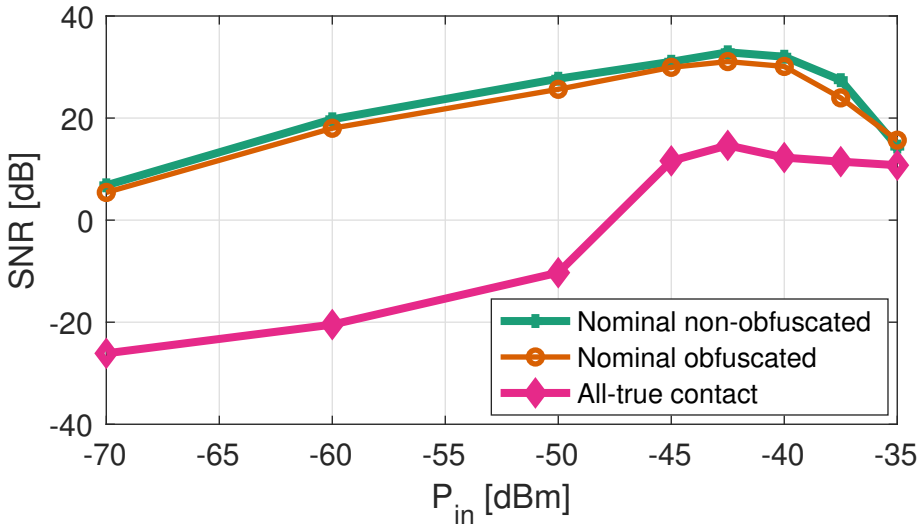


Figure 4.13: SNR vs. input power amplitude for the nominal non-obfuscated, nominal obfuscated, and all-true contact designs.

as an optimistic lower bound of simulation time per circuit instance, it still gives $\lambda_2 \geqslant 2.5 \times 10^{29}$ years. Due to the costly simulations, we simulated $n = 100$ random variants of the circuit and none of them passed the SNR specification, giving $\lambda_3 = 100\%$. We computed also $\lambda_4 = 139.11\%$ and $\lambda_5 = 8.56\%$.

Table 4.4 summarizes the obfuscation metrics. In conclusion, the practically zero-overhead obfuscation resulted in no measurable performance penalty for the nominal obfuscated design, in significant performance penalty for the all-true contact design, and in utterly impossible reverse engineering via a brute force attack on an ideally reduced search space.

4.8 CONCLUSION

We presented an obfuscation methodology for analog ICs via sizing camouflaging making use of fake contacts. We proposed two camouflaging design flows that consider camouflaging of an existing design and involving camouflaging in the design phase. We demonstrated that for realistic and large-size circuits and systems the methodology results in remarkable security against a brute-force attack performed in a reduced space after some informed assumptions by the attacker. We demonstrated also that it suffices to obfuscate few components, which minimizes the overall camouflaging effort and yields practically zero area overhead and performance penalty. In terms of future work, we are planning to extend the library of obfuscated components and also study more extensively possible counter-attacks sketched in Section 4.5.

METRIC	EVALUATION
S'	2^{139}
λ_1	110 bits
λ_2	2.5×10^{29} years
λ_3	100 %
λ_4	139.11 %
λ_5	8.56 %
Nominal obfuscated performance penalty	no, see Fig. 4.13
Obfuscation area overhead	0 %
All-true contact performance penalty	significant, see Fig. 4.13

Table 4.4: Obfuscation metrics for the RF ΣΔ ADC.

5

BREAKING ANALOG BIASING LOCKING TECHNIQUES VIA RE-SYNTHESIS

While in the last Chapters we proposed techniques to provide security, in this Chapter we switch sides and attack previously proposed techniques. More specifically we demonstrate an attack to break all analog circuit locking techniques that act upon the biasing of the circuit [111]. The attack is based on re-synthesizing the biasing circuits and requires only the use of an optimization algorithm. It is generally applicable to any analog circuit class. For the attacker the method requires no in-depth understanding or analysis of the circuit. The attack is demonstrated on a bias-locked LDO regulator. As the underlying optimization algorithm we employ a Genetic Algorithm (GA).

5.1 INTRODUCTION AND CONTEXT

Analog biasing locking is a security technique for analog circuits that has received a considerable amount of attention from the research community. The previous work in this field is elaborated in Section 2.2.1.2. There are several reasons for the popularity of analog biasing locking: (a) it is generally applicable to all analog circuits; (b) an incorrect key will have a dramatic impact on the performance trade-off of the circuit; (c) the lock mechanism is added at the most outside layer of the circuit, thus locking is dissociated from the design of the core circuit and does not incur any performance penalties.

Recently the first attack targeting analog biasing locking techniques was proposed in [97]. Compared to the attack in [97], the attack proposed in this Chapter is generally applicable to any analog circuit class and alleviates significantly the assumptions made regarding the capabilities of the attacker. It is inspired from analog circuit synthesis and design exploration methods [98]–[106], [108]. The underlying idea is to remove the locked biasing circuit and re-synthesize it so as to recover the circuit's intent performance trade-off. The attacker will only need to rely on an optimization algorithm to complete the attack and does not need to be knowledgeable in analog circuit design. In our implementation, we use a GA, in particular the NSGA-II algorithm [112]. The attacker can find online a free and open-source Matlab implementation of the NSGA-II algorithm. The proposed attack is demonstrated on a LDO regulator protected with biasing locking.

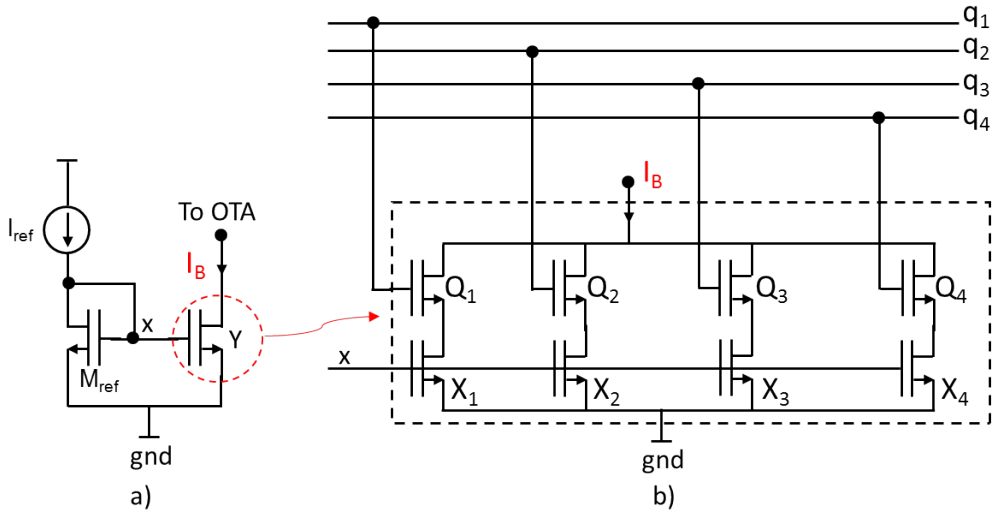


Figure 5.1: Locking a current mirror using the technique from [58].

This Chapter is structured as follows. In Section 5.2, we review the prior attack in [97]. In Section 5.3, we present the proposed attack. In Section 5.4, we compare the prior and proposed attacks. In Section 5.5, we discuss the implementation of the GA. In Section 5.6, we present the results of our case study. Section 5.7 concludes the Chapter.

5.2 PRIOR ATTACK

The attack proposed in [97] is based on Satisfiability Modulo Theory (SMT) and is shown to break the biasing locking techniques proposed in [57]–[59]. We refer to it as SMT-based attack.

The threat model assumes that the attacker has access to the netlist of the locked circuit, but does not know the valid key. The attacker is also in possession of the PDK of the technology and of the circuit data-sheet which specifies the performances and their specifications. The attacker will need to develop test benches for measuring the performances and perform circuit simulations. The most tedious and difficult aspect of the attack is that the attacker will need to write several circuit equations as will be described next. To solve these equations, the attacker will need to have access to an SMT-solver.

In particular, in a circuit with D locked biases, for the i-th locked bias b_i, $i = 1, \cdots, D$, we write an equation:

$$y_i = \phi_i(\mathbf{q}^i), \tag{5.1}$$

where $\mathbf{q}^i$ is the string of key-bits, i.e., $\mathbf{q}^i = [q_1^i, \cdots, q_k^i]$. For example, in the case of a locked current mirror using the technique shown in Fig. 5.1 [58], y_i is the aspect ratio of transistor Y, i.e., $y_i = (W/L)_Y$, and

$\phi_i(\mathbf{q}^i) = \sum_{j=1}^{k} q_j^i \, (W/L)_{X_j}$, where $(W/L)_{X_j}$ is the aspect ratio of transistor X_j and $q_j^i \in \{0, 1\}$.

