Advances in Side-Channel Security

Habilitationsschrift

Fakultät für Elektrotechnik und Informationstechnik
Ruhr-Universität Bochum

vorgelegt von

Amir Moradi

aus Hamedan

Bochum
September 2014

Amir Moradi
Place of birth: Hamedan, Iran
Author’s contact information:
amir.moradi@rub.de
http://www.emsec.rub.de/chair/_staff/amir-moradi/

Printed in Germany.

To my wife Shakila
and to Parviz and Nahid
for their love and endless support.

Preface

The work presented here was conducted during my time as a post-doctoral researcher at the group of Embedded Security (EmSec), Faculty of of Electrical Engineering and Information Technology at Ruhr-Universität Bochum. This cumulative habilitation thesis is a summary of 18 articles published in peer-reviewed journals and conference proceedings. The original papers are not included in the published version of this thesis due to copyright reasons. A list of the papers with link to the publisher's website can be found at the end of this thesis.

There are many people that I would like to express my appreciation, without whom this thesis would not be possible. My first thanks go to my supervisor and mentor Prof. Christof Paar for his munificent assistance and continuous support. Without his guidance, advice, and kindness I would definitely not have come so far.

Next in line are all the people who I got to know during (in total) more than seven years which I spent at EmSec. First, thanks go to Irmgard Kühn, who plays a super significant role in keeping the group running, further to Horst Edelmann, who often surprises me with his technical expertise. Second, acknowledgments go to the former and present colleagues (in alphabetic order) to Georg Becker, who brought chocolate brownie and milk shake once per year when he visited EmSec, to Benedikt Driessen, who becomes crazy by the taste of rosewater in Iranian cookies, to Thomas Eisenbarth, who gave me the first opportunity to take part in a German wedding, to Marc Fyrbiak, who is our new coffee guru, to Tim Güneysu, who reminds me his multi-tasking skill by typing and talking about two different topics at the same time, to Stefan Heyse, who assisted me to buy my first car in Germany, to Gesine Hinterwälder, who reminds me riding the bike even in the rain, to Markus Kasper, who surprised me with his ability to deal with many pets, to Timo Kasper, who always has a lot to do and never enough time, to Elif Bilge Kavun, who always brought delicious Baklava every time she visited her homeland, to Gregor Leander, the heart of any symmetric block cipher starting with "P", to Ingo von Maurich, who was a master of getting papers accepted at conferences with nice venue, to Oliver Mischke, with whom I experienced my first diving in ocean, to Martin Novotný, who is a symbol of kindness and emotion, to David Oswald, who can sing even the Serbian lyrics, to Axel Poschmann, who, as a nice friend, lent me his car for a long trip, to Thomas Pöppelmann, who loves traveling, to Bastian Richter, who assisted me with photographic and camera related issues, to Pascal Sasdrich, who has become our new VHDL guru, to Falk Schellenberg, who, as a super nice person, can never say NO to any request, to Tobias Schneider, who always surprises me with this talent and diligence, to Daehyun Strobel, who reminds me his wish to have a soccer team of his own children, to Pawel Swierczynski, whose last name demotivated me to learn Polish, to Alexander Wild, who showed me the first (Alex, Alex) combination, to Marko Wolf, with whom I shared my first office at EmSec, to Tolga Yalcin, whom I remember every time that I make Turkish coffee at home, to Christian Zenger, who reminds me the Barney Stinson's way of being *awesoooome*, and to Ralf Zimmermann, who introduced me the scooter motorcycle.

Last but not least, I would like to express my appreciation to my parents Parviz and Nahid, and to my wonderful wife Shakila for their endless love and support. Without you, Shakila, I could not have done it for sure. Words are not enough to express my gratefulness.

Table of Contents

Chapter 1

Introduction

The field of cryptography, the science of writing secretly, consists of many different sub-fields, each of which deals with the design and implementation of cryptographic primitives. These building blocks, e.g., block ciphers, stream ciphers, signature schemes, hash functions, etc., are employed to satisfy certain properties such as confidentiality, authenticity, and integrity – in a word – to achieve *security.* By contrast, in the field of cryptanalysis, the goal is to mathematically examine these primitives, in order to update the level of security that they can reliably maintain.

Currently, we are surrounded by an ever-growing number of electronic systems in which the integrated security primitives are amongst the fundamental and necessary features. Although this evolution offers many benefits, the embedded security-enabled devices are in the hands and under the control of legitimate users, who can also play the role of an adversary. It opens serious risks with respect to system security and user privacy that are not limited only to the weakness of the underlying cryptographic algorithms. The implementation attacks, which have become serious threats for pervasive applications, in particular, can turn a theoretically robust system into a completely broken setup.

The implementation attacks are a kind of cryptanalysis tool that does not target a cryptographic algorithm but, instead, its implementation (the so-called a cryptographic device), in order to recover the secret material that is stored inside and contributes in its processes. These kinds of attacks are based on an assumption that is not far from reality, because the adversary has physical access to the cryptographic device. Although the history of implementation attacks applied by intelligence agencies is not completely clear, these issues were brought to the attention of academia in the mid-90s, through timing attacks [61], fault-injection attacks [24], and power analysis attacks [62].

A fault-injection attack assumes that the adversary may disturb the computation of the cryptographic device to obtain faulty results and, hence, recover a secret. The implementation attacks cover a large range of cryptanalysis tools with various assumptions and requisites, but *side-channel analysis* (SCA) attacks, as the most common category, have attracted the greatest attention. Side-channel analysis attacks refers to those implementation attacks that do not manipulate the cryptographic device or its functionality. Following the classification of [69], side- channel analysis attacks are *passive* and *non-invasive* because they do not disturb the computation of the target device, and do not require either its temporary or permanent alteration. Hence, no evidence of an attack is left behind, and these attacks pose a serious practical threat to the security of cryptographic devices.

Indeed, the first side-channel analysis attack was presented to the scientific community by [61], where execution time is introduced as a side channel. Subsequently, power analysis [62] and electromagnetic (EM) analysis [39, 108] attacks were demonstrated, and the scientific community began to investigate these cryptanalysis tools from a variety of perspectives and to develop countermeasures. Amongst the known side channels, power consumption and, consequently, power analysis attacks, have received most attention from researchers, due to their effectiveness and easiness to mount. As demonstrated by numerous side-channel analysis attacks on real-world applications, e.g., [36], securing ubiquitous systems that can be controlled in a hostile environment is essential. Since avoiding side-channel leakages is not a trivial task, many countermeasures have been developed and introduced, with the aim of defeating or hardening certain side-channel analysis attacks.

The aim of this cumulative habilitation thesis is to address the main challenges on the way to providing resistance against side-channel analysis attacks, mainly for hardware platforms, and to develop appropriate solutions. The first part deals with introducing the concept behind power analysis attacks and the corresponding countermeasures (Chapter 2). In order to reach this goal, Field-Programmable Gate Array (FPGA)-based case studies (with respect to the underlying hardware platform of this thesis) are presented.

Most of the side-channel analysis attacks make use of a hypothetical model to estimate the behavior of the cryptographic device or e.g., the power consumption it needs to perform a certain operation. The efficiency of such attacks strongly depends on the soundness of the underlying model or the corresponding assumptions, which are not usually straightforward to obtain, especially for real-world targets. Hence, using an inaccurate assumption or model can lead to a false positive result in a robustness evaluation of a cryptographic device. Chapter 3 partially deals with this issue, where the relaxation of the necessity of employing such an accurate hypothetical model is attempted. The schemes presented in Chapter 3 can be used for both evaluation and attack purposes because they mainly target specific statistical moments of side-channel leakages. Evaluation labs can make use of the presented techniques to investigate the vulnerability of a product in such a way that – regardless of a hypothetical model – the resistance of the underlying product e.g., against first-order attacks, can be assessed.

Side-channel countermeasures are built on the basis of a theory that makes certain assumptions about the leakages and side-channel characteristics of the target device. The countermeasure is designed according to the considered assumptions, and its security is proven, leading to a provably-secure countermeasure. Examining the validity of such assumptions in practice is partially the topic of Chapter 4. It is shown that many more leakage sources – in hardware platforms – exist than those assumed by the theory of side-channel countermeasures. This does not break the proven security of a countermeasure, but it simply shows that there are more opportunities for an adversary to gain side-channel leakages, thereby breaking a device equipped with a countermeasure whose security is proven.

Hence, providing resistance against side-channel analysis attacks, especially for hardware platforms, is not a trivial task. Those countermeasures which are effective have very high overheads that turn the design into a software-type platform where the performance is limited. Masking schemes, which, more than the other side-channel countermeasures, have attracted researchers, face several difficulties when realized in hardware platforms. In part, Chapter 5 deals with the realization and evaluation of a sound first-order masking of the PRESENT and AES ciphers for a hardware (FPGA) platform. In this chapter, we go one step deeper into FPGA architecture and provide dedicated solutions to FPGAs, in order to harden side-channel analysis attacks. Here, specific countermeasures for modern Xilinx FPGAs are presented that make use of fundamental building blocks of the underlying FPGAs to mitigate side-channel leakages.

This thesis concludes with a brief summary of the main results and provides an outlook for future challenges in Chapter 6. This cumulative habilitation thesis consists of 18 articles published in peer-reviewed journals and conference proceedings that are reviewed and complemented in the following chapters. The list of the original papers, including the links to the publisher's web page, can be found at the end of this thesis. Based on this list, the relevant papers of each chapter are denoted at the beginning of the corresponding sections in a circled form, e.g., $\textcircled{a1}$.

Chapter 2

Preliminaries

2.1 Basics

This chapter deals with the fundamental concepts of power analysis attacks, the corresponding definitions, and a survey of the available techniques to defeat them. Although this thesis will be as solid as possible, some preliminary knowledge might be necessary to follow the topics covered. Hence, the interested reader who finds the basics provided here not adequate, is referred to the – to date only – book dedicated to power analysis attacks: [69].

In whole of this thesis we frequently use the term *cryptographic device*, with which we refer to a – normally CMOS – circuitry that performs a certain cryptographic algorithm. A cryptographic device can be a general-purpose microprocessor, in which a set of instructions realizes a cryptographic algorithm, or it can be a dedicated piece of hardware, e.g., an FPGA or an Application-Specific Integrated Circuit (ASIC), which has been specifically designed to – usually – speed up the execution of a cryptographic algorithm. We also often use the term *device under attack* or *device under test*. From an adversary point of view, a cryptographic device, the "victim" in which a secret material is stored, is a device under attack. The goal of a side-channel adversary is to recover some information about the secret stored in the device under attack. On the other hand, when the designer of a cryptographic device adopts the role of an adversary, in order to investigate how much information can be obtained by observing side-channel information, the underlying cryptographic device is considered as a device under test, because the designer (hypothetical adversary) is aware of all its detailed information.

2.1.1 Power Consumption

Sometimes the concept of power consumption used in the side-channel analysis domain is confused with that of energy consumption in a certain period of time. In the field of side-channel analysis, the instantaneous energy consumption of a cryptographic device is measured. Because a side-channel analysis attack searches for a difference between energy consumption of the device associated with various operations, the actual amount of energy consumed by the device is not of great importance. Therefore, the current passing through the cryptographic device is usually measured, to provide an estimate of the level of its energy consumption. The measurements are performed with a digital oscilloscope, usually with sampling rates of 100 MS/s up to 10 GS/s.

Measurement Methods

Several ways of measuring such current exist. The most common method is to place a shunt resistor ($R\,\Omega$) in the GND path (see Figure 2.1) and to measure the voltage drop V over the resistor, which directly yields a factor of the current $V = I \cdot R$. In some cases, placing the shunt resistor in the Vdd path can lead to better results (see Figure 2.2). This is due to two issues:

(1) Measurements performed in the GND path also contain the current necessary for I/O activity. Hence, in devices in which Vdd of the I/O and Vdd of the core are separated, measuring the current passing through the core Vdd path excludes the current associated with the I/O.

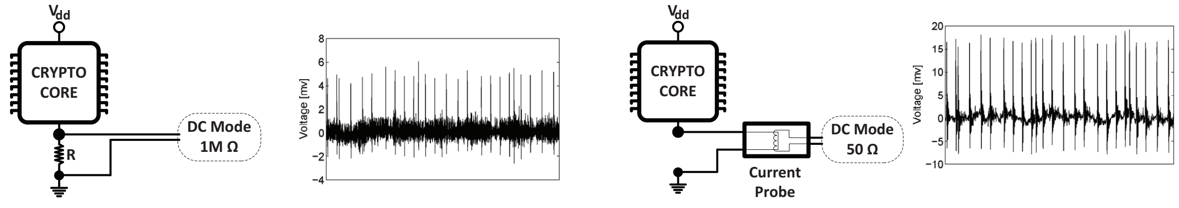
(2) According to Figure 2.2, one option for removing the DC shift of the signal is to measure the voltage over the cryptographic device in the AC mode or to employ a DC blocker (e.g., `BLK-89-S+` from Mini-Circuits[1]). In this case, the measured signal has a negative polarity compared to that measured through the GND path. We stress that, in each clock cycle, when the cryptographic device needs a large amount of current, the voltage regulators (which are supposed to maintain a certain level of voltage e.g., for the core Vdd) are not able to drive such high current and, usually, there is a small voltage drop at their output. Therefore, when measuring in the Vdd path over the cryptographic device, the voltage drops of the regulator's output, which are actually related to the device's activities, are also observable. For this reason, e.g., doubling the shunt resistor usually does not lead to a doubling of the peak-to-peak magnitude of the measured signal. For the same reason, using a very small $\approx 0.5\,\Omega$ shunt resistor usually suffices to obtain useful signals.

The Printed Circuit Board (PCB) and the measurement setup usually form a low-pass filter, due to the small parasitic capacitances that are made by the PCB layouts. Hence, measuring the power signals with a broad bandwidth simply leads to strong environmental noise. The bandwidth of the oscilloscope is hence commonly limited to a range of 20 MHz, which significantly reduces the noise. Further, using a differential probe to measure the voltage drop over the shunt resistor, when it is placed in the Vdd path, is not favorable, because these active probes also need an external power source that contributes its own noise (see Figure 2.2(c)). It is recommended to use a passive probe with 1:1 attenuation, e.g., a coaxial cable (with BNC to SMA) connector that can directly connect an oscilloscope channel to the Vdd of the cryptographic device. In case the peak-to-peak signal amplitude is very low, one or two low-noise AC amplifier(s), e.g., `ZFL-1000LN+` from Mini-Circuits, can be employed.

One more point is related to the format of the collected signals. The digital oscilloscope samples the voltage level in a quantized form by means of an 8-bit (or recently-available-in-market 12-bit) analogue-to-digital converter (ADC). Hence collecting the signals (so-called power traces) in voltage domain is not necessary, and storing the 8-bit ADC outputs satisfies the demands. This statement is true in cases that the settings of the oscilloscope do not change during the measurements. In other words, the gain and offset used to convert the ADC output to actual voltage value should not alter during collection of all necessary power traces.

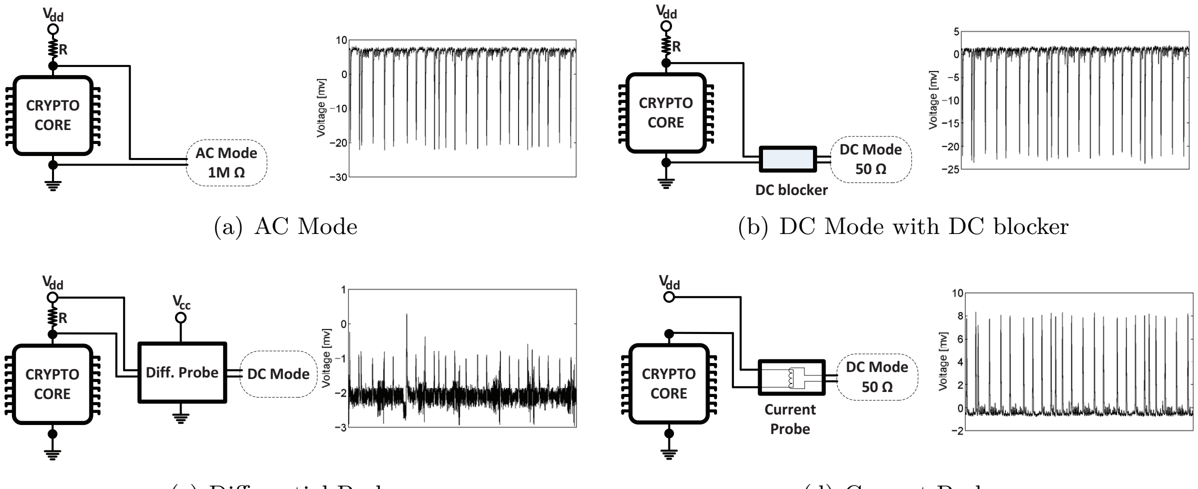
One more point is related to the format of the collected signals. The digital oscilloscope samples the voltage level in a quantized form by means of an 8-bit (or, more recently 12-bit) analog-to-digital converter (ADC). Hence, collecting the signals (so-called power traces) in the voltage domain is not necessary, and storing the 8-bit ADC outputs satisfies the demands. This statement is true for cases in which the settings of the oscilloscope do not change during the

[1] `www.minicircuits.com`



(a) DC Mode (b) Current Probe

Figure 2.1: Measurement in GND path



(a) AC Mode (b) DC Mode with DC blocker

(c) Differential Probe (d) Current Probe

Figure 2.2: Measurement in Vdd path

measurements. In other words, the gain and offset used to convert the ADC output to actual voltage value should not vary during collection of all necessary power traces.

It is also noteworthy that electromagnetic (EM) traces, which are closely proportional to the corresponding power traces, can be measured in the same way by means of an EM probe. The specification of the required EM probe varies according to the characteristics of the device under attack. Further, since the frequency of EM traces is usually higher than that of the power traces, no bandwidth limit should be set on the oscilloscope, and the sampling should be usually performed with a high sampling rate ($\geq$ 1GS/s).

Platforms

Unless stated otherwise, a hardware platform – usually an FPGA – is considered as the implementation platform of the case studies presented in this thesis. We mainly use different versions of SASEBO [3] and SAKURA [4] boards as an evaluation platform. These boards consist of two chips (usually two FPGAs), one of which plays the role of a cryptographic device. The other one is responsible for communicating with a PC e.g., via UART, as well as for transferring the requested data to and from the cryptographic device. The side-channel traces of the cryptographic device are measured from the dedicated places provided in the board for a

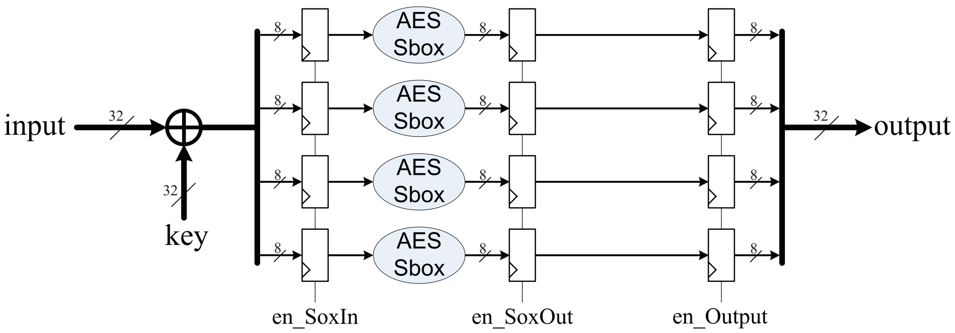


Figure 2.3: Block diagram of the first case study

shunt resistor. Because FPGA devices have different Vdd pins (for the I/O, internal core, and auxiliary), measuring the power through the Vdd path (Vdd_INT) is favored over the GND path.

2.1.2 Case Studies

In order to introduce the concept of power analysis attacks, two case studies are presented below.

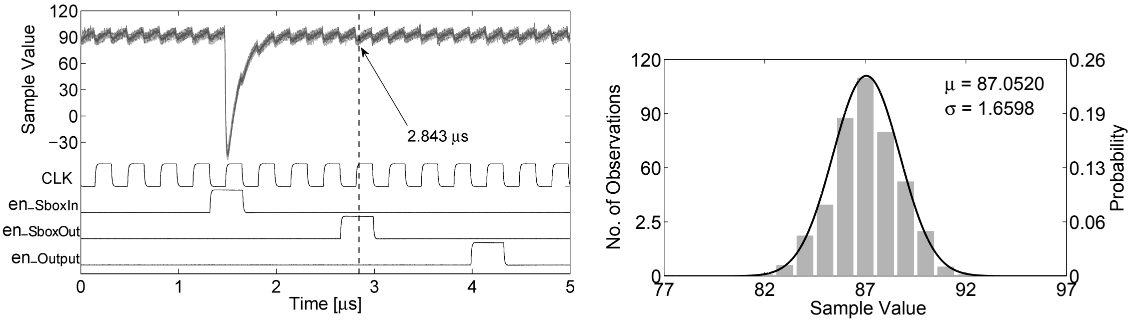
Case Study 1

For the first case study, we consider a circuit realizing a part of the AES encryption algorithm. Figure 2.3 shows its block diagram in which a 32-bit key XOR and four AES Sboxes are implemented. Each Sbox circuit is surrounded by SboxIn and SboxOut registers, whereas no logic exists between the SboxOut registers and output registers. All registers are driven by a global clock signal, and each register step can be controlled separately by its dedicated enable signal, i.e., `en_SboxIn`, `en_SboxOut`, and `en_Output`.

The circuit is implemented on the cryptographic device of SAKURA-G (Spartan-6 FPGA), and the very compact design of Canright [28] was taken to realize the AES Sbox circuits. The 32-bit input is given by the control FPGA, and the 32-bit output is sent by the cryptographic device to the control FPGA. The 32-bit key is hardwired inside the cryptographic device.

The lower part of Figure 2.4(a) depicts the timing diagram followed for the measurements. The measurements were performed with a passive

The lower part of Figure 2.4(a) depicts the timing diagram followed for the measurements. The measurements were performed with a passive coaxial probe and a LeCroy WaveRunner HRO 66Zi digital oscilloscope with a sampling rate of 1 GS/s. The bandwidth was limited to 20 MHz, and the output of the embedded amplifier of the SAKURA-G board was measured as the power signal. The employed oscilloscope is facilitated by a 12-bit ADC, but we have used only its 8-bit output. For each measurement two 32-bit input values are sent from the control FPGA. The first one is given to the cryptographic device and the timing diagram given in Figure 2.4(a) is followed. The associated power trace to these two 32-bit inputs is measured while the second input is given to and processed by the cryptographic device (following the same timing diagram).



(a) Superimposition of 500 traces and the corresponding timing diagram

(b) Histogram of the sample points at 2.843 μs

Figure 2.4: Result of the first case study

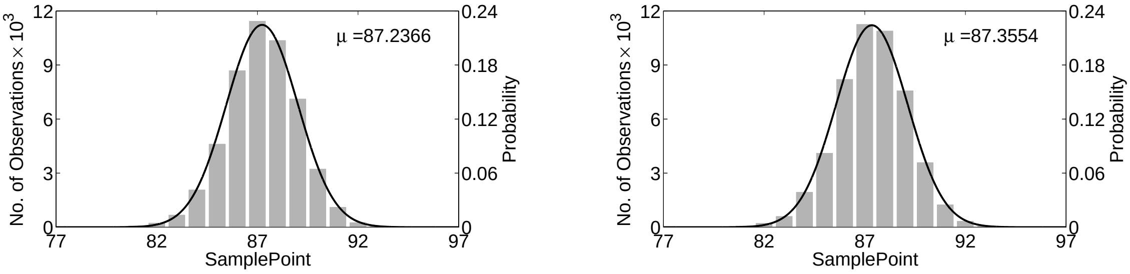
During the measurements we collected different sets of power traces. In the first set we collected 500 traces for a fixed pair of 32-bit inputs; a superimposition of 500 traces is shown in the upper section of Figure 2.4(a).

