Low-cost countermeasures against physical attacks on cryptographic algorithms implemented on altera FPGAs

Maxime Nassar

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

TELECOM ParisTech

2012-ENST-010

EDITE - ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

TELECOM ParisTech

Spécialité « Électronique et Communication »

présentée et soutenue publiquement par

Maxime NASSAR

le 09 Septembre 2012

Contre-mesures à bas coût contre les attaques physiques sur algorithms cryptographics implémentés sur FPGA Altera

Directeurs de thèse : **Jean-Luc DANGER et René MARTIN**
Co-encadrement de la thèse : **Sylvain GUILLEY**

Jury

Mme Elisabeth OSWALD, University of Bristol, UK	Rapporteur
M. Lionel TORRES, LIRMM, France	Rapporteur
M. Jens-Peter KAPS, George Mason University, USA	Examinateur
M. Victor LOMNÉ, ANSSI, France	Examinateur
M. Habib MEHREZ, LIP6, France	Président du jury

TELECOM ParisTech
école de l'Institut Télécom - membre de ParisTech

THÈSE

To Laure and Nabil.

Acknowledgements

This PhD was performed in association between the *Trustway* group of Bull and *SEN* (Systemes Electronique Numérique) group of Department COMELEC (Communications & Electronique) of Telecom-ParisTech. These three years gave me the rare opportunity to both learn the essence of research and understand the ways of industrial engineering. This was made possible by meeting and interacting with those two groups of very smart, experienced and friendly people.

First and foremost, I would like to deeply thank my two PhD directors, Professor Jean-Luc Danger of Telecom-ParisTech and René Martin of Bull for trusting and guiding me through this three years. I also extend my sincere thankfulness to Dr. Sylvain Guilley and Patrick Le-Quéré for their numerous teachings and advices in both research and technical fields. It has been my great pleasure to work and interact with those gentlemen who are all specialists in their domains, but more importantly have great human skills.

Furthermore, I am extremely grateful to all the members of my jury who took the time to thoroughly analyse my work. I had the chance to be approved and perform my PhD defense before such renowned specialists for which I feel deeply honoured.

Finally I was fortunate to have the constant support and encouragement of my friends and family without which none of this would have been possible.

Abstract

Side-Channel Analysis (SCA) and Fault Attacks (FA) are techniques to recover sensitive information concealed in cryptographic embedded systems by exploiting unintentional physical leakage, such as the power consumption or the radiated magnetic field. As such attacks are low-cost and easily set–up in practice, they prove to be a serious threat to most sensitive devices. Therefore, when a high level of security is required, specific countermeasures must be manually deployed by the designer, as such functionality are generally not available in current CAD software tools.

FPGA technology is often chosen for low and middle volume applications even for high-end embedded systems where high performances and flexibility are mandatory as well as state of the art security. However, it has been put to light that such devices show great intrinsic vulnerabilities against SCAs and should therefore be protected with adequate countermeasures, able to compensate for those weaknesses.

This thesis has two main goals. On one hand, a review of the state of the art of FPGA-compatible countermeasure against SCA for both symmetrical and asymmetrical ciphering, in order to compare them in terms of performances, area and security level. In the event where none fit the required specifications, alternate countermeasures should be designed. On the other hand , the implementation of the selected protections for standard or customized algorithms with the minimum area overhead, either with automatised design flows or as a hand-made Intellectual Properties (IPs).

Symmetrical algorithms, specially AES, are first studied and several vulnerabilities of usual protections, namely Dual-rail with Precharge Logic (DPL) and masking are analysed, as well as the issue of performance and area overheads. In this context, three new countermeasures are considered:

1. Balance placement and routing (PAR) strategies aiming at enhancing existing DPLs robustness when implemented in modern FPGAs.

2. A new type of DPL called Balanced Cell-based Dual-rail Logic (BCDL), based on the use of a global precharge signal and synchronisation schemes to thwart most of the known DPL weaknesses. BCDL also possess a built-in fault resilience mechanism for simple stuck-at faults and provides implementation optimisations, achieving competitive performances and area overhead.

3. The Rotating S-Box Masking (RSM), a new masking technique for the AES. On the one hand, the frequency output and robustness level of RSM against first-order SCAs are equivalent to a state of the art masking scheme. On the other hand, it brings a significant reduction of the area overhead, as well as robustness to other SCAs like the Variance Power Analysis (VPA), known to be efficient against usual masking. Like most masking schemes, RSM is however not naturally protected against fault attacks, implying that a classical fault detection

technique, for instance a deciphering and comparison, should be additionally implemented.

Regarding asymmetrical algorithms, current countermeasures already offer a robustness level and implementation cost suitable for industrial applications, against both passive and fault attacks. Therefore, rather than devising new mechanisms, a simple Elliptic Curve Cryptography (ECC) core is implemented on FPGA and SCAs are performed to verify the theoretical security level of the chosen protections.

Finally, as designing an efficient countermeasure generally requires a fine understanding of the attack process and as classical SCA may not be sufficient to properly evaluate the robustness of every countermeasure, several new SCAs are presented and evaluated. Firstly the "Rank Corrector" (RC), rather than an actual SCA, is a SCA enhancement algorithm, designed to reduce the number of observations required to perform a successful attack. RC can be used to complement most existing SCAs like Differential Power Analysis (DPA) and Correlation Power Analysis (CPA). Secondly, The First Principal Components Analysis (FPCA), introduces a novel SCA distinguisher based on the Principal Component Analysis (PCA). FPCA proves to be more efficient than classical DPA or CPA on an unprotected DES implementation and VPA on a masked DES architecture. Then, combinations of either acquisition methods or SCA distinguishers are discussed and show significant decrease in the number of measurements required to perform a successful attack.

Contents

List of Figures

List of Algorithms

Glossary

AES:	Advanced Encryption Standard
ASIC:	Application Specific Integrated Circuit
BCDL:	Balanced Cell based Dual-rail Logic
CED:	Concurrent Error Detection
CMOS:	Complementary Metal Oxide Semiconductor
CPA:	Correlation Power Analysis
CPK:	Current Predicted Key
CRC:	Cyclic Redundancy Check
CV:	Cumulative Variance
DES:	Data Encryption Standard
DFA:	Differential Fault Attack
DLP:	Discreet Logarithm Problem
DoM:	Difference of Means
DPA:	Differential Power Analysis
DPL:	Dual-rail with Precharge Logic
DRSL:	Dual-rail Random Switching Logic
DWDDL:	Double Wave Dynamic Differential Logic
ECC:	Elliptic Curve Cryptography
EPE:	Early Propagation Effect
EM:	Electro-Magnetic
EMA:	Electro Magnetic Analysis
FA:	Fault Attack
FK:	False Key
FPCA:	First Principal Component Analysis
FPGA:	Field Programmable Gate Array
GE:	Guessing Entropy
GF:	Galois Field
HD:	Hamming Distance
HW:	Hamming Weight
ICA:	Independent Component Analysis
IMDPL:	Improved Masked Dual-rail Precharge Logic
IP:	Intellectual Propertie
IWDDL:	Isolated Wave Dynamic Differential Logic
LAB:	Logic Array Bloc
LDA:	Linear Discriminant Analysis
LUT:	Look-Up Table
MDPL:	Masked Dual-rail Precharge Logic
MIA:	Mutual Information Analysis
MIM:	Mutual Information as a Metric
MTD:	Measurements To Disclose
NIST:	National institute of standard and technology
PAR:	Placement And Routing
PC:	Possibility of Combination

PCA:	Principal Component Analysis
PK:	Predicted Key
RC:	Rank Corrector
RAM:	Random Access Memory
ROM:	Read Only Memory
RPA:	Refined Power Analysis
RSA:	Rivest Shamir Adleman
RSM:	Rotating S-Box Masking
S:	Stability
SCA:	Side Channel Attack
SDDL:	Simple Dynamic Differential Logic
SE:	Safe Error
SG:	Security Gain
SK:	Secret Key
SNR:	Signal to Noise Ratio
SPA:	Simple Power Analysis
SR:	Success Rate
STH:	Stability Threshold
STTL:	Secure Triple Track Logic
VPA:	Variance Power Analysis
WDDL:	Wave Dynamic Differential Logic
ZPA:	Zero-value Point Attack

Chapter 1

Introduction

1.1 Context

As personal and sensitive information are increasingly stored in electronic devices, the need for security grows the same way. Typically, cryptographic devices are used to protect and conceal such data through several algorithms, such as the Advanced Encryption Standard (AES), Rivest Shamir Adleman (RSA) and Elliptic Curve Cryptography (ECC), which are the main focus of this work. Since their introduction, most of them have been thoroughly studied and proven secure from the cryptanalytic standpoint (although research is still ongoing for the ECC). However, in the past few years, a new class of attack have risen, which do not target the mathematical properties of such algorithms but rather their physical implementation itself. Such schemes, best known as Side-Channel Attacks (SCA) pose a very serious threat to secure designs as they are usually low-cost and easy to set–up. As can be expected, the development of SCAs have been coupled with research on appropriate countermeasures. From the sole academic standpoint, a perfect security is generally sought, considering the optimization of area and performances as an important but secondary matter. By contrast, this work operates in an industrial scope, where high level of security and low resource usage are both mandatory. In that regard, the main goal of this thesis is not to find the most secure countermeasure against all possible SCA (as a matter of fact with the present thrive for development of new SCA no countermeasure could be labeled that way), but rather to find the best possible trade-offs between robustness and overheads in terms of area and performances.

1.2 Organization

This document is structured as follows:

In chapter 2 a state of the art of Side-Channel Attacks is drawn, for both symmetrical and asymmetrical algorithms. Passive SCAs, mostly exploiting power, electromagnetic radiations or timing variations as their side-channel leakage are discussed as well as fault injection attacks.

Conversely Chapter 3 deals with the state of the art of SCA countermeasure and their limitations. Schemes targeting symmetrical ciphers, especially the AES are described separately from those aimed at asymmetrical ones, namely RSA and ECC.

The last three chapters present the main contributions of this work. Chapter 4 focus on new countermeasures for symmetrical algorithms and is divided in four parts. First, Section 4.1 gives a detailed study of the vulnerabilities of state of the art DPLs, all the more when implemented on FPGA, illustrated by a successful DPA on a WDDL 3DES. Second, specific constrained placement strategies are proposed in Section 4.2 as a possible way to enhance the security level of DPLs, taking the example of WDDL. Then a new DPL style so-called BCDL is developed in Section 4.3, in the goal of optimizing the trade-off between robustness and resource consumption. A thorough experimental study of that scheme is undertaken putting to light its advantages and weaknesses, then a comprehensive comparison of most known DPLs and their specificity is given in Section 4.4. Eventually the drawbacks of classical masking techniques are highlighted in Section 5.1 before the presentation of a new masking scheme for AES so-called Rotating S-Box masking (RSM) in Section 5.2. Experimental results as well as a thorough theoretical proof (Section 5.3) are then given to validate its security level against SCAs.

An ECC design is implemented in the fourth chapter 4 in order to properly assess that using an FPGA do not induce unexpected vulnerability or resource consumption. Therefore two classical countermeasure, namely the *Double and Add Always* and *random Splitting of Scalar* are implemented on an Altera StratixII. Simple Power Analysis (SPA) as well as *Doubling Attack* (presented in Section 2.2.1) are then performed on the three implementations to experimentally verify their robustness against such SCAs.

In the last chapter 7, three new SCA schemes are introduced. First, a novel attack so-called First Principal Component analysis (FPCA) is described in Section 7.1, using the Principal Component Analysis (PCA) as a side-channel distinguisher. Then two combined SCA are presented that exploit respectively a combination of measurements and distinguishers (Section 7.2 and 7.3). Theoretical and experimental results are given in that regard, showing a significant decrease in the number of side-channel observations or Measurement To Disclose (MTD) needed to perform successful attacks. Finally, a generic algorithm to enhance existing SCAs based on key ranks analysis, so-called *Rank Corrector* (RC) is proposed in Section 7.4. Although this scheme is not an actual attack, it can be combined with most classical SCA and reduce their required number of MTD.

Chapter 2

Physical Cryptanalysis on FPGA, State of the art

Cryptographic devices are typically used to protect and conceal sensitive data. Although classical algorithms, especially AES, RSA and ECC, are secure from the cryptanalytic standpoint, their actual implementation in hardware designs creates a vulnerability to physical attacks, namely Side-Channel Analysis (SCA) and Fault Attacks (FA).

SCAs and FAs are based on exploiting sensitive information, unintentionally leaked by the target device.There are numerous ways of accessing such information such as monitoring the power consumption or analysing erroneous outputs resulting from fault injection. Globally, a SCA can be characterised by two properties:

- *Passive* or *Active*: a passive attack will not disturb the device's behaviour in any way, while active ones will shift the target from its regular behaviour and analyse its response.
- *Intrusive* or *Non-Intrusive*: Intrusive techniques require physical tampering with the device like depackaging, while non-intrusive schemes only exploit directly available information, for instance the electro-magnetic emanations.

As of now, two categories of physical attacks are mainly studied: on one hand the passive/non-intrusive SCAs which make use of execution time [97], power consumption [98, 115] or electromagnetic radiations [59] as their side-channel and on the other hand active ones, namely FA.

2.1 Generic SCAs

The root of the vulnerability to SCAs lies in the behaviour of CMOS cells of which most recent electronic devices, specially FPGAs, mainly consist of. As a matter of

fact, a clear difference in the power consumption of those elements occurs between a transition from $0 \xrightarrow{*} 1$ or $1 \xrightarrow{*} 0$, when the value changes and one from $0 \xrightarrow{*} 0$ or $1 \xrightarrow{*} 1$, when it does not. This property gives rise to power-based SCAs, as an adversary may be able to observe those transitions via the global activity of the target circuit and use it to retrieve sensitive information. Moreover, one can also exploit the electro-magnetic (EM) field as the side-channel, as they are directly related to the current variations. Unlike power, EM radiations can be collected in a localized manner (depending on the used probe), which may enable the adversary to isolated specific part of the design and reduce the noise ratio. Nonetheless, as most schemes can be conducted using both power and EM measurements, both shall be treated together and referred to as power attacks in the remainder of this manuscript.

In the last few years a wealth of such attacks has been proposed and successfully conducted on several platforms, software and hardware, including FPGAs. They can be classified in two categories: those which exploit a single side-channel measurement, namely Simple Analyses and those which require numerous observations, denoted in the following by Statistical Analyses.

2.1.1 Simple Analyses

Simple Power Analysis (SPA), first introduced by Kocher and al.in [97, 98], relies on the observation of a single side-channel measurement or *trace*. Depending on the type of the target algorithm different information can be retrieved by such schemes. In hardware implementations, when analysing the execution of a block cipher, SPA is generally used to derive some characteristics of the design, for instance the number of rounds, which can in some cases reveal the nature of the algorithm. Figure 2.1 illustrates this behaviour with a SPA of a DES cryptoprocessor, where all the sixteen rounds are clearly visible, within one loading and one output additional cycles.

By contrast, SPA can be an extremal powerful tool to break designs of asymmetrical ciphers, namely RSA or ECC, where no countermeasure against SCAs is implemented. As a matter of fact, those algorithms display a specific iterative behaviour, with regards to the consecutive bits of the secret key. Their respective core operations: the modular exponentiation (Algorithm 1) and the scalar multiplication (Algorithm 2), which are the targets of most SCA, take one key bit into account at each round and perform different computations depending on its value.

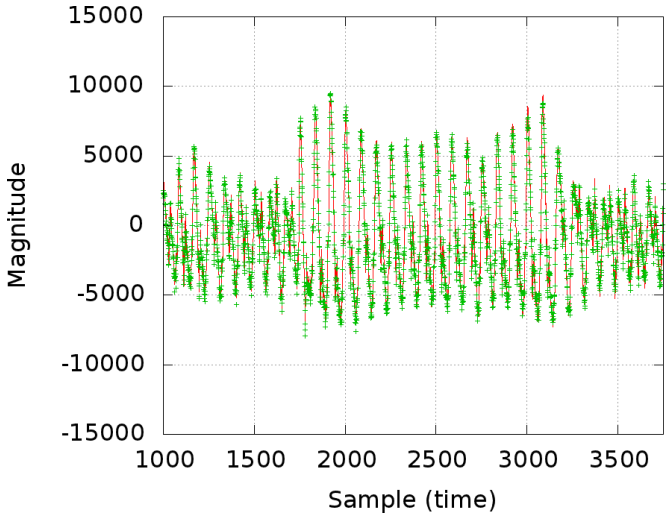


Figure 2.1: SPA on DES.

Algorithm 1 Exponentiation Algorithm.