Next, we write an equation to express the relationship between the bias b_i and y_i:

$$b_i = \psi_i(y_i). \tag{5.2}$$

For example, referring to Fig. 5.1, we have $b_i = y_i' \cdot I_{ref}$, where y_i' is the normalized aspect ratio of transistor Y with respect to transistor M_{ref}.

In addition, based on the m performances $\mathbf{p} = [p_1, \cdots, p_m]$ in the data-sheet, we write m equations linking each performance p_j to several biases b_i:

$$p_j = \theta_j(\mathbf{b}), \tag{5.3}$$

where $\mathbf{b} = [b_1, \cdots, b_D]$.

Thereafter, an SMT-solver is used to find a combination of keys $\mathbf{q} = [\mathbf{q}^1, \cdots, \mathbf{q}^D]$ that satisfies the combined equations:

$$p_j = \theta_j([\psi_1(\phi_1(\mathbf{q}^1)), \cdots, \psi_D(\phi_D(\mathbf{q}^D))]). \tag{5.4}$$

The SMT-based attack presents the following difficulties which limit its practicality:

(a) Deriving the functions θ_j is hardly possible for system-level performances. For example, considering an ADC main performances include SNR, Differential Non-Linearity (DNL), Integral Non-Linearity (INL), etc. An analog circuit's biases have a quantifiable impact on its DC operating point but not an easily quantifiable impact on such complex system-level performances that require transient or AC analysis and that arise due to non-idealities, higher order effects, and noise. For such complex circuits, the data-sheet does not list block-level performances where the SMT-based attack could apply since those are irrelevant for the end-user.

(b) Functions ϕ_i, ψ_i, and θ_j can only be derived assuming simplified transistor model equations, i.e., the Spice level 1 model (aka square-law model). This introduces large imprecision and, thereby, one needs to consider margins on the bias. Due to these margins, the attack may result in a large set of possible keys, all satisfying Eq. (5.4). With the correct key ideally being within this set, the attacker is left with the task of launching a brute-force attack, simulating the circuit using this set of keys, in order to single out the valid key. If the key search space is large and the circuit has long simulation times, the brute-force attack turns out to be very time-consuming.

(c) Deriving the functions θ_j requires a very high expertise by the attacker. In other words, this attack makes very strong assumptions about the capabilities of the attacker.

5.3 PROPOSED ATTACK

To remedy the difficulties of the SMT-based attack we propose a novel alternative attack that leverages analog circuit synthesis.

The underlying observation is that to break biasing locking techniques it suffices to search for the correct biases instead of the key, which is arguably a much easier problem since typically there are few biases that additionally only operate within a limited range of values. Unlike the SMT-based attack that aims at unlocking the biasing circuit by extracting and applying the correct key, the proposed attack first removes the locked biasing circuit and replaces it with a "fresh" non-locked biasing circuit that is sized accordingly so as to produce the desired biases.

In our threat model, similar to the SMT-based attack, we assume that the attacker has access to the circuit netlist and PDK, knows the target performances and their specifications, and has developed test benches for measuring the performances. However, unlike the SMT-based attack which requires writing circuit equations, the proposed attack treats the circuit as a black-box. The attacker will only need to run an optimization algorithm with comprehensive decision variables and objectives.

The proposed attack has 2 versions that are described in detail below. Their high-level descriptions are illustrated in Fig. 5.2. In the first step, common to both versions, the attacker identifies the locked biasing circuits by tracing the key-bits.

Version (a): The attacker replaces the locked biasing circuits with original non-locked biasing circuits. For example, in the case of a locked current mirror shown in Fig. 5.1, the attacker will reinstate the original schematic of the locked current mirror, i.e., remove the multi-branch structure shown in the right-hand side and replace it with a single transistor *Y*. However, the desired biases are unknown, i.e., bias current I_B in Fig. 5.1, and, thereby, the sizing of the components of the biasing circuits, i.e., the sizing of transistor *Y* in Fig. 5.1, are unknown. The attacker will run a multi-objective optimization algorithm to synthesize, i.e., size, the biasing circuits in the context of the complete circuit with the aim of satisfying all performance specifications. The decision variables are the values of the components in the biasing circuits, i.e., transistor aspect ratios, resistor values, etc. For example, in the case of a locked current mirror shown in Fig. 5.1, there is a single decision variable, i.e., the width *W* of transistor *Y* since the length L is left untouched by locking. The objective function is a metric of performance trade-off

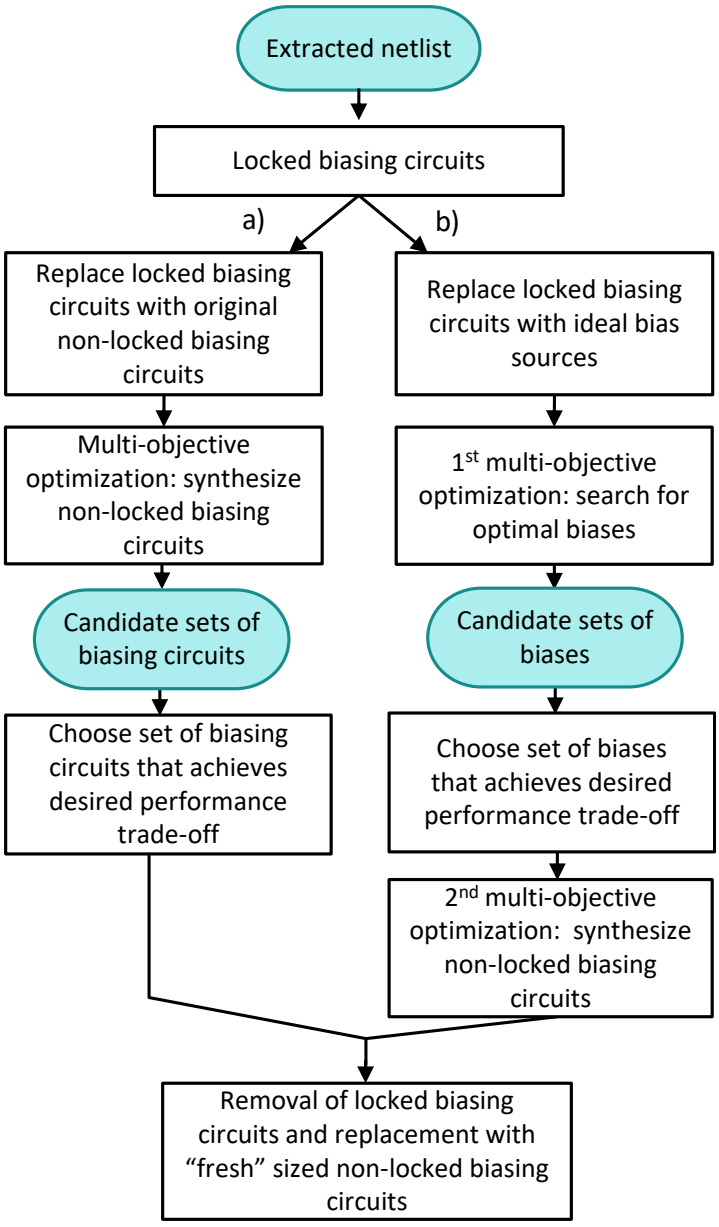


Figure 5.2: Flowchart of the two versions of the proposed attack.

for the circuit, i.e., the Euclidean distance from the specification boundaries measured with the $\|\cdot\|_2$ norm. During optimization, for every visited set of decision variables, the optimizer generates the corresponding circuit netlist and queries the circuit simulator to measure the performances using the test benches and compute the objective function. The optimization algorithm will return a Pareto front with p Pareto optimal solutions of performance trade-offs. A solution is considered Pareto optimal or non-dominated when no performance can be improved further without degrading any other performance. Each Pareto solution is produced by a specific set of sized biasing circuits. Then, the attacker has the freedom to choose a Pareto optimal solution that best meets the performance trade-off goal. It is noteworthy that the biases found may be different from the original obfuscated biases, but will still achieve similar or even better performance trade-offs. Finally, the attacker proceeds with the removal of the locked biasing circuits and their replacement with the corresponding sized biasing circuits resulting from optimization.

Version (b): The attacker removes the locked biasing circuits and replaces them with ideal bias sources. In this case, a first optimization is run using the biases as decision variables. The attacker will choose a Pareto optimal solution that meets the performance trade-off goal, and, in a second optimization step, the attacker will synthesize original non-locked biasing circuits independently of the core circuit such that they produce the biases resulting from the first optimization. Finally, the attacker will combine the sized non-locked biasing circuits with the core circuit and will run a confirmatory simulation so as to verify the performance trade-off.

So far we have not made any mention to the underlying optimization algorithm. In fact, any optimization algorithm can be used. In our implementation we use a GA that will be described in more detail in Section 5.5.

5.4 COMPARING THE ATTACKS

Conceptually, the difference between the proposed attack and the SMT-based attack is that the proposed attack is a removal attack aiming at automatically redesigning the locked biasing circuits, whereas the SMT-based attack aims at extracting the valid key.