Although all inputs and the processes associated with these measurements are identical, the traces are slightly different. This phenomenon, extensively explained in [69], is due to environmental noise (so-called *electrical noise* [69]). Observing the probability distribution of a sample point (e.g., by a histogram shown in Figure 2.4(b)) reveals that the electrical noise follows a Gaussian distribution; thus, it can be easily modeled [69]. The variance of this distribution might be different for each sample point, due to the noise induced by internal signal activities, e.g., the clock.

In the second set of the measurements, we collected 100 000 traces while the 32-bit inputs were randomly selected for each trace. We targeted the SboxOut registers, i.e., the sample points in the power traces around 2.843 μs (when the SboxOut registers store the output of the Sboxes). In general, a CMOS circuit consumes energy when the content of a signal changes; this requires charging and discharging tiny internal parasitic capacitances. In hardware platforms, these changes are due to the flips of the register bits. Therefore, we categorize the collected traces, based on a change in one of the bits of the SboxOut register. Since the 32-bit key, as well as the consecutive 32-bit inputs are known to us, we can easily compute whether, for each trace, the selected bit of the SboxOut register flipped or not. When we observe the distributions at sample point 2.843 μs of the categorized power traces (see Figure 2.5), we conclude that each histogram again fits to a Gaussian distribution, but the mean μ of these two distributions are slightly different.

A question here is how much these two means μ are different and how easy it is to distinguish between them. Such a question exists in many scientific fields, and a straightforward solution, i.e., Student's t-test, is widely used. One of the mostly used applications is in research performed on patients to examine the effectiveness of a certain medicine, e.g., to reduce the size of tumors.

The aim of a t-test is to provide a quantitative value as a probability that the mean μ of two sets of data are different. In other words, a t-test makes it possible to examine the validity of the *null hypothesis* as the samples in both sets were drawn from the same population, i.e., the two sets are indistinguishable.



(a) The Sbox output bit unchanged

(b) The Sbox output bit flipped

Figure 2.5: Histograms of sample points at $2.843\,\mu$s categorized based on an Sbox output bit flip, $\Delta\mu = 0.1188$ (for the first case study)

Hence let S_0 and S_1 indicate two sets that are under the test. Let also μ_0 (or μ_1) and δ_0^2 (or δ_1^2) stand for sample mean and sample variance of the set S_0 (or S_1) respectively, and n_0 and n_1 for the cardinality of each set. The t-test value and the degree of freedom v are computed as

$$\mathrm{t} = \frac{\mu_0 - \mu_1}{\sqrt{\frac{\delta_0^2}{n_0} + \frac{\delta_1^2}{n_1}}}, \qquad v = \frac{\left(\frac{\delta_0^2}{n_0} + \frac{\delta_1^2}{n_1}\right)^2}{\frac{\left(\frac{\delta_0^2}{n_0}\right)^2}{n_0-1} + \frac{\left(\frac{\delta_1^2}{n_1}\right)^2}{n_1-1}}.$$

In the final step, we estimate the probability for accepting the null hypothesis with Student's t cumulative distribution function. In other words, based on the degree of freedom v the Student's t distribution function is drawn

$$f(t) = \frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\,\Gamma(\frac{\nu}{2})} \left(1 + \frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}},$$

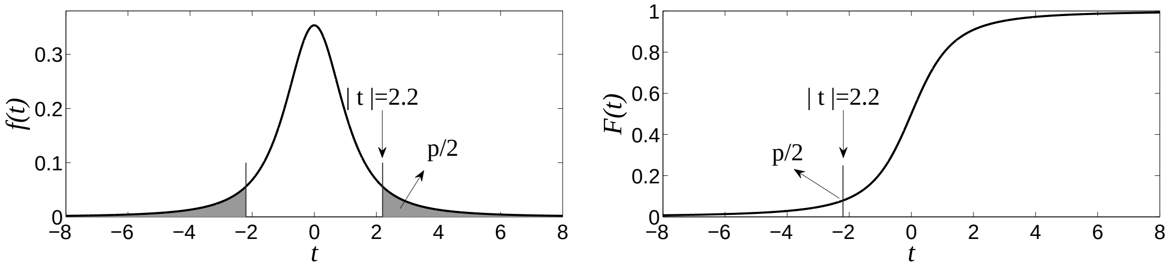
where Γ is the gamma function. Based on a two-tailed Welch's t-test, the desired probability is calculated as $p = 2 \times \int_{|\mathrm{t}|}^{\infty} f(t)$ (see Figure 2.6). As an alternative, we can make use of the corresponding cumulative distribution function

$$F(t) = \frac{1}{2} + t\Gamma\left(\frac{\nu+1}{2}\right) \times \frac{{}_2F_1\left(\frac{1}{2}, \frac{\nu+1}{2}; \frac{3}{2}; -\frac{x^2}{\nu}\right)}{\sqrt{\pi\nu}\,\Gamma\left(\frac{\nu}{2}\right)},$$

where ${}_2F_1$ is the hypergeometric function. Hence the result of the t-test can be estimated as $p = 2 \times F(-|\mathrm{t}|)$ (see Figure 2.6). Note that such a function is available amongst the MATLAB embedded functions as `tcdf(·,·)`.

Hence, small p values (alternatively large t values) provide evidence for rejecting the null hypothesis and concluding that the sets were drawn from different populations. For the sake of simplicity, usually a threshold for $|\mathrm{t}|$ as > 4.5 is defined to reject the null hypothesis without considering the degree of freedom and the cumulative distribution function. Following this concept we calculated the t-test values of the traces – at each sample point independently – based on an Sbox output bit flip [2]. The result shown in Figure 2.7 indicates that, when using all

[2] It is noteworthy that the t-test is currently being used in evaluation labs to examine the side-channel vulnerability of products [48].



(a) Probability density function

(b) Cumulative distribution function

Figure 2.6: Student's t distribution functions and two-tailed Welch's t-test (examples for $v = 2$)

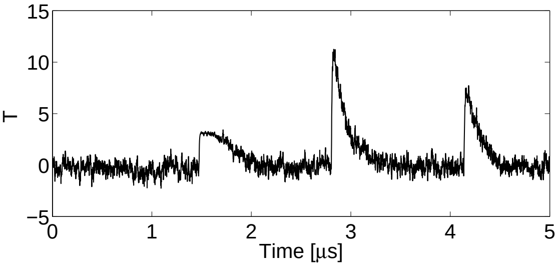


Figure 2.7: t-test based on an Sbox output bit flip (for the first case study)

100 000 traces, the t-test value is higher than the threshold when the Sbox output is stored in its corresponding register. This experiment shows the – albeit small – contribution of a single-bit flip of a register in power consumption associated with a 32-bit register.

Based on this concept, i.e., the dependency of the instantaneous power consumption of a cryptographic device on its internal process/change, several power analysis attacks have been introduced in the literature. Differential power analysis (DPA) [62] follows the concept explained above that, based on a key guess, the power traces are categorized into two sets and – based on a Gaussian assumption – the difference between the means is observed. The largest difference should indicate the correct key guess if the device under attack has such power consumption characteristics, and a sufficient number of traces was used in the attack. The latter is due to the fact that we need to estimate the means μ by averaging, and a small number of traces (with respect to the amount of noise) prevents the estimation of the means with sufficient accuracy.

Correlation power analysis (CPA) [25] requires a hypothetical model to roughly estimate the amount of power consumption for a certain operation of the device under attack. Hence the result of the hypothetical model for a key guess is correlated to the power traces independently at each sample time. The highest correlation coefficient is expected to recover the correct key guess again if the number of traces suffices and the hypothetical model is relatively proportional to the actual characteristics of the device under attack. For more information the interested reader is referred to [69].

Mutual information analysis (MIA) [45] overcomes two drawbacks of CPA: (1) the necessity for a linear relationship between the hypothetical model and actual power consumption of the cryptographic device, and (2) the Gaussian assumption which supposes that the dependency

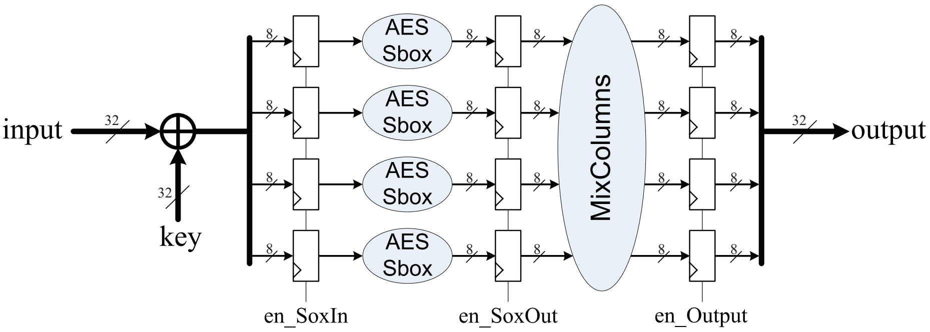


Figure 2.8: Block diagram of the second case study

of the power consumption on the processed data is represented by the difference between the mean μ of the distributions. Based on its concept, at each sample point independently a MIA estimates the mutual information between the measured power traces and an abstraction of the internal process/change based on a key guess. Hence there are many options for the adversary (or for the evaluator) for estimating the probability distributions required to estimate mutual information (see [12]).

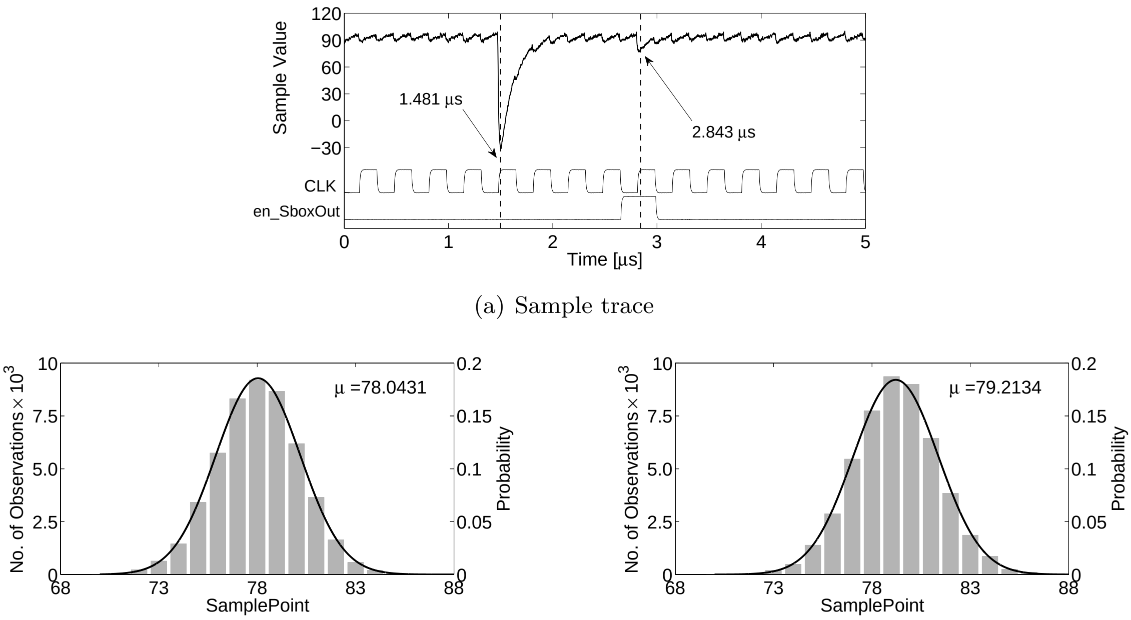
Case Study 2

The main reason for energy consumption in a CMOS circuit are glitches, that occur in a combinatorial circuit. A glitch refers to an unintentional transient change of a signal. For example, consider an XOR gate whose inputs change from “01” to “10”. Since the arrival time of the inputs to the gate are not exactly the same (even if both are driven from the same register), the output of the gate probably changes, for a short time, to “0” and then back to “1”. Now, consider a large combinatorial circuit, e.g., an AES Sbox, whose 8-bit input at a certain point in time changes from x to y. Several glitches will occur at the output of the gates, which form the underlying combinatorial circuit, until the output is stable and can be saved, e.g., in a register. The interesting issue is that, during this period of time, the glitches are propagated in such a way that the gates close to the input signal experience fewer glitches than those close to the output signal. The output signal may toggle several times (in a range of 50) until it becomes stable.

As stated, a change of an internal signal consumes energy. Hence, the more glitches that occur in a circuit, the more energy it consumes. In order to observe such an effect, we change the first case study slightly and add a MixColumns circuit between the SboxOut and the output registers (see Figure 2.8).

Repeating the last experiment, i.e., measuring 100 000 traces and categorizing them based on a bit flip in the SboxOut register, led to the two histograms shown in Figure 2.9 (where a sample trace is also presented). Compared to the last experiment, the power consumption at 2.843 μs (when the SboxOut register stores the Sbox output) is higher, which is clearly due to the glitches occurring in the MixColumns circuitry. Additionally, the difference between the mean μ of the two Gaussian distributions is more distinct than previously.

As an another observation we can examine the distributions at sample point 1.481 μs when the traces are categorized based on a bit flip in the SboxIn register, i.e., Figure 2.10. It is of

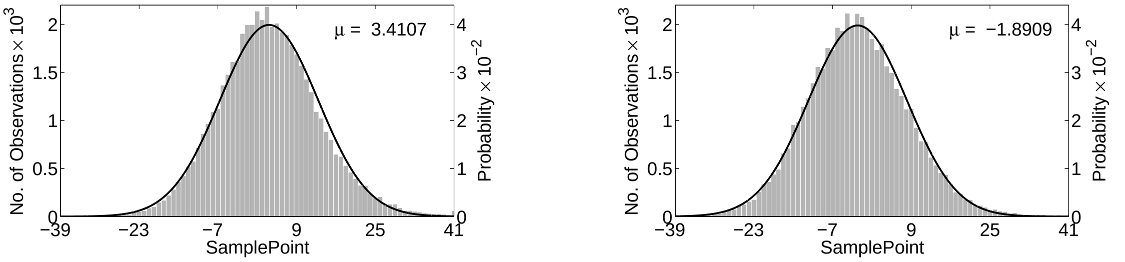


(a) Sample trace

(b) The Sbox output bit unchanged

(c) The Sbox output bit flipped

Figure 2.9: Histograms of sample points at $2.843\,\mu$s categorized based on an Sbox output bit flip, $\Delta\mu = 1.1703$ (for the second case study)



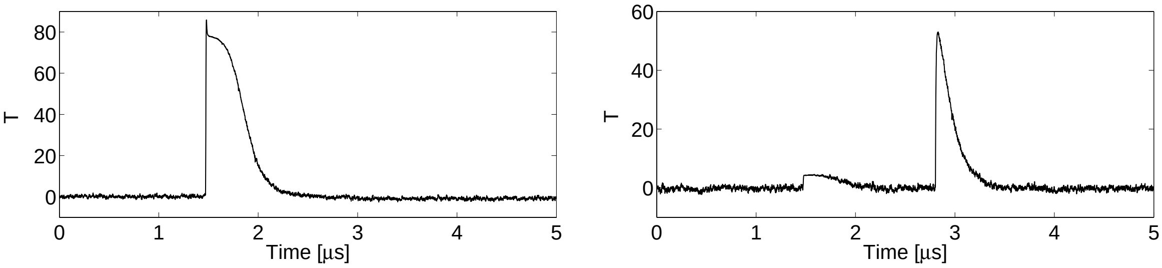
(a) The Sbox input bit unchanged

(b) The Sbox input bit flipped

Figure 2.10: Histograms of sample points at $1.481\,\mu$s categorized based on an Sbox input bit flip, $\Delta\mu = 5.3016$ (for the second case study)

interest that the mean μ of the distributions are more distinguishable, compared to those based on the SboxOut register. The same principle is observed when examining the corresponding t-test curves shown in Figure 2.11. This is due to the size of the underlying combinatorial circuit. Sine the Sbox circuit is much more complex than the MixColumns, more glitches occur inside the Sbox circuit. Hence, its energy consumption is higher, which can lead to a stronger dependency of power traces on processes/changed data.

In this case study, we aimed at explaining the reason behind the common selection of the Hamming distance (HD) model as the hypothetical power model in a CPA attack on hardware-based cryptographic devices, e.g., [36]. As explained, a bit flip in a register does not significantly



(a) Based on an Sbox input bit flip

(b) Based on an Sbox output bit flip

Figure 2.11: t-tests (for the second case study)

change the power consumption of the device unless the register drives a combinatorial circuit in which several toggles follow this bit flip.

2.2 Countermeasures

As explained, the power consumption of a cryptographic device depends on the data it processes; this is referred to as the data dependency of power traces. There are other kinds of dependency, e.g., operation dependency, which is related to the fact that each operation performed by a cryptographic device needs a specific amount of energy. This dependency differs from device to device and from platform to platform; similar to data dependency, it also changes at different sample points.

The goal of side-channel countermeasures is to avoid or mitigate such dependencies. Based on the taxonomy presented in [69], power analysis countermeasures can be divided into two main categories: *hiding* and *masking*. The aim of a hiding countermeasure is to

- cover the data or operation dependency by either randomizing the operations or adding a considerable amount of noise, or
- equalize the amount of power consumption for each operation and/or for each processed data.

As shown in previous case studies, the power traces should be aligned in such a way that the same operation is performed at a certain sample point of all power traces. Hence, shuffling, i.e., randomizing the order of operations (program flow), is amongst the well-known hiding countermeasures [55, 69] that can harden a power analysis attack. As already stated, noise addition [51, 69] techniques can impose an extra workload for a side-channel adversary, but this can be overcome by employing more traces, because the added noise usually follows a Gaussian distribution. Other hiding countermeasures try to solve the problem from scratch, i.e., by avoiding the data dependency. These countermeasures at the cell level, called DPA-resistant logic styles, aim at equalizing the power consumption of a cryptographic device regardless of any input, intermediate, or output value. For some examples, we refer to [31, 120, 121] (for more information, see Chapter 7.3 of [69]).

2.2.1 Masking

Amongst the most studied countermeasures, masking [30, 33, 69, 105] has attracted the most research interest. By randomizing the secret internals, it aims at cutting the relationship between the side-channel leakages and predictable processed data. The concept of masking is the same as *secret sharing*, in which secret data is split into several shares randomly, with the property that a way of constructively recombining the shares, to recover the secret, exists. A classical sharing is the first-order Boolean additive masking, where a sensitive variable x is split in two shares s_0 and s_1 in such a way that $x = s_0 \oplus s_1$. In a corresponding scheme one of the shares, e.g., s_0 is drawn *uniformly* randomly and is called the *mask*. Hence, the other share s_1 is computed as $x \oplus s_0$ and is called *masked data*.

The idea behind masking is to – with respect to the underlying cryptographic algorithm – perform the processes on the shares and then to combine them at the end of the algorithm. Therefore, a side-channel adversary should not be able to predict the processed data as long as the mask follows a uniform distribution and it cannot be predicted by the adversary. It is well known that any linear operation $l(\cdot)$ is easy to implement in this fashion. It is indeed sufficient to compute $l(\cdot)$ on each share individually as, $l(x) = l(s_0) \oplus l(s_1)$. However, this does not apply to non-linear operations, such as the computation of an Sbox, i.e, $S(s_0) \oplus S(s_1) \neq S(x)$. Most of the research effort in the field of masking has thus been devoted to this topic.

Historically, the masking strategy was initially named *Sbox precomputation*. For each unique Sbox table in the design, e.g., one in case of AES, and eight in case of DES, two random variables s_0 and s'_0 are drawn in order to mask the Sbox input and output respectively. Hence, a so-called masked Sbox $S'(\cdot)$ is computed [72] in such a way that $S'(x \oplus s_0) = S(x) \oplus s'_0$. Now, such a table can be used to securely traverse the Sbox of the first share $s_1 = x \oplus s_0$ as $s'_1 = S'(s_1)$. Indeed, (s_0, s_1), as a sharing of x, is transformed to (s'_0, s'_1), as a sharing of $S(x)$. In this setting, neither x nor $S(x)$ appears unmasked; hence, the data dependency should not longer exist.

Problems

The Sbox precomputation has a significant drawback: "efficiency". Ideally, each Sbox call in a shared form needs a prior precomputation. Hence, several techniques have been considered to deal with such high overhead. The options are: sharing a masked Sbox $S'(\cdot)$ for a couple of Sbox calls, e.g., for all Sboxes in the entire encryption, or for all (or some) Sboxes in a cipher round. Each of these techniques brings its own risk and vulnerability. As an important issue, careful attention should be paid to dealing with different x and y values that have been shared with the same mask s_0. For instance, if two masked data s_1^x and s_1^y are consecutively written in a register, following the Hamming distance (HD) model, $\mathrm{HD}(s_1^x, s_1^y) = \mathrm{HW}(x \oplus s_0 \oplus y \oplus s_0) = \mathrm{HW}(x \oplus y)$ is easily detectable by an adversary. Such a case is potentially probable during the ShiftRows operation of AES. As another example, an attack has been introduced in [32] to overcome the implemented masking if a masked Sbox is shared.

One further issue relates to the high number of masks required for the Sbox precomputations. It is relatively common and **incorrect** to take the same value for s_0 and s'_0 for the Sbox precomputation, which significantly eases the further computations (for example see [100] and [69] § 9.2.1). In such a case, the drawback becomes evident the masked Sbox $S'(\cdot)$ is being used. If the Sbox input is replaced by its corresponding output (which is a common method in software implementations), the HD model will again give the adversary a strong chance of succeeding,

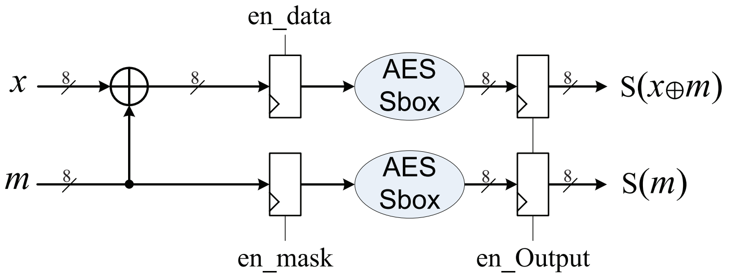


Figure 2.12: Block diagram of the third and fourth case studies

because $\mathrm{HD}(s_1, S'(s_1)) = \mathrm{HW}(x \oplus s_0 \oplus S(x) \oplus s_0) = \mathrm{HW}(x \oplus S(x))$. Here, we refer to [69] where, in Chapter 9.2.1, it is stated: “Note that setting $s'_0 = s_0$ does not make attacks easier in general”. The same statement is also presented in [100]: “for simplicity, s'_0 is often chosen to be equal to s_0”.