1: **Input**: $M, k = (1, k_{l-2}, \ldots, k_0)_2$
2: **Output**: $Q = M^k$
3: $Q \leftarrow M$
4: **for** $i = l - 2$ *downto* 0 **do**
5: $\quad Q \leftarrow M^2$
6: $\quad$ **if** $k_i = 1$ **then**
7: $\quad\quad Q \leftarrow Q.M$
8: Return Q

Algorithm 2 Scalar Multiplication.

1: **Input**: $P, k = (1, k_{l-2}, \ldots, k_0)_2$
2: **Output**: $Q = [k].P$
3: $Q \leftarrow P$
4: **for** $i = l - 2$ *downto* 0 **do**
5: $\quad Q \leftarrow 2.Q$
6: $\quad$ **if** $k_i = 1$ **then**
7: $\quad\quad Q \leftarrow Q + P$
8: Return Q

For instance, in Algorithm 1, both a modular multiplication and modular squaring are performed when a key bit is equal to 1, while only the squaring is computed in the other case. Therefore, if an adversary is able to distinguish between the side-channel leakages of those two operations, one measurement is theoretically sufficient to retrieve the entire secret key. An example of SPA on RSA is given in Figure 2.2 to illustrate this behaviour.

Here the two different operations, denoted by *M* and *S* are clearly visible, therefore one can directly deduce the successive value of the processed secret key bits.

In summary, SPA is the most easily mounted SCA, has it only requires one side-channel measurement (note that in practice, an average of the same trace is performed when possible in order to reduce the noise) and can be used as it stands, without the need of complex analysis software. Therefore providing protection against such scheme is mandatory for device aiming to achieve a high level of security. Fortunately

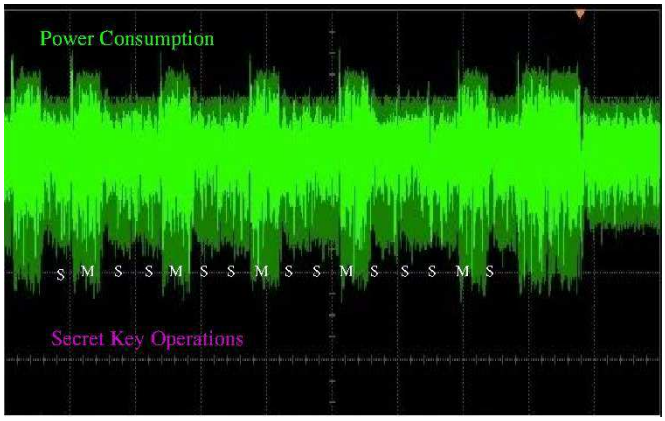


Figure 2.2: SPA on RSA.

several countermeasures have been proposed to thwart this kind of SCA and will be presented in Section 3.

2.1.2 Timing Attack

The concept of *timing attack* was put to light by Kocher and al. in [97] on software implementations of RSA and applied to hardware smart-card version of this algorithm by Dhem and al. in [52]. Other interesting attack methodology on RSA were presented by Schindler and al. in [150, 153]and by Toth and al. in [179].

The timing attack is a statistical SCA based on exploiting the execution time of a system as the side-channel leakage. As a matter of fact, when the implementation is careless especially in software designs, timing differences correlated to the value of the secret key can appear depending on the value of the inputs. Such correlation can then be exploited to recover a part or even the entire key.

In [153], the authors proven than block ciphers like AES can be broken by such schemes through the exploitation of specific flaws in the architecture, in this case the fact that the execution time of the MixColumns operation can be linked to partial values of the secret. On asymmetrical algorithms like RSA, the attack runs iteratively, recovering one secret key bit at a time. Supposing that the first key bits are known, a set of inputs are generated taking into account both an hypothesis on the value of the next bit and a certain knowledge of the design, such that timing measurement recorded during the corresponding executions will enable the adversary to confirm the validity of this hypothesis. The timing variations can be for instance due to RAM cache hits or branch instructions, in software architectures. In hardware designs those differences are less obvious but can be exploited to mount a successful attack, as shown in [52]. In this paper, the authors used the fact that the Montgomery multiplication, that usually runs in fixed time, will perform a final modular reduction when the intermediate result is too large, hence depending on the input. In the end, they show that given

the right conditions, a 512-bit RSA key can be recovered with 300000 timing measurements and a few minutes of calculation, proving that *timing attacks* poses a real threat to cryptographic devices.

However this kind of SCA requires knowledge of the architecture, to be carried out, as well as having a hand on the input messages. The authors of [52] emphasize that a complete control of the input is not mandatory to mount such attacks. Moreover the adversary must be able to perform precise timing measurements, for instance in [52] the acceptable error margin was of a few clock cycles.

Timing attack are not applicable on algorithms running in fixed time, however this property is difficult to ensure, specially for software designs using cache. An other way of thwarting such attacks is to randomize either the key or the input, as will be discussed in Section 3.3.

2.1.3 Statistical SCAs

In opposition to the SPA, statistical SCAs require the collection of a large number of measurements, processed with mathematical tools, in order to retrieve the target's secret key. To mount such a successful attack, a few hypotheses must be validated. First, the target algorithm must be known. Second the adversary must be able to at least observe, if not deliberately input, a variable data, which is generally the plaintext in the case of an attack on the first round (e.g. on DES), or the ciphertext when targeting the last round (e.g. on AES). Finally she should seek to uncover a fixed, unknown data, in most cases a part of the secret key (i.e. a sub-key). Then, the goal of such attack is globally to compare and find dependencies between key-dependant estimations of the side-channel leakage and experimental measurements. This leakage is formalized by a model on variable that is correlated to the secret. As illustrated in Figure 2.3, those SCAs usually unfold according to the following scenario:

1. Side-channel traces or observations ($\mathcal{O}_v$) are recorded during the execution of the target algorithm. The known variable data (plaintext or ciphertext) is denoted by v.

2. Models ($\mathcal{L}_k$) of the corresponding leakages are derived for each possible sub-key hypothesis k, based on v.

3. Those models are confronted to ($\mathcal{O}_v$) by a distinguisher δ which sorts those hypotheses and singles out the most probable one.

Those three points denote the critical parts of most SCAs which are the acquisition process, as well as the choice of leakage model and distinguisher.

The leakage model, denoted by $\mathscr{L}$ is the tool used by the adversary to create $\mathcal{L}_k$. It targets a key-dependent variable. Most commonly used leakage models are the

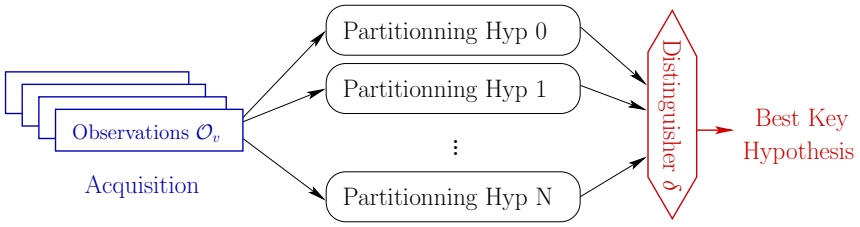


Figure 2.3: Statistical SCA global framework.

Hamming Weight (HW):

$$HW(X) = \sum_{i=1}^{n} x_i.$$

where the variable X is such as $X = (x_1, \ldots, x_n)_2$ and the Hamming Distance (HD):

$$HD(Z) = HW(X \oplus Y) = \sum_{i=1}^{n} x_i \oplus y_i.$$

where X and Y are two consecutive states of the variable Z. For instance the two consecutive values of a register. In hardware, all the more in FPGAs, the latter is favoured [33, 169] as it well matches the actual consumption, which mainly comes from the bit switching.

The distinguisher δ is a statistical tool used to compare the models with actual traces. Several distinguishers have been proposed in the literature, giving rise to a similar number of SCAs. The first was the Difference of Means (DoM), described by Kocher in [98], the corresponding SCA so-called Differential Power Analysis (DPA), was latter referred to as *mono-bit DPA*. The most classical attacks and their distinguishers are listed in Table 2.1 where the observations ($\mathcal{O}_v$) and models ($\mathcal{L}_k$) are respectively denoted by $\mathcal{O}$ and $\mathcal{L}$ and considered as random variables of variance σ.

Table 2.1: Classical SCAs and their distinguishers

Distinguisher	Attack
Covariance $\delta = cov(\mathcal{O}, \mathcal{L})$ [21]	multi-bit DPA
Pearson Correlation $\delta = \rho = \frac{cov(\mathcal{O},\mathcal{L})}{\sigma_{\mathcal{O}}.\sigma_{\mathcal{L}}}$ [33]	Correlation Power Analysis (CPA)
Mutual information [18, 112]	Mutual Information Analysis (MIA)
Variance [78, 106, 111, 167]	Variance-based Power Analysis (VPA)
Likelihood [38]	Bayesian attacks
Least squares [154]	Stochastic Attacks, Clavier's attack (DPA contest v1)

Note that the CPA can also be performed with other correlation coefficient like those of Spearman [17] or Kendall [183].

The mutual information $I(\mathcal{O}, \mathcal{L})$ between the observations and the leakage $\mathcal{L}$ can be used as a distinguisher for the MIA, but also as a metric to evaluate the information leakage, measured in bits and referred to as "Mutual Information as a Metric" (MIM), as described by Veyrat-Charvillon and Standaert in [185]. MIM can be computed such as:

$$I|\mathcal{O}; \mathcal{L}| = H(\mathcal{O}) - H(\mathcal{O}|\mathcal{L}).$$

Where $H(\mathcal{O})$ is the entropy of $\mathcal{O}$ and $H(\mathcal{O}|\mathcal{L})$ the conditional entropy of $\mathcal{O}$ knowing $\mathcal{L}$.

The main challenge when dealing with mutual information is the estimation of the entropy H. In [138] Prouff an Rivain compared several methods such as kernel, histograms and density functions, most of which require significant resources and computation time. In Section 4.3 and 5.2, MIM will be used to estimate information leakages of two new countermeasures. In order to perform this task in reasonable time, we will use the Gaussian estimation and assume that the distribution of both $\mathcal{O}$, $\mathcal{L}$ and their joint distribution are Gaussian. With this approximation the entropy $H(\mathcal{O})$ can be computed as:

$$H(\mathcal{O}) = log(\sigma_o.\sqrt{2\pi e}).$$

2.1.4 Profiling Attacks

By contrast to the previously presented SCAs, profiling attacks are divided in two distinct phases: first the training or profiling step makes use of a clone device, in order to precisely characterize the physical leakage model. Then in the matching or exploitation phase, the adversary takes advantage of the previously gathered information to efficiently recover the secret key of the actual target device.

Two different types of profiling SCAs have been described in the literature. On one hand the *Template Attack* was first proposed by Chari and al. in [38] and later developed in several works [2, 11, 34, 76, 141]. On the other hand, the *Stochastic Attack* was presented by Schindler and al. in [152, 154]. The main difference between those two schemes lies in the profiling phase, where the latter aim at exploiting the knowledge of the physical device, to better characterize the leakage.

Confronting those two methods is a delicate matter due to a number of issues like the possibility to chose the inputs on the target device, the considered leakage model and the level of control with the clone. Nevertheless a framework for evaluating profiling SCAs is given in [170] and a comparison of those attacks is performed in [64], which leads to the conclusion that *Stochastic Attacks* may be more efficient when the

available number of measurements during the profiling phase is small, while *Template Attacks* shows better results on large pools of traces.

Although profiling attacks have been presented as the most powerful SCAs, the necessity of possessing and controlling a clone device can become a serious disadvantage. Nonetheless, as discussed in [24, 95], those schemes can also be employed as a powerful evaluation tools for SCA countermeasures.

2.2 Specific SCAs on Asymmetrical Algorithms

Asymmetrical ciphers, namely RSA and ECC, display a specific iterative behaviour, with regards to the consecutive bits of the secret key. As a matter of fact, their respective core algorithms: the modular exponentiation (Algorithm 1) and the scalar multiplication (Algorithm 2), which are the targets of most SCA, take one key bit into account at each round and perform different operations depending on its value.

Due to their particular iterative behaviour, with regards to the secret key, asymmetrical ciphers are a target of choice for ingenious SCAs. As different operations are performed depending on the value of a single key bit, a direct correlation appears between the computations at a given iteration and the corresponding bit, a property that has been often exploited to mount powerful dedicated attacks.

2.2.1 Doubling attack

Presented in [57] by Fouque and Valette as new type of SCA targeting asymmetrical algorithms, namely the exponentiation and scalar multiplication, the doubling attack is situated between simple and differential analyses. Indeed it is based on studying the difference between two SPAs with chosen inputs. The adversary can compute $Q_1 = m^k$ and $Q_2 = (m^2)^k$ for RSA and respectively $Q_1 = [k].P$ and $Q_2 = 2.[k].P$ for ECC, where m and P are the inputs and k the secret key. The idea is that during those execution, if the same intermediate values may appear depending on the key bits. For instance, when considering the case of ECC, the doubling operation at iteration i in the computation of Q_1 is the same as the doubling operation at iteration $(i-1)$ for Q_2, if and only if $k_{i-1} = 0$. This behaviour is more clearly illustrated in Table 2.2, where the colored intermediate results are equal for Q_1 and Q_2 at different iterations. This Table describes the computation of $Q_1 = [k].P$ and $Q_2 = 2.[k].P$ with $k = (1001110)_2$ using the aforementioned countermeasure, namely both point doubling and addition are computed at each iteration.

As can be observed, by shifting the power curve of Q_1 by one iteration and performing the difference between the two curves (of Q_1 and Q_2), minimums should appear for the two colored point doubling, revealing the value of the corresponding key bits. Consequently, all the secret key bits (except the least significant one) can theoretically be deduced from the analysis of only two power consumption curves.

This scheme has also been proven to be efficient even against known DPA countermeasures, as will be discussed in Section 3.3, hence posing a serious threat to secure cryptographic designs.. Nevertheless, it can only be applied to "right-to-left" algorithms and some SPA countermeasures, for instance the *Montgomery Ladder* algorithm, are not vulnerable to such attack.

Table 2.2: *Doubling Attack* principle

Iteration i	k_i	$Q_1 = [k].P$	$Q_2 = 2.[k].P$
1	1	$2 * 0$ $0 + P$	$2 * 0$ $0 + 2P$
2	0	$2 * P$ $2P + P$	$2 * 2P$ $4P + 2P$
3	0	$2 * 2P$ $4P + P$	$2 * 4P$ $8P + 2P$
4	1	$2 * 4P$ $8P + P$	$2 * 8P$ $16P + 2P$
5	1	$2 * 9P$ $18P + P$	$2 * 18P$ $36P + 2P$
6	1	$2 * 19P$ $38P + P$	$2 * 38P$ $76P + 2P$
7	0	$2 * 39P$ $78P + P$ return $78P$	$2 * 78P$ $156P + 2P$ return $156P$

2.2.2 Comparative Power Analysis

This attack was proposed by Homma and al. in [77] targeting the modular exponentiation, but can also be applied to the ECC. It can be viewed as a generalization of the *doubling attack*. As a matter of fact, it can be performed on both left-to-right and right-to-left algorithms and is efficient on most SPA countermeasures including the *Double-and-add always* or *Montgomery Ladder* algorithms. Let's take the example of the ECC with *Double-and-add always*, like in the previous Section. The idea is the following: supposing that the first few bits of the secret key are known, one must chose two points P and P_1, with $P_1 = n.P$ ($n \in \mathbb{N}^*$), such as $[k].P_1$ at iteration j_1 equals $[k].P$ at iteration j where k_{j_1+1} is known and k_{j+1} is unknown. Then, by monitoring the power consumption of $[k].P$ at iteration $(j + 1)$ and comparing it with that of $[k].P_1$ at iteration $j_1 + 1$, the next key bit (k_{j+1}) can be found. The whole key is then found recursively.

Table 2.3 illustrate this principle, with $n = 9$, $j = 4$ and $j_1 = 2$, supposing the first four bits of the secret key k are known. In this example, the attack focus on finding the fifth bit of k. As $[k].P = 9.P$ at iteration 4, $P_1 = 9.P$ is chosen. Then by comparing the power consumption of $[k].P$ at round 5 and $[k].P_1$ at round 3, the value of k_5 can be found.

Table 2.3: *Comparative Power Analysis* principle

Iteration i	k_i	$[k].P$	$[k].P_1 = [k].9P$
1	1	$2 * 0$ $0 + P$	$2 * 0$ $0 + P_1 = 9P$
2	0	$2 * P$ $2P + P$	$2 * 9P$ $18P + P$
3	0	$2 * 2P$ $4P + P$	
4	1	$2 * 4P$ $8P + P = 9P$	
5	**?**	**if** $k_i = 0$**:** $2 * 9P$ $18P + P$	

This attack is more complex to mount than the *doubling attack*, but proves to be theoretically efficient on a wider range of architectures, including left-to-right algorithms and *Montgomery Ladder*.