A direct comparison regarding the required capabilities of the attacker for the two attacks is shown in Table 5.1. First, the SMT-based attack requires deriving circuit equations, which is hardly possible for many circuit classes, as discussed in Section 5.2. In contrast, the proposed

	PROPOSED ATTACK	SMT ATTACK [97]
Attack type	Removal	Key recovery
Shared requirements	PDK, data-sheet, netlist, testbenches, circuit simulator	
Additional requirements	Optimization algorithm, e.g., GA	Equations ϕ, ψ, θ, SMT-solver
Attacker expertise	Low	High

Table 5.1: Requirements for analog bias locking attacks.

attack requires only a common optimization algorithm and is generally applicable to any circuit class. Second, deriving circuit equations is a complex task requiring high analog design expertise, whereas with the proposed attack the attacker does not need to have any analog design expertise. In fact, the circuit can be handled as a black-box. Therefore, the proposed attack can be implemented by a “weak" attacker and, thus, it is considerably more powerful than the SMT-based attack.

5.5 GENETIC ALGORITHM

In our implementation, we employ a GA for performing heuristic multi-objective optimization. GAs are global optimization methods inspired by natural selection in nature. Mechanisms such as survival of the fittest, mutation and crossover are applied to a population to create descendant populations with ever-improving performances. Performances of individuals are evaluated through objective functions (aka fitness functions). In particular, we employ a Matlab implementation of the NSGA-II GA [112]. It is a free and open-source code with comprehensive parameters to be set, thus it can be readily used by any adversary to perform successfully the attack. The NSGA-II GA is configured as follows:

- A decision variable, i.e., component values such as transistor width for version (a) of the attack and bias values for version (b) of the attack, is represented with a gene. A gene is real-coded with real-valued floating point vectors. The concatenation of all genes represents the chromosome corresponding to a biasing circuit netlist for version (a) of the attack and biases for version (b) of the attack.
- We constrain the optimization by defining loose boundaries in the decision variables space so as to avoid unrealistic values, i.e., gigantic transistor widths and negative biases.

- We define loose boundaries for the performances set at a value equal to the specification multiplied by a factor of two. For chromosomes that have performances outside this range we penalize their fitness function such that they are disregarded from the next generation. In this way, the search is constrained within meaningful performance trade-offs.
- The selection function uses a binary tournament, where two randomly chosen candidates of the parent generation compete. The individual dominating the other, i.e., with a better rank, wins the tournament. When candidates with equal ranks are competing the crowding distance is decisive, by preferring solutions that are in less crowded regions. We are computing the crowding distance in the decision variable space which ensures a diverse population.
- The number of elite individuals to be reused in a new generation is controlled in order to preserve a diverse population. We limited the number of individuals on the Pareto front, i.e., of rank 1, to 35 % of the population.
- The chosen crossover operator uses line recombination, meaning that the offspring lies on a random point on the line between its two parents. 80 % of a new generation - excluding elite children - are generated using the crossover operator.
- We chose a Gaussian mutation operator, i.e., a random number chosen from a Gaussian distribution is added to each gene of a parent. 20 % of a new generation - excluding elite children - is generated using the mutation operator.
- Other settings are as follows: population size 50, number of generations 100.

5.6 CASE-STUDY AND RESULTS

Our case-study is an LDO regulator whose role is to provide stable supply voltages to sensitive circuits and is therefore widely applied in all sorts of ICs. The LDO regulator must compensate for variations in, e.g., load current, temperature or the global supply voltage. This makes an LDO regulator an interesting circuit to lock, especially in the context of a SoC with many sub-circuits that require stable supply voltages within small margins to operate correctly. With an incorrect key, a locked LDO regulator will provide out-of-spec supply voltages and, thereby, disrupt functionality of parts or the entire SoC.

We designed an LDO regulator in a 65 nm CMOS technology using the free and open-source OCEANE tool [113]. The block-level schematic

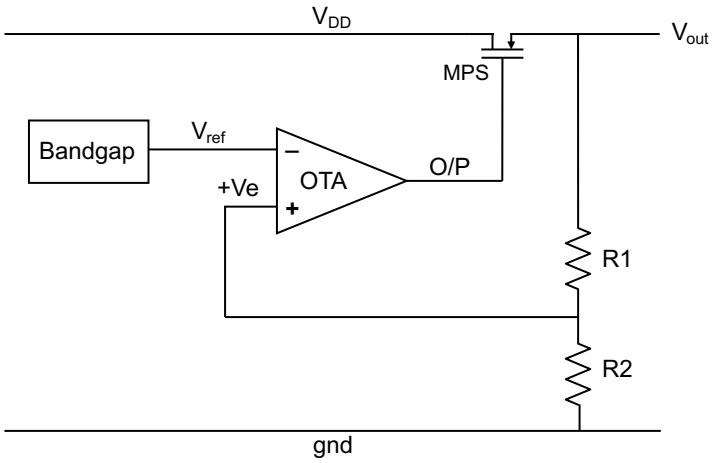


Figure 5.3: LDO regulator block-level schematic.

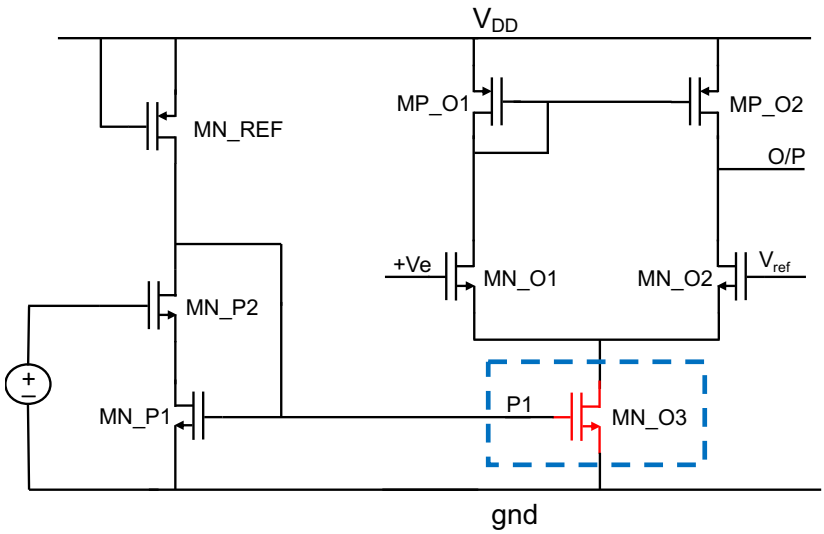


Figure 5.4: Error amplifier circuit inside the LDO regulator.

of the LDO regulator is shown in Fig. 5.3. The transistor-level schematics of the error amplifier circuit, implemented with an OTA, and the bandgap voltage reference circuit are shown in Figs. 5.4 and 5.5, respectively. Going down in the LDO regulator hierarchy, the bandgap voltage reference circuit includes an internal amplifier implemented with a self-biased OTA (SOTA) whose transistor-level schematic is shown in Fig. 5.6. The error amplifier in Fig. 5.4 and internal amplifier in Fig. 5.6 require proper biasing provided through current mirrors.

The LDO regulator is locked via locking these two current mirrors using the technique shown in Fig. 5.1 [58]. In Figs. 5.4 and 5.6 we highlight the obfuscated mirror transistor, i.e., transistor Y referring to Fig. 5.1.

The main performances of the LDO regulator that we consider include:

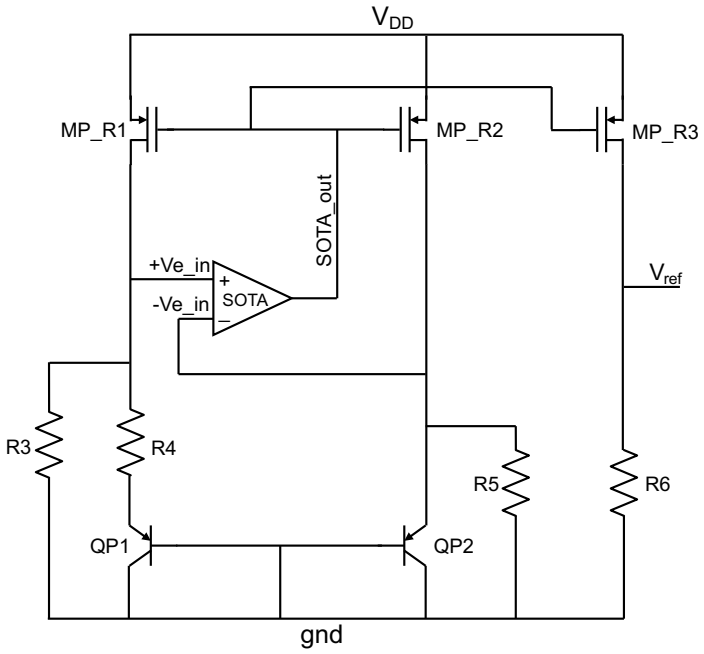


Figure 5.5: Bandgap voltage reference circuit inside the LDO regulator.