As a side note, we refer to [101] and [122], in which a few techniques targeting Sbox precomputations have been introduced. They basically aim at recovering the masks s_0 and s'_0 during the precomputation of $S'(\cdot)$, and making use of the revealed masks to predict the intermediate values of the masked algorithm. Further, some cryptographic functions are not Boolean, e.g., RSA which is based on modular arithmetic and hence is preferably masked in some ring $\mathbb{Z}_N$. There are also situations where the mask cannot be injected additively, but rather multiplicatively [47], or via a homographic function [34].

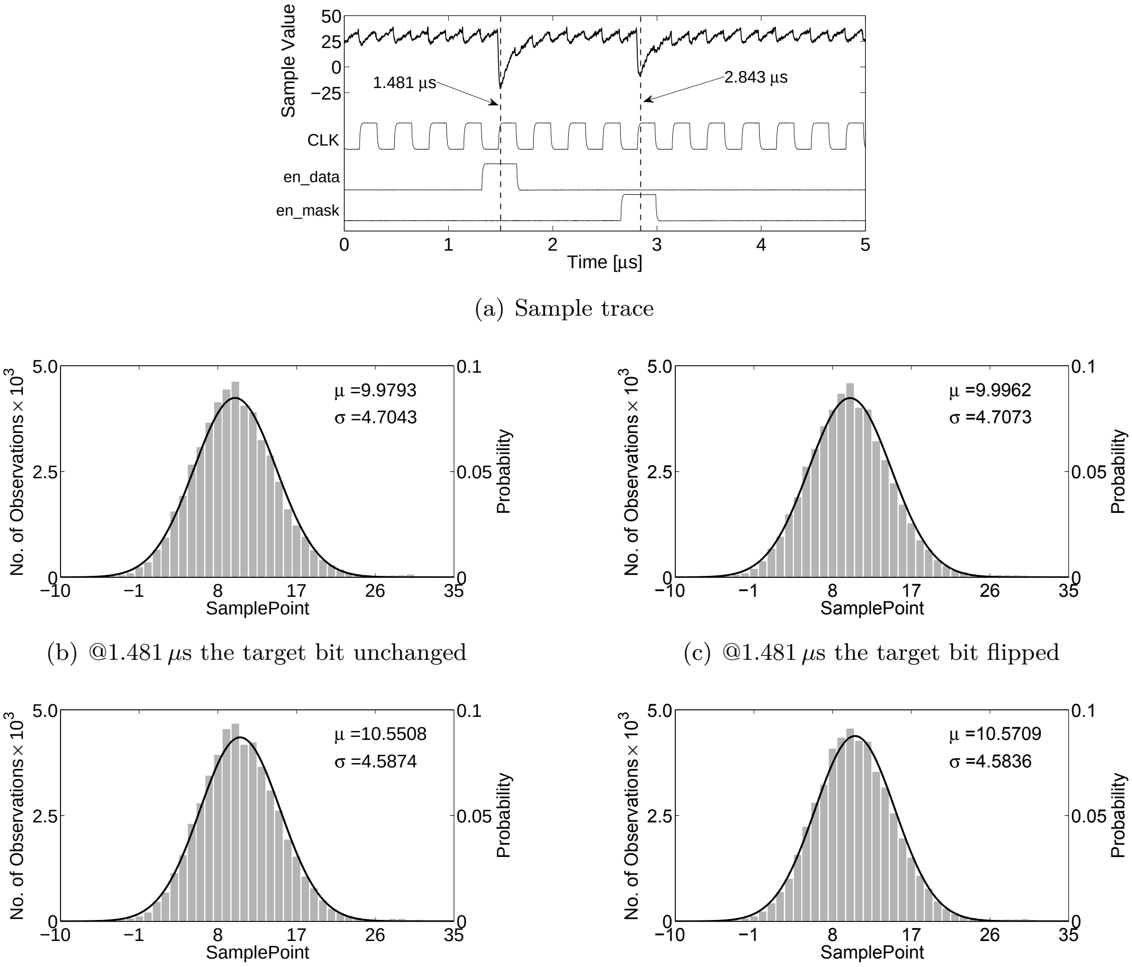
2.2.2 Case Studies

Regardless of the aforementioned issues here we provide two case studies to present how a masking scheme can prevent the leakages and how a corresponding implementation can be attacked. Before moving towards the details of the case studies, we should mention that the presented case studies do not realize (even partially) a masked implementation, and they are only presented to show how the protection and the attacks work. Further, we use the same platform (SAKURA-G) that was employed in the former case studies.

Case Study 3

For both case studies presented here, we use a design whose block diagram is shown in Figure 2.12. Two 8-bit values x (as data) and m (as mask) are given to the cryptographic device, which are instantly XORed to make masked data $x \oplus m$. The mask and masked data are stored in the corresponding registers whose enable signals can be controlled independently, i.e., `en_data` and `en_mask`. Each of these two registers drives an AES Sbox circuit (Canright design [28]), and their outputs are saved by two other dedicated registers that provide the 16-bit output of the design.

For the measurements, which have been performed similarly to the previous case studies, we selected both data x and mask m randomly (from a uniform distribution). We collected 100 000 traces, one of which is shown in Figure 2.13(a), where the timing diagram followed during the measurements is also shown. In this case study, the data register and the mask register are not active at the same time. Indeed, we emulate a sequential process on shares, which is a common



(a) Sample trace

(b) @1.481 μs the target bit unchanged

(c) @1.481 μs the target bit flipped

(d) @2.843 μs the target bit unchanged

(e) @2.843 μs the target bit flipped

Figure 2.13: Histograms of two sample points at 1.481 μs and 2.843 μs categorized based on a bit flip in consecutive data value x (for the third case study)
(@1.481 μs: $\Delta\mu = 0.0169$ and $\Delta\delta = 0.0030$)
(@2.843 μs: $\Delta\mu = 0.0201$ and $\Delta\delta = 0.0038$)

case in software platforms. Once more, we stress that employing two Sboxes on $x \oplus m$ and m does not produce any form of $S(x)$ or $S(x) \oplus m$. These two circuits have been considered in the design to increase the leakage (as explained, due to the glitches).

As before, we categorize the traces based on a register bit flip, but here the mask m is supposed to be unknown to the adversary. Hence, we consider a bit flip in consecutive data x values to categorize the traces. As shown in Figure 2.13(a) – with respect to the time instances when the data register and the mask register are active – two sample points are taken into account to show the probability distributions (histograms). Figure 2.13 indicates that the distributions are very similar to each other. In other words, based on the means μ and standard deviations δ,

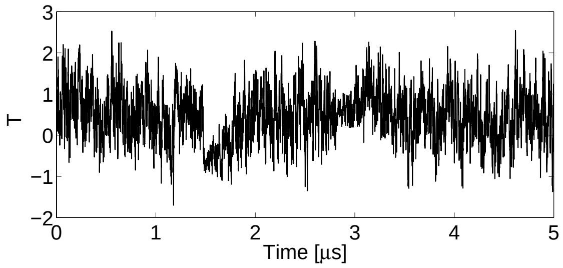


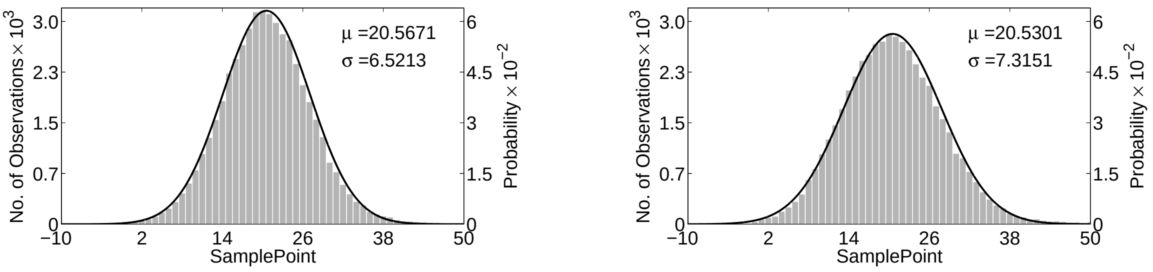
Figure 2.14: t-test based on a bit flip in consecutive data value x (for the third case study)

processed data cannot be categorized, as expected. This is also confirmed by the corresponding t-test curve (shown in Figure 2.14) that confidently accepts the null hypothesis.

Following the concept of secret sharing, secret data x can be recovered when having both shares $x \oplus m$ and m. In our case, information exists about $x \oplus m$ at sample points around $1.481\,\mu$s and about m around $2.843\,\mu$s. Therefore, combining this information (side-channel leakages) should give us some information about x. The combination of side-channel leakages can be performed in many different ways, but – as stated in [115] – the best choice is to make the leakages mean free and then multiply them together. In other words, for each trace, the power values at $1.481\,\mu$s and $2.843\,\mu$s are subtracted from their corresponding means μ and then multiplied to give a singular value associated with that power trace. This combination is also called *centered product*.

As two choices for the combination function, we consider addition and centered product, whose corresponding results are shown in Figure 2.15 and Figure 2.16. As demonstrated by the histograms, the mean μ of the two distributions after the addition are not distinguishable (as the related t statistic is also very small). However, the variances (or standard deviation δ) of these two distributions are not as closed as that of those shown in Figure 2.13. On the contrary, the means μ of the histograms after the centered multiplication (which do not fit to Gaussian anymore) are clearly discernible as the t statistic is around 40.

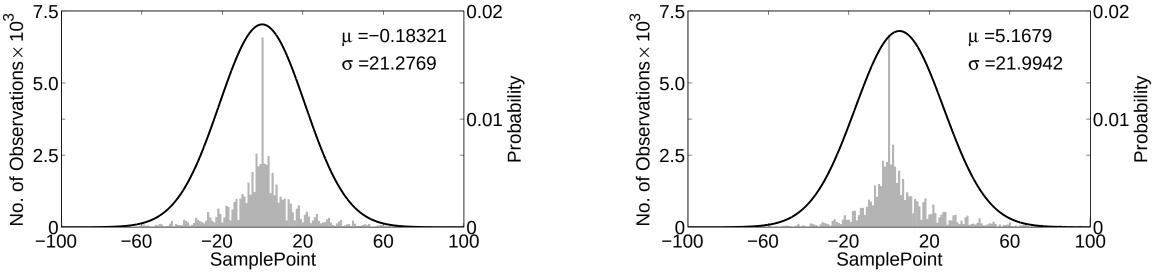
These examples are in the direction of the second-order attacks. Before giving the definition of a higher-order attack, we define the order of a masking scheme. A masking scheme that splits a secret into $n+1$ shares is of order n. The example given above, where x is split to $x \oplus m$ and m, is a first-order masking. In order to break a masking scheme of order n, one usually needs to perform an attack of order $n+1$ that combines the leakages associated with the $n+1$ shares. This uncertainty is due to the underlying secret sharing scheme and the implementation platform. For instance, if a secret sharing scheme splits a secret into $n+1$ shares, but having $m < n+1$ shares gives information about the secret, the corresponding masking scheme can be broken by an attack of order m. Further, consider an implementation of a masking scheme of order n that provides leakages related to a product (of any form) of at least two shares. In order to attack such an implementation, one does not necessarily need to combine $n+1$ leakages, each of which is related to each share, but can make use of the already-combined leakages provided by the implementation. Hence, such an implementation does not necessarily need to be attacked by an $(n+1)$-order attack. As an example, we refer to [70], [74], and [98], which are discussed in detail in Section 3.1 and Section 5.1.



(a) The target bit unchanged

(b) The target bit flipped

Figure 2.15: Histograms of addition of sample points at 1.481 μs and 2.843 μs categorized based on a bit flip in consecutive data value x (for the third case study) $\Delta\mu = 0.0370$ and $\Delta\delta = 0.7938$, t-test=0.8434



(a) The target bit unchanged

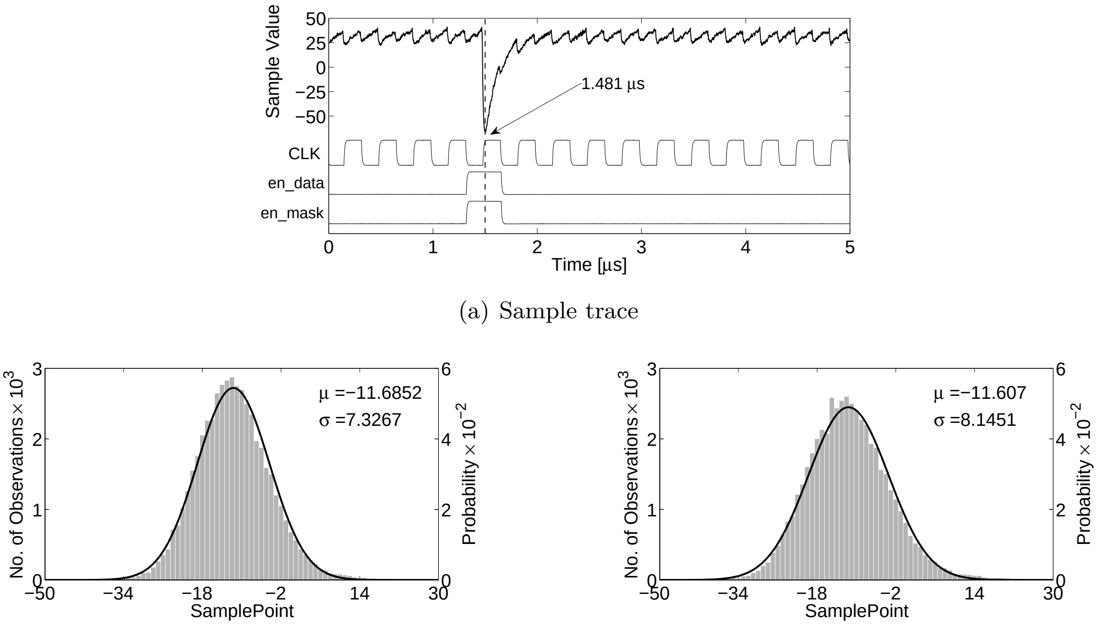
(b) The target bit flipped

Figure 2.16: Histograms of centered product of sample points at 1.481 μs and 2.843 μs categorized based on a bit flip in consecutive data value x (for the third case study) $\Delta\mu = 5.3511$ and $\Delta\delta = 0.7173$, t-test=39.1043

Case Study 4

For last experiment, we consider the previous (third) case study with a small difference in control signals. Now, both the mask and data registers are enabled at the same time, as shown in Figure 2.17(a). In fact, the leakage observed at sample point around 1.481 μs is the sum of those associated with the bit flips in the data register as well as those related to the mask register. Indeed, the leakages of the two shares are added by the implementation platform.

Performing the same scenario as before and categorizing the traces based on a bit flip in consecutive data values x yields the histograms and distributions shown in Figure 2.17. Similar to the results shown in Figure 2.15 (combining the leakages by the addition function), the mean μ of the two sets are not distinguishable, but the variances are. Here, since the leakages are already added together, a second-order investigation only needs to square (actually centered square) the sample points to observe a difference in the means μ. This phenomenon is illustrated by the histograms presented in Figure 2.18. For the sake of completeness, the corresponding t-test curves are shown in Figure 2.19 that confirm the statement given above. It is noteworthy that this concept and a corresponding attack was known as *zero-offset second-order attack* [124].



(a) Sample trace

(b) The target bit unchanged

(c) The target bit flipped

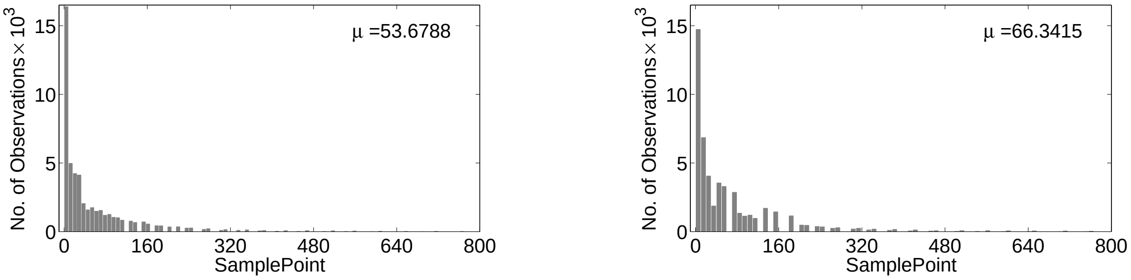
Figure 2.17: Histograms of sample points at 1.481 μs categorized based on a bit flip in consecutive data value x (for the fourth case study)
$\Delta\mu = 0.0782$ and $\Delta\delta = 0.8184$

In the literature, there are two distinct definitions for what the order of an attack is. Since the cryptographic devices initially considered for side-channel experiments were based on software platforms, some previous works defined the order of an attack via the number of leakage points combined by a combination function. This was due to the sequential processing nature of such platforms, where the shares could not usually be processed simultaneously. Other works define the order of an attack via the applied statistical moment. Here, we give the following definition:

- An attack which combines v different time instances – usually in v different clock cycles – of each power trace is called a v-variate attack.
- The order of an attack – regardless of v – is defined by the order of the statistical moments that are considered in the attack.

For instance, a CPA [25] that combines two points of each power trace by summing them up is a bivariate first-order attack, and a CPA that applies the squared values of each point of each trace is a univariate second-order attack. Those attacks in which no specific statistical moment is applied, e.g., mutual information analysis (MIA) [45], are distinguished only by v like univariate or bivariate MIA [12].

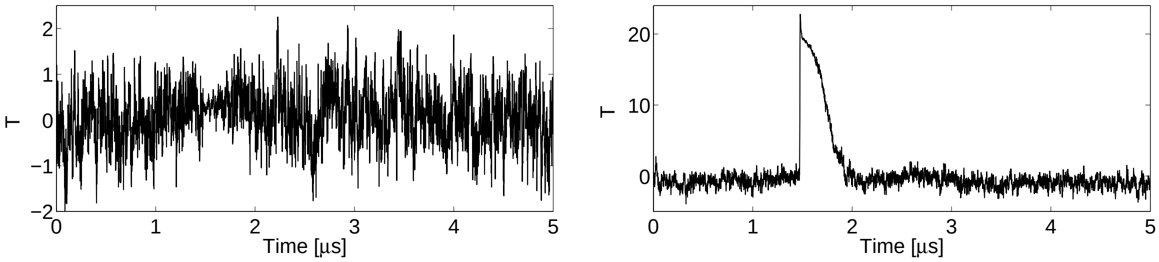
Based on the definition given above, to attack the presented third case study, a bivariate second-order attack should be mounted, because it needs to combine two points by the (centered) multiplication. Along the same lines, a univariate second-order attack is required to be mounted on the fourth case study.



(a) The target bit unchanged

(b) The target bit flipped

Figure 2.18: Histograms of mean-free squared sample points at 1.481 μs categorized based on a bit flip in consecutive data value x (for the fourth case study)
$\Delta\delta = 12.6627$



(a) Non-preprocessed

(b) Centered squared

Figure 2.19: t-tests based on a bit flip in consecutive data value x (for the fourth case study)

Remarks

We recall that producing random numbers is difficult and costly. Indeed, in theory, the realization of masking requires independent and uniformly distributed masks. Even if masks are practically produced by an algorithmic pseudo-random generator, e.g., a stream cipher or a block cipher in counter mode, this operation obviously consumes resources. Therefore, research efforts have been undertaken to mitigate such a necessity. It is interesting to note that some simplifications successfully managed to maintain a notion of security while avoiding the precomputation stage and reducing the required pool of entropy. For example, reducing the entropy of the mask is the concept followed by [94, 95]. Use of fewer mask values allows precomputing all masked look-up tables and fitting them to the small-size platforms.

If masking is correctly realized in software, it can significantly increase the complexity of a successful attack. The goal of most of the techniques is to prove the impossibility of first-order attacks. As stated above, the significant overhead needed to realize the masking schemes is amongst their major drawbacks. This overhead is mainly due to the on-the-fly recomputation of the masked look-up tables that are responsible for realizing the non-linear operations, e.g., AES S-box.

Many solutions have been proposed to relax this problem. Some have focused on avoiding look-up tables for non-linear operations, e.g., with secure multiplication (the interested reader is referred to [40, 110]). Because masking schemes usually assume uniformly distributed random masks, on-the-fly recomputation of the Sbox is unavoidable, unless a huge memory is available to precompute all the necessary tables [106].

Realization of masking in hardware platforms faces many challenges. The initial attempts to make a masked AES Sbox, e.g., [99], failed in practice [70]. Such a failure is due to the multiplier modules that partially multiply the mask and masked data [71]. This is unavoidable, because the masked Sbox circuit needs to have both shares in its processes in order to provide the masked Sbox output. Hence, the implementation platform already provides a leakage associated with a product of two shares, that is amplified because of glitches in the relatively large combinatorial circuit. Since the underlying secret sharing scheme is a first-order Boolean masking, as explained before, a first-order attack is capable of recovering the secret in this situation.

In fact, realizing a masked Sbox in hardware is not as easy as it seems. Most of the attempts could not satisfy the level of security that they are supposed to provide. The main reason behind such a collapse is the glitches which (1) are not easily controllable, and (2) are not considered in the assumptions when the countermeasure is designed. Detailed information about these issues is given in Section 4.2 and Section 5.1.

Chapter 3

Introducing Novel Attack Schemes

Introduction

In this chapter, we deal with three related topics. In the first part, we introduce the steps taken to develop the generic schemes for examining the side-channel vulnerability of a cryptographic device. This is important because most of the previously known attack techniques are based on a hypothetical model that strongly affects their efficiency. The second part deals with how to make use of these schemes in another physical attack. The underlying attack is based on measuring the critical path delay of a combinatorial circuit that partially realizes a cryptographic algorithm. It will be shown that, with the generic attacks we have developed, many AES ASIC cores can be broken. Finally, the last part of this chapter provides a discussion about other side channels that have recently been recovered. We focus mainly on static power consumption and give detailed information about its advantages and disadvantages, compared to the commonly-known dynamic power side channel.

3.1 Generic Techniques for Vulnerability Analysis a1 a3

As explained in Section 2.1.2, a CPA attack needs a hypothetical model to estimate a factor related to the amount of power consumption of the device under attack. Such a model is relatively easy to obtain in multi-purpose applications such as a microprocessor where, due to long data and/or address buses, a Hamming weight (HW) model usually suffices to efficiently mount a successful attack. Further, in such cases, a classical DPA (which targets a single-bit dependency) would also succeed. In hardware platforms – as stated – the Hamming distance (HD) model can be chosen to estimate the power, based on the bit flips.

However, these facts are based on observations obtained by experience. If the actual power consumption of the device under attack is not linearly proportional to the chosen hypothetical model, the attack fails, although there is a dependency between the device's processes and its power consumption. For such cases, one solution is profiling, in which a device similar to the one under attack is required to examine and obtain the actual power consumption model. Gaining access to such a device whose secrets are known or/and can be adjusted is not always possible. Hence, an alternative solution is to mount an MIA that relaxes the necessity for a linear relationship between the hypothetical and the actual power models. Although an MIA brings many advantages, its efficiency may be altered by a set of inappropriate parameters required for probability estimations, as explained in Section 2.1.2. Regardless of such parameters, the most general case for the hypothetical model is called *identity*, where the selected intermediate values are used directly as the output of the model. This model is beneficial only if the selected intermediate value does not have a one-to-one relationship with the targeted key. For instance, if the output of an Sbox in an implementation of DES is chosen as the targeted intermediate value, for any certain x, the function $f : k \mapsto y : \mathrm{S}(x \oplus k)$ is a many-to-one mapping. Hence, selecting y as the intermediate value and, consequently, the same value y as the output of the model in a MIA would not be problematic. In contrast, this does not apply if the device under attack is an implementation of AES as the AES Sbox is a bijection. Hence, instead of the identity model, one needs to select a model that follows the same concept, i.e., a many-to-one function. This can be easily done by, e.g., omitting one bit of the targeted intermediate value y, but this issue has been extensively investigated in [123] and the conclusion was that cases exist – depending on the device actual power model – for which omitting one bit is not beneficial for mounting a successful MIA.