2.2.3 Address-bit DPA

This is a specific variant of DPA that is based on targeting the registers addresses rather than their actual values. It was first proposed by Messerges and al. in [121] and can be deployed on designs implementing a countermeasure like the *double-and-add-always* algorithm, where the intermediate values do not depend on the secret key bits, but their addresses do. Indeed, the power consumption and electro-magnetic radiations of a system are affected by the manipulated registers addresses. In [81], Itoh and al. perform such a successful attack on an ECC architecture protected against both SPA and DPA by the *Montgomery ladder* algorithm and the *random projective coordinates* (those two countermeasures are respectively described in Sections 3.3.1 and 3.3.3), proving that care should be taken to provide countermeasures against such scheme.

2.2.4 Carry-Leakage based Attack

The carry-leakage attack proposed by Fouque and al. in [56] is an ingenious scheme to break asymmetrical algorithms implemented with a countermeasure called *scalar*

randomization (fully described in Section 3.3.2) that involves adding a specific random value to the secret key before each execution. The principle of this attack is not to target the cryptographic algorithm but the countermeasure itself and especially the initial addition with the random, in order to recover the secret key. As a matter of fact, the authors use the property that when adding a random value to a fixed one (i.e. the secret key), the probability of appearance of a carry flag only depends one the fixed value. Moreover, long additions of integers are usually performed on smaller size elementary blocs, hence generating several carry flags, correlated to the value of the key. Considering that the raising of those flags can be detected by side-channel observation, if an adversary is able to produce a sufficient number of measurements, the entire key could be recovered.

In practice, the number of broken key-bits depends on one hand on the elementary adder size: the smaller the adder, the more information is retrieved and on the other hand on the key length. Depending on those factors, the number of required measurements and computational cost may vary. Indeed, as the entire key is usually not directly recovered, additional analysis, for instance an exhaustive search of the remaining bits, is mandatory.

The evaluation presented in [56] suggest that RSA keys can be fully recovered up to a key length of 2024 for 32-bit adders, while the attack on ECC should be possible on keys of 160 bits and less.

2.2.5 RPA and ZPA

The Refined Power Analysis (RPA) proposed by Goubin in [66] is a statistical chosen-input SCA specific to the ECC. It is based on the fact that the apparition of "special points" during the scalar multiplication can be detected using side-channel measurements. The property of such points is to have an affine or projective coordinate equal to 0.

The attack proceeds as follows: supposing the first key bits are known, an hypothesis on the next bit (k_i) is made and an input point is generated such as $P = [{d_{k_i}}^{-1}(mod \#E)].P_0$, where P_0 is a special point, $\#E$ the cardinal of the considered curve E and d_{k_i} depends on both the known key bits and the hypothesis. Then a number of measurements are recorded during several execution of $[k].P$ and the mean curve is computed. If the hypothesis was correct, the resulting curve will show consumption peaks, in other cases it won't. Eventually all secret key bits can be recovered iteratively.

This attack is very powerful as it is efficient on classical SPA countermeasures like the *double and add always* algorithm as well as any DPA countermeasure that does not affect the special points (this will be discussed in Section 3.3). However it requires the possibility to chose the input and the existence of a special point on the curve, which is not always the case.

This attack was latter generalized by Akishita et al. in [5] where the concept of special point is extended to points inducing intermediate calculation values to be equal to 0. Therefore the Zero-value Point Attack (ZPA) is an extension or the RPA, that can be applied to a wider range of elliptic curves.

2.3 Generic Fault Attacks

When a cryptographic device performs erroneous computations, the faulty output becomes a side-channel and may leak sensitive information. This gives rise to another class of powerful SCA, namely fault or perturbation attacks (FA). The main goal of such scheme is to physically tamper with a target device in order to switch its regular behaviour to a faulty one and exploit the results to retrieve secret data.

Although the necessary means to deploy such attacks, all the more on FPGA are clearly more complex and expansive than those for passive SCAs, FA have been shown to be a very realistic threat to cryptographic designs. Moreover, most countermeasure against DPA and other passive SCA do not induce robustness towards FA.

In the literature both fault injection techniques and fault attacks are actively discussed, although not necessarily together. Fault induction is performed by changing the environment of the target device. The most classical ways to do so are *power spikes* [16], *clock glitches* [29, 93], *light* [162], *lasers* [101] and *eddy current* [140].

Several criterion can be used to characterized those faults. First of all they are either *transient* meaning that they only occur for a given period of time, generally during a computation due to shifts in voltage or clock frequency, or *permanent* if the hardware is definitely modified, for instance by cut wires. These attacks may also require a certain precision in either time or space, inducing a wide range of complexity for the corresponding experimental setups. Finally fault injection can result in several type of errors like bit flips, when the value of a bit is switched or stuck-at faults where the targeted bit is permanently set to a fixed value.

2.3.1 Differential Fault Attack

The most commonly used FA is the Differential Fault Attack (DFA), first introduced by Biham and Shamir in [27] which demonstrated its efficiency on DES. It consist in encrypting the same input twice, once without inducing any faults and once with a fault injection. By analysing the faulty/non-faulty couple an adversary is able discard some key hypotheses. These steps are then repeated until the full key is recovered. For instance in [27], the authors showed that this recovery was possible with 50 to 200 input plaintexts.

The concept of DFA have been thoroughly studied and adapted to most cryptografic algorithms like AES [29]. In [132], Piret and Quisquater proposed a DFA on AES which allows an adversary to retrieve the entire secret key with two well located faults. Asymmetrical algorithms can also be the siege of such attacks. Boneh and al.

presented such an application to RSA in [30] and Biehl and al. extended the concept to the ECC in [26]. In this case, the idea is to switch the computation from the mathematically strong elliptic curve to a weaker one by flipping a single register bit during the scalar multiplication.

2.3.2 Safe Error Attack

The *safe error attacks* concept was proposed by Joye and Yen in [90] and [190], targeting the RSA exponentiation algorithm, but can also be applied on the scalar multiplication and even on AES as shown by Blömer and Seifert in [29].Unlike classical fault attacks, it is not based on the study of erroneous outputs, but rather on the apparition, or not of a faulty result. As a matter of fact the principle of this scheme is to induce a fault in such a way that depending on a given part of the key, this fault will either be cleared or propagated till the output. There are two kinds of *safe error* (SE), on the one hand the *computational SE* consist in inducing a temporary fault on the computational part of a device, generally during dummy operations introduced by countermeasures like the *double and add always* algorithm. For instance if such a fault is generated during a dummy multiplication of an exponentiation and the result is valid, the adversary can immediately deduce that the corresponding key bit was 0. On the other hand, *memory SE* target registers and memory blocs aiming for the same behaviour as before, namely that the output will be faulty or not, depending on the key.

In practice, given that an adversary is able to induce such faults with the right timing and precision, safe-errors require almost no analysis and can recover the key relatively fast, hence posing a real threat to both software and hardware systems. Nevertheless inducing precise localized faults on FPGA implementations should prove to be much more difficult than on smart-carts, given the architecture of such devices.

2.4 Specific Fault Attacks on ECC

Due to the mathematical properties of the ECC, several specific fault attacks have been proposed, usually targeting the curve parameters or input point in order to move the scalar multiplication from a cryptographically strong curve to a weak one, where the discreet logarithm problem (DLP) is easier to solve.

2.4.1 Invalid Point Attacks

Let the generic equation of an elliptic curve E defined over field $\mathbb{K}$ be such as:

$$E_{\mathbb{K}} : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{2.1}$$

The first attack of this type was described by Biehl and al. in [25] by observing that a curve parameter, namely a_6 was not used in the scalar multiplication of a point P with a key k such as $Q = [k].P$. Therefore an adversary can chose a weak curve

E' defined over a field $\mathbb{K}'$, where only a_6 differs from E. Then a new input point P' situated on E', is used to compute $Q' = [k].P'$. By solving the corresponding DLP, partial information on the secret key can be recovered and this process is repeated in order to retrieve the entire key. This attack shows that naive architectures, not taking into account the possible treat of SCA, can usually be broken with simple attacks, not requiring specific equipment or time consuming analyses.

The straightforward way to prevent such attack is to verify that the input point in actually on the right curve. In this case the attack is still possible, but will require inducing a fault after the verification, which requires a relatively high precision in both location and timing.

2.4.2 Invalid Curve Attacks

In [43], Ciet and Joye generalized the concept of Biehl [25], by demonstrating that random unknown faults could be efficiently exploited. Two fault models are considered: *transient* faults induced for instance on a bus, during the reading of a parameter and *permanent* ones due to lasting modifications of non-volatile memory, where those parameters are stored. Then the authors showed that by altering either the curve parameters, the point or the field representation, partial and even total recovery of the secret key was possible.

2.4.3 Twist Curve Attack

The *twist-curve attack* proposed by Fouque and al. in [55] can be applied on systems that do not use the y-coordinate during the computation of the scalar multiplication, for instance when using the classical *Montgomery ladder* algorithm. The idea is to move the computation from a strong curve E, to its quadratic twist denoted by $\tilde{E}$. As a matter of fact, even though if the curve E is strong, its twist is usually not and the DLP may be possible to solve. This attack is also based on the fact that a given x-coordinate corresponds either to a curve or its twist with a 50 % probability. Therefore, by inducing a random fault on the abscissa of a point P of curve a, an adversary can move the computation to its twist with a 50 % chance.

Chapter 3

FPGA Countermeasures, State of the art

3.1 Generic Countermeasures

Several SCA countermeasures can be implemented independently of the considered cryptographic algorithm and often without modifying its netlist.

- *Obfuscation* techniques e.g. noise generators are a classical way to reduce the Signal to Noise Ratio (SNR) in order to increase the number of side-channel measurements required to perform a successful SCA. However, given that recent technologies may allow an adversary to record numerous traces in reasonable time, such devices should only be use in combination with other countermeasures, in order to slow down potential attacks.

- *Desynchronization* schemes, for instance the *Random Delay Insertion* [109] which consists in adding dummy operations of random length throughout the algorithm are commonly used in software to prevent an adversary to properly synchronize the side-channel observations. In hardware designs, such schemes are more complex to implement and may induce significant performance loss. In the end the global effect is similar to that of a noise generator and they should not be implemented as the sole countermeasure against SCAs. Moreover, several synchronisation techniques have been described to thwart such countermeasure [51, 74, 161].

- *Key updating*, first proposed in [98] consists in regularly altering the secret key in order to restrict the potential number of side-channel measurements that can be recorded by an adversary. For instance, if a DPA is known to require 10000 traces to be successfully performed, one should update the key before this threshold. However, with the constant evolution in the SCA domain, this number is bound to decrease. Moreover, such countermeasure will not be robust against attacks

that can be mounted with very few measurements (e.g. a DFA an AES can be successful with only two traces), as the update cannot be performed on a too frequent basis for performance reasons. Finally several cryptographic protocols do not allow modification on the key in which case such countermeasure in impracticable.

All those countermeasure are not by themselves sufficient to ensure robustness against SCAs, but should rather be used in addition to others, in order to increase the overall security of the target device. As a matter of fact with the increasing research in the SCA field, no existing countermeasure can be labeled as perfectly secure against all SCA and a robust design should include a superposition of several schemes in order to reach the desired trade-off between complexity, performance and security.

3.2 Countermeasures Against Passive Attacks on Symmetrical Algorithms

Whichever the type, the main goal of these attacks is always to exploit a correlation between a passive side-channel leakage (power, electromagnetic fields, ...) and sensitive data (usually a secret key). As of now, two major ideas of countermeasure have been described to remove those dependencies:

1. Hiding [115, Chap. 7] [15, Chap. 4]: flatten the global power consumption, making it uniform and constant.

2. Masking [115, Chap. 9] [15, Chap. 4]: randomize the sensitive data.

3.2.1 Masking

The concept of masking, was independently introduced in 1999 by Chari and al. in [37] as well as Goubin and Patarin in [68], although the term "masking" was not yet employed. The first consists in a theoretical study of this countermeasure, while the latter present a general method for protecting block ciphers, taking the example of DES. Since then, masking has become one of the most widely used countermeasures against SCAs. The Section hereafter will first deal with the global principle of masking, followed by the presentation of specific schemes dedicated to the AES and finally, the High-Order masking concept and implementation techniques will be discussed.

Masking Principle

Globally, the masking technique relies upon the concealment of a sensitive variable x by one or several masks $(m_0, m_1, \dots)$ which take random values. The internal variable x no longer exist as a net in the cryptosystem but can be reconstructed by a couple of signals $(m_0, m_1, \dots)$, $x_m = \theta(x, m_0, m_1, \dots)$ where x_m is the masked variable and θ a

specific operation which will define the actual type of masking. Nowadays, the most common operator is the exclusive-or *xor* which gives rise to boolean masking such as $x_m = x \oplus m_0$, at the first-order. Other schemes include arithmetic and multiplicative masking [7], respectively corresponding to modular addition and multiplication, as well as affine masking [58] which is a combination of the multiplicative and boolean methods, such as $x_m = m_0.x \oplus m_1$.

Let's take the example of the boolean masking. The implementation of this countermeasure is straightforward for a function f that has the following linearity property:

$$f(x \oplus m) = f(x) \oplus f(m)$$

As a matter of fact, the value of $f(x)$ can be reconstructed from the application of f on $x \oplus m$ and m, hence the computation of $f(x)$ can be extracted at the very end of the algorithm. This avoids direct leakage of information as $x \oplus m$ and m are decorrelated with x. In hardware designs, the masking is thus implemented as two parallel paths, one for the masked sensitive data and one for the mask itself. This way the actual unmasked result can be obtained by combining the final values of both paths.

Unfortunately, symmetrical algorithms, like DES and AES, are composed of both linear and non-linear operations, namely the S-Boxes denoted in the following by S. In this case, the classical S-Box is used in the computation of the masked data, however the structure becomes more complex when dealing with the mask path, as $S(x)$ cannot be reconstructed mathematically from $S(x \oplus m)$ and $S(m)$. As of now, two major ideas have been proposed to deal with this issue:

- The *Global Look-up Table* scheme as described by Proof and al. in [136] for software purpose, consists in generating a new S-Box noted S' addressed by both the mask m and the mask data x_m. For hardware architectures, this concept is the most natural one and has already been used in actual FPGA implementation like [111], such as:

$$\begin{aligned} S'(x_m, m) &= m', \\ and\ S(x_m) &= S(x) \oplus m'. \end{aligned} \tag{3.1}$$

  Where m' is a new mask reusable for the next round.

  However, this method have two major drawbacks. First of all, the complexity in terms of area brought by S' may fit block ciphers with small S-Boxes like PRESENT or even DES, but is unrealistic for algorithms with relatively large S-Boxes, like AES. Indeed an 8-input classical AES S-Box must be transformed into a 16-input masked one which, considering they are stored in RAM or ROM, would require 2^{16} memory bits for all sixteen of them. Although recent FPGA technology may allow the implementation of such a scheme, it is unfit for industrial designs including complex architecture and numerous Intellectual Property

(IPs). Second, the S-Boxes are addressed by the masked data and the mask, thereby leaking at second-order in the same of zero-offset attacks [187, §4.1].

- The table *Re-computation Method* proposed by Messerges in [119] is, as of now, mostly deployed in software designs and consists in pre-computing an already masked S-Box, denoted by S'' in the following, with new random values before each ciphering, such as:

$$S''(x) = S(x \oplus m_0) \oplus m_1. \tag{3.2}$$

  Where m_0 and m_1 are two random masks.

  This S-Box is then used in combination with other masks, dedicated to the linear part, in order to perfectly secure the whole algorithm against first-order attacks. For more detailed information the reader is referred to [119] and [136]. It is noteworthy that this methods is most relevant on algorithms where all S-Boxes are identical, like AES, as only one pre-computation is required before each ciphering.

  Although this countermeasure proves to be quite efficient in software implementations, especially when only one S-Box is implemented, its direct application is unfit for hardware architectures where the pre-computation ans memory storage of one masked S-Box would be more costly in terms of performances than an entire AES ciphering.

In the end, the hardware, all the more the FPGA implementation of a full masked block cipher and especially AES is not a trivial matter when a high level of security as well as competitive performances and area consumption are required.

AES-specific Schemes

As mentioned before, the most sensitive part of the masking process and in fact of most SCA countermeasure, are the S-Boxes. In the case of AES, which is the main focus of this thesis in terms of symmetrical algorithms, several specific schemes have been proposed throughout the years to mask the S-Boxes, exploiting their mathematical properties. As a matter of fact, there is only one type of AES S-Box, unlike DES for instance and it is composed of two distinct operations: a linear affine transformation and an inversion over $GF(256)$. The main challenge thus lies within the masking of this inversion.

The first AES-dedicated scheme was presented by Akkar and Giraud in [7]. The idea was to use boolean masking on the linear operations and switch to multiplicative masking for the inversion, which is possible due to the $GF(256)$ ring structure. However, this method does not protect the "0" value and was later proven to be vulnerable to a first-order SCA so-called zero-value attack [65, 128], which exploit this flaw. A

similar and simplified version of this countermeasure was proposed by Trichina and al. in [181], which possessed the same vulnerability.