(a) Output voltage dependence on temperature variations from $-55\,°C$ to $125\,°C$, i.e., $\frac{\Delta V_{out}}{\Delta T}\left[\frac{mV}{°C}\right]$;

(b) Output voltage dependence on supply voltage variations from $V_{DD} = 1.5\,V$ to $3\,V$, i.e., $\frac{\Delta V_{out}}{\Delta V_{DD}}\left[\frac{mV}{V}\right]$;

(c) Output voltage offset from the nominal output voltage of $1.18\,V$ under full and zero load current, denoted by ΔV_{out} @ $0\,mA$ $[mV]$ and ΔV_{out} @ $50\,mA$ $[mV]$, respectively;

(d) Regulated output voltage overshoot as a response to a sudden variation of the load current from $0\,mA$ to $50\,mA$.

Table 5.2 shows the nominal performances and 5 diverse Pareto solutions produced by each attack version. All performances have an upper specification. Figs. 5.7, 5.8 and 5.9 plot the LDO regulator characteristic measurements for the nominal design, a locked design when applying a random incorrect key, and the unlocked design using Pareto solution #2 of version (a) of the attack. As it can be seen from Table 5.2, both versions of the attack are capable of fully recovering the circuit functionality, resulting in good performance trade-offs compared to the nominal one. In fact, thanks to the optimization, the attack is even capable of finding improved performance trade-offs.

To visualize the Pareto front and appreciate the diversity of Pareto solutions, we run the attack by considering only two performances,

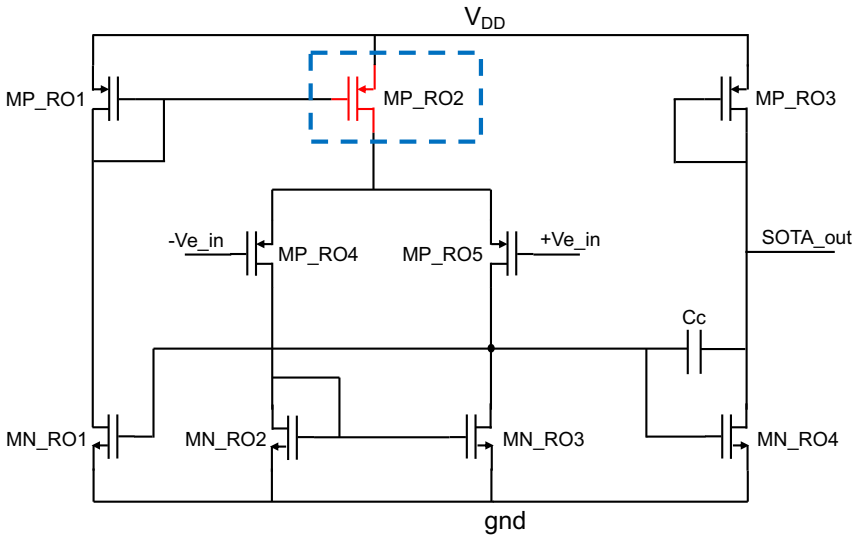


Figure 5.6: Internal amplifier circuit inside the bandgap voltage reference circuit.

namely ΔV_{out} @ 50 mA and V_{out} overshoot. The Pareto front is illustrated in Fig. 5.10.

A single call to the simulator to extract all performances takes approximately 5.8 seconds on an Intel Xeon E5-2640 @ 2.40 GHz with 128 GB of RAM. To perform the single optimization in version (a) of the attack and the first optimization in version (b) of the attack, we run a GA using 100 generations with a population of 50 per generation, totaling in 5000 simulator calls. The second optimization in version (b) of the attack sizes the standalone biasing circuit for which simulation time is very small. In this case, the GA terminates in less than 5 minutes. Therefore, the time of version (a) of the attack is 8.06 hours and the simulation time of version (b) of the attack is only slightly longer.

5.7 CONCLUSION

We demonstrated an attack that breaks any biasing locking technique. The attack is based on removing the locked biasing circuits and re-synthesizing them. The attack applied to a protected LDO regulator is completed in about 8 hours and recovers an LDO regulator with excellent performance trade-off. It requires only the use of an optimization algorithm, thus it can be performed even by “weak" attackers that have no analog design expertise. In addition, the attack is generally applicable to any analog circuit.

	$\frac{\Delta V_{out}}{\Delta T}\left[\frac{mV}{°C}\right]$	$\frac{\Delta V_{out}}{\Delta V_{DD}}\left[\frac{mV}{V}\right]$	ΔV_{out}@ 0 mA [mV]	ΔV_{out}@ 50 mA [mV]	V_{out} over-shoot[mV]
Nominal	8.11	33.45	1.12	3.15	47.97
Version a) Pareto sol.					
#1	5.73	34.74	0.91	4.09	35.45
#2	8.69	32.95	0.98	3.11	51.20
#3	7.94	35.89	0.35	3.60	38.52
#4	5.58	35.17	1.14	3.85	36.57
#5	6.24	35.42	0.62	3.83	36.74
Version b) Pareto sol.					
#1	10.43	33.26	0.30	3.11	51.67
#2	4.83	35.00	2.05	3.64	38.06
#3	12.53	36.54	3.27	3.40	41.02
#4	8.99	36.05	0.99	3.53	39.27
#5	9.27	34.59	0.14	3.22	45.20

Table 5.2: Recovered LDO regulator performance trade-offs resulting from the attack.

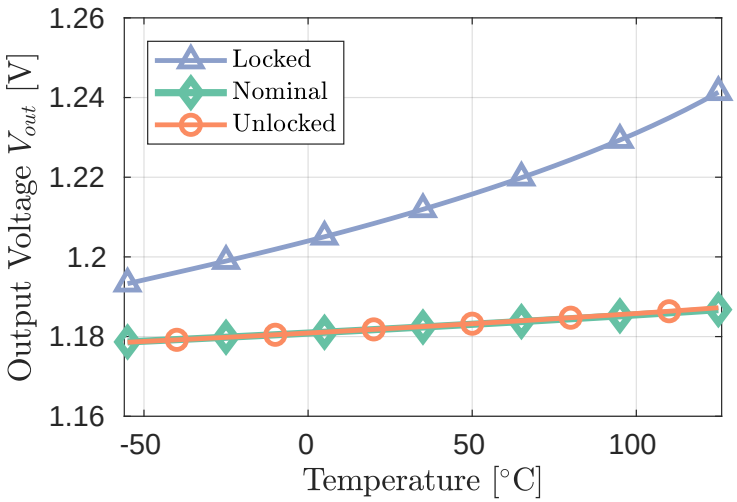


Figure 5.7: LDO regulator output voltage vs. temperature variations.

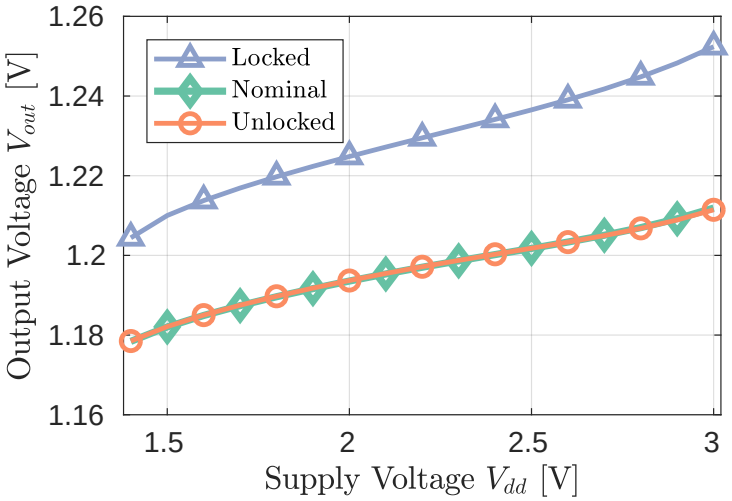


Figure 5.8: LDO regulator output voltage vs. supply voltage variations.

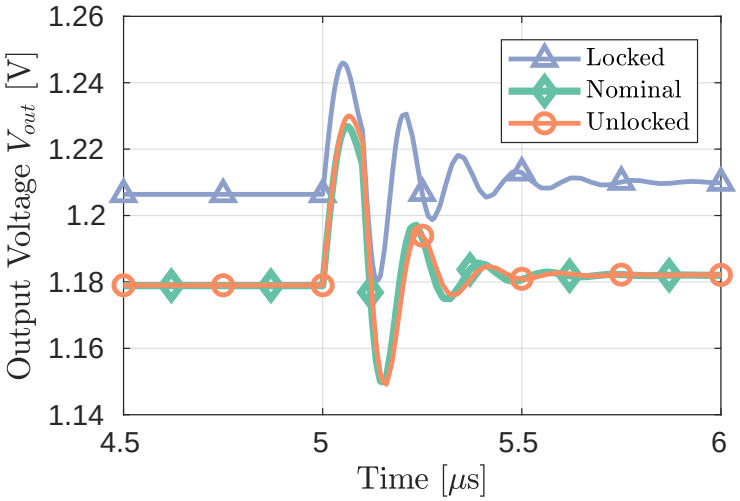


Figure 5.9: LDO regulator output voltage vs. time showing the response to a sudden variation of the load current.