A question that arises here is whether such an implementation, which cannot be successfully attacked by a common hypothetical power model, can be observed in practice. As an example, we refer to [70], where an ASIC implementation of a masked AES, based on [99], is attacked. It has been shown that none of their attempts to mount an ordinary CPA could succeed, but they built a more specific power model as *toggle count* that estimates the amount of the power by counting the number of toggles occurring on a simulated combinatorial circuit (as stated in Section 2.1.2, due to glitches). Using such a power model, they demonstrated that the implementation is vulnerable to a first-order attack. As explained at the end of Chapter 2, such a vulnerability is due to the masked hardware in which the mask and mask data are partially multiplied. This example represents a case in which the actual power model cannot be easily guessed, but it can be obtained by having access to the device netlist (for simulation and, hence, the toggle count model) or it can be achieved by profiling.

3.1.1 Correlation-Enhanced Collision

We have attempted to develop a scheme that can relax the requirements mentioned above. Our development is in the direction of side-channel collision attacks. Such attacks make use of side-channel leakages to detect a collision, thereby recovering some information about the device's internals. Here, a collision might be defined in terms of various criteria. The most general case – called a *linear* collision [20] – considers a case in which the output of two Sboxes in a cipher round (commonly the first or the last) are identical. In the case of the AES encryption at the first round, if such a collision is detected, i.e., $\mathrm{S}(x_1 \oplus k_1) = \mathrm{S}(x_2 \oplus k_2)$, due to the one-to-one mapping of the AES Sbox, it can be directly translated to $x_1 \oplus k_1 = x_2 \oplus k_2$, where x_i and k_i stand for a plaintext byte and a key byte, respectively. Since the plaintexts are supposed to be known to the adversary, he/she thus reveals the *linear* difference between the targeted key bytes $k_1 \oplus k_2 = p_1 \oplus p_2$. Repeating this scenario for the other key bytes – in the case of AES-128 – the adversary can construct at most 15 (out of all $\binom{16}{2} = 120$) independent linear equations, thereby shrinking the key space from 2^{128} to 2^8.

An important question here is how feasible it is to detect such a collision. If the Sbox or other modules dealing with its output provide side-channel leakages in several clock cycles, e.g., a software platform, such a collision detection might be feasible, because the target of most of the investigations in this area was an Atmel microcontroller [19, 20, 22, 112, 113]. However, if a collision is wrongly detected, the constructed system of equations probably omits the correct key. Therefore, several techniques have been developed to deal with false collision detections [19, 20, 22]. An article on how to combine classical DPA and collision attacks [21] is also available. Finally, a generic scheme based on the coding theory that avoids all the heuristics of the former works, has been introduced in [42].

Our scheme [84] initially targets a compact masked AES Sbox by Canright [27] implemented on an FPGA. Similar to [70], all of our attempts to mount a first-order attack by means of ordinary power models failed. However, using the "Correlation-Enhanced Collision Attack" (CECA), which we introduced in [84], we were able to mount a collision attack and shrink the key space drastically. Our scheme is based on the principle that, without profiling, we can obtain a form of the device's actual power model when, e.g., an Sbox over $x_1 \oplus k_1$ is operated. In the next step, we make use of this model and perform the attack when the same Sbox module over $x_2 \oplus k_2$ is performed.

Using a part of the power traces that corresponds to the first Sbox computation, we estimate the means as $\mu_{x_1=\mathrm{x}} = \mathsf{E}(\mathrm{l}_{x_1}|x_1 = \mathrm{x})$, where $\mathsf{E}(.)$ is the expectation operator and l_{x_1} represents a (univariate) random variable as the leakages at the time instance when the first Sbox is computed. Correspondingly, we can define μ_{x_2} as the estimated means, based on the value of x_2 when the second Sbox is performed. In other words, each μ_{x_1} and μ_{x_2} at each sample point contains a vector of 256 elements (in the case of AES, because each x_1 and x_2 represents 8 bits of the plaintext). As the last step, μ_{x_1} is permuted, based on a guessed key difference Δk, and the correlation between the two vectors is estimated

$$\hat{\rho}(\mu_{x_1 \oplus \Delta k}, \mu_{x_2}).$$

The correct Δk should be indicted by the highest correlation, based on certain assumptions:

(1) Enough traces are used to estimate the means μ_{x_1} and μ_{x_2}.

(2) The leakages associated with the first and the second Sboxes are aligned. For instance, if the targeted Sboxes are performed in consecutive clock cycles, the leakages of the second Sbox should be shifted one clock cycle (to the left), in order to be aligned with those of the first Sbox.

(3) There is a dependency between the **means** of the leakages associated with an Sbox, e.g., μ_{x_1} and its corresponding input (resp. output) $x_1 \oplus k_1$.

The last point mentioned above restricts the scheme to first-order leakages, but it can detect any kind of such leakages if the assumptions listed above hold. We have applied this technique to many different applications that are supposed to provide the first-order resistance. The first application was the masked AES Sbox [27], whose first-order leakage is not extractable by means of commonly known HW/HD models. This technique has been also used to verify the first-order resistance of various schemes. For example, the realizations of threshold implementation (discussed in Section 5.1), which is supposed to maintain the first-order resistance, have been practically examined by CECA [74, 90]; they confirm the correctness of the underlying proofs in practice. In Section 4.2, we present a detailed description of the applications for which our technique has been used to evaluate their robustness. In fact, it avoids the necessity for any hypothetical model that might not be found straightforwardly. This plays an important role during the evaluation process of a product (by evaluation labs), because many hypothetical models – e.g., for a CPA – should be used to verify its resistance, which may lead to a false positive conclusion.

Although CECA has been developed for non-profiled (collision) settings, it can also be used when profiling is possible. In such a case, the attack, following the procedure mentioned above – in contrast to a collision case – can recover the value of the key. However, using CECA in a profiled setting might be not favorable, because many other analysis schemes can make use of the profiling phase and develop efficient attacks.

Despite the numerous features and abilities of CECA, it has a few disadvantages:

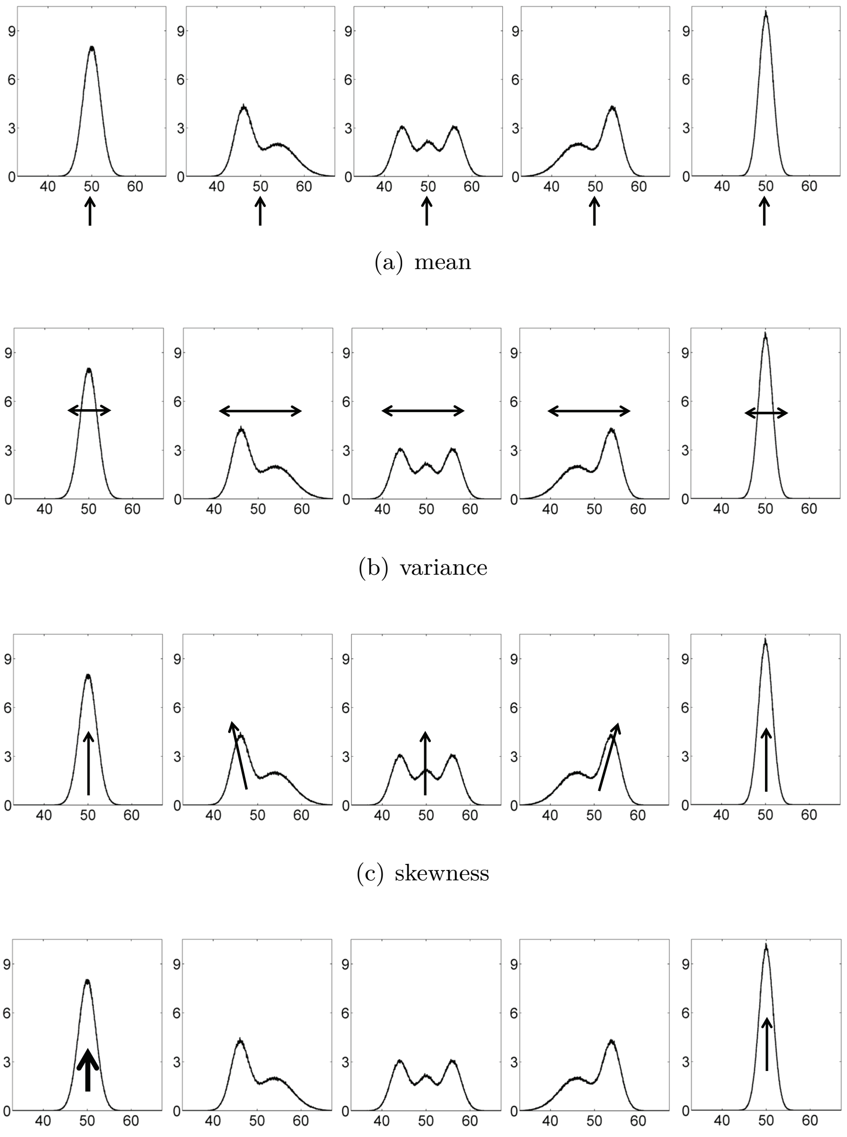
- It is restricted only to first-order leakages. Hence, an implementation with second-order leakage seems to be robust.
- The estimated means μ_{x_i} are correlated to each other. In the case of AES, each vector contributes to the estimation of correlation and has 256 elements. This becomes more problematic when the Sbox size is small. For instance, when an implementation of PRESENT cipher [23] is targeted, each vector would have only 16 elements, which results in a less accurate estimation of correlation.
- Although it can detect any kind of first-order leakage, it cannot be used as a metric. It also cannot provide any information on how hard the attack is or any number indicating the data complexity of the attack, i.e., the number of traces required to successfully mount the attack.

With respect to the last point, one needs to restrict the number of traces and re-do the complete process of the attack to obtain an overview of its minimum number of required traces.

In a subsequent work [74], this scheme has been extended to consider higher-order leakages. For the sake of simplicity, consider the five probability distributions shown in Figure 3.1. In the case of, e.g., a threshold implementation, the mean of all the probability distributions is the same (Figure 3.1(a)). However, their higher-order moments are different. Because their variance, indicating the width of the distribution, are different, they can be distinguished. The same holds for their skewness and kurtosis, which are the measure of the asymmetry and the sharpness of the probability distributions, respectively. Therefore, one can easily consider higher statistical moments, instead of only means, to perform a CECA. It is noteworthy that we assume that the leakages are univariate. Otherwise, the multivariate leakages should be added together, to create univariate leakages, thereby examining higher-order statistical moments. Furthermore, simple ways exist to combine multivariate leakages during the measurements, if they appear in adjacent clock cycles [83]. We provide a detailed description of this work in Section 4.2. We also refer to [74], where we developed a more general scheme that does not consider any statistical moment. It deals with the probability distribution functions (pdf) of the measurements and attempts to detect the collisions by examining the distance/difference between the pdfs (a comprehensive list of the known methods can be found in [29]).

3.1.2 Moments-Correlating DPA

Higher-order CECA eliminates the first disadvantage of the original scheme listed above, but two other drawbacks still remain. Hence, in the last work we have undertaken in this direction [91], we introduced *Moments-Correlation DPA* (MCDPA), which avoids all the aforementioned disadvantages. In MCDPA, the moments, e.g., means μ_{x_i}, are not correlated together. Instead, the estimated moments μ_{x_1} are used as a power model to perform a CPA. In a non-profiling



(a) mean

(b) variance

(c) skewness

(d) kurtosis

Figure 3.1: Probability distributions and statistical moments

(collision) setting, the moments μ_{x_1} are estimated when the first, e.g., Sbox is calculated, and are used to mount the attack on the traces associated with the second Sbox calculation, i.e., l_{x_2} as

$$\hat{\rho}(\mu_{x_1 \oplus \Delta k}, \mathrm{l}_{x_2}).$$

Thus, the number of elements in each correlating vector is not restricted to the size of the underlying Sbox. Instead, like a CPA attack, the number of traces used in the attack determines the size of each vector. Therefore, the drawback related to the accuracy of the estimation of correlation is no longer relevant. Furthermore, the value of the correlation – in case of a successful attack – can directly turn into an approximation of the number of required traces. This is based on the rule-of-thumb, proposed in [68, 114], to approximate the data complexity based on the squared inverse of the estimated correlation, i.e., $n \propto \frac{1}{\hat{\rho}^2}$.

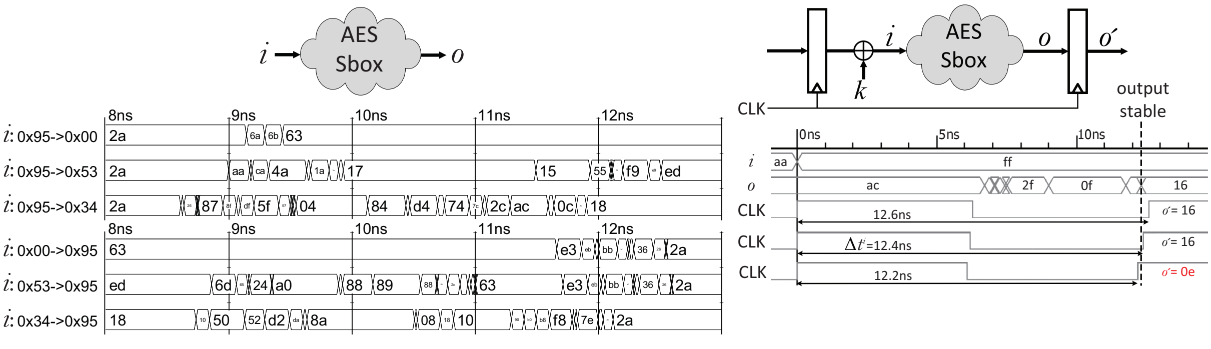
In order to extend MCDPA to apply higher-order statistical moments, rather than only means, in the first step, one can estimate the variance, skewness, kurtosis, etc., based on x_1, similar to a higher-order CECA. In the next step, the traces related to the second part – based on x_2 – must be preprocessed. In the case of the second-order moment, similar to the estimation of the variance as the centered second-order moment, the traces l_{x_2} are preprocessed as $(\mathrm{l}_{x_2} - \mu_{x_2})^2$. For the third- and higher (n)-order moments, the preprocessing is performed as $(\frac{\mathrm{l}_{x_2}-\mu_{x_2}}{\sigma_{x_2}})^n$, because the skewness and kurtosis represent the third- and fourth-order standardized statistical moments, with σ_{x_2} as the standard deviation $\sqrt{\mathsf{E}((\mathrm{l}_{x_2} - \mu_{x_2})^2)}$.

One interesting feature of MCDPA is that the amount of leakages through different statistical moments can be compared. Thanks to the rule-of-thumb, the amount of leakage – with respect to the number of traces – for two statistical moments at different orders can be compared by a ratio of their corresponding estimated correlations. Amongst the known side-channel evaluation/attack techniques, this feature is available solely for MCDPA. Indeed, MCDPA is the correct form of CECA and can be used as a metric to examine and quantify the amount of information leakage of an implementation. It can be also used in a profiling mode by estimating the statistical moments during the profiling phase. Once again, it should be stressed that MCDPA – similar to CECA – targets univariate leakages, and for a multivariate analysis, a prior combination is inevitable when using these schemes.

3.2 Timing in the Range of Gates' Delay a2 a4

Let us consider a combinatorial circuit that realizes an AES Sbox (see Figure 3.2(a)). When its input changes – again due to glitches, as explained in Section 2.1.2 – it takes a certain amount of time t until its output becomes stable. An interesting point is that t differs, based on the consecutive input values. Three examples are shown in Figure 3.2(a), where the input change `0x53` to `0x95` needs a longer time, compared to the other cases. Indeed, the longest t defines a term in hardware logic design as the *critical path delay* that determines the maximum frequency of the circuit.

In short, the time required for a combinatorial circuit to propagate an input change to its output depends on the old and new inputs. Such a dependency was – for the first time – investigated in [64] and is known as *fault sensitivity analysis* (FSA). The main question here is how to measure and extract this dependency from a circuit. The general concept is shown in Figure 3.2(b), where the output of the underlying circuit, e.g., the Sbox, is stored by a register. If the duration of the clock at a specific clock cycle is very short (shorter than the critical path delay), an incorrect value is stored by the register. In fact, the concept followed in [64] is to create a clock glitch when the Sbox circuit is active. The length of the clock glitch Δt is adjusted in such a way that it is slightly smaller than the critical path delay. Hence, for some Sbox input values/changes, the length of the clock glitch is enough to store a fault-free Sbox output, whereas, for some others, it does not suffice. In other words, an abstract of the time required for the Sbox computation is measured. The result of each measurements is a single bit indicating that the result is faulty or fault-free, which is directly translated to the conclusion



(a) (b)

Figure 3.2: AES Sbox combinatorial circuit, (a) the concept of the glitches, and (2) how to measure the circuit delay

that the time required by the Sbox for the associated input value/change is greater or smaller than a certain value Δt.

The authors of [64] proposed applying a distinguisher, such as CPA; i.e, correlating the result of the measurements (a stream of bits) using a hypothetical model. Based on the authors‘ observations, an AES Sbox circuit realized based on PPRM1 [92] (1 stage Positive Polarity Reed-Muller) – on average – needs more time when HW of its output is higher, compared to the cases with low HW. In other words, the time the Sbox circuit needs to make its output stable is approximately proportional to its output HW. Hence, the HW model – similar to common CPA attacks – could be used to mount successful FSA attacks.

In such an attack, the adversary model defined below is considered:

- The adversary can access and control the clock signal that triggers the registers providing the input and saving the output of the target combinatorial circuit (e.g., the Sbox).
- He/She knows in which clock cycle the target combinatorial circuit processes the desired data, e.g., the known or guessed input or output.
- He/She is equipped with appropriate instruments to shorten the duration of the clock glitch with suitable accuracy.
- He/She is able to guess a hypothetical model that is linearly related to the time required by the target combinatorial circuit for different input values/changes.

The last point above actually reflects the limitations of DPA/CPA attacks addressed in Section 3.1: their efficiency depends on the correctness/suitability of the hypothetical model. This issue has been discussed in [64]: using a HW model, an FSA is not successfully mounted on the AES cores whose Sbox is implemented without following the PPRM1 concept.

In order to relax this requirement, we performed extensive analyses on several ASIC implementations of AES [87, 88]. Firstly, we extend the measurement scenario to observe the timings more accurately than single bit values. In order to do so, we need to shorten the duration of the clock glitch Δt while keeping the same input to the target combinatorial circuit, in order to find the exact Δt that, e.g., the Sbox requires. Therefore, we add a further point to the definition of the adversary model:

- He/She can control the target device in such a way that the same input value is repeatedly processed by the target combinatorial circuit during shortening the time interval of the clock glitch.

In fact, for different input values, we are measuring the timing of the combinatorial circuit, allowing us to obtain more accurate information, compared to the original scheme of FSA.

Secondly, we try to relax the necessity for a hypothetical model that can be obtained by (1) a profiling phase, or (2) a simulation including detailed post-place-and-route information. As a straightforward solution, we applied CECA [84] in a non-profiling (collision) setting. In short, the timings obtained from one Sbox computation are compared to those of a second Sbox. The comparison is performed by following the concept of CECA that permutes one of the timings and correlates it to the other one to find the most probable Δk. It is noteworthy that this scheme (in general, CECA and its improved version MCDPA) is feasible when the same target module (e.g., Sbox) is reused (first on $x_1 \oplus k_1$ and then on $x_2 \oplus k_2$). However, if the implementation consists in more instances of the target module (e.g., 16 instances of the Sbox to realize a round-based architecture), the feasibility of the attack depends on the similarity between the two targeted instances. Based on our observations, e.g., in [85], if the Sboxes are even differently placed and routed, their power consumption characteristics still are similar enough to mount a successful CECA. The same has been observed when verifying our FSA-based CECA on the AES cores developed by the SASEBO team [3]. All the AES implementations covered by LSI chips as SASEBO LSI2 [2] in 130 nm and 90 nm technology and SASEBO LSI3 [5] in 65 nm technology have a round-based architecture, but the attack we developed succeeded on all the cores.

It has been shown in [87] that the combination that we made between FSA and CECA could overcome the security provided by not only the unprotected cores but also by DPA-resistant and fault-attack-protected cores. 14 AES cores exist in each targeted LSI. One of the cores of each LSI is dedicated to a fault-detection technique [111], which is supposed to provide security against fault-injection attacks. With a minor adjustment, our attack could break such an implementation (on all three LSIs) that is known as the only attack possible on the underlying fault-detection scheme [111].

In the case of other fault detection schemes (see [73] for a survey of the known methods), our attack can be still feasible, depending on the underlying architecture and implementation method. For example, a concurrent error detection (CED) scheme, namely *invariant-based* CED, is presented in [52], which provides a reasonable rate for fault detection with a very low area overhead. In a subsequent work [73], we adopted our attack and provided techniques to defeat the protection achieved by the invariant-based CED.

In short, we stress that these kinds of attacks, which need a clock glitch to somehow measure the timing of a combinatorial circuit, can be defeated by an embedded module that detects a clock glitch. This countermeasure is usually considered by smartcard vendors when issuing tamper-resistant products. However, such a countermeasure can affect the performance of the device because the clock frequency is restricted to be not higher than a threshold that is slightly lower than the maximum operation frequency of the device. A practical experiment in this direction has been reported in [37]; it demonstrates a possible way to detect such clock glitches. It also describes the behavior of the presented scheme in different supply voltages and shows its relatively strong effect on the maximum clock frequency of the circuit at low voltage supply values.

3.3 New Side Channels (a5)

Execution time [61], power consumption [62], and electromagnetic emanation [6, 39, 108] are very well-known side channels that have attracted the attention of researchers. However, the side channels are not only limited to this list. Acoustic [41], optical emission [38], and temperature [57] side channels are amongst those that have also been brought to the attention of scientific communities. However, due to their efficiency, low-cost, and simplicity, power consumption and electromagnetic emanation side channels have been widely investigated more then the others and applied in academia as well as in industry. Not all, but most of the activities in the areas of side-channel analysis have been undertaken based on the principle of CMOS circuits, i.e., focusing on the main power consumption factor of the circuits, namely, *dynamic* power consumption. Therefore, the attacks and countermeasures are mainly based on the dynamic power consumption of the underlying circuit.

During recent years, by shrinking the technology, the VLSI community has reported the dependency of *static* power consumption of a CMOS circuit on its internals (see [46, 65]). Moreover, interesting results are presented in [8] and [9], where an attack using static power is called *Leakage Power Analysis* (LPA). This issue has been reported mainly based on simulation results. Practical results exist in [9] and [46] where a weak dependency between the static power and the content of registers is shown. However, none of the aforementioned articles report a key-recovery attack based on static power in practice.