In [65], Golic and Tymen described a possible way to remove the issue of the "0" value, by randomly embedding $GF(256)$ in another ring, where the specific masking of this value can be performed. Blömer and al. proposed in [28] to use the polynomial representation of the inversion over $GF(256)$ and perform its computation with an exponentiation. Other schemes were presented by Goubin and Courtois in [48], Piret and Standaert in [133] and Prouff and Rivain in [136]. However all those methods are mostly software-oriented and hardly usable for FPGA implementation, as they would generally induce unreasonable overheads in terms of area and/or performances.

Methods specifically focused on hardware implementations were described for instance by Trichina and al. in [182], which dealt with an AES cryptoprocessor tailored for low cost devices. In [129], Oswald and al. presented a secure design of the AES S-Box, based on breaking down the computation of the inversion in several operations over $GF(4)$ and $GF(16)$, where the masking is easily performed. Oswald and Schramm also extended this countermeasure to software architectures in [130] and Canright and Batina proposed a similar ASIC-oriented method in [35].

To our best knowledge, few papers have dealt with the actual implementation of a fully-fledged masked AES design in FPGA. In [118], Mentens and al. proposed such an implementation, combining Boolean and multiplicative masking. However, as stated before, this type of countermeasure has been shown to be susceptible to so-called zero-value attacks, that exploit the absence of masking on the `0x00` byte value. Last year, Regazzoni and al. [142] developed a full Boolean masking scheme optimized for recent FPGAs and based on the work of Oswald and Schramm [130].Their experimental results on a Xilinx Virtex5, show an area consumption of roughly three times the unprotected one and a performance penalty of 50%. Although relatively expensive compared to an unprotected implementation, this design is nevertheless suitable for actual industrial applications and provably secure against first-order SCAs, proving that masking can be a sound countermeasure for FPGA architectures.

However, as stated in Section 2.1, robust schemes against first-order SCAs like DPA or CPA can still be the siege of high-order attacks. Therefore, over the past few years, several high-order masking techniques have been proposed to counteract such threats.

High Order Masking

Over the past few years, high-order SCAs have become an increasingly realistic threat. However, few papers have dealt with countermeasures to such attacks and especially for hardware implementations. The first high-order masking scheme so-called

"Unique Masking Method" (UMM), was presented by Akkar and Goubin in [8] targeting a software implementation of DES. The idea is to compute new S-Boxes, depending on two random 32-bit values and use them to create several possible rounds that can be used in different order. Later this first method was attacked one one hand by Akkar and al. in [6] and on the other hand by Jiqiang Lv and Yongfei Han in [110], that both proposed enhancements to remove the UMM vulnerabilities. Nevertheless these schemes are based on the fact that the new S-Box computation does not provide any side-channel leakage, which could be a wrong assumption, thus inducing a security flaw.

In [80], Ishai and al. proposed an hardware oriented masking technique, theoretically robust at any order. It it based securing the target algorithm at the gate level, by decomposing it in basic *AND* and *XOR* gates with at most two fan-ins and three fan-outs. In this case security against *t-order* attacks can be achieved in $O(nt^2)$ where n is the number of gates. Obviously such a scheme would be extremely costly for FPGA implementations where the gate level decomposition is hardly feasible and would require large waste of resources, considering that recent FPGAs are build with at least 4- to 6-inputs look up tables.

In [155], Schramm and Paar presented a method based on the *table re-computation* [119] discussed beforehand, where four masks are used instead of two, in order to extend the security to high-order attacks. However, it was put to light in [47] that this countermeasure was only effective against second-order SCAs. Nevertheless, the complexity of the overall scheme is greatly increased, with regards to the corresponding first-order countermeasure, making it unfit for high performance FPGA implementations.

Other optimized software countermeasures were introduced by Rivain and al. in [143] against second-order SCAs and by Rivain and Prouff in [144] against any high-order attack. In the latter, the authors adapted the idea of Ishai and al. [80] to software implementations, in order to create a generic high-order masking scheme with provable security.

*

To summarize, masking is maybe the most widely spread countermeasure against SCAs, especially for software targets. As a matter of fact, several schemes can be implemented in software to fit the requirements in terms of complexity, performances and security, even against high-order attacks. However, as long as hardware designs are concerned, all the more on FPGA, implementing a fully-fledged masked crypto-processor in not a trivial task. Moreover, industrial application of cryptographic algorithms are usually embedded in complex designs performing numerous tasks, thus low area consumption and relatively high performances are mandatory. Therefore, existing high-order masking schemes seem mostly not applicable to such usage, as does several first-order countermeasures discussed in the SCA literature. As of now, the

most fitted architecture for FPGA seems to be the one proposed in [142] which shows an area increase of a factor 3 and performance loss of 50%. Therefore, additional research should be carried out in order to find optimal trade-offs between security and implementation costs.

3.2.2 DPLs

DPL (Dual-rail with Precharge Logic) tends to make the global power consumption constant, independently of the inputs, hence removing any correlation with the sensitive data. This method relies on 2 major principles:

1. The duplication of the datapath in 2 dual paths ("True" and "False"). Each cell C is replaced by a couple (C_t,C_f) performing functions (f_t,f_f) such as: $f_f(x) \equiv \overline{f_t}(\overline{x})$. This way, when one gate switches states, its dual does not and vice-versa.

2. A 2-phases protocol to ensure there is one and only one transition for each dual couple, at each cycle:

   (a) Precharge: every signal is set to the same value, generally '0'.

   (b) Evaluation: effective computation.

This protocol is illustrated in Figure 3.1 with the theoretical timings for a 2-input AND/OR couple. In this example, a (a_t, a_f) and b (b_t, b_f) are the inputs, while o (o_t, o_f) is the output. As can be seen in this example, one and only one transition occurs on each signal, at each clock cycle. Therefore the global consumption should be constant throughout the computation and this type of countermeasure should be sound against SCAs.

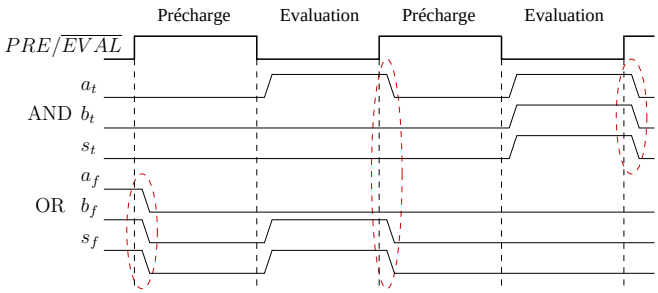


Figure 3.1: Precharge and evaluation of DPLs.

However, several vulnerabilities (that will be thoroughly discussed in Section 4.1) have been found in the past few years, especially when dealing with FPGA implementations, where both hardware and synthesis tool cannot be perfectly controlled (unlike ASICs). Thus many different flavors of DPLs have been proposed, usually

aiming at removing previously found weaknesses, or optimising area consumption and/or performances.

In the following, we present a list, as exhaustive as possible, of existing DPLs susceptible to be implemented on FPGA and their characteristics in terms of surface, performances and theoretical robustness against Side-Channel Attacks when available.

WDDL

WDDL (Wave Dynamic Differential Logic), was introduced by Kris Tiri in 2004 [176, 178]. The specificity of this scheme lies within the way of achieving the precharge state. As a matter of fact, the combinatorial part of WDDL is composed solely of logical **AND** and **OR** gates, thus the '0' value can simply be introduced once, in the registers and will automatically be propagated through all the logical gates like a "wave".

The transformations between single- and dual-rail for WDDL are depicted in Figure. 3.2. Each logical gate is replaced by a couple including itself and its dual, registers are duplicated twice, to allow the 2-phase computation on both "True" and "False" parts and inverters simply become a crossing of dual nets.

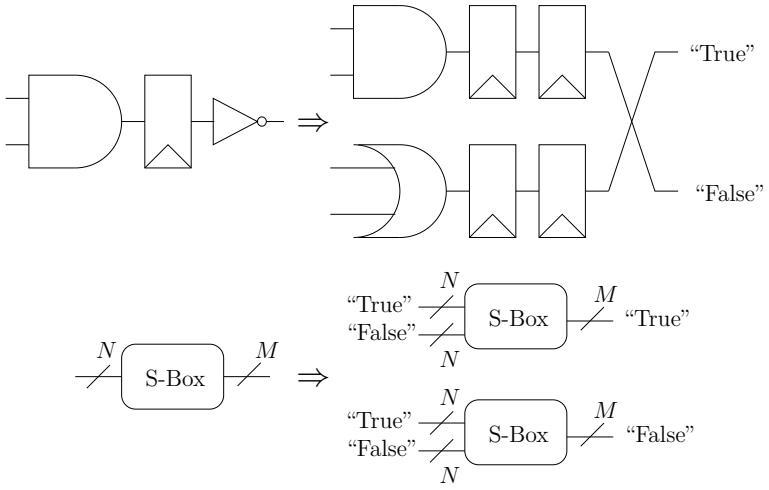


Figure 3.2: Single-Rail to Dual-Rail for WDDL.

In WDDL, logical functions must be positive in order to avoid glitches. It is therefore almost impossible to use functions with more than 4 inputs, as the computation of *N-inputs* functions with $N > 4$ is too complex on a mathematical point of view. Furthermore WDDL theoretically needs a perfectly balanced placement & routing between dual nets. While this can be achieved when carefully designing an ASIC, it is, as of now, not feasible on FPGAs.

Table 3.1 discloses, the theoretical ratios in terms of surface and performances for WDDL, with regards to an unprotected implementation. Area estimation is done sep-

arately for the registers, S-Boxes and the linear combinatorial part, as those usually differs for most DPLs.

Table 3.1: Area and performance ratios for WDDL.

Registers	S-Box	Combi.	Speed
4	4-x^{2n}	$\geq$2	$<$0.5

WDDL with Divided Back-end Duplication (DBD)

This method, introduced in [14] by Baddam and Zwolinski, aims at completely separating the "True" and "False" halves of WDDL, in such a way that the placement and routing could be made for one part and copied/pasted for the other, hence achieving an strong balance level between the two parts. However, as it was shown in Section 3.2.2, inverters in the WDDL logic must be replaced by wire crossings, hence forbidding a separation of the two halves.

To cope from this issue, inverters in this logic are replaced by *XOR* gates, that acts as: buffer during the precharge phase and inverters during the evaluation phase, when their second input is connected to a negated precharge signal. This is a global signal, called $\overline{prch}$, which is equal to 0 during the precharge and 1 during the evaluation. This way the design can properly operate while being totally separated. The transformation between WDDL and this logic is illustrated in Figure 3.3.

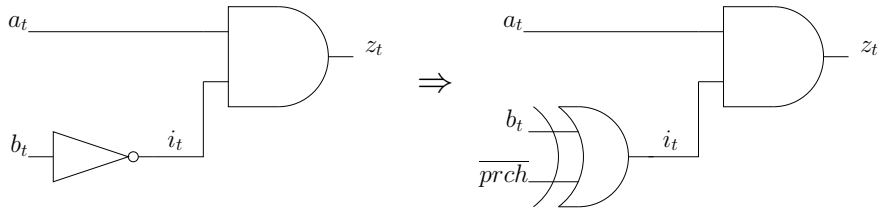


Figure 3.3: From WDDL to WDDL with Divided Back-end Duplication.

However the clean separation in two distinct path may leave the design open to located EM attacks, especially with the recently described cartography techniques. As a matter of fact if a foe can isolate the EM radiation of one part, an attack should be possible.

In [14] authors foresee an overhead in area consumption of 25% with regards to basic WDDL and a speed decrease of about 20%. As it is a derivation of WDDL, functions must be positive.

IWDDL

Isolated WDDL (IWDDL), introduced by McEnvoy and al. in [117] is another scheme aiming at separating the "True" and "False" parts of WDDL, allowing a symmetrical placement and routing. This method consists in keeping the inverters in order to enable the separation and "superpipelining", i.e. inserting registers after each one, to avoid glitches and to allow the precharge.

From the robustness standpoint this architecture is appealing, however its overhead in terms of area is significant and the pipelining results in a decrease of the performances by a ratio of $1/(2\dot{n})$, n being the number of inverters in the critical path. Moreover it could also be subject to isolated EM attacks.

SDDL

Simple Dynamic Differential Logic (SDDL) presented by Tiri and Verbauvhede in [176] is also a separable DPL, in which all functions are usable, even negative ones (unlike WDDL). However, in order to allow the precharge to propagated through the entire design, such functions are coupled with dedicated logic as illustrated in Figure 3.4 in the case of a 2-input *XOR* gate. In this example, the $prch$ signal is set to 1 during the precharge phase, in order to force both output to 0.

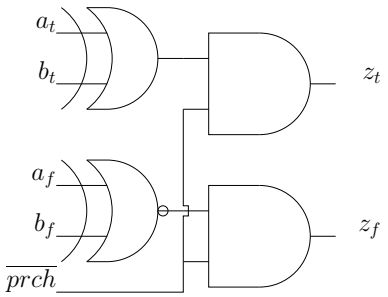


Figure 3.4: SDDL *XOR* Gate.

As it is separable, "True" and "False" parts could be copied and pasted, allowing a symmetrical placement and routing. However the issue of located EM attacks remains, as in most separable DPL. Moreover the use of negative logic introduces possible glitches, as in unlike WDDL, a gate could switch more than once per clock cycle.

In [184], Velegalati and Kaps recently proposed an improved implementation of SDDL AES on a Xilinx FPGA, by adding specific placement constraints, resulting in a security improvement, in terms of Measurements To Disclose (MTD), of a factor 27 with regards to the unprotected architecture and 2.3 w.r.t. the basic SDDL design.

Partial DDL

The idea of Partial Dynamic Differential Logic, presented in [92] by Kaps and al., is to protect only a fraction of the datapath, in order to decrease the area overhead of DPLs. As of now, DPLs generally protects only the datapath, where sensitive information are manipulated, while the control part is still single-railed. Moreover, SCA are always based on a leakage model and some part of the datapath are harder to model than other, hence less susceptible to be the target of such attacks. Therefore, Partial DDL only dualize the easily modeled portions, thus reducing the area consumption to less than 2 times the unprotected one.

As the chosen DPL in [92] is SDDL, drawbacks of this countermeasure, namely glitches and possible located EM attack remain an issue.

DWDDL

Double WDDL (DWDDL) was proposed by Yu and Schaumont in [192], to counteract any imbalance between the "True" and "False" networks, induced by the FPGA implementation of WDDL. As a matter of fact, DWDDL consists in duplicating a WDDL design, preserving the placement and routing by a copy/paste mechanism similar to SDDL, while inverting all signals, hence creating two opposite WDDL instances, with the exact same routing, but inverted "True" and "False" paths.

Although the security gain is significant with regards to a simple WDDL architecture, the overhead in terms of area, two times that of WDDL, makes it unfit for complex cryptographic FPGA designs.

MDPL

Masked Dual-rail Precharge Logic (MDPL), presented by Popp and Mangard in [135], is a dual-rail with precharge logic mixed with a masking scheme.

All logic operators are build from the MDPL *AND*, based on two majority gates, as shown in Figure 3.5(a). Figure 3.5(b) displays an MDPL register. Majority gates are used to ensure that all outputs switch values simultaneously. Every logic gate also has an additional input: a one-bit mask whose role is to randomly switch paths for true and false nets. Considering that the mask is plugged to a RNG, these two signals will randomly take one of the two possible routes. Therefore the security of this scheme does not rely on tedious placement or routing constraints, as all imbalances are randomized and should thus not be exploitable.

However, Schaumont an Tiri showed in [148] that the combination of DPL and masking in such a way could be inefficient. As a matter of fact, analysis of the power measurements probability density functions (*pdf*) can allow an adversary to distinguish between the different mask values. The effect of the mask can then be removed in order to perform a so-called *folding attack*.

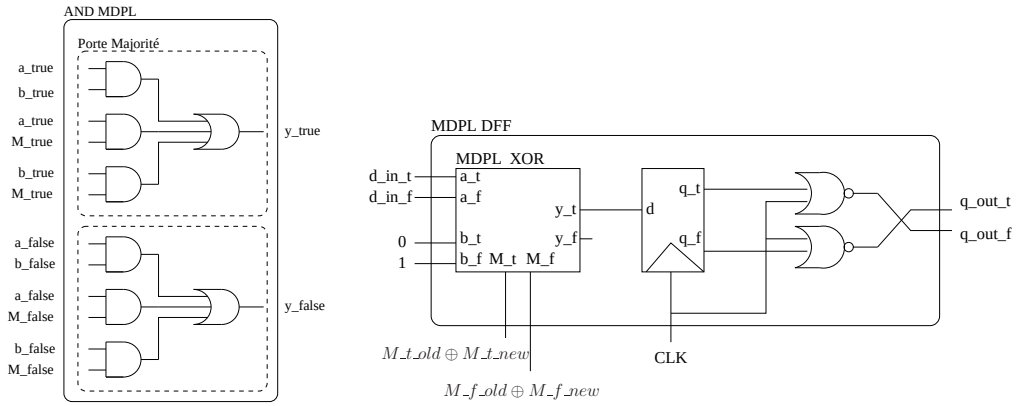


Figure 3.5: (a) MDPL AND. (b) MDPL DFF.