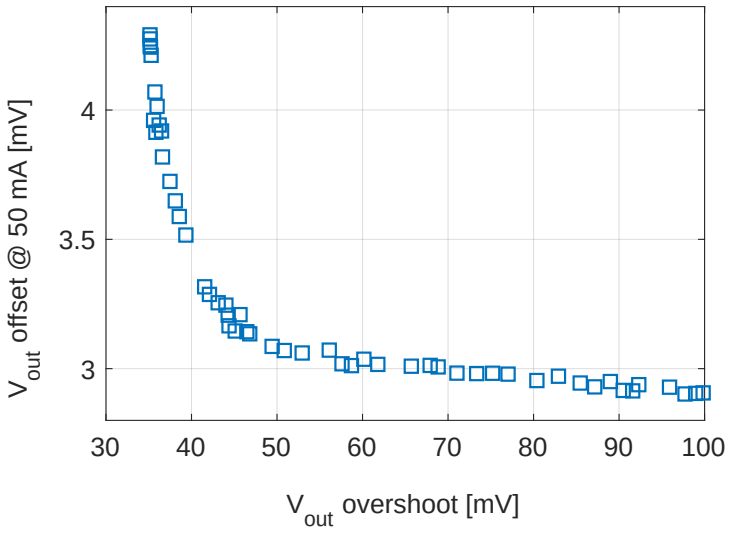


Figure 5.10: Pareto front showing trade-off between two optimization objectives: ΔV_{out} @ 50 mA vs. V_{out} overshoot.

6

CONCLUSION & OUTLOOK

6.1 CONCLUSION

In the foreseeable future globalization will continue to pressure the semiconductor industry to conduct research in areas that were not prioritized in the last decades: The rise of the SoC, driven by applications such as the internet-of-things, or the extreme costs of building and maintaining IC manufacturing facilities, are giving rise to fabless and IP-heavy business models, massively outsourcing production steps. Consequently the semiconductor supply chain finds itself scattered all over the planet and for an IC design house, to assure every single contractor and sub-contractor in this chain is trustworthy, seems like a herculean task. Suddenly, becoming the target of IP theft, hardware Trojans or counterfeiting campaigns from within that very supply chain is not a highly improbable scenario anymore. This endangers not only the software that runs security-critical processes on at-risk hardware, but possibly even more the reputation and revenue of entire companies.

This thesis demonstrates that establishing security and trust for analog IPs and ICs, while still in its infancy, is feasible. Nevertheless it can be considered a hard research topic, whose advancement is slowed by the nature of analog circuits such as their sensitivity to any sort of modification within the design, or the non-automated and largely manual design flow. The techniques presented in the literature and published as a result of this thesis already have the potential to protect analog and mixed-signal circuits against subsets of all the possible attack scenarios. This makes a clear threat model to understand the risks a specific IP/IC faces indispensable, thereby allowing to find a suitable security method for an IP/IC.

6.2 CONTRIBUTIONS OF THE THESIS

Three main contributions to the state of the art are the result of this thesis and can be compiled as follows:

IN CHAPTER 3 we proposed *MixLock* [69], the first technique to allow securing mixed-signal circuits against IC piracy threats, including overproduction, cloning, reverse engineering, etc. via logic locking of the circuit's digital section. We explored the impact of digital security methods on the overall system and derived security metrics for the analog and the digital domain, highlighting that the two

must be co-optimized to achieve holistic security. Two different state-of-the-art logic locking mechanisms, namely SFLL and DisORC & TRLL were implemented in the ΣΔ ADC case study's digital decimation filter. The efficiency of locking in the analog domain was analyzed by leveraging SNR as performance measure. Furthermore we demonstrated *MixLock* by applying it to hardware in an experiment with a ΣΔ ADC. We recorded the output bitstream of a silicon-implemented ΣΔ modulator and processed the bitstream on an FPGA implementation of the locked digital decimation filter. We thereby confirmed the theoretical and simulational foundations laid out before and showed the capabilities of *MixLock* in a hardware experiment. Finally in an audio-demonstrator real-world audio signals were processed by a locked ADC and their audio-quality was assessed [88].

IN CHAPTER 4 we demonstrated an analog obfuscation technique specifically for analog circuits to combat reverse engineering related threats by carrying out only minute changes within the circuit's layout [89]. We camouflaged effective transistor geometries by leveraging fake interconnects, thereby assuring that reverse engineering attacks would reveal an incorrect netlist to the attacker. We presented a library of obfuscated layout components and gave recommendations for effective application of the camouflaging technique. Furthermore we proposed security metrics for evaluating the difficulty of reverse engineering. We applied the technique to two case studies: an operational amplifier and a ΣΔ ADC. We showed that although the netlists recovered through reverse engineering appear structurally correct they present inferior performances.

IN CHAPTER 5 we switched sides and took the role of the attacker. We proposed an easy to set up, low effort attack [111] to break all those defense mechanisms that leverage analog biasing locking, which thus far constitutes the most researched security mechanism for analog circuits. We compared our removal attack, based on replacing and re-synthesizing the locked biasing circuits, to the state of the art, stressing its few requirements. Finally we demonstrated the attack on a locked LDO voltage regulator while employing a genetic algorithm for means of multi-objective optimization for circuit synthesis.

6.3 FUTURE WORK & OUTLOOK

The contributions made within the scope of this thesis could be extended and improved in a number of ways. As first perspective applying *MixLock* to larger scale mixed-signal circuits would be a very instructive exercise,

especially with regard to system-level behavior. Locking a PLL via its digital loop-filter would have highly interesting effects on its stability and dynamic behavior, while locking digital blocks in an RF transceiver may impact adherence to spectral mask specifications or the BER. Layout obfuscation on the other hand may be extended to not only protect an analog circuit's netlist from reverse-engineering but also its architecture. This could be achieved by placing entire fake-components in vicinity of functional components, seemingly connecting them and thereby suggesting to an attacker a different circuit architecture than actually implemented. The latter two techniques could thereafter be applied in a compound technique to protect analog and digital sections of a mixed-signal circuit, ideally in the scope of a silicon implementation.

What remains to be evaluated is whether the scientific community will be able to find a generic way to apply means of hardware security to analog and mixed-signal circuits, i.e., can there be a mechanism that is suitable to secure an op-amp but also a PLL? Similar problems arise in the domain of analog testing, where currently every type of circuit requires a specific test-setup, which is thus not generic, in contrast to testing of digital circuit.

As a new research direction we envisage the protection of Artificial Intelligence (AI) hardware accelerators. AI and in particular Deep Neural Networks (DNNs) find numerous applications in various fields, including computer vision, natural language processing, robotics, medicine, automotive, etc. DNNs contain multiple layers of different types, comprise several thousands of synapses and neurons, and make a myriad of operations in one single forward pass. To deal with the complexities of AI workloads, running DNN models on a CPU is not efficient and specific hardware AI processors are required to accelerate training and inference. Moreover, there is a large incentive for moving the AI hardware accelerators from the cloud onto edge IoT devices, in order to address bandwidth, data privacy, energy consumption, and availability concerns. Widely used AI hardware accelerators are GPUs and FPGAs, but orders of magnitude of performance improvement in terms of speed, area, and energy consumption can be achieved with ASICs. ASIC implementation is an active research topic and several mostly proprietary designs have surfaced in the recent years. Examples are Google's TPU [114], Intel's Loihi [115], and IBM's True North [116]. As dedicated ASIC AI processor designs become a reality and enter high-volume manufacturing, they will be a subject of piracy. At this stage, the literature on anti-piracy methods is non-existing and research is needed to add security features for protecting not only the intellectual property of the hardware but also the DNN model itself that is loaded onto the hardware.

Part II

APPENDIX

BIBLIOGRAPHY

[1] H. Hung, Y. Chiu, and M. Wu, "Analysis of competition between IDM and fabless–foundry business models in the semiconductor industry," *IEEE Transactions on Semiconductor Manufacturing*, vol. 30, no. 3, pp. 254–260, 2017 (cit. on pp. 3, 4).

[2] Bill McClean, *Fabless Company Share of IC Sales to Set New Record in 2020 at 32.9%*, Dec. 2020. [Online]. Available: https://www.icinsights.com/data/articles/documents/1328.pdf (visited on 01/27/2021) (cit. on pp. 3, 4).

[3] Harald Bauer, Ondrej Burkacky, Stephanie Lingemann, Klaus Pototzky, Peter Kenevan, and Bill Wiseman, *Semiconductor design and manufacturing: achieving leading-edge capabilities*, Aug. 2020. [Online]. Available: https://www.mckinsey.com/industries/advanced-electronics/our-insights/semiconductor-design-and-manufacturing-achieving-leading-edge-capabilities (visited on 12/21/2020) (cit. on p. 3).