Due to the very small scale of the signal amplitude (of static power consumption), which cannot be easily measured in practice by means of the currently-available facilities and equipment, the static power side channel was not taken as a serious threat by the side-channel community. Indeed, the belief of the community – which is not far removed from reality – is that the information available through the dynamic power consumption channel is much more extensive and much easier to detect, compared to that of the static power.

The first questions are how to measure the static power and what is the difference in comparison to dynamic power measurements. We refer to Figure 2.1 and Figure 2.2 in Section 2.1.1, where several techniques for measuring dynamic power consumption are presented. In the case of static power measurements, the desired component to be measured is the DC part of the power signals. Hence, options such as using a current probe (Figure 2.1(b) and Figure 2.2(d)), using a DC blocker (Figure 2.2(b)), and any kind of AC coupling (Figure 2.2(a)) are not available for static power measurements. Furthermore, AC amplifiers which remove the DC shift, e.g., `ZFL-1000LN+` (Mini-Circuits) or `PA303` (Langer EMV-Technik[1]), cannot be employed. The only available options are measuring (1) by DC coupling in the GND path (Figure 2.1(a)) and (2) in the Vdd path using a differential probe (Figure 2.2(c)).

In [75], we have demonstrated the first practical results of a side-channel analysis using information leakage through static power consumption. The presented experiments are based on Xilinx FPGAs, i.e., three FPGA families (Virtex-5, Spartan-6, Kintex-7) with three different process technologies, namely 65 nm, 45 nm, and 28 nm. For all the practical results shown, the

[1] `www.langer-emv.com`

static power was measured in the Vdd path using an AP 033 differential probe from LeCroy equipped with an × 10 amplifier, which is helpful, due to the very low amplitude of the static power signal.

The initial experiments showed that the value of I/O signals have a strong effect on the amount of the measured static power. Therefore, a measurement methodology has been developed that should be followed during the static power measurements. It implies that all I/O signals should be set to a deterministic value and that the communication channels, e.g., UART, are off during the measurements.

The results of the preliminary experiments of [75] show a clear dependency between the static power and the value stored/processed by the fundamental components of the FPGAs, i.e., register, LookUp Table (LUT), and signal routing. An AES encryption engine (with and without DPA countermeasures, i.e., shuffling and masking) has also been considered in the experiments, which indicates the feasibility of key-recovery attacks thorough static power.

According to the results of this work, it can be summarized that *side-channel attacks using static power consumption are possible*, but many issues and difficulties remain and are listed below:

- Currently-available measurement setups, which are optimized for dynamic power analysis, are not suitable for static power analysis. Hence, a dedicated measurement setup must be designed that matches the characteristics of static power consumption signals.

- In contrast to the case of dynamic power, the measurement environment has a dominant impact on the static power measurements. Hence, a sophisticated measurement environment, e.g., a climate chamber, that precisely controls the temperature and humidity, should be used and adjusted for the requirements of static power measurement.

- In contrast to ASIC platforms, the signal routings are realized by switch boxes in FPGAs. Therefore, the length of signals routed in the FPGAs has a considerable effect on the amount of its static power consumption. This issue is expected to be relaxed in ASIC platforms.

Here, a highly important issue is related to the security of univariate-resistant masking schemes, e.g., [107], when static power is considered as a side channel. In contrast to the case of dynamic power, the sum of leakages associated with all shares – of a masking (secret sharing) scheme – is observable through static power. Hence, the univariate-resistant masking schemes are expected to be more easily vulnerable through static power than multivariate attacks through dynamic power.

Although many novel concepts have been identified by the results of [75], it is a preliminary study and much more research in this area (e.g., the vulnerability of univariate-resistant masking schemes) should be performed. This requires a deep understanding of side-channel attacks through static power which is partially (only for FPGA devices) given by [75]. The adaptation and re-design of the currently available countermeasures, so that they can provide robustness against power analysis attacks considering both dynamic and static power side channels, is desirable.

Chapter 4
Evaluation of Theory in Practice

Introduction

This chapter covers two topics, both of which deal with the evaluation of side-channel resistance of either commercial or prototype cryptographic devices. The first part presents the result achieved by the evaluation of the vulnerability of the bitstream encryption feature of Xilinx and Altera FPGAs. In the second part, we discuss the evaluation of some countermeasures, which have been designed mainly for hardware platforms. It is shown that, in most of the cases, although sound theoretical models support the countermeasures, their implementation in practice usually fails. The reasons behind such a failure are discussed, and the challenges one may face when realizing a masking algorithm in hardware are introduced.

4.1 Security Evaluation of FPGA Bitstream Encryption b1 b2 b7

Most of today's FPGAs are (re)configured with bitstreams that determine the complete functionality of the device. In most cases, FPGAs produced by the dominant vendors use volatile memory, e.g., SRAM, to store the bitstream. The bitstream is stored in an external, non-volatile memory, e.g., EEPROM or Flash, and is transferred to the FPGA on each power-up. One of the disadvantages of FPGAs, especially with respect to custom hardware such as ASICs, is that an attacker who has access to the external memory can easily read out the bitstream and clone the system, or extract the IP of the design. The solution that industry has provided for this issue is a security feature called *bitstream encryption*. This scheme is primarily based on symmetric cryptography, in order to provide confidentiality of the bitstream data. Currently, FPGA families exist that provide authentication based on either symmetric (HMAC) or asymmetric cryptographic primitives.

In bitstream encryption, the designer encrypts the generated row bitstream with a secure symmetric cipher, such as AES, using a secret key k. The encrypted bitstream can now be safely stored in the external memory. The FPGA possesses an internal decryption engine and uses the previously stored secret key k to decrypt the bitstream before configuring the internal circuitry. Thus, wire-tapping the data bus or dumping the content of the external memory containing the encrypted bitstream does not yield useful information for cloning or reverse engineering the device, given that the adversary does not know the secret key k.

4.1.1 Xilinx

The cryptographic scheme used by Xilinx FPGAs, starting from the old Virtex-II pro family through to the recent 7 series, is Triple-DES (3DES) or AES-256 in Cipher Block Chaining

(CBC) mode. Altera made use of AES-128 (in previous families) and AES-256 (in the recent and currently active families) in Counter (CTR) mode. Until the publication of our findings – first reported in [77] – the bitstream encryption feature of Xilinx FPGAs was considered unbreakable.

The documents related to the bitstream encryption, which Xilinx made publicly available, sufficed to understand the structure of an encrypted bitstream and verify all the detail of its format by examining a few examples whose key k were known to us. For the first targeted family, we considered a Virext-II pro FPGA embedded on the early versions of SASEBO [3]. Because the SASEBO platforms provide a reasonably suitable situation for power measurements, we have made use of them for our initial analysis. Based on the knowledge obtained from the Xilinx public data-sheets, we developed a microcontroller-based device to play the rule of a Xilinx program. In addition to the basic features for configuring the underlying FPGA, it provides trigger signals to align the measurements properly.

Because no information about the internal architecture of the decryption module inside the FPGA is available, all of our analyses are based on trial and error. Indeed – similar to the case of KeeLoq [36] – it is a real-world target to perform side-channel attacks in a black-box scenario. Since, supposedly, the decryption module is one of the small components inside the large FPGA circuitry, its contribution to the power consumption of the chip is very small. Nevertheless, by means of a dedicated measurement setup, as well as digital filtering, we could obtain (filtered) power traces that seemed suitable for an attack. Due to the strong dependency of the efficiency of a CPA attack on the underlying power model, we had to guess the architecture of decryption module in order to develop an attack. Finally, we have discovered that the decryption module follows a round-based architecture that performs each round of the 3DEC decryption in a clock cycle. Based on these findings, we could select a suitable intermediate value (and HD model) to mount successful attacks that recovered all the key bits stored in the FPGA. Due to the DES cipher structure, each step of the attack recovers 6 bits of the key by searching in a space of 2^6 candidates.

This approach has been continued to evaluate the vulnerability of other Xilinx families more recent than Virtex-II pro. Because the underlying cryptographic algorithm had changed from 3DES to AES-256, the same attack as before could not be straightforwardly mounted on Virtex-4 and subsequent families. Although the measurement setup and the developed tools could be partially used with minor adjustments, due to the differences between the bitstream encryption of the old and new Xilinx families, the internal architecture of the AES-256 decryption module had to be guessed. In fact, this was the most time consuming task, because many different methods exist to realize such a decryption function in hardware. Finally, we ended up with a round-based architecture, where – similar to the case of Virtex-II pro – every decryption round is performed in a clock cycle. Compared to the previous attack, here, after an optimization step, we had to search in a space of 2^{32} in each part of the attack to recover 32 bits of the key. Performing a CPA attack with 32-bit key hypotheses is challenging, hence, we made use of Graphic Processing Units (GPU) to accelerate the attack up to 33 minutes. It should be noted that this attack was performed on a single point of 60 000 power traces, because it was discovered during the analysis with a known key.

We observed the same power pattern during the analysis of bitstream encryption in Virtex-5, Spartan-6, and Kintex-7 devices. Indeed, the same attack succeeds on these devices with the same hypothetical architecture for the decryption module. This implies that the same decryption module has been embedded in all Xilinx FPGA families that utilize the AES-256 decryption function. Although, due to reliability issues, it makes sense to use the same module in different designs, in this case, by performing a successful attack on one of the devices, the others also become potentially vulnerable.

4.1.2 Altera

As stated, the Altera bitstream encryption feature – called *design security* – employs AES-128 or AES-256 in CTR mode. Contrary to Xilinx, (to date) no public documents are available regarding the details of this feature of Altera FPGAs. Therefore, we had to recover all the details by reverse engineering [89]. The victim was the Altera development tool, i.e., Quartus II software application, which can generate encrypted bitstreams. After recovering all the details and developing the measurement setup, as well as a dedicated device to perform the configuration (feeding the encrypted bitstream to the FPGA), we could start the measurements and verify the correctness of our hypothetical architectures for the encryption module. Note that, because the underlying mode of operation is CTR mode, an encryption function is used by both parties, i.e, Quartus II development tools as well as the FPGA chip.

Compared to the Xilinx case, here we had to somehow send chosen plaintexts to the FPGA during the measurements. This requirement is due to the CTR mode, because most of the input bits of the plaintexts given to the encryption module during one power up stay unchanged. Hence, we had to repeatedly reset the configuration process and send a random initial value (IV) for the CTR mode. In this way, we could collect power traces associated with different (random) plaintexts.

As the last step, by using the correlation with the known key, the internal architecture of the encryption module has been revealed. Accordingly, a feasible CPA attack with a suitable power model was developed that, at most, needs to search in a space of 2^{16} at each step. Our first target was a Stratix-II FPGA, in which AES-128 is employed. The result of our attack [89] is known, to date, as the only possible attack on design security of Altera FPGAs and it confirms the feasibility of a full key-recovery attack. In the recent families, e.g., Stratix-III, Altera made use of AES-256 as the encryption function. Further, based on our reverse engineering of the Quartus II application, the CTR mode was not realized following a common scheme, and the input of the AES-256 encryption function is not easily incremented just arithmetically. Instead, a proprietary function, which seems to be an LFSR, is embedded to update the plaintexts for the encryption function [118]. This new development actually eases our side-channel analysis because a constant part in consecutive plaintexts is no longer observed. Hence, the adversary does not need to send random IVs. Instead, the measurements related to a normal power-up of the targeted FPGA suffice to obtain useful traces for the attack.

Comparison

In summary, we show that, up to now, we could successfully break the security feature of all Xilnix FPGA families that offer a bitstream encryption functionality. Although we usually examined only one device per family, i.e., Virtex-II pro, Virtex-4, Virtex-5, Spartan-6, and Kintex-7, we are confident that the other devices in each family are embedded with the same decryption engine. Hence, we expect that our developed attacks will, with minor adjustments, succeed on other devices from the same family, e.g., Virtex-6. In the case of Altera, we targeted only Stratix-II and Stratix-III FPGAs. Because Altera has moved to the AES-256 encryption function since the generation of Stratrix-III, we believe that the same encryption module is embedded in recent families, e.g., Stratix V and Arria II. Indeed, we have verified this assumption by the result of our reverse engineering efforts on the Quartus II application.

In brief, we now compare the attacks and the associated difficulties that an adversary faces when attacking the Xilinx and Altera bitstream encryption:

- **Measurement scenario**: the power traces of all Xilinx FPGAs can be measured during their normal power-up, but an Altera Stratix-II FPGA should be repeatedly reset with random IV during the measurements; this also requires detailed knowledge about the encrypted bitstream structure. Stratix-III and later Altera families do not face such an issue, due to their underlying counter-update function.

- **Attack**: recent Xilinx families need to be attacked by searching in a space of 2^{32}, but Altera FPGAs can be attacked, in most cases, by guessing a 16-bit secret. The attack on Xilinx devices can be improved by not following the common measurement scenario. In other words, if the FPGA is repeatedly reset and is fed by chosen ciphertexts (contrary to the normal power-up), the search space can be significantly reduced (to 2^{16}).

New Xilinx families, e.g., Virtex-6 and all FPGAs from 7 series, include an authentication feature (HMAC using AES-256) that is supposed to check the integrity of the encrypted bitstreams. However, no dedicated place to save the HMAC key is reserved in the FPGAs. Instead, the first block of the encrypted bitstream contains the HMAC key. Further, the correctness of the HMAC is examined at the end of the loading, when the encrypted bitstream is fully loaded into the FPGA. Hence, this feature does not affect the feasibility of our developed attacks. Even if random ciphertexts are fed to the FPGA to ease the attack procedure, the integrated HMAC does not prevent or harden the attacks.

In fact, all our analyses in this direction (vulnerability analysis of real-world devices in practice) indicate the feasibility of side-channel attacks, even in black-box scenarios. Our results demonstrate that, after more than 15 years since the introduction of side-channel analysis, many security-enabled devices exist in which no dedicated countermeasures to defeat such attacks are embedded. This highlights the necessity for side-channel countermeasures for real-world applications that may fall into the hands of an adversary.

4.2 Failure of Masking in Hardware b3 b4 b5 b6 b8 b9

As mentioned above, after practical evaluation of a Boolean masked AES Sbox [99] in hardware [70], it became clear that, due to the glitches, realizing masked hardware is not easily feasible. Further, as shown in Section 3.1.1, generic schemes (e.g., CECA) can exploit the information leakage through specific statistical moments without being limited to a certain hypothetical power model. Hence, any leakage (at certain order) that is made available by complex masked hardware [27] can be examined and exploited to mount a successful key-recovery attack. Therefore, researchers have invested effort in designing and introducing various schemes to deal with this problem and prevent such leakages by hardware.

4.2.1 AES Dual Ciphers

As one of the initial schemes in this direction, a few works have focused on making use of the AES properties to randomize the intermediate values. They are mostly based on altering the underlying irreducible polynomial. Indeed, they follow the concept of *dual ciphers*, initially proposed in [10], because two ciphers E and E' are called dual ciphers if they are isomorphic. In other words, invertible linear transformations $f(\cdot)$, $g(\cdot)$ and $h(\cdot)$ exist such that

$$\forall P, K \qquad E_K(P) = f(E'_{g(K)}(h(P))),$$

where plaintext and key are denoted by P and K, respectively.

By a simple squaring (in GF(2^8) and under the AES irreducible polynomial), a dual cipher can be created [10]. By iterating this process, 8 dual ciphers for each possible irreducible polynomial in GF(2^8) can be derived. Since it is also shown in [10] how to create dual ciphers by porting the cipher to use one of the other 30 irreducible polynomials in GF(2^8), a total of 240 non-trivial dual ciphers for AES exists. In fact, compared to original AES, all the constants, including the replacement of the irreducible polynomial, the coefficients of the MixColumns, and the affine transformation in the Sbox, are changed to realize an AES dual cipher. Note that all generators, polynomials, and constants of each of the 240 AES dual ciphers can be found in [11]. Also, it has been shown in [109] that one can include field mappings from GF(2^8) to GF$(2)^8$, as well as intermediate isomorphic mappings to GF(2^2) and GF(2^4), to build 61 200 similar AES dual ciphers.

Since the intermediate values of the dual ciphers are different from those of original AES, it is mentioned in [10] that one can randomly change the constants of the cipher, thereby realizing different dual ciphers and provide security against power analysis attacks. This idea was taken up by [43, 44, 58, 59, 125] that present the performance and area loss when the scheme is realized to increase the security against side-channel attacks.

In [82], we examined this scheme, i.e., random selection of constants to choose a dual cipher out of 240, from a side-channel point of view. We have implemented an evaluation circuit realizing the scheme on a Virtex-5 FPGA by means of precomputed matrices and constants that is able to perform AES computations using randomly chosen 240 dual ciphers. Besides providing practical evidence of the vulnerability of this original dual cipher implementation to several side-channel attacks, we have also described some of the general flaws of the scheme when it is considered as a side-channel countermeasure.

In fact, the most problematic property of such schemes is their inability to mask the zero value. This is a general problem in multiplicative masking schemes because, regardless of the mask m, the input value $x = 0$ never gets masked. Therefore, a CPA attack using the zero-value power model [7] can easily overcome the protection. The same problem applies to the dual cipher approach. Because of the linearity of the transformations, i.e., multiplication by a matrix in GF(2), e.g., $h(.)$, zero input is always transformed to itself in all 240 cases.

In addition, we have shown that the vulnerability of dual cipher implementations is not only limited to the zero value. They also suffer from the presence of elements such as the mask reuse, the concurrent operations on both mask and masked data, and the violation of the *balance* property. In short, even by ignoring the large area overhead of the circuit, compared to other lighter masking schemes, AES dual ciphers are unsuitable as a side-channel countermeasure and can be broken using modest efforts and simple attack models.

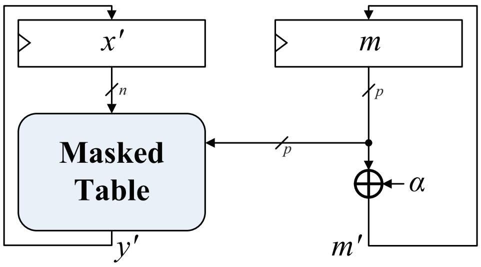


Figure 4.1: The underlying scheme of [67]

4.2.2 Altering the Mask Pull

Suppose an AES Sbox $S(.)$, and x as its input. When the input is represented by $x' = x \oplus m$ and m, the output of the masked Sbox is represented by $y' = S(x) \oplus m'$ and m'. This mapping is done with a pre-computed look-up table T as $(x', m) \mapsto (y', m')$, where the output mask m' is made by a deterministic function over the input mask m. Therefore, the look-up table T maps a $2n$-bit input to a $2n$-bit output and is of the size of $2n \cdot 2^{2n}$ bits. This scheme, which is a common way of realizing a masked Sbox if the table fits into the memory space, is called a "conventional" scheme.

The problem that is observed in [67] is a univariate leakage caused when the registers containing the look-up table input (x', m) are updated by its output (y', m'). The bit flips in the registers are represented by $(\Delta x, \Delta m) = (x \oplus S(x) \oplus m \oplus m', m \oplus m')$. Therefore, a univariate MIA or a univariate second-order CPA can reveal the relationship between the bit flips and x. In order to overcome this problem, a new scheme has been introduced in [67].

In the new scheme, x is represented by two shares $x' = x \oplus F(m)$ and m, where the bit length of m (denoted by $p < 2n$) is no longer equal to n, and F is a deterministic function from $\mathbb{F}_2^p$ to $\mathbb{F}_2^n$. The look-up table T^* maps a $(n + p)$-bit input (x', m) to an n-bit output $y' = S(x) \oplus F(m')$, and the output mask is computed easily as $m' = m \oplus \alpha$, where α is a non-zero p-bit constant. The table T^* needs $n \cdot 2^{n+p}$ storage bits and still is comparable to the size of T for values of p close to n. In this case, the register update – the same scenario as before – leads to $(\Delta x, \Delta m) = (x \oplus S(x) \oplus F(m) \oplus F(m'), m \oplus m')$. The mask difference is constant, i.e., $\Delta m = \alpha$, but in order to appropriately choose the function F, two constructions have been proposed in [67].

Both proposed constructions satisfy the required conditions, i.e., that Δm be constant and the distribution of $F(m) \oplus F(m')$ be uniform. The underlying concept is to keep T^* small enough to be able to fit it into ordinary platforms with limited memory and, at the same time, avoid the leakage associated with the register updates and, consequently, prevent univariate attacks of any order [67]. An overview of the scheme is shown in Figure 4.1.

In [81], we presented the corresponding practical realization and evaluation results. In addition to the limited practical relevance of the scheme[1], we have shown that it can resist attacks that target the register update model. Indeed, we considered a CECA with register update model to examine the first- and the second-order leakages without any restrictions on the

[1] Usually, in hardware platforms, the Sbox output is not saved in a register containing its corresponding Sbox input.

power model. We indeed confirm the proven security notations of [67]. However, the leakages available in hardware platforms are not solely restricted to register updates. Therefore, a similar CECA with, e.g., an Sbox input model, recovers the secret from the implementation of both constructions of [67]. We showed that, although the scheme can prevent the attacks targeting register updates, a simple modification to the target of the attack can make it successful.

We have thus demonstrated that more leakage sources exist in hardware platforms than those assumed by the theory of side-channel countermeasures. We did not break the proven security of the underlying countermeasure, but we simply showed that there are many more opportunities for the attacker to gain side-channel leakages, thereby breaking a device equipped with a countermeasure whose security is proven.

Low-Entropy Masking

It was discussed, in Section 2.2.1 ,that the random masks m are taken from a uniform distribution. Otherwise, masked data $x' = x \oplus m$ do not cover the entire space $\{0,1\}^n$ (n: number of bits of x and m, typically 8). Hence, observing the distribution of x' may directly yield information about x. In this direction, some works exist in which attempts were made to reduce the mask entropy while maintaining a certain level of security that the masking scheme can provide.

For example, we refer to [49] and [66], where the concept as *low-entropy masking* is studied. The basic idea is based on coding theory. Suppose that the masks are drawn from a binary linear code C of length n, dimension k, and minimum distance d as $[n,k,d]$. It can be proven that such a masking scheme can provide security against first-order attacks if the algebraic degree of the actual leakage function (of the cryptographic device) $< d$. In formal terms

$$\mathsf{E}(\mathscr{L}\,(X \oplus (M \in C)) \mid X = x)$$

is constant for any pseudo-Boolean function $\mathscr{L}$ of algebraic degree $< d$, where X represents any n-bit random variable, M is a random variable uniformly distributed over the binary linear code C, and $\mathsf{E}(.)$ represents expectation. As an example, selecting an optimal $[16,8,5]$ binary code leads to the following conclusions:

- It can resist any first-order attacks if $\mathscr{L}$ algebraic degree < 5.
- Similarly, it can resist univariate second-order attacks (without considering the leakage of the mask M) for all leakage functions with algebraic degree < 3.
- However, a univariate third-order attack should succeed if the leakage function is not linear.

Rotating Sbox Masking (RSM) is an example of such low-entropy masking schemes [95]. The mask values are chosen in such a way that the leakage caused by the masked data $x' = x \oplus m$ depends on x only if $\mathscr{L}$ algebraic degree at least is 4. It is explained in [14] that such security can be achieved if the masks are distributed as the 16 codewords of the $[8,4,4]$ binary linear code (extension with one parity bit of the $[7,4,3]$ Hamming code).