In [123] De Mulder and al. performed actual attacks on a prototype chip, illustrating on one hand the concept of the *folding attack* and proposing, on the other hand another scheme to break masked DPLs, so-called *subset attack*. By studying histograms of the mean power consumption for specific time samples, the authors are able to sort the measurements is several groups depending on the mask. Classical DPA is then successful when performed on a given group.

Moreover, Saeki and Suzuki presented another attack in [145] exploiting imbalances of the mask signal itself as an alternate SCA vulnerability.

As no FPGA implementation has been proposed yet, the overhead in terms of security and performances is ambiguous on such devices. The ASIC implementation results given in [135] show an increase of 4.5 times in terms of area and a decrease of 50% inn terms of speed.

iMDPL

Improved MDPL (iMDPL) was proposed by Popp and al. in [134]. This enhanced version of MDPL consists in adding synchronization logic before the basic MDPL gates, in order to avoid a particular DPL vulnerability so-called *early propagation effect* (EPE) (note that this topic will be properly discussed in Section 4.1). The result is a greatly improved robustness against classical first-order SCA like DPA, however the *folding attack* of [148] and *subset attack* of [123] may still be a threat to such countermeasure.

In terms of area consumption, the authors of [134] estimate an increase of roughly three times, with regards to the regular MDPL implementation.

DRSL

Dual-rail Random Switching Logic (DRSL) described in [39] by Chen and Zhou, also mix masking techniques and dual-rail logic. It is derived from MDPL and RSL (Random Switching Logic), which is a single-rail logic using an "enable" signal to synchronize the inputs before the evaluation. Figure 3.6 shows an example of a DRSL gate. The $NAND$ and OR gates are used to perform the synchronization and the 2 $RSL\ NAND$ gates are randomly selected depending on the mask $m_{(t,f)}$.

Nevertheless the attacks of [148], [123] and [145] are still applicable to DRSL. Moreover a possible glitch can occur due to lack of synchronization before the precharge as will be described in Section 4.1.4.

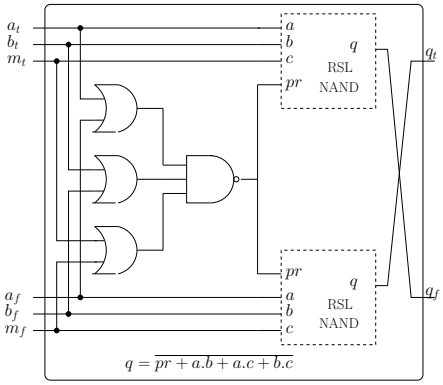


Figure 3.6: DRSL gate example.

STTL

Secure Triple Track Logic (STTL), presented by Soreas and al. in [164] is a triple-rail logic, which uses a specific synchronisation signal on every gate. This mechanism is illustrated in Figure 3.7(a). The ("X_v") signals must be slower than any other, as they represent the validity of the gate outputs. As a matter of fact, all inputs must be valid for a gate to actually evaluate or precharge. Therefore this technique provides a high level of security, as it shows quasi data-independent consumption and propagation delays and is completely glitch-free. Figure 3.7(b) shows and example of the implementation of an STTL AND gate.

Regarding this countermeasure, the authors foresee an area overhead of 11 times the surface of an unprotected implementation and a performance downgrade of a factor 5.

*

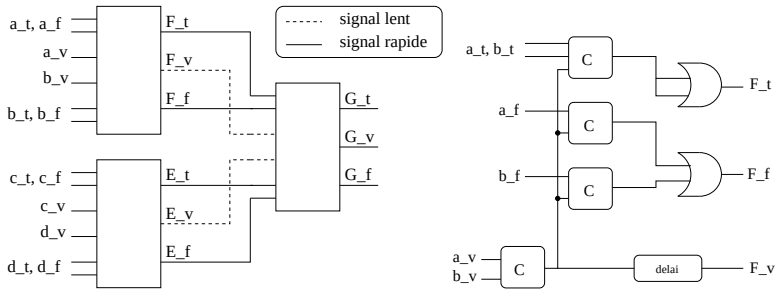


Figure 3.7: (a) STTL Gate Example . (b) STTL AND.

As of now, several types of DPLs have been proposed, introducing ingenious schemes to make up for known DPL vulnerabilities and thwart Side-Channel Attacks. However, all of them still present either a significant overhead in terms of area and/or performances, a lingering sensitivity to SCAs, or even both. Therefore additional research is mandatory in order to produce a DPL properly fitted for industrial applications. This could either be done by finding ways to enhance existing DPLs security and/or implementation cost, or by designing new DPL schemes aiming for both high robustness and low area and performance overhead.

3.3 Countermeasures Against Passive Attacks on Asymmetrical Algorithms

As described in Section 2 Side-Channel Attacks on asymmetrical ciphers target the core algorithms, respectively the *Exponentiation* 1 for RSA or the *Scalar Multiplication* 2 for ECC. Therefore, most SCA countermeasures focus on protecting these algorithms, generally by altering them or the operations within.

As before, the concept of adding randomness is widely exploited and comes in many flavors, potentially targeting several parts of these algorithms. However, the "hiding" idea, namely DPLs has never been evoked to protect asymmetrical ciphers.

In the following, countermeasures against passive attacks are discussed, most of them common to RSA and ECC, in which case both algorithms are given side by side.

3.3.1 Against Simple Analyses

As shown in Section 2.1.1, when no protection is present, the simplest attack can prove to be the most powerful. As a matter of fact, a Simple Power Analysis (SPA), namely one power trace, is theoretically sufficient to recover the whole secret key during an *Exponentiation* or *Scalar Multiplication*. Therefore, many countermeasures have been proposed to resist such attacks. They can be sorted in two main categories:

1. Those which tend to perform the same operations independently of the key bits value.

2. Those which aim at making the different computation indistinguishable.

Double-and-Add Always

The first idea is to remove the key-dependent conditional branch and compute both basic operations, squaring and multiplication, or point addition and doubling, for each key bit. This can be done by adding "dummy" operations as proposed by Coron [46], which gives the "Double-and-add always" (Algorithm 4) respectively "Square and Multiply Always" (Algorithm 3). This way both operations are computed at each step of the algorithm and a single SCA measurement will always display a steady succession of those operations.

Algorithm 3 Square and Multiply Always.

1: **Input**: $M, k = (1, k_{l-2}, \ldots, k_0)_2$
2: **Output**: $Q = M^k$
3: $Q_0 \leftarrow P$
4: **for** $i = l - 2$ **downto** 0 **do**
5: $\quad Q_0 \leftarrow Q_0^2$
6: $\quad Q_1 \leftarrow Q_0.M$
7: $\quad Q_0 \leftarrow Q_{k_i}$
8: Return Q_0

Algorithm 4 Double-and-add always.

1: **Input**: $P, k = (1, k_{l-2}, \ldots, k_0)_2$
2: **Output**: $Q = [k].P$
3: $Q_0 \leftarrow P$
4: **for** $i = l - 2$ **downto** 0 **do**
5: $\quad Q_0 \leftarrow 2.Q_0$
6: $\quad Q_1 \leftarrow Q_0 + P$
7: $\quad Q_0 \leftarrow Q_{k_i}$
8: Return Q_0

As one dummy operation (multiplication/addition) is added roughly 50% of the time, the expected downgrade in terms of performances is about 33%. Moreover a register must be added to store the dummy results, which is no negligible in the case of large input RSA or ECC.

It is noteworthy that this scheme itself is not sufficient to thwart all types of simple analyses. As a matter of fact, it is still sensitive to the so-called *doubling attack*, *Comparative Power Analysis*, "Safe-error Attack" and "Address-bit DPA" discussed in Section 2.1.1.

Montgomery Ladder

The Montgomery Ladder [122] algorithm offers another way of performing both operation at each iteration, independently on the key value, for either RSA (Algorithm 5) or ECC (Algorithm 6). The interesting characteristic is that no dummy operation is required, hence it it less sensitive to *computational safe-error attacks* how ever it can still be the siege of *memory safe-error attacks*, as shown by Kim and al. in [96].

Algorithm 5 Montgomery Ladder on RSA.

1: **Input**: $M, k = (1, k_{l-2}, \ldots, k_0)_2$
2: **Output**: $Q = M^k$
3: $Q_0 \leftarrow M$
4: $Q_1 \leftarrow M^2$
5: **for** $i = l - 2$ **downto** 0 **do**
6: $\quad Q_{1-k_i} \leftarrow Q_0.Q_1$
7: $\quad Q_{k_i} \leftarrow Q_{k_i}^2$
8: Return Q_0

Algorithm 6 Montgomery Ladder on ECC.

1: **Input**: $P, k = (1, k_{l-2}, \ldots, k_0)_2$
2: **Output**: $Q = [k].P$
3: $Q_0 \leftarrow P$
4: $Q_1 \leftarrow 2.P$
5: **for** $i = l - 2$ **downto** 0 **do**
6: $\quad Q_{1-k_i} \leftarrow Q_0 + Q_1$
7: $\quad Q_{k_i} \leftarrow 2.Q_{k_i}$
8: Return Q_0

However, by itself, it is still vulnerable to *Comparative Power Analysis* and "Address-bit DPA" discussed in Section 2.1.1.

As for the Double-and-Add Always, the expected overhead in terms of performance is of 33% and an additional register is required.

Using Only One Type of Operation

The security of the previously described algorithms is based on the fact that the same succession of basic operation, namely square and multiply, or double and add, take place at each iteration, independently of the key. This concept can be generalised by using only one type of operation for the entire calculation.

The "Universal Exponentiation Algorithm" was presented by Clavier and al. in [44]. The idea is to use a representation based on addition chains, hence reducing the security proof only to verifying that the basic operation is secure.

Before the actual algorithm, an addition chain $\mathbf{C}(k)$ of length l is computed for the key k. At step i, k is represented by the register sequence:

$$\Gamma(k) = (\gamma(i) : \alpha(i), \beta(i))_{1 \leq i \leq l} \tag{3.3}$$

Meaning that the contents of register $R[\alpha(i)]$ must be added/multiplied by $R[\beta(i)]$ and the result stored in $R[\gamma(i)]$. the "Universal Exponentiation Algorithm" (Algorithm 7 for RSA and Algorithm 8 for ECC) takes advantage of this representation to efficiently compute any exponentiation.

Algorithm 7 "Universal Exponentiation Algorithm" on RSA.

Input: $M, \Gamma(k) = (\gamma(i) : \alpha(i), \beta(i))_{1 \leq i \leq l}$
Output: $Q = M^k$
$R[\alpha(i)] \leftarrow M$
$R[\beta(i)] \leftarrow M$
for $i = 1$ **to** l **do**
    $R[\gamma(i)] \leftarrow R[\alpha(i)].R[\beta(i)]$
Return $R[\gamma(l)]$

Algorithm 8 "Universal Exponentiation Algorithm" on ECC.

Input: $P, \Gamma(k) = (\gamma(i) : \alpha(i), \beta(i))_{1 \leq i \leq l}$
Output: $Q = [k].P$
$R[\alpha(i)] \leftarrow P$
$R[\beta(i)] \leftarrow P$
for $i = 1$ **to** l **do**
    $R[\gamma(i)] \leftarrow R[\alpha(i)] + R[\beta(i)]$
Return $R[\gamma(l)]$

This method present the advantage of being provably secure against SPA (if the elementary operation $R[\gamma(i)] \leftarrow R[\alpha(i)]./ + R[\beta(i)]$ is leakage-free). Its performances are also interesting, as, given a well calculated addition chain, it can require about $1.25n$ additions/multiplications (n being the length of the key) which is far better than the *Double an Add Always* which requires $1n$ of each basic operations.

However, as the addition chain must be recomputed for each new key, this method is hardly applicable to systems that often change their secret keys. Moreover, the computation of the addition chain must be performed in a secure environment, as its sole knowledge would allow an adversary to recover the key.

The atomicity concept, proposed by Chevalier-Mames and al. in [40] is another way of using only one type of operation. It is based on performing the computation with "atomic blocs" that are indistinguishable from the side-channel point of view. A simple example is given in Algorithm 9 in the case of RSA, considering that multiplication and a squaring are performed with the same hardware.

Algorithm 9 Atomic Square and Multiply.

Input: $M, k = (1, k_{l-2}, \ldots, k_0)_2$
Output: $Q = M^k$
$Q_0 \leftarrow 1$
$Q_1 \leftarrow M$
$i \leftarrow l - 2$
$b \leftarrow 0$
while $i \geq 0$ **do**
    $Q_0 \leftarrow Q_0.Q_b$
    $b \leftarrow b \oplus k_i$; $i \leftarrow i + k_i - 1$
Return Q_0

In [40], author further develop this concept to smaller atomic blocs containing basic modular operations. This way such methodology can be applied to a wide range

of algorithms, including the ECC. In terms of performances and complexity, this countermeasure is almost equivalent to an unprotected design. However, the assumption that the basic operations are indistinguishable is hard to satisfy, specially for the ECC, as shown in the next Section 3.3.1 and care has to be taken to ensure that the computation of $b \leftarrow b \oplus k_i$ and $i \leftarrow i + k_i - 1$ does not leak any sensitive information (the authors propose to mask such operations). Moreover, this scheme should still be vulnerable to chosen-input simple analyses, namely *doubling attack* and *Comparative Power Analysis*.

Making Addition and Doubling Indistinguishable

The vulnerability to most simple analyses comes from the differences in the computation of the core operations. In the domain of Elliptic Curve Cryptography, several schemes have been proposed to remove possible distinction between formulas for point addition and doubling. They rely either on mathematical modification of those formulas (by using additional modular calculations), or on intrinsic properties of specific curves.

Unified addition and doubling formulas were first described by Brier and al. in [32]. They allow to use the same calculations for both addition and doubling, however they unintentionally create a loop hole, by returning an error in the particular case where the two input points have opposite y-coordinate.

Izu an al. exploited this behaviour to propose an attack so-called "Exceptional Procedure Attack" in [83]. Walter also presented an attack in [188], based on the fact that modular multiplication algorithms (like Montgomery's for instance) often compute a conditional subtraction just before the result depending on the input. Then, supposing a foe can detect such an operation, it can lead to distinguishing between the operations with one type of input, namely squaring and doubling and the others.

Finally Amiel and al. proposed another attack, effective only on the "Double and Add" algorithm, based on the fact that when performing a doubling, some modular multiplication will become modular squaring and that such a different can be exploited by an adversary.

In order to avoid such vulnerability, Dechene and al. described a generalisation of Brier's formulas in [79], removing the specific behaviour that lead to Izu's attack.

However, Stebila and al. showed in [172] that Walter's attack [188] was still applicable. Moreover, Amiel's attack is also effective on those formulas, when the "Double and Add" algorithm is employed.

Specific curves possess the intrinsic property of having unified addition and doubling formulas, this is the case for Jacobian [107], Hessian [88] and Edward's [54] curves. However, these are not standard curves, which implies that their security against cryptanalytic attacks has not been proven.

3.3.2 Against Statistical Attacks

Protection against simple analyses and statistical attacks are two different topics, indeed none of the previously presented SPA countermeasure is efficient against DPA and other similar attacks, however both type of countermeasure are mandatory to develop a secure design.

As of now, the main concept for protecting asymmetrical algorithms against such attacks is the randomisation. It comes in many flavors, but in the end the idea is either to randomise the secret key, the input data or, in the case of ECC, the curve itself. In most cases, the exponentiation (respectively scalar multiplication) algorithm is modified in order to combine SPA countermeasures as well as randomisation and sometimes even fault detection mechanisms.

Blinding the Input

The first countermeasure was proposed by Kocher in [97, 98] to protect RSA exponentiation against timing attacks and extended to the ECC by Coron in [46]. The general idea is close to the masking for symmetrical algorithms.

Before the computation, a pair of scalars (r_i,r_f) such as $r_f^{-1} = r_i^k \mod n$, respectively points ($R_i = (x_i, y_i)$,$R_f = (x_f, y_f)$) such as $R_f = [-k].R_i \mod p$ (where n is the public modulus and the p the ECC prime number) are chosen randomly. Then, before launching the algorithm, the input is "blinded" by this random entity by respectively multiplying the input message by $r_i \mod n$ or adding $R_i \mod p$ to the input point. Finally, after the computation, the actual result can be extracted by multiplying the output with $v_f \mod n$, respectively adding $R_f \mod p$.