[4] K. Xiao, D. Forte, Y. Jin, R. Karri, S. Bhunia, and M. Tehranipoor, "Hardware Trojans: lessons learned after one decade of research," *ACM Transactions on Design Automation of Electronic Systems*, vol. 22, no. 1, 6:1–6:23, 2016 (cit. on p. 5).

[5] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: identifying and classifying hardware trojans," *Computer*, vol. 43, no. 10, pp. 39–46, 2010 (cit. on pp. 5, 13).

[6] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, "Hardware trojan attacks: threat analysis and countermeasures," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1229–1247, 2014 (cit. on pp. 5, 13).

[7] S. Skorobogatov and C. Woods, "Breakthrough silicon scanning discovers backdoor in military chip," in *Cryptographic Hardware and Embedded Systems – CHES 2012*, E. Prouff and P. Schaumont, Eds., ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2012, pp. 23–40 (cit. on p. 5).

[8] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache, "Fault injection attacks on cryptographic devices: theory, practice, and countermeasures," *Proceedings of the IEEE*, vol. 100, no. 11, pp. 1207–1228, 2012 (cit. on p. 6).

[9] H. Choukri and M. Tunstall, "Round reduction using faults," *Workshop on Fault Diagnosis and Tolerance in Cryptography*, vol. 5, pp. 13–24, 2005 (cit. on p. 6).

[10] I. Polian, "Security aspects of analog and mixed-signal circuits," in *Proc. IEEE International Mixed-Signal Testing Workshop*, 2016 (cit. on pp. 6, 15).

[11] K. Tiri and I. Verbauwhede, "A VLSI design flow for secure side-channel attack resistant ICs," in *Proc. Design, Automation & Test in Europe*, 2005 (cit. on p. 6).

[12] N. Beringuier-Boher, K. Gomina, D. Hely, J.-B. Rigaud, V. Beroulle, A. Tria, J. Damiens, P. Gendrier, and P. Candelier, "Voltage glitch attacks on mixed-signal systems," in *Proc. Euromicro Conference on Digital System Design*, 2014, pp. 379–386 (cit. on p. 6).

[13] B. Lippmann, M. Werner, N. Unverricht, A. Singla, P. Egger, A. Dübotzky, H. Gieser, M. Rasche, O. Kellermann, and H. Graeb, "Integrated flow for reverse engineering of nanoscale technologies," in *Proc. Asia and South Pacific Design Automation Conference*, 2019, pp. 82–89 (cit. on pp. 6, 8).

[14] R. Torrance and D. James, "The state-of-the-art in semiconductor reverse engineering," in *Proc. IEEE/ACM Design Automation Conference*, 2011, pp. 333–338 (cit. on pp. 6, 8).

[15] U. Guin, K. Huang, D. DiMase, J. M. Carulli, M. Tehranipoor, and Y. Makris, "Counterfeit integrated circuits: a rising threat in the global semiconductor supply chain," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1207–1228, 2014 (cit. on p. 7).

[16] S. E. Quadir, J. Chen, D. Forte, N. Asadizanjani, S. Shahbazmohamadi, L. Wang, J. Chandy, and M. Tehranipoor, "A survey on chip to system reverse engineering," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 13, no. 1, pp. 1–34, 2016 (cit. on p. 8).

[17] J. Markoff, "F.B.I. says the military had bogus computer gear," *The New York Times*, May 2008. [Online]. Available: https://www.nytimes.com/2008/05/09/technology/09cisco.html (visited on 01/25/2021) (cit. on p. 8).

[18] A. Greenberg, "Your guide to russia's infrastructure hacking teams," *Wired*, Dec. 2017. [Online]. Available: https://www.wired.com/story/russian-hacking-teams-infrastructure/ (visited on 01/26/2021) (cit. on pp. 8, 9).

[19] ——, "How an entire nation became russia's test lab for cyberwar," *Wired*, Jun. 2017. [Online]. Available: https://www.wired.com/story/russian-hackers-attack-ukraine/ (visited on 01/25/2021) (cit. on pp. 8, 9).

[20] David Kushner, "The real story of Stuxnet," *IEEE Spectrum*, Feb. 2013. [Online]. Available: https://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet (visited on 01/26/2021) (cit. on pp. 8, 9).

[21] M. Pecht and S. Tiku, "Bogus: electronic manufacturing and consumers confront a rising tide of counterfeit electronics," *IEEE Spectrum*, vol. 43, no. 5, pp. 37–46, 2006 (cit. on p. 9).

[22] Semiconductor Industry Association, *Winning the battle against counterfeit semiconductor products*, 2013. [Online]. Available: https://www.semiconductors.org/wp-content/uploads/2018/06/SIA-Anti-Counterfeiting-Whitepaper-1.pdf (visited on 01/25/2021) (cit. on p. 9).

[23] S. Adee, "The hunt for the kill switch," *IEEE Spectrum*, vol. 45, no. 5, pp. 34–39, May 2008. [Online]. Available: https://spectrum.ieee.org/semiconductors/design/the-hunt-for-the-kill-switch (visited on 01/26/2021) (cit. on p. 9).

[24] M. Rostami, F. Koushanfar, and R. Karri, "A primer on hardware security: models, methods, and metrics," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1283–1295, 2014 (cit. on p. 9).

[25] M. Tehranipoor and C. Wang, Eds., *Introduction to Hardware Security and Trust*. Springer-Verlag New York, 2012 (cit. on p. 9).

[26] Mark M. Tehranipoorr, Ujjwal Guin, and Swarup Bhunia, "Invasion of the hardware snatchers: cloned electronics pollute the market," *IEEE Spectrum*, Apr. 2017. [Online]. Available: https://spectrum.ieee.org/computing/hardware/invasion-of-the-hardware-snatchers-cloned-electronics-pollute-the-market (visited on 01/26/2021) (cit. on p. 9).

[27] B. Razavi, *RF Microelectronics, Second Edition*. Mc Graw Hill, 2012 (cit. on p. 10).

[28] Y. Jin and Y. Makris, "Hardware trojans in wireless cryptographic ICs," *IEEE Design & Test of Computers*, vol. 27, no. 1, pp. 26–35, 2010 (cit. on p. 13).

[29] Y. Liu, Y. Jin, and Y. Makris, "Hardware trojans in wireless cryptographic ICs: silicon demonstration & detection method evaluation," in *IEEE/ACM International Conference on Computer-Aided Design*, 2013, pp. 399–404 (cit. on p. 13).

[30] Y. Liu, Y. Jin, A. Nosratinia, and Y. Makris, "Silicon demonstration of hardware trojan design and detection in wireless cryptographic ICs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 4, pp. 1506–1519, 2017 (cit. on pp. 13, 15, 16).

[31] K. S. Subramani, A. Antonopoulos, A. A. Abotabl, A. Nosratinia, and Y. Makris, "ACE: adaptive channel estimation for detecting analog/RF trojans in WLAN transceivers," in *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2017 (cit. on pp. 13, 15).

[32] K. S. Subramani, N. Helal, A. Antonopoulos, A. Nosratinia, and Y. Makris, "Amplitude-modulating analog/RF hardware trojans in wireless networks: risks and remedies," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3497–3510, 2020 (cit. on p. 13).

[33] L. Lin, W. Burleson, and C. Paar, "MOLES: malicious off-chip leakage enabled by side-channels," in *IEEE/ACM International Conference on Computer-Aided Design*, 2009, pp. 399–404 (cit. on p. 13).

[34] D. Chang, B. Bakkaloglu, and S. Ozev, "Enabling unauthorized RF transmission below noise floor with no detectable impact on primary communication performance," in *Proc. IEEE VLSI Test Symposium*, 2015 (cit. on p. 13).

[35] S. Chang, G. Bhat, U. Ogras, B. Bakkaloglu, and S. Ozev, "Detection mechanisms for unauthorized wireless transmissions," *ACM Transactions on Design Automation of Electronic Systems*, vol. 23, no. 6, 70:1–70:21, 2018 (cit. on pp. 13, 15, 16).

[36] M. Elshamy, G. Di Natale, A. Pavlidis, M.-M. Louërat, and H.-G. Stratigopoulos, "Hardware trojan attacks in analog/mixed-signal ics via the test access mechanism," in *Proc. IEEE European Test Symposium*, 2020 (cit. on p. 14).

[37] K. Yang, M. Hicks, Q. Dong, T. Austin, and D. Sylvester, "A2: analog malicious hardware," in *Proc. IEEE Symposium on Security and Privacy*, 2016, pp. 18–37 (cit. on p. 14).

[38] M. M. Bidmeshki, K. S. Subramani, and Y. Makris, "Revisiting Capacitor-Based Trojan Design," in *2019 IEEE 37th International Conference on Computer Design*, IEEE, 2019, pp. 309–312 (cit. on p. 14).

[39] X. Cao, Q. Wang, R. L. Geiger, and D. J. Chen, "A hardware trojan embedded in the inverse widlar reference generator," in *Proc. IEEE International Midwest Symposium on Circuits and Systems*, 2015 (cit. on p. 14).