In other words, the masks are drawn from a set with 16 elements, each of which has 8 bits. An AES implementation of the scheme, which keeps a certain order of the masks with respect to the 16 Sboxes of each encryption round, has been introduced in [95], and the corresponding traces became publicly available through the DPA contest V4 [119].

According to the underlying binary code, an attack (e.g., CPA) with a model as bit(s) (or HW) of the Sbox input or output should not be feasible, which is already one step toward first-order security with very limited mask entropy. This has been practically confirmed by the analysis we presented in [78]. However, since in software platforms the registers containing the Sbox input are usually overwritten by its corresponding Sbox output, we have shown in [78] that successful attacks can be mounted targeting the bit flips in the aforementioned registers.

Indeed, we have demonstrated that $x' \oplus y' = (x \oplus m_{\text{in}}) \oplus (S(x) \oplus m_{\text{out}})$ can be represented as $(x \oplus S(x)) \oplus (m_{\text{in}} \oplus m_{\text{out}})$, which means that $x \oplus S(x)$ is, conceptually, masked by $m' = m_{\text{in}} \oplus m_{\text{out}}$. However, the set formed by m' is not a binary linear code, and therefore the bits of m' are not balanced. As a result, the bits of m' are not uniformly taken from $\{0, 1\}$, and a classical attack using bits of $x \oplus S(x)$ succeeded by considering less than 1 000 measurements.

With this work, we again demonstrated that the implementation platform causes many leakage sources that are not considered in the underlying theoretical model. We should mention that, in some cases, it is not easy and straightforward to consider all leakage sources of the platform. This actually highlights the importance of designing countermeasures without strong assumptions about the actual leakage model of the platform.

The work presented in [26] is in the same direction; it demonstrates how to make use of the wire-tap code approach in side-channel protection. Similar to the low-entropy masking (leakage squeezing) concept, a binary linear code $C: \ [n, k, d]$ is defined. C consists in a generator matrix G of size $(n - k) \times n$ with rows of $(g_1, \ldots, g_{n-k})$, where each row is an n-bit binary vector. A codeword $c \in C$ is generated as $m \cdot G$, where m is a $(n - k)$-bit binary vector. Indeed, all 2^{n-k} codewords of C can be generated by m passing through the entire space vector $\{0, 1\}^{n-k}$. C also contains a $k \times n$ (binary) parity-check matrix H as $\forall c \in C, \ c \cdot H^T = 0$.

In order to encode a k-bit message x, it should be first expanded to n bits by means of a $k \times n$ matrix L as $x \cdot L$. Rows of $L = (l_1, \ldots, l_k)$ are n-bit binary vectors that are linearly independent of each other, and none of them is a codeword. The encoding of x is determined by adding (modulo 2) a randomly selected codeword to the result of the expansion as

$$x' \ = \ x \cdot L \ \oplus \ m \cdot G.$$

For the case of AES with $k = 8$ and the binary linear code $[16, 8, 5]$, the authors of [26] showed how to implement the AES encryption function accordingly. It has been shown that the scheme can provide security against power analysis attacks while it also contains a feature that can detect certain faults. Our analysis, presented in [76], considers a hardware implementation of the scheme and addresses the difficulties one may face in realizing such a scheme. The problems are mainly due to the size of the masked Sbox because the input and output bit widths are no longer 8 bits. In fact, the most challenging issue is related to the Sbox input size, i.e., 16 bits, because the expansion matrix L and the underlying binary linear code have a length of 16 bits. Therefore, we provided two different solutions to fit such large masked Sboxes into the memories available in the FPGAs, but both solutions seem to turn the scheme into a classical Boolean masking where the desired features of the wire-tap code are no longer present.

The practical evaluation of [76] indeed confirms the resistance of the scheme to attacks up to a certain order (with respect to the minimum distance of the underlying code). However, its area overhead is much higher than a classical Boolean masking on the same platform, and some features of the scheme, e.g., removing the mask without knowing it, are negligible, due to the specification of the underlying algorithm AES.

Despite its high area overhead, it seems that a wire-tap code can be taken as a potential candidate to satisfy resistance against both side-channel and fault-injection attacks. It is noteworthy that its ability to detect faults has not been extensively investigated, and it is not yet clear to which type of fault attacks and under which fault models it resists.

4.2.3 Univariate-Resistance Masking

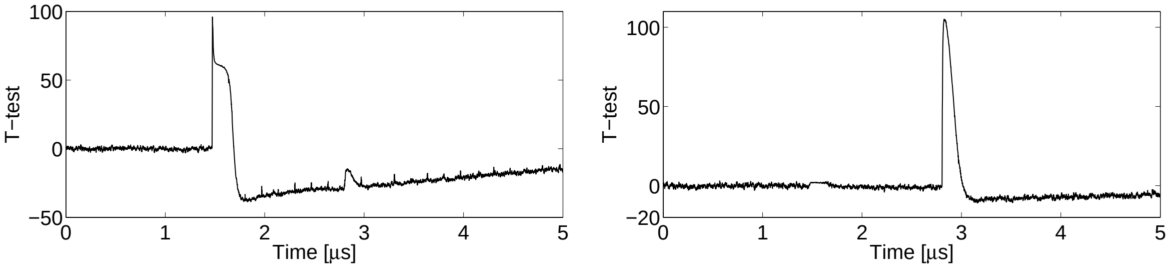
Most of the masking schemes in hardware target protection at a certain order. This is because the shares are usually processed at the same time in hardware platforms, which results in a univariate leakage (preferably at higher orders). The simultaneous processing of the shares is due to one of the main features of the hardware platforms, i.e., efficiency. In contrast, software platforms – due to their sequential nature – usually process the shares in different clock cycles.

The only masking scheme known, to date, that can provide univariate resistance of any order is [107], which is based on the Shamir secret-sharing scheme and a multi-party computation protocol. According to its construction, a minimum of three shares is required to achieve univariate resistance. Each share is given to its corresponding module and, later, the modules communicate with each other to finish the computation. Such communications are specifically protected by fresh new masks to avoid the leakage.

Based on our evaluation results [83] to partially implement the AES encryption function (actually only the Sbox), the scheme's overhead is relatively high. This overhead is not restricted to the area; its timing overhead is also significant because only one Sbox computation needs at least 132 clock cycles. It actually turns the hardware platform into a design with software characteristics, because several sequential operations must be conducted, which is an apparent contradiction to a major purpose of the hardware platforms, i.e., parallel processing and efficiency. However, the practical evaluations showed that – in contrast to many masking schemes – it can provide univariate resistance. This means that if the adversary is not able to combine the leakages in different clock cycles, no attack of any order would be possible.

It is commonly believed that it is favorable for the adversary to run the device under attack at a low frequency, e.g., up to 10 MHz. Otherwise, the power peaks at consecutive clock cycles overlap each other [69], and it is not easily possible to distinguish the clock cycles from the power traces. However, the shares of the scheme of [107] are processed in consecutive clock cycles by different modules that realize the multi-party computation concept. Therefore, in this case, if the device runs at a high frequency, the adjacent power peaks associated with the shares of a secret are partially added together. Hence, a univariate attack would be possible. We have examined this issue in [83], and demonstrated that, by running the device at a frequency of 24 MHz, a second-order univariate attack is possible. Note that since the combination function – caused here by overlapping the adjacent peaks at a high operation frequency is an additive combination, a first-order attack is not successful.

As another alternative to running the device fast, we reported, in [83], an interesting feature of the measurement setup that can also lead to overlap of the adjacent power peaks. We observed that if a DC blocker (e.g., Figure 2.2(b)) and/or an (AC) amplifier (examples in Section 2.1.1) is employed in the measurement setup, the power peaks at each clock cycle also partially include information about the process of the previous clock cycles, due to the capacitive elements at the input of such components.



(a) based on an Sbox input bit flip

(b) based on an Sbox output bit flip

Figure 4.2: (the second case study + BC block + amplifier) *t*-tests

To find evidence for such an effect, we repeated the experiments of the second case study presented in Section 2.1.2. In the new experiments, we changed the measurement setup by employing a DC blocker `BLK-89-S+` and an AC amplifier `ZFL-1000LN+`. After collecting the same number of traces, i.e., 100 000, and performing the same *t*-test (based on an Sbox input and output bit flip, the result shown in Figure 4.2 was obtained. Compared to Figure 2.11, it is clear that the dependency of the power traces on the targeted Sbox input bit flip is still observable after more than several clock cycles. This effect is more obvious when the Sbox input bit-flip is targeted, that is due to the height of the power peak when the Sboxes are computed (see Figure 2.9(a)). Based on this issue, and the result of [83], we conclude that moving further in this direction to develop univariate schemes in hardware does not seem to be useful. An alternative could be a scheme in which the shares would not have to be processed in consecutive clock cycles, and a reasonable time distance between the processing of the shares of a secret would be considered.

4.2.4 Masking at Cell Level

As explained in Section 2.2, a few works exist that aim at providing side-channel resistance by – ideally – equalizing the power consumption independent of any process or data. These schemes, which make use of dual-rail pre-charge logic, require a balance between the capacitance and delay of dual rails. Otherwise, the arrival time (or power) would depend on the value the dual rail transfers. This propagates to a combinatorial circuit (implemented by the underlying logic style) and the power traces would be related to the processed values.

In order to avoid such an issue, which needs a custom routing process, MDPL has been introduced [103]. It employs a single mask bit for the entire circuit to randomize the selection of the routes by each dual rail. However, after a proper evaluation [116], it was discovered that most of the proposed logic styles, such as WDDL [121] and MDPL [103], suffer from the *early propagation* effect. This phenomenon, also called *data-dependent time-of-evaluation*, refers to cases in which a gate fires (evaluates) its output at different time instances, depending on the value of its input. It becomes more problematic when several such gates are cascaded to realize a combinatorial circuit. Hence, it causes the power consumption pattern of the circuit to have a clear dependency on its input value. Indeed, practical evaluations reported in [102] confirm this issue, and show that an MDPL circuit can be attacked (relatively) similarly to the corresponding unprotected circuit. Hence, the authors have proposed a modified version, i.e., iMDPL [102], which prevents early propagation at both evaluation and precharge phases.

Based on an iMDPL prototype chip [102], it has been shown that CPA attacks using common power models, e.g., HW/HD, face many difficulties before they can succeed. By contrast, we have evaluated the same prototype chip with several other advanced and sophisticated schemes [80].

In summary, we admit that the classical attacks are significantly hardened, but a few other techniques can exploit the leakage. For example, CECA could reveal the linear difference between many key bytes, which indeed indicates a first-order leakage of the iMDPL circuit. As an interesting point, we could perform several attack after preprocessing the traces by integration over a fixed-size window. Even a classical CPA attack by HW model could be successfully mounted after such a preprocessing. One more finding related to this work relates to the detectability of the mask bit. We have observed that, because the mask bit (also in dual-rail form) is routed to all logic gates, it is the longest route and the line consuming the most power (due to its large parasitic capacitance). Also, because no particular attention was paid to the routing of the rails of the mask bit, their capacitances (or their delay and their power) are not necessarily the same. Therefore, with careful observation we could exploit the value of the mask bit at each clock cycle of each power trace. As the last step, the traces are categorized on the basis of the exploited mask bit value, and then a classical CPA attack (by HW model) could be successfully mounted.

The result of our evaluations [80], in fact, predicates the necessity of balancing the dual rails of the mask bit. However, this contradicts the concept of iMDPL, because it targets “dual-rail without routing constraints”.

Chapter 5

Development of Novel Countermeasures

Introduction

In this chapter, we deal with dedicated countermeasures for hardware platforms that range between masking and hiding techniques. The threshold implementation that can solve the problem of masking in hardware is explained first. Afterwards, we concentrate mainly on hiding techniques that have been developed for FPGA devices.

5.1 Threshold Implementation $\textcircled{c1}$ $\textcircled{c2}$

As discussed in previous chapters, realizing masking in hardware is not straightforward and some previous attempts (see above) have failed in practice. Therefore, the research community has invested effort in this direction to provide the schemes that – in the presence of glitches – can provably satisfy the desired level of security. Threshold Implementation (TI) is amongst these approaches and – with a sound theoretical model – it can resist certain side-channel attacks [98].

The idea of TI is based on the algebraic degree of a non-linear function, e.g., an Sbox, whose masked version should be implemented. Let us suppose a function $f : (x_1, \ldots, x_n) \mapsto y$, where each x_i is an input bit. Writing the algebraic normal form (ANF) of $f(\ldots)$ leads to

$$\begin{aligned}
&k_0 \ + \ k_1 \cdot x_1 \ + \ k_2 \cdot x_2 \ + \ \ldots k_n \cdot x_n \ + \\
&k_{1,2} \cdot x_1 x_2 \ + \ k_{1,3} \cdot x_1 x_3 \ + \ \ldots \ + \ k_{1,n} \cdot x_1 x_n \ + \ \ldots \ + \ k_{n-1,n} \cdot x_{n-1} x_n \ + \\
&\ldots \\
&\ldots \\
&k_{1,2,\ldots,n} \cdot x_1 x_2 \ldots x_n.
\end{aligned}$$

The coefficients $k_0, \ldots, k_{1,2,\ldots,n} \in \{0, 1\}$ form the function $f(\ldots)$, and a coefficient $k_{i_1,i_2,\ldots,i_a} \neq 0$ with the largest $a \leq n$ defines its algebraic degree. In other words, a function with the algebraic degree a has terms with, at most, a input bits in its ANF.

If each input bit is represented by s Boolean shares as $x_i = \bigoplus\limits_{j=1}^{s} x_i^j$, replacing each x_i by its corresponding shares in $f(\ldots)$ results in an ANF in which, in most a, shares form a term. Therefore, the number of shares must be $s > a$, to ensure that no term exists in the ANF of the shared $f(\ldots)$ that multiplies all s shares. This is necessary, due to the concept of higher-order attacks explained in Section 2.2.1. Otherwise, the implemented circuit will have a first-order leakage because it already provides leakages that depend on the multiplication of all shares, i.e., the combination of leakages associated with each share is performed by hardware itself.

Now, suppose that the output of the shared function $f(\ldots)$ is split into t component functions $f^1(\ldots),\ f^2(\ldots),\ \ldots,\ f^t(\ldots)$ in such a way that $f(\ldots) = \bigoplus\limits_{j=1}^{t} f^j(\ldots)$. One requirement of TI [98] is that each component function $f^j(\ldots)$ must be independent of at least one share. In other words, all of the terms of the ANF of $f^j(\ldots)$ each must not contain a certain input share. For simplicity, we can suppose that $f^j(\ldots)$ is independent of the input shares $x_1^j,\ x_2^j,\ \ldots,\ x_n^j$. This property is referred to as *non-completeness* [98]. Otherwise, the leakage associated with $f^j(\ldots)$ will depend on all shares, which results, again, in a first-order leakage. Therefore, as a direct consequence, the number of output shares must be $t\ >\ a$.

As explained earlier, in Section 2.2.1, the $n-bit$ masks must be drawn uniformly from $\{0,1\}^n$, unless the low-entropy masking concept has been followed (Section 4.2.2). Therefore, the output shares $f^1(\ldots),\ f^2(\ldots),\ \ldots,\ f^t(\ldots)$ or a subset of them must be uniform if they are given as input to another shared function. A few definitions for the uniformity in this context have been presented in [96, 97, 98]. Below, a general formalization is given:

Suppose that, for a certain input $(\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_n)$, all possible sharing $\Big\{(x_1^1, x_2^1, \ldots, x_n^1),$ $(x_1^2, x_2^2, \ldots, x_n^2),\ \ldots\ (x_1^s, x_2^s, \ldots, x_n^s)\Big\}$, where $\Big(\mathrm{x}_1\ =\ \bigoplus\limits_{j=1}^{s} x_1^j,\ \mathrm{x}_2\ =\ \bigoplus\limits_{j=1}^{s} x_2^j,\ \ldots,\ \mathrm{x_n}\ =\ \bigoplus\limits_{j=1}^{s} x_n^j\Big)$, are given to the component functions $f^1(\ldots),\ f^2(\ldots),\ \ldots,\ f^t(\ldots)$. The tuple $\big(f^1(\ldots),\ f^2(\ldots),$ $\ldots,\ f^t(\ldots)\big)$ should be drawn uniformly from the set $\big\{(y^1, y^2, \ldots, y^t)\big\}$ as the total possible sharing of $y = f(\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_n)$.

This definition is based on a single-bit output function $f(\ldots)$, which can be easily extended to one with higher output bits. In short, each TI representation of an Sbox should fulfill the above-mentioned criteria to provide the first-order resistance. However, the Sboxes (as the non-linear part of the ciphers) usually have a large algebraic degree $>\ 2$, which leads to the use of at least $s\ =\ 4$ input shares. Hence, as a solution, the Sbox $S(.)$ is decomposed into two (or more) functions as $S(x)\ =\ h(g(x))$, where both $h(.)$ and $g(.)$ are surjective. If $S(.)$ is bijective, both $h(.)$ and $g(.)$ are also bijective. The decomposition is performed in such a way that the algebraic degree of each of $h(.)$ and $g(.)$ is less than that of $S(.)$, hence, the TI representation of each of $h(.)$ and $g(.)$ can be made using a lower number of input shares s. However, the glitches of the first stage (shared function $g(.)$) should not be propagated to the next stage. Therefore, a register should be placed between the two stages to buffer the $g(.)$ outputs, i.e, the $h(.)$ inputs.

TI is a way of implementing Boolean masking at algorithmic level in hardware. In fact, the linear parts of each algorithm are easy to share as $l(x)\ =\ l(x^1) \oplus l(x^2) \oplus \ldots l(x^s)$. All the difficulties of this scheme relate to dealing with the non-linear parts, e.g., Sboxes. Therefore, it should be designed and adjusted for each algorithm independently, and the challenging task is to make a representation that fulfills the uniformity.

The authors of [98] have applied the scheme on NOEKEON Sbox [35] and provided correct TI representation. Subsequently, we attempted to realize the PRESENT Sbox under the definitions of this scheme, and realized the first practical (FPGA-based) implementation and side-channel evaluations [104]. At the time this work was undertaken, the generalized evaluation schemes (presented in Section 3.1), e.g., CECA and MCDPA, were not available, and we only performed state-of-the-art CPA attacks with the commonly known HW/HD model. Later, we made use of this implementation and the corresponding measurements to perform suitable analyses using CECA and presented in [74]. In fact, we have practically confirmed the soundness of the theory behind the TI scheme for providing first-order security regardless of the actual as well as the hypothetical power model.

The TI representation of the Sboxes with a low algebraic degree (up to 4), i.e., 3-bit and 4-bit Sboxes, can be achieved with moderate effort, because the original developers of the scheme have presented many useful constructions in [18]. However, for larger Sboxes, it becomes very challenging, e.g., forAES, because it has an algebraic degree of $a = 7$; hence, for a first-order secure implementation, at least $s = 8$ input shares are required. This makes the uniformity check computationally impossible.

In this direction, one can concentrate on the building block of the AES Sbox, because it is formed by a $\mathrm{GF}(2^8)$ inversion followed by an affine transformation. Similar to a linear function, the affine part is easy to share, but the challenge is the inversion part: $\mathrm{Inv}(x) = x^{-1} \bmod \mathrm{Poly}_{\mathrm{AES}}$. It can be written as x^{254} as $x \times x^{254} = 1$. For a simple decomposition, one can write $x^{254} = x^{vw}$, as $v \cdot w \bmod 255 = 254$. Since the algebraic degree of a function $f : x \mapsto x^v$ is identified by the Hamming weight of v, restricting v and w to specific groups with small Hamming weights would help in decomposing the inversion function. Unfortunately no (v, w) with each HW < 3 exist that fulfill this requirement. Further, the mapping $f : x \mapsto x^v$ does not form a bijection $\forall v$; $\mathrm{HW}(v) = 2$. However, eight candidates exist when the HW of both v and w is 3. The candidates are $(13, 98)$, $(19, 161)$, $(26, 49)$, $(38, 208)$, $(52, 152)$, $(67, 137)$, $(76, 104)$, and $(134, 196)$.

Therefore, it is possible to share each decomposed part with a minimum of 4 input shares, but this technique has not been implemented, to date. Instead, we have focused on a tower field representation of the AES Sbox in [90], and provided the TI representation of each sub-module. Further, we had to insert several registers between the TI sub-modules to avoid the propagation of the glitches and realize a pipeline architecture. More importantly, we have observed that the input of the TI sub-modules does not follow a uniform distribution; hence , we made use of the *re-masking* technique which employs random *fresh* masks to maintain the uniformity. In short, the design presented in [90] is not a straightforward TI representation of the AES and, with several tricks, we could provide an architecture to maintain the first-order security. A disadvantage of our design is its large number of required fresh mask bits per clock cycle, which – depending on the target platform and the underlying application – might be problematic. Along the same lines, compared to our design, a more efficient implementation of the AES with the same technique has been presented in [16] that reduces the number of required fresh masks and needs less area for implementation. The reduced area overhead is mainly due to switching the number of shares, i.e., the linear parts of the cipher (MixColumns) are shared by two shares and, at the start of the non-linear parts (by means of fresh masks), the shares are extended to three or more. On one hand, this reduces the area overhead, and on the other hand, it makes a second-order attack easier because the non-linear parts are only shared by two shares.

It is noteworthy that a more general formalization of TI has recently been presented in [17] that extends the concept to provide resistance against higher-order attacks. Although the number of required input shares increases extremely for the desired higher orders, it is known as the only available theoretically-sound approach for hardware platforms to provide higher-order resistance.

5.2 FPGA-Dedicated Countermeasures c3 c4

As stated above, different techniques in the direction of hardening side-channel analysis attacks exist. Each of them is usually suitable for a certain implementation platform. For instance, shuffling, which randomizes the order of the operations without priority, is commonly considered in software applications [56], due to its sequential execution nature. Shuffling can also be applied in hardware if the operations are also executed sequentially, e.g., [86], where a serialized architecture that performs SubBytes operation in 16 clock cycles is presented.

The question still remains about how to generate noise with high impact on power consumption in order to hide the data-dependent leakages. Such techniques are usually considered in hardware applications and cannot be easily embedded into software platforms by the end user. In [51], we tried to make use of the components available in Xilinx FPGAs for side-channel protection. In the category of hiding (with a focus on noise generation), we provided three different techniques, all of which are driven by a (pseudo-)random number generator (PRNG):

- Certain Look-Up-Tables (LUTs) can be configured as shift registers. We employed such LUTs and formed rotating registers initially filled by `1010...1010`. If the rotating register is clocked, all bits toggle once, hence consuming the largest amount of power solely due to the register bit flips. Because the registers are clocked at each clock cycle but enabled by the PRNG, we could amplify the leakage related to the output of each embedded PRNG.

- The routings are realized by switch boxes in FPGAs. A switch box is a complex circuit that provides the determined and selectable possibilities of connecting a certain input signal to dedicated output(s). By configuring the switch boxes, each of which is close to one CLB[1], routing between the LUTs, FFs, and slices is realized. We followed the idea presented in [13] to make short circuits by misconfiguring the switch boxes, which should be done by low-level tools (at XDL[2] level). Hence, by connecting the PRNGs to the misconfigured switch boxes, we could again amplify the leakage of the PRNG outputs.