The evaluation of overheads in terms of area and performances for this countermeasure, depends on the way to update the random number.

As a matter of fact, the basic idea is to draw a new random entity at each iteration of the core algorithm. In this case, the additional steps that must be undertaken, with regards to an unprotected algorithm can be described as follows:

1. Generate the random r_f respectively R_i. While r_i is a simple random number such as $r_i < k$, k being the secret, R_i is a point of the considered elliptic curve. Generating such a point can be done mainly in two different manners: by choosing a random x-coordinate and computing the corresponding y-coordinate using the equation of the curve (note that this method does not systematically lead to the generation of a valid point, as there is no way of ensuring that the random number actually corresponds to a valid x-coordinate). Or by performing a scalar multiplication between a random number and the curve generator, which allows the systematic creation of a valid point, at the cost of an additional scalar multiplication, hence a doubling of the computation time, as the chosen random number should be of approximately the same length as the secret k.

2. Computing $r_i = (r_f^{-1})^e \mod n$, where e is the public exponent, via a modular inversion (which is known to be extremely costly in terms of performances) and an exponentiation. And respectively $R_f = [-k].R_i \mod p$ with another scalar multiplication (obviously care should be taken to protect this operation against SPA).

3. After performing the blinded exponentiation: $Q = (m.r_i)^k \mod n$ or scalar multiplication: $Q = [k].(P + R_i$, the actual results are obtained by a multiplication with r_f (mod n), respectively a point subtraction with R_f.

As the input is randomized, this method can resist any chosen-input attack like the *Comparative Power Analysis*, as well as statistical SCAs like DPA and timing attacks. Moreover it also thwarts the RPA and ZPA in the case of ECC, as control of the input point is also mandatory for those to be successful.

Eventually, the generation of a new random at each iteration proves to add a significant computational cost, of roughly one exponentiation plus one modular inversion for RSA and two scalar multiplications for ECC, which leads to a performance downgrade of at least a factor 2, respectively 3 (not taking into account other operations that can be considered as negligible).

Therefore, Kocher [97], respectively Coron [46], proposed a way to refresh the random pairs (r_i,r_f) and (R_i,R_f), without drawing a new one at each iteration. Indeed, on one hand (r_i,r_f) can be updated by computing $r'_i = r_i^2$ and $r'_f = r_f^2$ and the other hand a new (R'_i,R'_f) can be computed such as $R'_i = [(-1^b).2].R_i$ and $R'_f = [(-1^b).2].R_f$, where b is a random bit.

However, it has been proven that these methods induce a vulnerability to *doubling attack* and *Comparative Power Analysis*. As a matter of fact, if an adversary can chose the input, he could successively compute $Q_1 = (m.r_i)^x$ and $Q_2 = (m^2.r_i^2)^x = ((m.r_i)^x)^2 = Q_1^2$ and perform a successful "Doubling Attack" on RSA. This is also true for ECC with $Q_1 = [k].(P + R_i)$ and $Q_2 = [k].(2.P + 2.R_i) = [k].2.(P + R_i) = 2.Q_1$ with a 50% chance due to the random b.

In the end, the blinding technique can be considered as a robust countermeasure against chosen-input attacks and statistical SCAs when the refreshing of the blinding entity (scalar or point) is performed in a random manner at each iteration. In this case the overhead in terms of performance is significant, indeed the designer can expect a downgrade by at least a factor 2 and 3 for respectively RSA and ECC. Another refreshing method exists to greatly reduce the performance loss, however it brings out a vulnerability to chosen-input SCAs.

BRIP Algorithm

Several papers have dealt with the combination of blinding techniques and SPA resistant algorithms. The interesting properties of such methods are on one hand to ensure robustness to both SPA and statistical attacks and on the other hand to reduce the impact of blinding on the design performances, by merging it with the algorithm. Indeed the additional exponentiation/scalar multiplication required to compute r_f and R_f as described in the second step of Section 3.3.2 can be directly performed within the algorithm, hence almost removing its effect on the global performances.

Such a scheme so-called BRIP was proposed by Mamiya and al. in [114]. This countermeasure was initially designed for ECC on smart-cards, but can perfectly be adapted to FPGA implementations and RSA. Algorithm 10 and Algorithm 11 give the definition of BRIP on both RSA and ECC. It is a subtle combination of "Double-and-Add Always" from MSB to LSB and blinding or Random-Initial-Point (RIP).

Algorithm 10 BRIP on RSA.

1: **Input**: $M, k = (1, k_{l-2}, \ldots, k_0)_2$
2: **Output**: $Q = M^k$
3: r = *random scalar*
4: Compute r^{-1}
5: $Q \leftarrow r$, $Q_0 \leftarrow r^{-1}$, $Q_1 \leftarrow M.r^{-1}$
6: **for** $i = l - 1$ **downto** 0 **do**
7: $\quad Q \leftarrow Q^2$
8: $\quad Q \leftarrow Q.Q_{k_i}$
9: $Q \leftarrow Q.Q_0$
10: Return Q

Algorithm 11 BRIP on ECC.

1: **Input**: $P, k = (1, k_{l-2}, \ldots, k_0)_2$
2: **Output**: $Q = [k].P$
3: R = *random point*
4: Compute $-R$
5: $Q \leftarrow R$, $Q_0 \leftarrow -R$, $Q_1 \leftarrow P - R$
6: **for** $i = l - 1$ **downto** 0 **do**
7: $\quad Q \leftarrow 2.Q$
8: $\quad Q \leftarrow Q + Q_{k_i}$
9: $Q \leftarrow Q + Q_0$
10: Return Q

An additional register is required, with regards to the basic "Double-and-Add Always" algorithm, in order to store the input random entity, which induce a non negligible area increase. Moreover, the generation of this random input and computation of its inverse are still mandatory, which corresponds to either a modular inversion or scalar multiplication for respectively RSA and ECC, as discussed in Section 3.3.2.

BRIP is theoretically robust against all types of simple attacks, even with chosen-input, as well a statistical SCAs including RPA and ZPA on ECC. However, some fault attacks should still be efficient, especially "Safe Errors". As a matter of fact, if an adversary could induce a fault in either register Q_0 or Q_1 before step 8 of the algorithm, the value of the corresponding key bit could directly be deduced from the faulty/non-faulty result.

Blinded Fault Resistant Algorithm

In [31], Bosher and al. proposed a new, right-to-left algorithm, initially designed for RSA but applicable to ECC, which includes a fault detection mechanism, in addition to SPA protection and input blinding.

This scheme, described by Algorithm 12 and Algorithm 13 is as robust to simple and statistical SCAs as the BRIP presented in Section 3.3.2, but can also thwart several fault attacks, including "Safe Errors", thanks to the additional test performed in steps that verifies the validity of the three registers used during the computation before outputting a result.

Algorithm 12 Blinded Fault Resistant Algorithm on RSA.

1: **Input**: $M, k = (1, k_{l-2}, \ldots, k_0)_2$
2: **Output**: M^k
3: r = *random scalar*
4: Compute r^{-1}
5: $Q_0 \leftarrow r$, $Q_1 \leftarrow r^{-1}$, $Q_2 \leftarrow M$
6: **for** $i = 0$ **to** $l - 1$ **do**
7: $\quad Q_{1-k_i} \leftarrow Q_{1-k_i}.Q_2$
8: $\quad Q_2 \leftarrow Q_2^2$
9: **if** $(Q_0.Q_1.M = Q_2)$ **then**
10: $\quad$ Return $(r^{-1}.Q_0)$
11: **else**
12: $\quad$ Return $(Error)$

Algorithm 13 Blinded Fault Resistant Algorithm on ECC.

1: **Input**: $P, k = (1, k_{l-2}, \ldots, k_0)_2$
2: **Output**: $[k].P$
3: R = *random point*
4: Compute $-R$
5: $Q \leftarrow R$, $Q_0 \leftarrow -R$, $Q_1 \leftarrow P$
6: **for** $i = 0$ **to** $l - 1$ **do**
7: $\quad Q_{1-k_i} \leftarrow Q_{1-k_i} + Q_2$
8: $\quad Q_2 \leftarrow 2.Q_2$
9: **if** $(Q_0 + Q_1 + P = Q_2)$ **then**
10: $\quad$ Return $(Q_0 - R)$
11: **else**
12: $\quad$ Return $(Error)$

In terms of area and performances, this algorithm should be similar to the BRIP (considering that the test is negligible), as one additional register is required with regards to the "Double-and-Add Always", as well as the generation of the random input.

Randomization of the Secret

First introduced by Kocher in [97] as a possible countermeasure against timing attacks on RSA then precised and extended to ECC by Coron in [46], this countermeasure consists in randomizing either the secret exponent or scalar before each computation. In the case of RSA, this is done by adding a multiple of the order of the elements $\phi(n)$ to the secret exponent k, such as $k_i = k + \lambda_i.\phi(n)$. Then, the protected exponentiation $Q' = M^{k_i} \mod n$ is performed and the result is such as $Q' = M^k \mod n$, as $M^{\lambda_i.\phi(n)} \mod n = 1$. In the same way, the protected scalar multiplication can be computed as $Q' = [k_j].P = [k + \lambda_j.\#E] \mod p = [k].P \mod p$, where $\#E$ is the cardinal of curve E and P a point of E.

The implementation cost of this countermeasure is low, as only an extra addition and multiplication are required and the existing ECC operators could likely be used to perform these computations. Nevertheless the performance downgrade is significant, for two reasons. First of all, the drawing of a large random number, however this is the case for almost all statistical SCA countermeasures on asymmetrical algorithm, as they are usually based on some kind of randomization. Second, the new key k_i is larger than the original k, indeed the length of k_i is the sum of those of k and λ_i. Therefore either λ_i is chosen short and the security level brought by this scheme is low, or it is long and the performance overhead will be significant.

In terms of security, the randomisation of the secret implies robustness against statistical attacks like DPA, as well as RPA and ZPA with regards to ECC. However, this countermeasure has been proven vulnerable to *doubling attack* in [9] assuming that no randomisation is added on the input. Moreover, Ciet [41] put to light that some high weight bits of the secret key may not be affected by this randomization, depending on the curve and more precisely on the characteristics of $\#E$.Finally, as discussed in Section 2.2.4 using a carry-leakage based attack [56], Fouque and al. recently showed that the entire secret can be recovered by targeting the initial addition with the random rather than the actual computation of the core algorithm.

Random Splitting of the Secret

This countermeasure was first proposed by Ciet and Joye in [42] concerning the ECC, but is perfectly applicable to RSA. The idea is to protect the secret exponent or scalar, by randomly splitting it in two before each execution, computing the core algorithm on the two parts in parallel and then merging those to retrieve the actual result. As a matter of fact, the exponentiation $Q = M^k \mod n$ can be computed as $Q = Q_1.Q_2 \mod n = M^r.M^{k-r}$ where k_1 is a random number. In the same way a scalar multiplication $Q = [k].P \mod p$ is splitted as $Q = Q_1 + Q_2 \mod p = [r].P + [k-r].P \mod p$.

It directly follows that the performances are roughly the same as an unprotected implementation, while the required area is multiplied by a factor two, indeed the initial and final computations, namely a subtraction and respectively a modular multiplication and point addition for RSA and ECC are negligible with regards to the core algorithms. Note that alternate way of performing the splitting have been discussed in [180] and [87]. In the latter, Joy propose to compute the scalar multiplication as $Q = Q_1 + Q_2 \mod p = [k \mod r].P + [k/r].([r].P) \mod p$, which induce a much more complex architecture, but can be used to reduce the size of r.

This countermeasure is secure against all existing statistical SCAs, including RPA and ZPA, as well as the chosen-input simple analyses, as the secret key is randomized at each computations.Moreover, it helps thwarting several fault attacks, as a fault

induced on one of the paths, even a safe error will not directly allow to draw a conclusion on the value of an actual secret key bit, but rather on a bit of a random key, only valid during one call to the core algorithm. The carry-leakage based attack of [56] does not apply in this case, as long as the random number r is not chosen such as $r = \lambda.\phi(n)$ or $r = \lambda.\#E$, for respectively RSA and ECC.As of now, the only potential attack scheme against this countermeasure has been proposed by Muller and Valette in [124], where they put to light a weakness to second-order safe-error attacks and address-bit DPA. Therefore in order to ensure a total security, the random splitting could be combined with other countermeasures against such attacks.

Against Address-bit DPA

As stated in Section 2.2.3 the address-bit DPA is a statistical attack that rather than targeting the value of the registers, focus on their addresses. Therefore, protection against such scheme can be achieved by randomizing the order in which those registers are used. Based on this idea, Itoh and al. proposed a modified scalar multipication algorithm in [82]. However this scheme was later attacked by Masami Izumi and al. [84], who presented an enhance version of the previous countermeasure, depicted in Algorithm 14.

Algorithm 14 Montgomery ladder with randomized addresses.

Input: $M, k = (1, k_{l-2}, \ldots, k_0)_2$
Output: M^k
r = *random scalar*
Compute r^{-1}
$Q_0 \leftarrow r$, $Q_1 \leftarrow r^{-1}$, $Q_2 \leftarrow M$
for $i = 0$ **to** $l - 1$ **do**
    $Q_{1-k_i} \leftarrow Q_{1-k_i}.Q_2$
    $Q_2 \leftarrow Q_2^2$
if $(Q_0.Q_1.M = Q_2)$ **then**
    Return $(r^{-1}.Q_0)$
else
    Return $(Error)$

This scheme makes use of an additional register in order to perform the address shuffling and requires the generation of a random number, the size of the secret key, for each new computation.

3.3.3 ECC Specific Countermeasures

Some countermeasures were specially developed, to take advantage of the mathematical properties of the ECC. The general goal is to add randomness to certain aspects

of the ECC representation in order to thwart statistical SCAs. As of now three major ideas have proposed and consist of randomizing either the projective coordinates, the curve itself or the field on which it is defined.

Random Projective Coordinates

Proposed by Coron in [46], this countermeasure can be deployed when the computation $Q = k.P$ is performed in projective coordinates [45], which is often the case for performance reasons. The input base-point $P = (x, y)$ is then represented by $P' = (\theta x, \theta y, \theta)$ in the case of homogeneous projective coordinates or $P' = (\theta^2 x, \theta^3 y, \theta)$ for Jacobian projective coordinates, with θ is such as: $\theta \in \mathbb{K}^*\theta$, where $\mathbb{K}$ is the finite field over which the curve is defined. For more clarity, let's note $P' = (X, Y, Z)$.

For the sake of simplifying the computations and increasing the performances, Z is usually chosen equal to 1. However, the idea behind this countermeasure, is to draw a random Z for each new input, compute $Q' = (X', Y', Z') = k.P'$ and then compute $Q = (X'/Z', Y'/Z')$ or $Q = (X'/Z'^2, Y'/Z'^3)$ for respectively homogeneous and Jacobian projective coordinates.

The main advantage of this countermeasure is the little impact on the system performances. Indeed, other than generating a random scalar and using the computation formulas without the simplifications brought by the choice of $Z = 1$, no modification is required.

As the input is randomized, this countermeasure is robust against classical statistical SCAs like DPA, however is can still be attacked by the ECC targeting RPA and ZPA [67] described in Section 2.2.5. As a matter of fact, the specific 0 value exploited by those schemes cannot be randomized by this technique, as $\theta.0 = 0, \ \forall\theta$.

Random Field Isomorphisms

This countermeasure was proposed by Joye an Tymen in [89] and applies to an elliptic curve E defined over a field $\mathbb{K} = GF(2^m) = GF(2)[X]/\Pi(X)$, where Π is an irreducible polynomial of degree m over $GF(2)$. The idea is that there are many such irreducible polynomials, so that $\mathbb{K}$ can be randomly replaced by an isomorphic field $\mathbb{K}'$. The computation of $Q = k.P$ is then performed as follows:

1. Choose a random irreducible polynomial Π' of degree m over $GF(2)$ and let $\mathbb{K}' = GF(2)[X]/\Pi'(X)$.
2. Let ϕ be the field isomorphism between $\mathbb{K}$ and $\mathbb{K}'$ and $P' = \phi(P)$.
3. Compute $Q' = k.P' \in \mathbb{K}'^2$, in $E_{/\mathbb{K}'}$.
4. Compute $Q = \phi^{-1}(Q') \in \mathbb{K}^2$.

However, this countermeasure is once again attackable by RPA and ZPA as the randomization does not affect all specific 0 values [67].

Random Elliptic Curve Isomorphisms

In [89], Joye an Tymen also presented this method, which consists in computing the scalar multiplication on a random isomorphic curve and then coming back to the original one. This method can be applied to any elliptic curve $E(\mathbb{K})$ such as $E_{/\mathbb{K}} : y^2 = x^3 + ax + b$, defined over field $\mathbb{K}$ of characteristic $\neq 2, 3$. Algorithm 15 precisely describes the different step of the countermeasure.