[40] Q. Wang, R. L. Geiger, and D. Chen, "Hardware trojans embedded in the dynamic operation of analog and mixed-signal circuits," in *Proc. National Aerospace and Electronics Conference*, 2015, pp. 155–158 (cit. on p. 14).

[41] Q. Wang, D. Chen, and R. L. Geiger, "Transparent side channel trigger mechanism on analog circuits with PAAST hardware trojans," in *IEEE International Symposium on Circuits and Systems*, 2018 (cit. on p. 14).

[42] Z. Liu, Y. Li, Y. Duan, R. L. Geiger, and D. Chen, "Identification and break of positive feedback loops in trojan states vulnerable circuits," in *Proc. IEEE International Symposium on Circuits and Systems*, 2014, pp. 289–292 (cit. on p. 14).

[43] C. Cai and D. Chen, "Performance enhancement induced trojan states in op-amps, their detection and removal," in *Proc. IEEE International Symposium on Circuits and Systems*, 2015, pp. 3020–3023 (cit. on p. 14).

[44] A. Antonopoulos, C. Kapatsori, and Y. Makris, "Security and trust in the analog/mixed-signal/RF domain: a survey and a perspective," in *Proc. IEEE European Test Symposium*, 2017 (cit. on pp. 15, 22).

[45] M. M. Alam, S. Chowdhury, B. Park, D. Munzer, N. Maghari, M. Tehranipoor, and D. Forte, "Challenges and opportunities in analog and mixed signal (AMS) integrated circuit (IC) security," *Journal of Hardware and Systems Security*, vol. 2, no. 1, pp. 15–32, 2018 (cit. on p. 15).

[46] A. Waksman, M. Suozzo, and S. Sethumadhavan, "FANCI: identification of stealthy malicious logic using boolean functional analysis," in *Proc. ACM Conference on Computer and Communications Security*, Association for Computing Machinery, 2013, pp. 697–708 (cit. on p. 15).

[47] M.-M. Bidmeshki, A. Antonopoulos, and Y. Makris, "Information flow tracking in analog/mixed-signal designs through proof-carrying hardware IP," in *Proc. Design, Automation and Test Conference in Europe*, 2017 (cit. on p. 15).

[48] S. Bhasin, J.-L. Danger, S. Guilley, X. T. Ngo, and L. Sauvage, "Hardware Trojan Horses in Cryptographic IP Cores," in *Workshop on Fault Diagnosis and Tolerance in Cryptography*, IEEE, 2013, pp. 15–29 (cit. on p. 15).

[49] M. Banga and M. S. Hsiao, "Trusted RTL: Trojan detection methodology in pre-silicon designs," in *IEEE International Symposium on Hardware-Oriented Security and Trust*, IEEE, 2010, pp. 56–59 (cit. on p. 15).

[50] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *IEEE Symposium on Security and Privacy*, IEEE, 2007, pp. 296–310 (cit. on pp. 15, 16).

[51] J. Rajendran, V. Jyothi, O. Sinanoglu, and R. Karri, "Design and analysis of ring oscillator based Design-for-Trust technique," in *Proc. of the 29th VLSI Test Symposium*, IEEE, 2011, pp. 105–110 (cit. on p. 16).

[52] S. Narasimhan, W. Yueh, X. Wang, S. Mukhopadhyay, and S. Bhunia, "Improving IC Security Against Trojan Attacks Through Integration of Security Monitors," *IEEE Design & Test of Computers*, vol. 29, no. 5, pp. 37–46, 2012 (cit. on p. 16).

[53] M. Elshamy, A. Sayed, M.-M. Louërat, A. Rhouni, H. Aboushady, and H.-G. Stratigopoulos, "Securing programmable analog ICs against piracy," in *Proc. Design, Automation and Test in Europe Conference*, 2020 (cit. on p. 17).

[54] S. G. R. Nimmalapudi, G. Volanis, Y. Lu, A. Antonopoulos, A. Marshall, and Y. Makris, "Range-controlled floating-gate transistors: a unified solution for unlocking and calibrating analog ICs," in *Proc. Design, Automation and Test in Europe Conference*, 2020 (cit. on p. 17).

[55] N. G. Jayasankaran, A. S. Borbon, E. Sanchez-Sinencio, J. Hu, and J. Rajendran, "Towards provably-secure analog and mixed-signal locking against overproduction," in *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2018 (cit. on p. 18).

[56] M. Yasin, A. Sengupta, M. T. Nabeel, M. Ashraf, J. Rajendran, and O. Sinanoglu, "Provably-secure logic locking: from theory to practice," in *Proc. ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1601–1618 (cit. on pp. 18, 25, 30, 31, 35).

[57] V. V. Rao and I. Savidis, "Protecting analog circuits with parameter biasing obfuscation," in *Proc. IEEE Latin American Test Symposium*, 2017 (cit. on pp. 18, 19, 21, 78).

[58] J. Wang, C. Shi, A. Sanabria-Borbon, E. Sánchez-Sinencio, and J. Hu, "Thwarting analog IC piracy via combinational locking," in *Proc. IEEE International Test Conference*, 2017 (cit. on pp. 18–20, 78, 85).

[59] D. H. K. Hoe, J. Rajendran, and R. Karri, "Towards secure analog designs: a secure sense amplifier using memristors," in *Proc. IEEE Computer Society Annual Symposium on VLSI*, 2014 (cit. on pp. 18, 19, 78).

[60] G. Volanis, Y. Lu, S. Govinda, R. Nimmalapudi, A. Antonopoulos, A. Marshall, and Y. Makris, "Analog performance locking through neural network-based biasing," in *Proc. IEEE VLSI Test Symposium*, 2019 (cit. on pp. 18–20).

[61] V. V. Rao and I. Savidis, "Transistor sizing for parameter obfuscation of analog circuits using satisfiability modulo theory," in *2018 IEEE Asia Pacific Conference on Circuits and Systems*, 2018, pp. 102–106 (cit. on p. 19).

[62] ——, "Mesh based obfuscation of analog circuit properties," in *IEEE International Symposium on Circuits and Systems)*, 2019, pp. 1–5 (cit. on p. 19).

[63] K. Juretus, V. V. Rao, and I. Savidis, "Securing analog mixed-signal integrated circuits through shared dependencies," in *Proc. ACM Great Lakes Symposium on VLSI*, 2019 (cit. on p. 21).

[64] A. Ash-Saki and S. Ghosh, "How multi-threshold designs can protect analog IPs," in *Proc. IEEE International Conference on Computer Design*, 2018, pp. 464–471 (cit. on p. 21).

[65] Y. Bi, J. S. Yuan, and Y. Jin, "Beyond the interconnections: split manufacturing in RF designs," *Electronics*, vol. 4, no. 3, pp. 541–564, 2015 (cit. on p. 21).

[66] X. Zhang and M. Tehranipoor, "Design of on-chip lightweight sensors for effective detection of recycled ics," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 5, pp. 1016–1029, 2014 (cit. on p. 22).

[67] K. Huang, Y. Liu, N. Korolija, J. M. Carulli, and Y. Makris, "Recycled IC detection based on statistical methods," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 6, pp. 947–960, 2015 (cit. on p. 22).

[68] K. He, X. Huang, and S. X.-D. Tan, "EM-based on-chip aging sensor for detection of recycled ics," *IEEE Design & Test*, vol. 33, no. 5, pp. 56–64, 2016 (cit. on p. 23).

[69] J. Leonhard, M. Yasin, S. Turk, M. Nabeel, M.-M. Louërat, R. Chotin-Avot, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos, "MixLock: securing mixed-signal circuits via logic locking," in *Proc. Design, Automation & Test in Europe Conference*, 2019 (cit. on pp. 25, 91).

[70] N. Limaye, E. Kalligeros, N. Karousos, I. G. Karybali, and O. Sinanoglu, "Thwarting all logic locking attacks: dishonest oracle with truly random logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2020 (cit. on pp. 25, 31, 32).

[71] Maxim Integrated, *MAX36051: DeepCover Security Manager with 128 Bytes of Nonimprinting Memory* (cit. on p. 26).

[72] Y. M. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security," in *Proc. USENIX Security Symposium*, 2007 (cit. on p. 26).

[73] C. Herder, M.-D. Yu, F. Koushanfar, and S. Devadas, "Physical unclonable functions and applications: a tutorial," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1126–1141, 2014 (cit. on p. 26).

[74] X. Zhang and M. Tehranipoor, "Design of on-chip lightweight sensors for effective detection of recycled ICs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 5, pp. 1016–1029, 2014 (cit. on p. 28).

[75] J. A. Roy, F. Koushanfar, and I. L. Markov, "Ending piracy of integrated circuits," *IEEE Computer*, vol. 43, no. 10, pp. 30–38, 2010 (cit. on pp. 28, 29, 32, 35).

[76] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri, "Fault analysis-based logic encryption," *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 410–424, 2015 (cit. on pp. 29, 32).

[77] M. Yasin, J. Rajendran, O. Sinanoglu, and R. Karri, "On improving the security of logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 9, pp. 1411–1424, 2016 (cit. on pp. 29, 32).