- The Xilinx FPGAs contain several Block-RAMs (BRAM) with, usually, 18 KBit or 36 KBit capacity. Each BRAM is equipped with two independent ports, through each of which the entire memory content can be accessed for both read and write. It was shown previously, in [50], that writing to the same location by different ports – so-called write collision – can lead to semi-random but device-dependent results. Hence, we have also used this feature of BRAMs to write randomly chosen data (provided by PRNG) at the same location, thereby consuming a random amount of energy, due to the unpredictable result of write collisions.

Our practical analysis in [51] shows that the noise generators made by 1) 16 rotating registers each with 576 bits, 2) 16 short circuit misconfigured switch boxes and 3) 16 BRAMs for write collision, provide approximately the same level of noise, because they increase the data complexity of an attack by a factor of approximately 3.

One more option in the category of hiding is to misalign the traces, hence hardening the attacks. The Xilinx FPGAs contain several Digital Clock Management (DCM) units and clock

[1] Configurable Logic Block: usually containing two slices in xilinx modern families. Each slice consists of four LUT6 and at least four registers.

[2] Xilinx Design Language: as a human-readable netlist of a design right before generation of the bitstream.

multiplexers. Therefore, we employed the DCMs to generate multiples of the incoming clock as well as of the phase-shifted ones. Also, using a few clock multiplexers controlled by a PRNG, we could derive a randomized clock to drive the crypto core, thereby misaligning the traces and significantly hardening the attacks.

By masking in hardware, which has extensively been discussed in Sectoin 4.2, we employed the BRAMs, whose use can remarkably reduce the area overhead with respect to the number of occupied LUTs and slices. We realized the same concept as Boolean masking in software. In other words, the BRAMs are used to save the masked Sboxes. However, the masks should be changed periodically; hence it is necessary to update the BRAM contents. We – one more time – used the dual-port feature of the BRAMs as well as their large capacity. In the developed technique, two copies of the Sbox are initially stored in a BRAM. The first port is used to read the Sbox output from a half of the BRAM, and the second port to update the second half, using new masks. Once the update process is completed, a context switch occurs, i.e., the second – updated – half is used for the encryption/decryption (always using the first port) and the first half is updated using the next new masks. Such a construction allows for implementing the masked (randomized) tables in hardware and it benefits from the high efficiency of a hardware platform. It is noteworthy that, in such a scenario – depending on the number of given e.g., plaintexts to be encrypted – a few consecutive encryptions may share the same mask. Although this can be avoided by introducing wait cycles between the consecutive encryptions, to ensure the termination of the update process, it has a significant effect on the throughput, which moves it slightly away from the main feature of the hardware platforms, i.e., efficiency.

5.2.1 Dual-Rail Precharge Logic

As explained in Section 2.2 and Section 4.2.4, a category of hiding countermeasures that aim to equalize the power consumption is known as DPA-resistant logic styles. Such schemes usually target an ASIC platform, because 1) specific components (designed to realize the functional gates) do not exist amongst the FPGA components, and 2) routing is not easily controlled in FPGAs. Nevertheless, a few works have tried to realize the dual-rail precharge concept in FPGA. Most of them are based on the *duplication* concept, in which a counterpart of the circuit is – usually vertically – copied to another location that can retain the structure and routing of the original part. The seminal work here is [126] that introduces Double WDDL (DWDDL). It duplicates a fully placed-and-routed WDDL circuit, resulting in a massive resource utilization. For further similar attempts, see [53, 54, 60].

On the other hand, realizing the circuits with a kind of logic style in which the `true` and `false` networks are not separated, is more challenging. In this field, we refer to [93] in which a logic style called Balanced Cell-based Dual-rail Logic (BCDL) by employing a global precharge signal is introduced. Such approaches have a drawback because all the components in a circuit simultaneously go to the precharge phase, resulting in a high power consumption peak [120]. Another related work is [15] that proposes a WDDL-like scheme without early propagation, that is called DPL-noEE. Recalling the early propagation concept, initially addressed in [117] and later discussed in [102], DPL-noEE can prevent the early propagation in the evaluation phase, but nothing is considered to deal with the start of the precharge phase. Moreover, no restrictions have been put into the placement and routing processes, and the imbalances of the dual rails have been ignored.

In [79], we developed a similar style (so-called AWDDL) that can prevent the early propagation in both evaluation and precharge phases. Our construction is based on forming an S-R latch by means of a feedback to the LUT, which keeps the evaluated output until all input signals are in precharge. Indeed, it is the first known style in FPGAs that can prevent early propagation in both phases.

Further, to mitigate the routing imbalances, we were the first to develop a customized router to find the best match with minimal routing differences for a design. Our tool is based on the report of the Xilinx FPGA Editor on the delay of the routes. In short, the steps listed below were followed:

- The circuit (expressed by an HDL code) is placed and routed automatically by Xilinx tools with a specific constraint to use a certain part of the target FPGA for the placement of the module, e.g., the encryption, which is implemented by our developed AWDDL.
- The generated NCD file is converted to XDL and the routings of the target module are removed.
- All possible single routings from a source node to a destination node are generated using RapidSmith [63].
- The delay of each possible routing is extracted by Xilinx FPGA Editor.
- A set of equations based on all possible routings and all necessary connections of the target module forms a Boolean satisfiability (SAT) problem.
- A solution for the routing is achieved by means of a SAT solver, e.g., CryptoMiniSat [1], and a set of restrictions on the maximum delay differences.
- The selected routings are set on the target module with the help of RapidSmith, and the final bitstream is generated by Xilinx development tools.

Our practical investigations show a remarkable reduction of the amount of exploitable leakage when the target module is realized by AWDDL. However, the leakage cannot be completely avoided. Reasons for such an imperfection are manyfold:

(1) Routings for all dual rails with null delay difference do not necessarily exist.

(2) If such routings exist, the selection of all of them without conflict is not necessarily possible.

(3) The customized router is based on the delay report provided by Xilinx FPGA Editor. This information is based on the worst-case simulation and may differ from reality. Such a difference may also be due to the process variation that affect the routing capacitances and, hence, their delay.

Chapter 6
Summary and Outlook

This habilitation thesis dealt with development, evaluation and investigation of novel side-channel-based cryptanalysis schemes as well as the countermeasures. For most of the attacks and countermeasures presented in this thesis, the target was a hardware implementation of a symmetric block cipher. In particular, the focus was mainly on FPGA-based platforms, either to examine the efficiency of the attacks or to realize the countermeasures and evaluate their robustness.

In this thesis, new analysis schemes have been introduced, each of which targets certain goals. By concentrating on a specific statistical moment, generic attacks – with respect to relaxing the necessity of a hypothetical model – have been presented. Amongst the advantages of the schemes presented here, we highlight their metric feature that can be applied at certain orders, thereby quantifying the amount of information available through each statistical moment separately. The presented schemes do this quantization with respect to the data complexity of an attack, i.e., the number of measurements required for a successful attack. For future work in this area, it would be beneficial to do such a quantization from an information theoretic point of view and identify the moments with the highest leakage. Furthermore, combining certain moments, e.g., the second and the third ones, for such a quantization is not possible with currently available tools.

Another direction that we have focused on was to make use of other side channels that have not been extensively studied, to date. In this area, we refer to the generic attacks we introduced to deal with the timing information of combinatorial circuits, which needs to be measured precisely e.g., by accurate clock glitches (in the range of ps). Such an attack (and its initial form, known as fault sensitivity analysis) highlighted the inevitability of corresponding countermeasures, e.g., a clock glitch detector [37]. In addition, a roadmap toward a sound evaluation of static power consumption as a side channel has been presented for the first time. Although the result of such an attempt is still considered as preliminary, it provides clear evidence that such attacks are possible in practice. Of course, if the currently-available measurement setup and facilities, which are very much optimized for dynamic power measurements, are used, the attacks using static power side channel are not as efficient as their dynamic counterparts. On the other hand, static power opens new doors for the analysis of schemes that provide a high level of resistance against dynamic power attacks. In this regard, we refer to masking schemes that aim at providing resistance against any univariate attacks. A further example is the masking schemes combined with noise addition to harden a higher-order attack. In such a case, static power can better focus on the leakage associated with the processed masked data and mitigate the effect of the noise generator. Hence, developing a suitable measurement setup dedicated to static power measurements, as well as their corresponding analysis schemes to better understand the concept of static power analysis are amongst the plans for the future. Examining the ability of currently known countermeasures to resist against fine-tuned static power analysis should also be considered for further activities in this area.

In the areas of bitstream encryption of FPGA devices, some side-channel attacks on various Xilinx and Altera FPGAs have been presented. It is now well known that the decryption modules deployed by Xilinx and Altera are not equipped with side-channel protection. However, some of our attacks had to search in a space of 2^{32}, which makes the key-recovery process computationally hard. In future work, we plan to reduce this complexity by feeding chosen ciphertexts to the decryption module, thereby limiting the unknown parts to be guessed. This scenario may have a negative impact on the measurement speed because, for each chosen ciphertext, the configuration process should be restarted.

A major part of this thesis dealt with the practical evaluation of – mostly – the masking schemes that have been proposed to the side-channel community. We have shown that – in most cases – the schemes that are theoretically sound fail in practice when they are employed in a hardware platform. Indeed, very few masking schemes exist whose claimed security can be confirmed in practice when considering a hardware implementation. As one of the conclusions in this area, we refer to the simplicity of overcoming the univariate masking schemes that process the shares in a sequential manner. As shown, with a small modification of the measurement setup, multivariate leakages can be turned to a univariate leakage exploitable by a univariate attack. In this direction, one of the future plans is to develop a measurement setup to, e.g., square the power signals in the analogue domain. This is feasible by means of analogue voltage multipliers that are common components in the areas of analogue amplification and filtering. With such a measurement setup, it will be possible to perform first-order attacks on the devices that realize a perfect first-order masking. Here, investigating the efficiency of such a scenario, compared to the commonly-known preprocessing of the traces in a discrete domain, is of crucial interest.

Along the same lines, from the constructions and evaluations we performed on threshold implementation, it can be concluded that it is, to date, the only promising scheme that has a reasonable overhead and is able to maintain the claimed security when implemented either on an ASIC or an FPGA platform. Obviously it is difficult to adjust it for every algorithm and, in some cases, it is not straightforward (or even impossible) to find a correct representation of the target function. For instance, realizing the AES Sbox in such a way that it fulfills all the requirements of the scheme is still unattainable. Hence, as explained in Section 5.1, there is a way to reduce the minimum number of input shares as well as to decompose the inversion part of the AES Sbox. Yet another path for a future work is to proceed in this direction in the hope of finding a uniform shared representation of the decomposed inversion function, thereby avoiding the several fresh random masks required for the currentlyknown threshold implementations of the AES Sbox. Further, more development with respect to higher-order resistance in hardware [17] is expected.

Last but not least, we developed several countermeasures dedicated for FPGAs covering a range from hiding (as noise generator, misalignment facilities, dual-rail precharge logic) up to masking (as randomized tables stored in BRAMs). According to our developmental and practical investigations – to the best of our knowledge – our solution to realize Boolean masking with periodicallyupdated BRAMs is a (first-order) secure and highly (area) efficient scheme for FPGAs. Our activities in the areas of dual-rail precharge logic made it clear that completely avoiding early propagation in both phases is possible. We also established that, even when employing a customized router, obtaining identical routes with the same delay for dual rails seems impossible. Further, the information available for a customized router has been developed from worst-case simulations (by the FPGA vendor) and is, hence, very conceptual. As a future plan, we will proceed in this direction by developing techniques to practically measure the difference between the delays of a dual rail. The result of such an experiment is expected to demonstrate a dependency on the process variation. If such a statement is correct, it will probably be impossible to achieve the same delay for even the routes shown to be identical, in terms of delay and shape, in FPGA vendor design tools. Under such conditions, power equalizing techniques (like dual-rail logics) in FPGAs will not have a bright future.

Bibliography

[1] CryptoMiniSat. Available as download here `https://gforge.inria.fr/frs/?group_id=1992`.

[2] ISO/IEC 18033-3 Standard Cryptographic LSI – with Side Channel Attack Countermeasures – Specification, ver 1.0. `http://satoh.cs.uec.ac.jp/SASEBO/resources/crypto_lsi/CryptoLSI2_Spec_Ver1.0_English.pdf`.

[3] Side-channel Attack Standard Evaluation Board (SASEBO). `http://satoh.cs.uec.ac.jp/SASEBO/en/`.

[4] Side-channel AttacK User Reference Architecture. `http://satoh.cs.uec.ac.jp/SAKURA/index.html`.

[5] Standard Cryptographic LSI Specification – Countermeasures against Side Channel Attacks (65nm) – Specification, ver 0.9. `http://satoh.cs.uec.ac.jp/SASEBO/resources/crypto_lsi/CryptoLSI3_Spec_Ver0.9_English.pdf`.

[6] Dakshi Agrawal, Bruce Archambeault, Josyula R. Rao, and Pankaj Rohatgi. The EM Side-Channel(s). In *CHES 2002*, volume 2523 of *LNCS*, pages 29–45. Springer, 2002.

[7] Mehdi-Laurent Akkar, Régis Bevan, and Louis Goubin. Two Power Analysis Attacks against One-Mask Methods. In *FSE 2004*, volume 3017 of *LNCS*, pages 332–347. Springer, 2004.

[8] M. Alioto, L. Giancane, G. Scotti, and A. Trifiletti. Leakage Power Analysis attacks: Well-defined procedure and first experimental results. In *Microelectronics 2009*, pages 46–49. IEEE, 2009.

[9] Massimo Alioto, Luca Giancane, Giuseppe Scotti, and Alessandro Trifiletti. Leakage Power Analysis Attacks: A Novel Class of Attacks to Nanometer Cryptographic Circuits. *IEEE Trans. on Circuits and Systems*, 57-I(2):355–367, 2010.

[10] Elad Barkan and Eli Biham. In How Many Ways Can You Write Rijndael? In *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 160–175. Springer, 2002.

[11] Elad Barkan and Eli Biham. The Book of Rijndaels. Cryptology ePrint Archive, Report 2002/158, 2002. `http://eprint.iacr.org/`.

[12] Lejla Batina, Benedikt Gierlichs, Emmanuel Prouff, Matthieu Rivain, François-Xavier Standaert, and Nicolas Veyrat-Charvillon. Mutual Information Analysis: a Comprehensive Study. *J. Cryptology*, 24(2):269–291, 2011.

[13] Christian Beckhoff, Dirk Koch, and Jim Torresen. Short-Circuits on FPGAs Caused by Partial Runtime Reconfiguration. In *FPL 2010*, pages 596–601. IEEE, 2010.

[14] Shivam Bhasin, Claude Carlet, and Sylvain Guilley. Theory of masking with codewords in hardware: low-weight *d*th-order correlation-immune Boolean functions. Cryptology ePrint Archive, Report 2013/303, 2013. `http://eprint.iacr.org/2013/303/`.

[15] Shivam Bhasin, Sylvain Guilley, Florent Flament, Nidhal Selmane, and Jean-Luc Danger. Countering early evaluation: an approach towards robust dual-rail precharge logic. In *WESS 2010*, page 6. ACM, 2010.

[16] Begül Bilgin, Benedikt Gierlichs, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. A More Efficient AES Threshold Implementation. In *AFRICACRYPT - 2014*, volume 8469 of *LNCS*, pages 267–284. Springer, 2014.

[17] Begül Bilgin, Benedikt Gierlichs, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. Higher-Order Threshold Implementations. In *ASIACRYPT 2014*, volume 8874 of *LNCS*, pages 326–343. Springer, 2014.

[18] Begül Bilgin, Svetla Nikova, Ventzislav Nikov, Vincent Rijmen, and Georg Stütz. Threshold Implementations of All 3 ×3 and 4 ×4 S-Boxes. In *CHES 2012*, volume 7428 of *LNCS*, pages 76–91. Springer, 2012.

[19] Andrey Bogdanov. Improved Side-Channel Collision Attacks on AES. In *SAC 2007*, volume 4876 of *LNCS*, pages 84–95. Springer, 2007.

[20] Andrey Bogdanov. Multiple-Differential Side-Channel Collision Attacks on AES. In *CHES 2008*, volume 5154 of *LNCS*, pages 30–44. Springer, 2008.

[21] Andrey Bogdanov and Ilya Kizhvatov. Beyond the Limits of DPA: Combined Side-Channel Collision Attacks. *IEEE Trans. Computers*, 61(8):1153–1164, 2012.

[22] Andrey Bogdanov, Ilya Kizhvatov, and Andrei Pyshkin. Algebraic Methods in Side-Channel Collision Attacks and Practical Collision Detection. In *INDOCRYPT 2088*, volume 5365 of *LNCS*, pages 251–265. Springer, 2008.

[23] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In *CHES 2007*, volume 4727 of *LNCS*, pages 450–466. Springer, 2007.

[24] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the Importance of Checking Cryptographic Protocols for Faults (Extended Abstract). In *EUROCRYPT 1997*, volume 1233 of *LNCS*, pages 37–51. Springer, 1997.

[25] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation Power Analysis with a Leakage Model. In *CHES 2004*, volume 3156 of *LNCS*, pages 16–29. Springer, 2004.

[26] Julien Bringer, Hervé Chabanne, and Thanh Ha Le. Protecting AES against side-channel analysis using wire-tap codes. *J. Cryptographic Engineering*, 2(2):129–141, 2012.

[27] D. Canright and Lejla Batina. A Very Compact "Perfectly Masked" S-Box for AES. In *ACNS 2008*, volume 5037 of *LNCS*, pages 446–459. Springer, 2008.

[28] David Canright. A Very Compact S-Box for AES. In *CHES 2005*, volume 3659 of *LNCS*, pages 441–455. Springer, 2005.

[29] Sung-Hyuk Cha. Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4):300–307, 2007.

[30] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards Sound Approaches to Counteract Power-Analysis Attacks. In *CRYPTO 1999*, volume 1666 of *LNCS*, pages 398–412. Springer, 1999.

[31] Zhimin Chen and Yujie Zhou. Dual-Rail Random Switching Logic: A Countermeasure to Reduce Side Channel Leakage. In *CHES 2006*, volume 4249 of *LNCS*, pages 242–254. Springer, 2006.

[32] Christophe Clavier, Benoit Feix, Georges Gagnerot, Mylène Roussellet, and Vincent Verneuil. Improved Collision-Correlation Power Analysis on First Order Protected AES. In *CHES 2011*, volume 6917 of *LNCS*, pages 49–62. Springer, 2011.

[33] Jean-Sébastien Coron and Louis Goubin. On Boolean and Arithmetic Masking against Differential Power Analysis. In *CHES 2000*, volume 1965 of *LNCS*, pages 231–237. Springer, 2000.

[34] Nicolas Courtois and Louis Goubin. An Algebraic Masking Method to Protect AES Against Power Attacks. In *ICISC 2005*, volume 3935 of *LNCS*, pages 199–209. Springer, 2005.

[35] Joan Daemen, Michaël Peeters, Gilles Van Assche, and Vincent Rijmen. Nessie proposal: NOEKEON. Submitted as an NESSIE Candidate Algorithm, 2000. `http://www.cryptonessie.org`.

[36] Thomas Eisenbarth, Timo Kasper, Amir Moradi, Christof Paar, Mahmoud Salmasizadeh, and Mohammad T. Manzuri Shalmani. On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoq Code Hopping Scheme. In *CRYPTO 2008*, volume 5157 of *LNCS*, pages 203–220. Springer, 2008.

[37] S. Endo, Y. Li, N. Homma, K. Sakiyama, K. Ohta, D. Fujimoto, M. Nagata, T. Katashita, J. Danger, and T. Aoki. A Silicon-Level Countermeasure Against Fault Sensitivity Analysis and Its Evaluation. *IEEE Trans. VLSI Syst.*, 23(8):1429–1438, 2015.

[38] Julie Ferrigno and Martin Hlavác. When AES blinks: introducing optical side channel. *IET Information Security*, 2(3):94–98, 2008.

[39] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. Electromagnetic Analysis: Concrete Results. In *CHES 2001*, volume 2162 of *LNCS*, pages 251–261. Springer, 2001.

[40] Laurie Genelle, Emmanuel Prouff, and Michaël Quisquater. Thwarting Higher-Order Side Channel Analysis with Additive and Multiplicative Maskings. In *CHES 2011*, volume 6917 of *LNCS*, pages 240–255. Springer, 2011.

[41] Daniel Genkin, Adi Shamir, and Eran Tromer. RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis. Cryptology ePrint Archive, Report 2013/857, 2013. `http://eprint.iacr.org/`.

[42] Benoît Gérard and François-Xavier Standaert. Unified and Optimized Linear Collision Attacks and Their Application in a Non-profiled Setting. In *CHES 2012*, volume 7428 of *LNCS*, pages 175–192. Springer, 2012.

[43] Felipe Ghellar and Marcelo Lubaszewski. A novel AES cryptographic core highly resistant to differential power analysis attacks. In *Integrated Circuits and Systems Design - SBCCI 2008*, pages 140–145. ACM, 2008.

[44] Felipe Ghellar and Marcelo Lubaszewski. A novel AES cryptographic core highly resistant to differential power analysis attacks. *Journal Integrated Circuits and Systems*, 4(1):29–35, 2009.

[45] Benedikt Gierlichs, Lejla Batina, Pim Tuyls, and Bart Preneel. Mutual Information Analysis – A Generic Side-Channel Distinguisher. In *CHES 2008*, volume 5154 of *LNCS*, pages 426–442. Springer, 2008.

[46] Jacopo Giorgetti, Giuseppe Scotti, Andrea Simonetti, and Alessandro Trifiletti. Analysis of data dependence of leakage current in CMOS cryptographic hardware. In *ACM Great Lakes Symposium on VLSI*, pages 78–83. ACM, 2007.

[47] Jovan Dj. Golic and Christophe Tymen. Multiplicative Masking and Power Analysis of AES. In *CHES 2002*, volume 2523 of *LNCS*, pages 198–212. Springer, 2002.

[48] G. Goodwill, B. Jun, J. Jaffe, and P. Rohatgi. A testing methodology for side channel resistance validation. In *NIST non-invasive attack testing workshop*, 2011. `http://csrc.nist.gov/news_events/non-invasive-attack-testing-workshop/papers/08_Goodwill.pdf`.

[49] Vincent Grosso, François-Xavier Standaert, and Emmanuel Prouff. Low Entropy Masking Schemes, Revisited. In *CARDIS 2013*, volume 8419 of *LNCS*, pages 33–43. Springer, 2013.

[50] Tim Güneysu. Using Data Contention in Dual-ported Memories for Security Applications. *Signal Processing Systems*, 67(1):15–29, 2012.

[51] Tim Güneysu and Amir Moradi. Generic Side-Channel Countermeasures for Reconfigurable Devices. In *CHES 2011*, volume 6917 of *LNCS*, pages 33–48. Springer, 2011.

[52] Xiaofei Guo and Ramesh Karri. Invariance-based concurrent error detection for advanced encryption standard. In *DAC 2012*, pages 573–578. ACM, 2012.

[53] Wei He, Eduardo de la Torre, and Teresa Riesgo. A Precharge-Absorbed DPL Logic for Reducing Early Propagation Effects on FPGA Implementations. In *ReConFig 2011*, pages 217–222. IEEE Computer Society, 2011.