Algorithm 15 Scalar multiplication with Random Elliptic Curve Isomorphism.

1: **Input**: $P(x, y), k = (1, k_{l-2}, \ldots, k_0)_2$.
2: **Output**: $Q = [k].P$.
3: r = *random element* of $\mathbb{K}^*$.
4: Form point $P'(r^{-2}.x, r^{-3}.y)$.
5: Compute $a' = r^{-4}.a$.
6: Compute $Q'(x'_Q, y'_Q) = [k].P'$ in $E'(\mathbb{K})$ such as $E'_{/\mathbb{K}} : y^2 = x^3 + a'x + b'^{\dagger}$.
7: **if** $(Q' = infinity(\mathcal{O}))$ **then**
8: Return $\mathcal{O}$.
9: **else**
10: Return $Q(r^2.x'_Q, r^3.x'_Q)$.

† note that b' is not required to perform the scalar multiplication.

An interesting property of this scheme is to be compatible with all types of projective coordinates. As a matter of fact, P' in step 4 of this algorithm can be defined as $P'(r^{-2}.x, r^{-3}.y, 1)$, corresponding to a projective representation with $Z = 1$.

However, this scheme is still vulnerable to RPA and ZPA, as the specific 0 values are not randomized [67]. For instance, a coordinate equal to 0 for an input point $P_{RPA}(x, 0)$, will not be affected by the randomization of step 4.

*

Numerous countermeasures have been developed in the past few years to thwart classical SCAs like timing attacks, SPA or DPA, inducing a wide range of possible overheads for both area and performances. However, as can be expected, several ingenious attack schemes were successful in breaking most of them, especially the relatively low-cost techniques like the second variant of point blinding presented in Section 3.3.2 which proves to be vulnerable to *doubling attack* and *Comparative Power Analysis*, or the random projective coordinates of Section 3.3.3 that can be broken using RPA or ZPA. As of now, only the basic blinding (where a random scalar or point is chosen at each execution of the core algorithm) and the random key splitting are secure against both chosen-input simple attacks and statistical analyses, including RPA and ZPA in the case of ECC, however they must also be combined with a classical SPA resistant scheme, as they alone are not perfectly secure against such attacks.

In other words, there is no perfect countermeasure that can resist all existing SCAs on asymmetrical algorithm and show very little overheads in terms of performance and area, but depending on the required level of security and acceptable implementation cost, one can combine two or more countermeasures to meet those expectations. Moreover, the protection choice should be customized, taking into account the specificity of each given design. For instance, if a device does not give the possibility of selecting the inputs in any way, then chosen-input SPAs are not a threat and the low-cost blinding can be deployed as a countermeasure against statistical attacks.

Nevertheless, a thorough security against SCA must also include fault detection mechanisms, as the robustness against passive SCAs does generally not induce resistance to perturbation attacks.

3.4 Countermeasures Against Fault Attacks

3.4.1 Generic Countermeasures

Redundancy

One of the most commonly used fault detection mechanism is the redundancy, which can be in time or space. The global idea is to compute several instances (usually two) of the considered algorithm and comparing the results. When those instances are both implemented and computed in parallel we speak of space redundancy, which induces an overhead in area consumption of a factor 2. An other possibility is to systematically perform encryption as well as decryption and verify that the results are coherent. In this case the performances are divided by 2.

Such a scheme was proposed for AES by Karri and al. so-called *Concurrent Error Detection* (CED) in [94]. The CED can be implemented at the algorithm, round or even operation level and detect both permanent and transient faults.

Another redundancy-based countermeasure was described by Maisitri and Leveugle in [113], called *Double-Data-Rate*. It consists in duplicating the registers while sharing the combinatorial parts, in order to compute the algorithm on both clock edges. The results are an area overhead of 36% and a throughput of $15 - 55\%$.

Parity an Cyclic Redundancy Check (CRC)

Adding parity bits or CRC are common techniques to detect and correct faults in several fields like the communication. Such methods can be used as SCA countermeasures as showed for instance by Bertoni and al. in [19, 20], which proposed a low-cost countermeasure for AES using a single parity bit. This scheme detects single fault with a coverage of 96.3% for an area overhead of 18%. In [189] Yen and al. presented another countermeasure using $(n+1, n)$ CRC on AES with $n \in 4, 8, 16$ for respectively a 8-, 32- and 128-bit architectures, that showed better performance in detecting multiple faults.

3.4.2 Specific Schemes for Asymmetrical Algorithms

Coherency Check

The *coherency check* can be used to verify the validity of intermediate results during the computation of an exponentiation of scalar multiplication. For instance when implementing the *Montgomery ladder* on ECC depicted in Algorithm 6, checking that $Q_1 - Q_0 = P$ is always true is a possible way to detect faults as described in [131]. Moreover the *coherency check* can be directly integrated in the algorithm, as stated in Section 3.3.2 at the cost of an additional register .

This countermeasure can efficiently detect *differential fault*, *sign-change* and *safe-error* attacks.

Curve Integrity Check (ECC)

Curve parameters can be the siege of fault attacks, therefore, they should be verified before the execution of the scalar multiplication, for instance using CRC (Cycling redundancy Check).

Point Verification (ECC)

This method verifies if a point actually lies on the right curve or not. Let $E : y2 + xy = x3 + ax2 + b$ be an elliptic curve defined over $GF(p)$, the validity of a point $P(x_p, y_p)$ can be checked by simply verifying the equation: $y_p^2 + x_p.y_p = x_p^3 + a.x_p^2 + b$. This simple verification does not induce a significant area or performance overhead, as only a few modular operations are required depending on the curve's equation. Moreover it is able to prevent several fault attacks, including *differential*, *invalid point* and *twist curve* attacks. In order to be efficient, it should be deployed at least at the beginning and the end of the scalar multiplication and, depending on the performance needs, within the algorithm.

*

With the increasing number of ingenious physical attack schemes against cryptographic algorithms, it is hardly possible to ensure the security of a device with only one type of countermeasure. In order to be secure against simple and statistical SCAs as well as FA and even specific dedicated attacks, designers should carefully choose a combination of protections, depending on the target algorithm and acceptable complexity.

Regarding perturbation attacks, most faults can be detected by time or space redundancy and parity bits as well as customized verification schemes implemented at different stages of the considered algorithms.

SCAs are usually regarded as a bigger threat, as they are relatively low-cost and undetectable by the target devices. The major types of countermeasures against such

attacks are *obfuscation*, *desynchronization*, *key updating*, *DPLs* and *masking*. The first two can be used independently of the algorithm, but only induce an increase of the required number of side-channel measurements. *Key updating* consists in limiting the available number of traces and can therefore be sound when confronted to known adversaries. However it is not compatible with several protocols and could be defeated by future SCAs potentially requiring less measurements to be successfully performed. A wealth of *DPLs* have be presented to protect symmetrical algorithms. Nevertheless, in most cases, their implementation on FPGA induces either a specific vulnerability against SCAs, significant overheads in terms of performance and complexity, or both. *Masking* is maybe the most commonly used countermeasure and almost the sole concept for thwarting statistical SCAs on asymmetrical algorithms. However it proves to be vulnerable to several SCAs (e.g. VPA) and also shows significant increase in resource consumption, with regards to unprotected architectures, when implemented on FPGA.

Chapter 4

New DPL Countermeasures for Symmetrical Algorithms

Although many countermeasures have been developed to protect symmetrical algorithms (specially DES and AES), most of them are still unfit for actual industrial usage. This can be due to unrealistic overheads in either area consumption or performances, as well as lingering vulnerabilities to specific attacks or security flaws induced by the implementation on FPGA. It is therefore mandatory to improve existing countermeasures or design new ones, aiming for the best trade-off between security, performances and area.

4.1 DPL Vulnerabilities

As depicted in Section 3.2.2, DPLs aim at making the power consumption constant therefore independent of any sensitive data. However, DPL logic can still leak information when "true" and "false" parts of the same instance evaluate at different times, which leads to possible attacks. Moreover, implementation on FPGA tends to exacerbate existing imbalance due to a lack of control on net routing and gate consumption. The following sections will describe the two major phenomenons causing those imbalances, called "early propagation" and "technological bias".

4.1.1 Early Propagation Effect

When a DPL gate switches between phases, input signals acquire their respective values. Switching input signals are likely to acquire respective logic value at different times due to differences in logical path, even if the gates are balanced. As a matter of fact, if the transition probability of the gate is unity, the gate will evaluate without waiting for all signals to acquire the right value, which causes the operation to start at different times in subsequent acquisition. This phenomena was first introduced by

Suzuki in 2006 [103, 173] as "early evaluation". Considering that such desynchronization may occur during either precharge or evaluation, we talk of respectively "early precharge" and "early evaluation". The global phenomena shall be referred to as *early propagation effect* (EPE) in the remainder of this manuscript.

Figure. 4.2 illustrates the principle of early evaluation for a 2-input AND gate and its dual 2-input OR gate, as represented on the Figure 4.1.

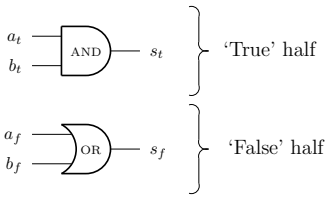


Figure 4.1: WDDL Gate Example.

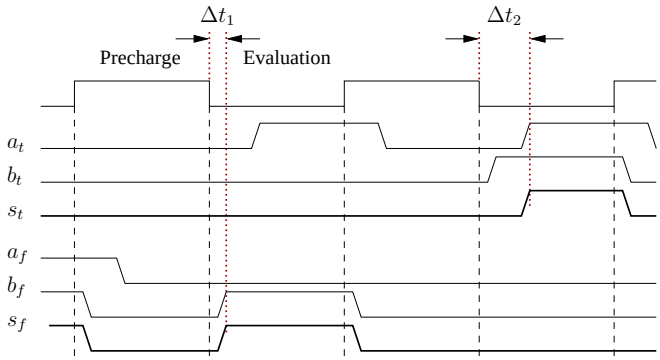


Figure 4.2: Early Evaluation.

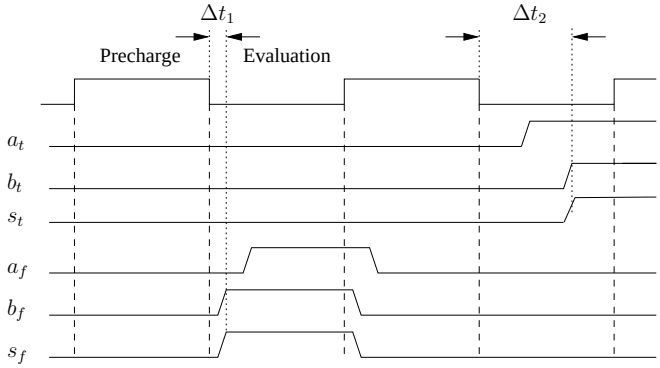


Figure 4.3: Early Evaluation Combined With Imbalance of Dual Nets.

In this example, it is clear that, *depending on the inputs value*, the switching times differs between AND and OR gates. On one hand, the OR gate evaluates as soon as one of its inputs is set to '1', on the other hand, the AND must wait until each of its entries are set to '1'. Of course, the opposite behavior would take place if the first available input had been '0' (the AND gate would have evaluated immediately and the OR gate would have waited for every inputs to be set).

Moreover, in dual-rail logic and specially when implemented on FPGA, the delay between switching times is strengthened by two main factors:

1. The difference of logical paths between several inputs of a given logical gate, due to the fact that they didn't pass through the same number of logical layers

as shown in Figure 4.4.

2. The imbalance between "true" and "false" inputs, due to placement and routing differences, also produces a timing delay between corresponding nets of a dual-rail signal. This phenomenon is exacerbated by the FPGA implementation, where routing is usually performed by CAD tools that do not provide straightforward ways to perfectly balance the routing of dual nets. Then, as shown on Figure 4.3, if the "true" input b_t is slower than its dual b_f, the delay Δt_2 could increase even more.

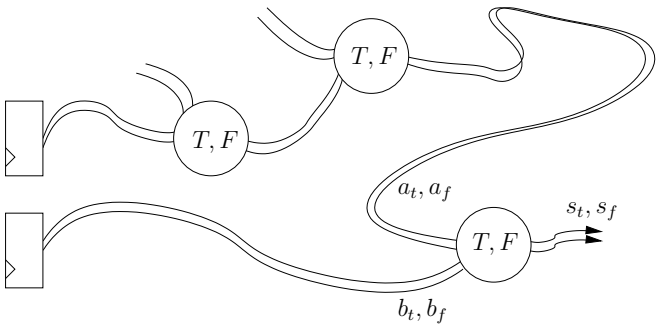


Figure 4.4: Unbalance in logical paths.

To summarize, if for a given signal, the delay Δt defined as $\Delta t \doteq \Delta t_1 - \Delta t_2$, can be detected within power measurements, the activity of the node will be monitored, which will lead to sensitive information leakage and possible attacks.

4.1.2 Technological Bias

The other major vulnerability in DPL circuits is the technological bias. It comes from three factors. Firstly, the power consumption of a gate and its complement are not necessarily the same when implemented on FPGA. Secondly, technological variations can occur resulting form slight shifts during the manufacturing process. Finally, non-identical routing of true and false paths induce a bias in the consumption of corresponding gates, due to different output capacitive load [177]. As stated in the previous Section 4.1.1 this phenomena also affect the switching times of dual nets, which increases the EPE.

In order to estimate the effect of this network imbalance, the 8 DES S-Boxes were implemented on a EP1S25 Stratix FPGA. We used compact S-Boxes, designed specifically for FPGA as described in [70]. This first study is a proof of concept, therefore performed only on the S-Boxes, which are the most sensitive part of most symmetrical algorithms in terms of SCA protection. Moreover, we seek to observe the impact of placement and routing imbalance alone, without mixing it with EPE, therefore statistics were performed on the absolute difference $|delay(true) - delay(false)|$ at the input

of each DPL gate, considering only the interconnect latency. This metric, referred to in the sequel by $\Delta_{TB}t$, is more suitable than a ratio, considering that a difference of say one nanosecond is always critical, independently of the paths length.

Timing information is extracted from the Standard Delay Format (*.SDF*) file, generated by the QuartusII software and intended to be used for back annotated simulation. The code snippet of Table 4.1 gives an example of the SDF syntax. The main information extracted from the SDF file are the interconnection delay and the propagation delay of each cell as placed and routed in the FPGA. `IOPATH Input_1 Output_1 (x:x:x) (x:x:x)` specifies the propagation time from *Input_1* to *Output_1*. `PORT Input_1 (x:x:x) (x:x:x)` specifies the interconnect delay modeled at input port *Input_1*. Those timings are expressed as two triples representing the minimum, typical and maximum delay for respectively rising and falling transitions (i.e. $0 \rightarrow 1$ and $1 \rightarrow 0$).

Table 4.1: Example of SDF timing information for a given `instance_name` gate.

```
(CELL
 (CELLTYPE "cell_type")
 (INSTANCE instance_name)
  (DELAY
    (ABSOLUTE
      (PORT Input_1 (min:typ:max) (min:typ:max))
      // ... etc ...
      (IOPATH Input_1 Output_1 (x:x:x) (x:x:x))
      // ... etc ...
    )
  )
)
```

To avoid side-effects, the input delays are set to zero by setting the *virtual pin assignment*, usually used for incremental compilation, on all ports. Still, these assignments generate dummy cells playing the role of sources for the inputs and of sinks for the outputs. It is thus necessary to manually remove them and to zero the corresponding propagation delays.

Additionally, the connectivity information is lost in SDF. In particular, the pins association at the input of the basic cell (`stratix_cell`) can be shuffled. Therefore, in order to compare dual signals, the netlist file must be parsed in parallel with the SDF. For instance, the analysis tools must be aware that one signal may arrive on `dataa` whereas the dual is actually connected to `datab` (`data{a,b,c,d}` are the four input names of `stratix_cell`).

Results displayed in Table. 4.2 show a max and mean delay of roughly 1 ns and 300 ps, which is non negligible considering the design runs at 50 MHz and should induce an exploitable source of side-channel leakage.

Table 4.2: Dual nets balance ($\Delta_{TB}t$), of DES WDDL S-Boxes (in picoseconds).

S-Box	Max	Mean	Std Dev
# 1	1041	351	347
# 2	1307	377	346
# 3	1009	359	337
# 4	1145	296	315
# 5	1113	291	331
# 6	1179	373	375
# 7	1518	342	392
# 8	877	196	196

As a matter of fact, the observation of those S-Boxes placement (via the QuartusII "ChipPlaner") suggest that in the absence of constraints, true and false parts of the netlist are almost randomly fitted in the chip. Figure. 4.5 illustrates such behaviour with the example of the fifth S-Box. It is indeed visible that the true and false part are completely scattered, thereby inducing such an imbalanced routing.