[78] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *Proc. IEEE International Symposium on Hardware Oriented Security and Trust*, 2015 (cit. on p. 30).

[79] M. Yasin, B. Mazumdar, J. Rajendran, and O. Sinanoglu, "SARLock: SAT attack resistant logic locking," in *IEEE International Symposium on Hardware Oriented Security and Trust*, 2016 (cit. on p. 30).

[80] Y. Xie and A. Srivastava, "Mitigating SAT attack on logic locking," in *Proc. International Conference on Cryptographic Hardware and Embedded Systems*, 2016, pp. 127–146 (cit. on p. 30).

[81] M. Yasin, B. Mazumdar, O. Sinanoglu, and J. Rajendran, "Removal attacks on logic locking and camouflaging techniques," *IEEE Transactions on Emerging Topics in Computing*, 2017 (cit. on p. 30).

[82] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "AppSAT: approximately deobfuscating integrated circuits," in *Proc. IEEE International Symposium on Hardware Oriented Security and Trust*, 2017, pp. 95–100 (cit. on pp. 30, 36).

[83] Y. Shen and H. Zhou, *Double DIP: re-evaluating security of logic encryption algorithms*, Cryptology ePrint Archive, Report 2017/290, 2017 (cit. on pp. 30, 36).

[84] L. Li and A. Orailoglu, "Piercing logic locking keys through redundancy identification," in *2019 Design, Automation Test in Europe Conference Exhibition*, 2019, pp. 540–545 (cit. on p. 32).

[85] P. Chakraborty, J. Cruz, and S. Bhunia, "SAIL: machine learning guided structural analysis attack on hardware obfuscation," in *2018 Asian Hardware Oriented Security and Trust Symposium*, 2018, pp. 56–61 (cit. on p. 32).

[86] D. Haghighitalab, D. Belfort, A. Kilic, A. Benlarbi-Delai, and H. Aboushady, "A 2.4 GHz ISM-band highly digitized receiver based on a variable gain LNA and a subsampled Sigma-Delta ADC," *Analog Integrated Circuits and Signal Processing*, vol. 95, no. 2, pp. 259–270, 2018 (cit. on p. 33).

[87] A. Sayed, T. Badran, M.-M. Louërat, and H. Aboushady, "1.5-to-3.0 GHz tunable RF ΣΔ ADC with a fixed set of coefficients and a programmable loop delay," *IEEE Transactions on Circuits and Systems - II: Express Briefs*, vol. 67, no. 9, pp. 1559–1563, 2020 (cit. on pp. 38, 72).

[88] J. Leonhard, M.-M. Louërat, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos, "Mixed-signal hardware security using MixLock: demonstration in an audio application," in *International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design*, 2019 (cit. on pp. 43, 92).

[89] J. Leonhard, A. Sayed, M.-M. Louërat, H. Aboushady, and H.-G. Stratigopoulos, "Analog and mixed-signal IC security via sizing camouflaging," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020 (cit. on pp. 49, 92).

[90] R. P. Cocchi, J. P. Baukus, L. W. Chow, and B. J. Wang, "Circuit camouflage integration for hardware IP protection," in *Proc. IEEE/ACM Design Automation Conference*, 2014 (cit. on p. 49).

[91] S. Chen, J. Chen, D. Forte, J. Di, M. Tehranipoor, and L. Wang, "Chip-level anti-reverse engineering using transformable interconnects," in *Proc. IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, 2015, pp. 109–114 (cit. on p. 49).

[92] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security analysis of integrated circuit camouflaging," in *Proc. ACM Conference on Computer and Communications Security*, 2013, pp. 709–720 (cit. on p. 49).

[93] S. Patnaik, M. Ashraf, J. Knechtel, and O. Sinanoglu, "Obfuscating the interconnects: low-cost and resilient full-chip layout camouflaging," in *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2017, pp. 41–48 (cit. on p. 50).

[94] Y. Tsividis, *Mixed Analog-Digital VLSI Devices and Technology*. World Scientific, 2002 (cit. on p. 51).

[95] M. E. Massad, S. Garg, and M. Tripunitara, "Integrated circuit (IC) decamouflaging: reverse engineering camouflaged ICs within minutes," in *Proc. Network and Distributed System Security Symposium*, 2015 (cit. on p. 62).

[96] C. Yu, X. Zhang, D. Liu, M. Ciesielski, and D. Holcomb, "Incremental SAT-based reverse engineering of camouflaged logic circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 10, pp. 1647–1659, 2017 (cit. on p. 62).

[97] N. G. Jayasankaran, A. S. Borbon, A. Abuellil, E. Sánchez-Sinencio, J. Hu, and J. Rajendran, "Breaking analog locking techniques via satisfiability modulo theories," in *Proc. IEEE International Test Conference*, 2019 (cit. on pp. 63, 77, 78, 83).

[98] D. M. Binkley, C. E. Hopper, S. D. Tucker, B. C. Moss, J. M. Rochelle, and D. P. Foty, "A CAD methodology for optimizing transistor current and sizing in analog CMOS design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 2, pp. 225–237, 2003 (cit. on pp. 64, 77).

[99] W. Daems, G. Gielen, and W. Sansen, "Simulation-based generation of posynomial performance models for the sizing of analog integrated circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 5, pp. 517–534, 2003 (cit. on pp. 64, 77).

[100] B. Liu, Y. Wang, Z. Yu, L. Liu, M. Li, Z. Wang, J. Lu, and F. V. Fernández, "Analog circuit optimization system based on hybrid evolutionary algorithms," *Integration*, vol. 42, no. 2, pp. 137–148, 2009 (cit. on pp. 64, 77).

[101] T. McConaghy, P. Palmers, P. Gao, M. Steyaert, and G. Gielen, *Variation-Aware Analog Structural Synthesis*. Springer, 2009 (cit. on pp. 64, 77).

[102] T. Y. Zhou, H. Liu, D. Zhou, and T. Tarim, "A fast analog circuit analysis algorithm for design modification and verification," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 2, pp. 308–313, 2011 (cit. on pp. 64, 77).

[103] A. Malak, Y. Li, R. Iskander, F. Durbin, F. Javid, J.-M. Guebhard, M.-M. Louërat, and A. Tissot, "Fast multidimensional optimization of analog circuits initiated by monodimensional global peano explorations," *Integration*, vol. 48, no. C, pp. 198–212, 2015 (cit. on pp. 64, 77).

[104] Y. Li, Y. Wang, Y. Li, R. Zhou, and Z. Lin, "An artificial neural network assisted optimization system for analog design space exploration," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2019 (cit. on pp. 64, 77).

[105] M. Eick and H. E. Graeb, "Mars: matching-driven analog sizing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 8, pp. 1145–1158, 2012 (cit. on pp. 64, 77).

[106] H. E. Graeb, *Analog Design Centering and Sizing*. Springer Netherlands, 2007 (cit. on pp. 64, 77).

[107] H. E. Graeb, Ed., *Analog Layout Synthesis: A Survey of Topological Approaches*. Springer, 2011 (cit. on p. 64).

[108] N. Lourenço, R. Martins, A. Canelas, R. Póvoa, and N. Horta, "AIDA: layout-aware analog circuit-level sizing with in-loop layout generation," *Integration*, vol. 55, pp. 316–329, 2016 (cit. on pp. 64, 77).

[109] H. Ou, K. Tseng, J. Liu, I. Wu, and Y. Chang, "Layout-dependent effects-aware analytical analog placement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 8, pp. 1243–1254, 2016 (cit. on p. 64).

[110] A. Vijayakumar, V. C. Patil, D. E. Holcomb, C. Paar, and S. Kundu, "Physical design obfuscation of hardware: a comprehensive investigation of device and logic-level techniques," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 64–77, 2017 (cit. on p. 65).

[111] J. Leonhard, M. Elshamy, M.-M. Louërat, and H.-G. Stratigopoulos, "Breaking analog biasing locking techniques via re-synthesis," in *26th Asia and South Pacific Design Automation Conference*, 2021 (cit. on pp. 77, 92).

[112] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002 (cit. on pp. 77, 83).

[113] J. Porte, *Outil pour la conception et l'enseignement d'électronique analogique (OCEANE)*, https://www-soc.lip6.fr/equipe-cian/logiciels/oceane/, Online (cit. on p. 84).

[114] N. P. Jouppi, C. Young, N. Patil, *et al.*, "In-datacenter performance analysis of a tensor processing unit," *ACM/IEEE International Symposium on Computer Architecture*, p. 17, 2017 (cit. on p. 93).

[115] M. Davies, N. Srinivasa, T. Lin, *et al.*, "Loihi: a neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, Jan. 2018 (cit. on p. 93).

[116] F. Akopyan, J. Sawada, A. Cassidy, *et al.*, "TrueNorth: design and tool flow of a 65 mW 1 million neuron programmable neurosynaptic chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537–1557, Oct. 2015 (cit. on p. 93).