[54] Wei He, Andrés Otero, Eduardo de la Torre, and Teresa Riesgo. Automatic generation of identical routing pairs for FPGA implemented DPL logic. In *ReConFig 2012*, pages 1–6. IEEE Computer Society, 2012.

[55] Christoph Herbst, Elisabeth Oswald, and Stefan Mangard. An AES Smart Card Implementation Resistant to Power Analysis Attacks. In *ACNS 2006*, volume 3989 of *LNCS*, pages 239–252. Springer, 2006.

[56] Christoph Herbst, Elisabeth Oswald, and Stefan Mangard. An AES Smart Card Implementation Resistant to Power Analysis Attacks. In *ACNS 2006*, volume 3989 of *LNCS*, pages 239–252. Springer, 2006.

[57] Michael Hutter and Jörn-Marc Schmidt. The Temperature Side Channel and Heating Fault Attacks. In *CARDIS 2013*, volume 8419 of *LNCS*, pages 219–235. Springer, 2013.

[58] Ming-Haw Jing, Jian-Hong Chen, Zih-Heng Chen, and Yaotsu Chang. The Secure DAES Design for Embedded System Application. In *EUC Workshops 2007*, volume 4809 of *LNCS*, pages 617–626. Springer, 2007.

[59] Ming-Haw Jing, Zih-Heng Chen, Jian-Hong Chen, and Yan-Haw Chen. Reconfigurable system for high-speed and diversified AES using FPGA. *Microprocessors and Microsystems*, 31(2):94–102, 2007.

[60] Jens-Peter Kaps and Rajesh Velegalati. DPA Resistant AES on FPGA Using Partial DDL. In *FCCM 2010*, pages 273–280. IEEE Computer Society, 2010.

[61] Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *CRYPTO 1996*, volume 1109 of *LNCS*, pages 104–113. Springer, 1996.

[62] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In *CRYPTO 1999*, LNCS, pages 388–397. Springer, 1999.

[63] Christopher Lavin, Marc Padilla, Philip Lundrigan, Brent E. Nelson, and Brad L. Hutchings. Rapid prototyping tools for FPGA designs: RapidSmith. In *FPT 2010*, pages 353–356. IEEE, 2010.

[64] Yang Li, Kazuo Sakiyama, Shigeto Gomisawa, Toshinori Fukunaga, Junko Takahashi, and Kazuo Ohta. Fault Sensitivity Analysis. In *CHES 2010*, volume 6225 of *LNCS*, pages 320–334. Springer, 2010.

[65] Lang Lin and Wayne Burleson. Leakage-based differential power analysis (LDPA) on sub-90nm CMOS cryptosystems. In *ISCAS 2008*, pages 252–255. IEEE, 2008.

[66] Houssem Maghrebi, Sylvain Guilley, and Jean-Luc Danger. Leakage Squeezing Countermeasure against High-Order Attacks. In *WISTP 2011*, volume 6633 of *LNCS*, pages 208–223. Springer, 2011.

[67] Houssem Maghrebi, Emmanuel Prouff, Sylvain Guilley, and Jean-Luc Danger. A First-Order Leak-Free Masking Countermeasure. In *CT-RSA 2012*, volume 7178 of *LNCS*, pages 156–170. Springer, 2012.

[68] Stefan Mangard. Hardware Countermeasures against DPA ? A Statistical Analysis of Their Effectiveness. In *CT-RSA 2004*, volume 2964 of *LNCS*, pages 222–235. Springer, 2004.

[69] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, 2007.

[70] Stefan Mangard, Norbert Pramstaller, and Elisabeth Oswald. Successfully Attacking Masked AES Hardware Implementations. In *CHES 2005*, volume 3659 of *LNCS*, pages 157–171. Springer, 2005.

[71] Stefan Mangard and Kai Schramm. Pinpointing the Side-Channel Leakage of Masked AES Hardware Implementations. In *CHES 2006*, volume 4249 of *LNCS*, pages 76–90. Springer, 2006.

[72] Thomas S. Messerges. *Power Analysis Attacks and Countermeasures for Cryptographic Algorithms*. PhD thesis, University of Illinois at Chicago, USA, 2000. 468 pages.

[73] Oliver Mischke, Amir Moradi, and Tim Güneysu. Fault Sensitivity Analysis Meets Zero-Value Attack. In *FDTC 2014*, pages 59–67. IEEE Computer Society, 2014.

[74] Amir Moradi. Statistical Tools Flavor Side-Channel Collision Attacks. In *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 428–445. Springer, 2012.

[75] Amir Moradi. Side-Channel Leakage through Static Power - Should We Care about in Practice? In *CHES 2014*, volume 8731 of *LNCS*, pages 562–579. Springer, 2014.

[76] Amir Moradi. Wire-Tap Codes as Side-Channel Countermeasure - An FPGA-Based Experiment. In *INDOCRYPT 2014*, volume 8885 of *LNCS*, pages 341–359. Springer, 2014.

[77] Amir Moradi, Alessandro Barenghi, Timo Kasper, and Christof Paar. On the vulnerability of FPGA bitstream encryption against power analysis attacks: Extracting keys from Xilinx Virtex-II FPGAs. In *ACM CCS'11*, pages 111–124. ACM, 2011.

[78] Amir Moradi, Sylvain Guilley, and Annelie Heuser. Detecting Hidden Leakages. In *ACNS 2014*, volume 8479 of *LNCS*, pages 324–342. Springer, 2014.

[79] Amir Moradi and Vincent Immler. Early Propagation and Imbalanced Routing, How to Diminish in FPGAs. In *CHES 2013*, volume 8731 of *LNCS*, pages 598–615. Springer, 2013.

[80] Amir Moradi, Mario Kirschbaum, Thomas Eisenbarth, and Christof Paar. Masked Dual-Rail Precharge Logic Encounters State-of-the-Art Power Analysis Methods. *IEEE Trans. VLSI Syst.*, 20(9):1578–1589, 2012.

[81] Amir Moradi and Oliver Mischke. How Far Should Theory Be from Practice? - Evaluation of a Countermeasure. In *CHES 2012*, volume 7428 of *LNCS*, pages 92–106. Springer, 2012.

[82] Amir Moradi and Oliver Mischke. Comprehensive Evaluation of AES Dual Ciphers as a Side-Channel Countermeasure. In *ICICS 2013*, volume 8233 of *LNCS*, pages 245–258. Springer, 2013.

[83] Amir Moradi and Oliver Mischke. On the Simplicity of Converting Leakages from Multivariate to Univariate - (Case Study of a Glitch-Resistant Masking Scheme). In *CHES 2013*, volume 8086 of *LNCS*, pages 1–20. Springer, 2013.

[84] Amir Moradi, Oliver Mischke, and Thomas Eisenbarth. Correlation-Enhanced Power Analysis Collision Attack. In *CHES 2010*, volume 6225 of *LNCS*, pages 125–139. Springer, 2010.

[85] Amir Moradi, Oliver Mischke, and Christof Paar. Practical evaluation of DPA countermeasures on reconfigurable hardware. In *HOST 2011*, pages 154–160. IEEE Computer Society, 2011.

[86] Amir Moradi, Oliver Mischke, and Christof Paar. Practical evaluation of DPA countermeasures on reconfigurable hardware. In *HOST 2011*, pages 154–160. IEEE Computer Society, 2011.

[87] Amir Moradi, Oliver Mischke, and Christof Paar. One Attack to Rule Them All: Collision Timing Attack versus 42 AES ASIC Cores. *IEEE Trans. Computers*, 62(9):1786–1798, 2013.

[88] Amir Moradi, Oliver Mischke, Christof Paar, Yang Li, Kazuo Ohta, and Kazuo Sakiyama. On the Power of Fault Sensitivity Analysis and Collision Side-Channel Attacks in a Combined Setting. In *CHES 2011*, volume 6917 of *LNCS*, pages 292–311. Springer, 2011.

[89] Amir Moradi, David Oswald, Christof Paar, and Pawel Swierczynski. Side-channel attacks on the bitstream encryption mechanism of Altera Stratix II: facilitating black-box analysis using software reverse-engineering. In *FPGA 2013*, pages 91–100. ACM, 2013.

[90] Amir Moradi, Axel Poschmann, San Ling, Christof Paar, and Huaxiong Wang. Pushing the Limits: A Very Compact and a Threshold Implementation of AES. In *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 69–88. Springer, 2011.

[91] Amir Moradi and François-Xavier Standaert. Moments-Correlating DPA. Cryptology ePrint Archive, Report 2014/409, 2014. `http://eprint.iacr.org/`.

[92] Sumio Morioka and Akashi Satoh. An Optimized S-Box Circuit Architecture for Low Power AES Design. In *CHES 2002*, volume 2523 of *LNCS*, pages 172–186. Springer, 2002.

[93] Maxime Nassar, Shivam Bhasin, Jean-Luc Danger, Guillaume Duc, and Sylvain Guilley. BCDL: A high speed balanced DPL for FPGA with global precharge and no early evaluation. In *DATE 2010*, pages 849–854. IEEE Computer Society, 2010.

[94] Maxime Nassar, Sylvain Guilley, and Jean-Luc Danger. Formal Analysis of the Entropy / Security Trade-off in First-Order Masking Countermeasures against Side-Channel Attacks. In *INDOCRYPT 2011*, volume 7107 of *LNCS*, pages 22–39. Springer, 2011.

[95] Maxime Nassar, Youssef Souissi, Sylvain Guilley, and Jean-Luc Danger. RSM: A small and fast countermeasure for AES, secure against 1st and 2nd-order zero-offset SCAs. In *DATE 2012*, pages 1173–1178. IEEE, 2012.

[96] Svetla Nikova, Christian Rechberger, and Vincent Rijmen. Threshold Implementations Against Side-Channel Attacks and Glitches. In *ICICS - 2006*, volume 4307 of *LNCS*, pages 529–545. Springer, 2006.

[97] Svetla Nikova, Vincent Rijmen, and Martin Schläffer. Secure Hardware Implementation of Non-linear Functions in the Presence of Glitches. In *ICISC - 2008*, volume 5461 of *LNCS*, pages 218–234. Springer, 2008.

[98] Svetla Nikova, Vincent Rijmen, and Martin Schläffer. Secure Hardware Implementation of Nonlinear Functions in the Presence of Glitches. *J. Cryptology*, 24(2):292–321, 2011.

[99] Elisabeth Oswald, Stefan Mangard, Norbert Pramstaller, and Vincent Rijmen. A Side-Channel Analysis Resistant Description of the AES S-Box. In *FSE 2005*, volume 3557 of *LNCS*, pages 413–423. Springer, 2005.

[100] Elisabeth Oswald and Kai Schramm. An Efficient Masking Scheme for AES Software Implementations. In *WISA 2005*, volume 3786 of *LNCS*, pages 292–305. Springer, 2005.

[101] Jing Pan, J. I. den Hartog, and Jiqiang Lu. You Cannot Hide behind the Mask: Power Analysis on a Provably Secure *S*-Box Implementation. In *WISA 2009*, volume 5932 of *LNCS*, pages 178–192. Springer, 2009.

[102] Thomas Popp, Mario Kirschbaum, Thomas Zefferer, and Stefan Mangard. Evaluation of the Masked Logic Style MDPL on a Prototype Chip. In *CHES 2007*, volume 4727 of *LNCS*, pages 81–94. Springer, 2007.

[103] Thomas Popp and Stefan Mangard. Masked Dual-Rail Pre-charge Logic: DPA-Resistance Without Routing Constraints. In *CHES 2005*, volume 3659 of *LNCS*, pages 172–186. Springer, 2005.

[104] Axel Poschmann, Amir Moradi, Khoongming Khoo, Chu-Wee Lim, Huaxiong Wang, and San Ling. Side-Channel Resistant Crypto for Less than $2,300$ GE. *J. Cryptology*, 24(2):322–345, 2011.

[105] Emmanuel Prouff, Christophe Giraud, and Sébastien Aumônier. Provably Secure S-Box Implementation Based on Fourier Transform. In *CHES 2006*, volume 4249 of *LNCS*, pages 216–230. Springer, 2006.

[106] Emmanuel Prouff and Matthieu Rivain. A Generic Method for Secure SBox Implementation. In *WISA 2007*, volume 4867 of *LNCS*, pages 227–244. Springer, 2007.

[107] Emmanuel Prouff and Thomas Roche. Higher-Order Glitches Free Implementation of the AES Using Secure Multi-party Computation Protocols. In *CHES 2011*, volume 6917 of *LNCS*, pages 63–78. Springer, 2011.

[108] Jean-Jacques Quisquater and David Samyde. ElectroMagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards. In *E-smart 2001*, volume 2140 of *LNCS*, pages 200–210. Springer, 2001.

[109] Håvard Raddum. More Dual Rijndaels. In *AES Conference 2004*, volume 3373 of *LNCS*, pages 142–147. Springer, 2004.

[110] Matthieu Rivain and Emmanuel Prouff. Provably Secure Higher-Order Masking of AES. In *CHES 2010*, volume 6225 of *LNCS*, pages 413–427. Springer, 2010.

[111] Akashi Satoh, Takeshi Sugawara, Naofumi Homma, and Takafumi Aoki. High-Performance Concurrent Error Detection Scheme for AES Hardware. In *CHES 2008*, volume 5154 of *LNCS*, pages 100–112. Springer, 2008.

[112] Kai Schramm, Gregor Leander, Patrick Felke, and Christof Paar. A Collision-Attack on AES: Combining Side Channel- and Differential-Attack. In *CHES 2004*, volume 3156 of *LNCS*, pages 163–175. Springer, 2004.

[113] Kai Schramm, Thomas Wollinger, and Christof Paar. A New Class of Collision Attacks and Its Application to DES. In *FSE 2003*, volume 2887 of *LNCS*, pages 206–222. Springer, 2003.

[114] F-X Standaert, Eric Peeters, Gaël Rouvroy, and J-J Quisquater. An overview of power analysis attacks against field programmable gate arrays. *Proceedings of the IEEE*, 94(2):383–394, 2006.

[115] François-Xavier Standaert, Nicolas Veyrat-Charvillon, Elisabeth Oswald, Benedikt Gierlichs, Marcel Medwed, Markus Kasper, and Stefan Mangard. The World Is Not Enough: Another Look on Second-Order DPA. In *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 112–129. Springer, 2010.

[116] Daisuke Suzuki and Minoru Saeki. Security Evaluation of DPA Countermeasures Using Dual-Rail Pre-charge Logic Style. In *CHES 2006*, volume 4249 of *LNCS*, pages 255–269. Springer, 2006.

[117] Daisuke Suzuki and Minoru Saeki. Security Evaluation of DPA Countermeasures Using Dual-Rail Pre-charge Logic Style. In *CHES 2006*, volume 4249 of *LNCS*, pages 255–269. Springer, 2006.

[118] Pawel Swierczynski, Amir Moradi, David Oswald, and Christof Paar. Physical Security Evaluation of the Bitstream Encryption Mechanism of Altera Stratix II and Stratix III FPGAs. *ACM Transactions on Reconfigurable Technology and Systems*, 7(4):34:1–34:23, 2015.

[119] TELECOM ParisTech SEN research group. DPA Contest (4th edition), 2013-2014. `http://www.DPAcontest.org/v4/`.

[120] K. Tiri, M. Akmal, and I. Verbauwhede. A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards. In *Solid-State Circuits Conference - ESSCIRC 2002*, pages 403–406, 2002.

[121] Kris Tiri and Ingrid Verbauwhede. A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation. In *DATE 2004*, pages 246–251. IEEE Computer Society, 2004.

[122] Michael Tunstall, Carolyn Whitnall, and Elisabeth Oswald. Masking Tables - An Underestimated Security Risk. In *FSE 2013*, volume 8424 of *LNCS*, pages 425–444. Springer, 2013.

[123] Nicolas Veyrat-Charvillon and François-Xavier Standaert. Generic Side-Channel Distinguishers: Improvements and Limitations. In *CRYPTO 2011*, volume 6841 of *LNCS*, pages 354–372. Springer, 2011.

[124] Jason Waddle and David Wagner. Towards Efficient Second-Order Power Analysis. In *CHES 2004*, volume 3156 of *LNCS*, pages 1–15. Springer, 2004.

[125] Shee-Yau Wu, Shih-Chuan Lu, and Chi-Sung Laih. Design of AES Based on Dual Cipher and Composite Field. In *CT-RSA 2004*, volume 2964 of *LNCS*, pages 25–38. Springer, 2004.

[126] Pengyuan Yu and Patrick Schaumont. Secure FPGA circuits using controlled placement and routing. In *CODES+ISSS 2007*, pages 45–50, 2007.

List of Original Publications

Introducing Novel Attack Schemes

a1 **Amir Moradi**, Oliver Mischke, and Thomas Eisenbarth. Correlation-Enhanced Power Analysis Collision Attack. In *Cryptographic Hardware and Embedded Systems – CHES 2010*, volume 6225 of *Lecture Notes in Computer Science*, pages 125–139. Springer, 2010.
DOI: 10.1007/978-3-642-15031-9_9 acceptance rate: 27% (of 108 submissions)

a2 **Amir Moradi**, Oliver Mischke, Christof Paar, Yang Li, Kazuo Ohta, and Kazuo Sakiyama. On the Power of Fault Sensitivity Analysis and Collision Side-Channel Attacks in a Combined Setting. In *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*, pages 292–311. Springer, 2011.
DOI: 10.1007/978-3-642-23951-9_20 acceptance rate: 26% (of 119 submissions)

a3 **Amir Moradi**. Statistical Tools Flavor Side-Channel Collision Attacks. In *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 428–445. Springer, 2012.
DOI: 10.1007/978-3-642-29011-4_26 acceptance rate: 21% (of 195 submissions)

a4 **Amir Moradi**, Oliver Mischke, and Christof Paar. One Attack to Rule Them All: Collision Timing Attack versus 42 AES ASIC Cores. *IEEE Transactions of Computers*, 62(9):1786–1798, 2013.
DOI: 10.1109/TC.2012.154 last-5-year impact factor: 1.726

a5 **Amir Moradi**. Side-Channel Leakage through Static Power - Should We Care about in Practice? In *Cryptographic Hardware and Embedded Systems – CHES 2014*, volume 8731 of *Lecture Notes in Computer Science*, pages 562–579. Springer, 2014.
DOI: 10.1007/978-3-662-44709-3_31 acceptance rate: 26% (of 127 submissions)

Evaluation of Theory in Practice

b1 **Amir Moradi**, Alessandro Barenghi, Timo Kasper, and Christof Paar. On the vulnerability of FPGA bitstream encryption against power analysis attacks: extracting keys from xilinx Virtex-II FPGAs. In *ACM Conference on Computer and Communications Security – CCS 2011*, pages 111–124. ACM, 2011.
DOI: 10.1145/2046707.2046722 acceptance rate: 14% (of 429 submissions)

b2 **Amir Moradi**, Markus Kasper, and Christof Paar. Black-Box Side-Channel Attacks Highlight the Importance of Countermeasures - An Analysis of the Xilinx Virtex-4 and Virtex-5 Bitstream Encryption Mechanism. In *The Cryptographers' Track at the RSA Conference 2012 – CT-RSA 2012*, volume 7178 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2012.
DOI: 10.1007/978-3-642-27954-6_1 acceptance rate: 23% (of 113 submissions)

b3 **Amir Moradi**, Mario Kirschbaum, Thomas Eisenbarth, and Christof Paar. Masked Dual-Rail Precharge Logic Encounters State-of-the-Art Power Analysis Methods. *IEEE Transactions on VLSI Systems*, 20(9):1578–1589, 2012.
DOI: 10.1109/TVLSI.2011.2160375 last-5-year impact factor: 1.252

b4 **Amir Moradi** and Oliver Mischke. How Far Should Theory Be from Practice? - Evaluation of a Countermeasure. In *Cryptographic Hardware and Embedded Systems – CHES 2012*, volume 7428 of *Lecture Notes in Computer Science*, pages 92–106. Springer, 2012.
DOI: 10.1007/978-3-642-33027-8_6 acceptance rate: 26% (of 120 submissions)

b5 **Amir Moradi** and Oliver Mischke. On the Simplicity of Converting Leakages from Multivariate to Univariate - (Case Study of a Glitch-Resistant Masking Scheme). In *Cryptographic Hardware and Embedded Systems - CHES 2013*, volume 8086 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2013.
DOI: 10.1007/978-3-642-40349-1_1 acceptance rate: 20% (of 132 submissions)

b6 **Amir Moradi** and Oliver Mischke. Comprehensive Evaluation of AES Dual Ciphers as a Side-Channel Countermeasure. In *Information and Communications Security – ICICS 2013*, volume 8233 of *Lecture Notes in Computer Science*, pages 245–258. Springer, 2013.
DOI: 10.1007/978-3-319-02726-5_18 acceptance rate: 20% (of 113 submissions)

b7 **Amir Moradi**, David Oswald, Christof Paar, and Pawel Swierczynski. Side-channel attacks on the bitstream encryption mechanism of Altera Stratix II: facilitating black-box analysis using software reverse-engineering. In *ACM/SIGDA International Symposium on Field Programmable Gate Arrays – FPGA 2013*, pages 91–100. ACM, 2013.
DOI: 10.1145/2435264.2435282 acceptance rate: 22% (of 106 submissions)

(b8) **Amir Moradi**, Sylvain Guilley, and Annelie Heuser. Detecting Hidden Leakages. In *Applied Cryptography and Network Security – ACNS 2014*, volume 8479 of *Lecture Notes in Computer Science*, pages 324–342. Springer, 2014.
DOI: 10.1007/978-3-319-07536-5_20 acceptance rate: 22% (of 147 submissions)

(b9) **Amir Moradi**. Wire-Tap Codes as Side-Channel Countermeasure - an FPGA-based experiment. In *Progress in Cryptology – INDOCRYPT 2014*, volume 8885 of *Lecture Notes in Computer Science*, pages 341–359. Springer, 2014.
DOI: 10.1007/978-3-319-13039-2_20 acceptance rate: 24% (of 101 submissions)

Developing Novel Countermeasures

(c1) Axel Poschmann, **Amir Moradi**, Khoongming Khoo, Chu-Wee Lim, Huaxiong Wang, and San Ling. Side-Channel Resistant Crypto for Less than 2,300 GE. *Journal of Cryptology*, 24(2):322–345, 2011.
DOI: 10.1007/s00145-010-9086-6 last-5-year impact factor: 1.196

(c2) **Amir Moradi**, Axel Poschmann, San Ling, Christof Paar, and Huaxiong Wang. Pushing the Limits: A Very Compact and a Threshold Implementation of AES. In *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 69–88. Springer, 2011.
DOI: 10.1007/978-3-642-20465-4_6 acceptance rate 18% (of 167 submissions)

(c3) Tim Güneysu and **Amir Moradi**. Generic Side-Channel Countermeasures for Reconfigurable Devices In *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 2011.
DOI: 10.1007/978-3-642-23951-9_3 acceptance rate: 26% (of 119 submissions)

(c4) **Amir Moradi** and Vincent Immler. Early Propagation and Imbalanced Routing, How to Diminish in FPGAs In *Cryptographic Hardware and Embedded Systems – CHES 2014*, volume 8731 of *Lecture Notes in Computer Science*, pages 598–615. Springer, 2014.
DOI: 10.1007/978-3-662-44709-3_33 acceptance rate: 26% (of 127 submissions)