In the end, basic FPGA implementations of WDDL (and in general most DPLs) shows several limitations in terms of balance and synchronisation, which should induce exploitable vulnerabilities against first-order SCAs.

4.1.3 Successful Attack on DES WDDL

Simulation

In order to better estimate how this vulnerability can be taken advantage of, we simulated the same eight DES S-Boxes as in Section 4.1.2 (implemented in WDDL logic) and analyzed the switching delay Δt for every node, therefore taking into account both early evaluation and routing imbalance.

The simulator is Mentor Graphics Modelsim. As the objective is to effectively attack the implementation on FPGA, simulation uses the post-place and route design written by the EDA Netlist Writer of QuartusII. The results of the simulation are stored in a Value Change Dump (VCD) file. The VCD is an ASCII file which contains header information, variable definitions and the value changes for specified variables. One VCD file is created for each S-Box. The code snippet of Table 4.3 illustrates the VCD syntax for two dual wires denoted by *wire_a1_false* and *wire_5e_true*. In this example, the ´ and ¨ characters are used to represent those wires. Then, when one of them

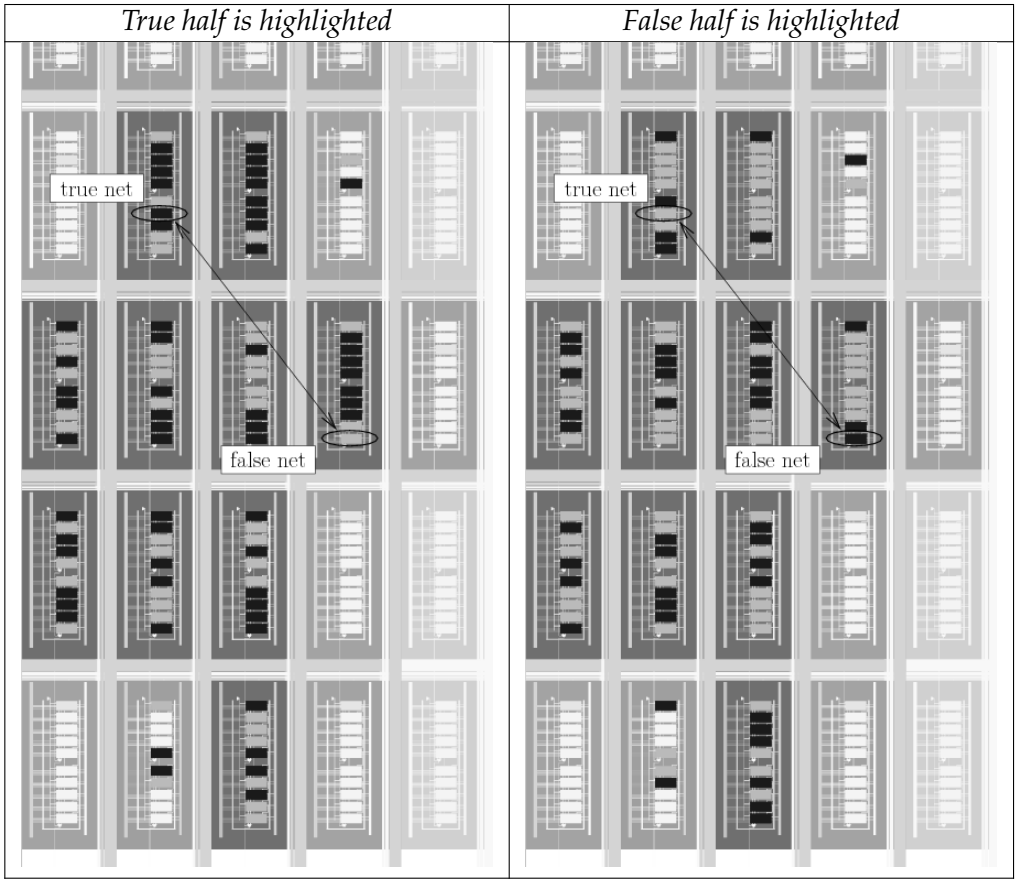


Figure 4.5: Unconstrained DES S-Box # 5 in Stratix.

changes its value, the corresponding time t is recorded and represented by $\#t$, followed by the new value for this net. By reading such file, one can thus know the precise times of all value changes for each considered net.

Information extracted from the VCD file, regarding both inputs and outputs of each logic node, can thus be compiled in order to assess the switching delay of each net during the precharge and evaluation phases. It is defined as the delay for one net to switch after the inputs are set. In this evaluation, the absolute difference between the mean switching delay of the true net and the mean switching delay of the false net is calculated for each node. As in Section 4.1.1 this difference is denoted by Δt. The nodes of the S-Box are then ordered in descending Δt. Table 4.4 presents an extract of the nodes classification for S-Box 5 and global results are plotted for the eight S-Boxes in Figure 4.6.

According to Figure 4.6, the most vulnerable S-Box seems to be box5 for two main reasons: Δt is the highest and its decrease is the slowest.

Table 4.3: Example of VCD file for a given gate.

```
$scope module i_dut_1 $end
$var wire 1 ! wire_a1_false $end
$var wire 1 " wire_5e_true $end
#0
$dumpvars
x"
x!
$end
#1761
0!
#2605
0"
#51915
1!
#101348
0!
#151915
1!
#201348
0!
#252274
1"
#301464
0"
```

This analysis yields a classification of the nodes according to their vulnerability, namely the Δt. In order to exploit this information for real attack, the following methodology is applied:

1. First selection: find nodes where Δt is higher than 1 ns (for experimental reasons: the timing difference should be visible with a sampling rate of 20 Gs/s and a bandwidth of 5 GHz).
2. Second selection: among the nodes in the first selection, find those for which the dispersion of the switching delay do not overlap.
3. Third selection: among the nodes in the second selection, keep the ones having the smallest dispersion.

Figure 4.7 presents the nodes thus selected. It displays the repartition in time of the switching delay for the "true" and "false" nets. The node in box5 with nets $wire\_e\_true$ and $wire\_1\_false$ is chosen as the most vulnerable and will be used for a DPA attack. As a matter of fact, the switching delays of the two dual nodes are clearly separated, with a difference of more than 1 ns The separation between "true" and "false" evaluation dates is especially eloquent for S-Boxes 3, 4 and 5.

Table 4.4: Extract of nodes classification for S-Box 5.

delta_t	wire/port name	mean delay(ps)	std dev
1336	wire_9a95_false	5224.00	738.12
	wire_656a_true	3888.00	318.50
1152	wire_e_true	1696.00	116.12
	wire_1_false	2848.00	7.00
1104	wire_9569_true	5096.00	469.32
	wire_6a96_false	3992.00	250.02
1024	wire_3a5c_false	4416.00	269.61
	wire_c5a3_true	3392.00	222.74
976	wire_69_false	4656.00	199.41
	wire_96_true	3680.00	197.14
848	wire_9a_false	4064.00	812.79
	wire_65_true	3216.00	322.51

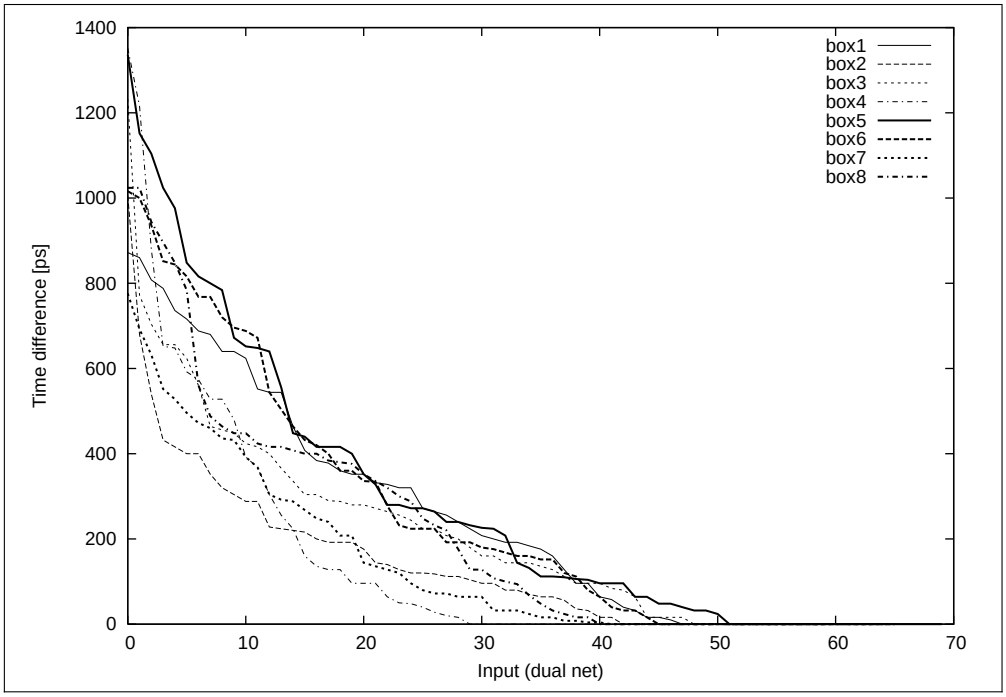


Figure 4.6: Δt decrease for the eight DES S-Boxes.

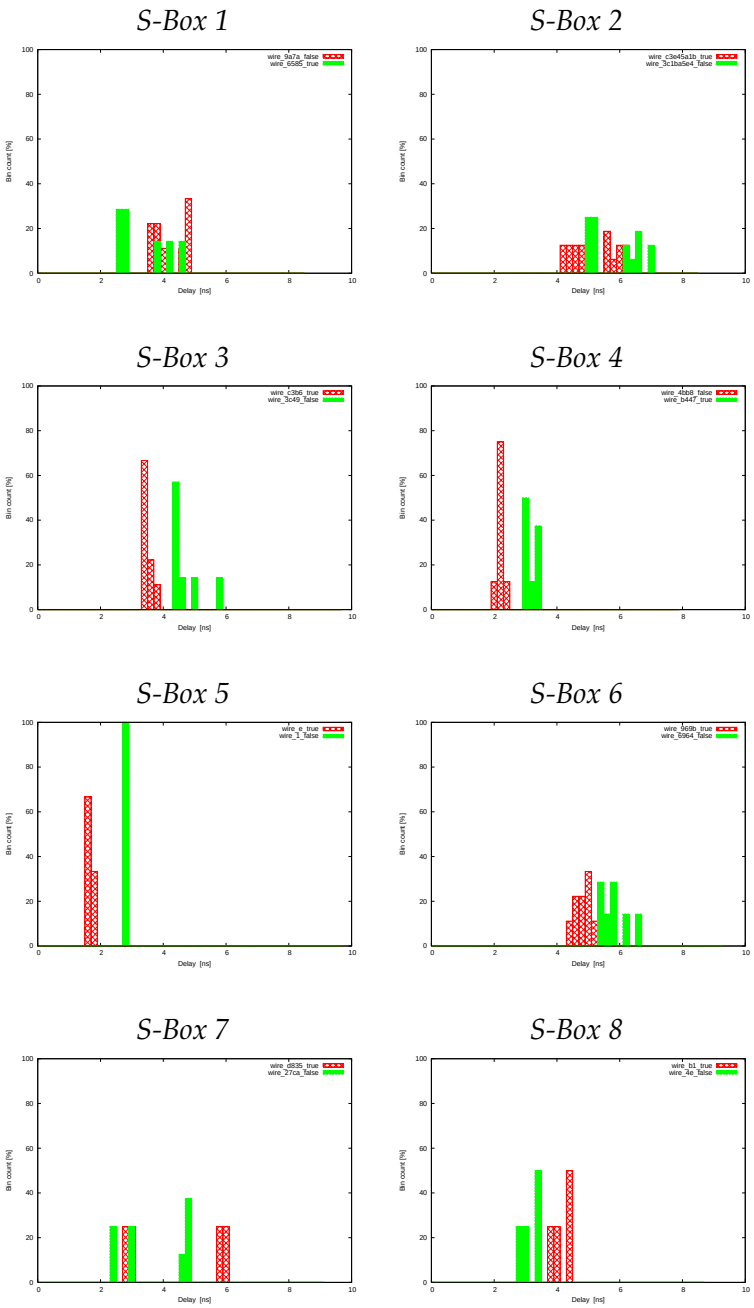


Figure 4.7: Switching delay of the most vulnerable nodes (true and false nets).

Known-Key Attack

In order to validate the results obtained by simulation, a DPA attack is performed on the node identified as the more sensitive. As our objective is firstly to validate that the vulnerabilities are indeed exploitable, we make the correct assumption for the key and perform DPA with the correct selection function.

The target FPGA is an Altera Stratix EP1S25, power consumption measurements were acquired, using a *differential probe* plugged to the positive rail of the FPGA core power supply through a 1 Ω shunt resistor, coupled with a *54855 Infiniium oscilloscope* from Agilent Technologies [4].

Figure. 4.8 shows the result of this attack. As can be observed, the results match the expectation, *e.g.* the DPA shows a spike that betrays the incriminated correlation.

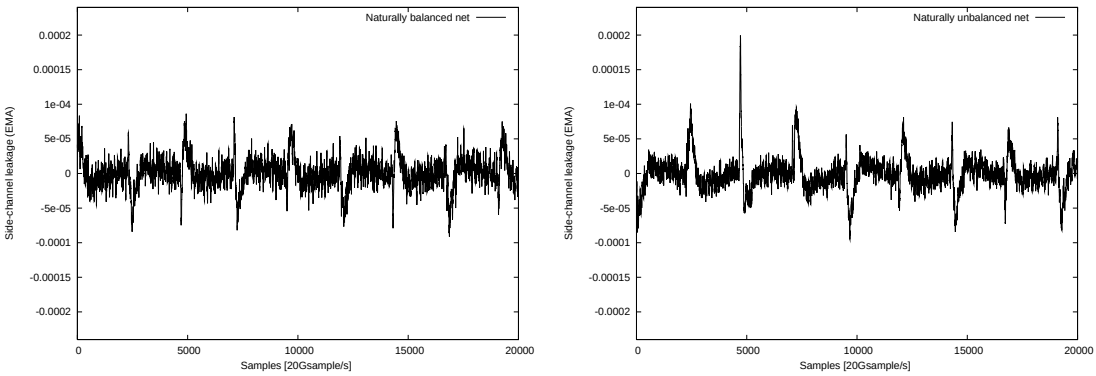


Figure 4.8: Experimental covariance between the power traces and a regular net (left – no leakage) & the most critical net value (right – peak around sample 5,000).

Actual Attack

Finally, with the same experimental setup than in the previous Section 4.1.3, CPA and DPA were performed on the full 3DES implementation both with WDDL and without (for reference).

The attack target was the value of the "R" register after the first round. Both Hamming Weight (HW) and Hamming Distance (HD) leakage models were used, targeting one to four bit for each sub-key.

Table 4.5 gives the results in number of Measurements To Disclose (MTDs). For the unprotected 3DES module, as expected, the best results are obtained with the CPA, by guessing the HD of four bit. For WDDL, because of the precharge, the best results are obtained by guessing the HW (the precharge value of the R register is 0). Targeting a single bit is more powerful than four bits. Indeed, the leakage in WDDL is caused by the imbalances between "True" and "False" networks and those imbalances may be opposite for different targeted bits, therefore counterbalancing themselves.

Table 4.5: DPA and CPA on 3DES WDDL.

(a) Unprotected DES Module - CPA Hamming Distance 4 bit								
S-Box	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
MTD	478,720	197,056	464,128	614,720	418,944	709,056	348,288	134,080

(b) WDDL DES Module - CPA Hamming Weight 1 bit								
S-Box	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
MTD	5,469,440	1,368,000	557,248	3,597,184	1,116,672	2,876,480	5,508,224	2,563,200

Measurements were not averaged, which explains the large number of traces involved. Gray shaded cells point out the sub-key with the highest resistance in that particular case, hence corresponding to the total number of MTD required to break the entire secret key. It is noteworthy that this was the first published successful attack on an FPGA implementation of 3DES WDDL, moreover with a full key recovery.

4.1.4 Counteracting DPL vulnerabilities

As the previous sections set light on the reality of DPL weaknesses and the feasibility of simple first-order attacks on such architectures, it is mandatory to find ways of removing those weaknesses or at least decrease their impact, in order to use DPLs as countermeasures for real industrial applications.

Early Propagation

To counter this phenomena, synchronization is systematically required before the gates switch between phases.

Isochronous logic can be used to obtain such synchronization. In that case the circuit is built with the goal of spending the exact same time within each phase. This should be achieved by specific customization of the cells. Therefore, such technique is limited to designs implemented on ASIC e.g. SecLib [71].

An other method, applicable to both ASIC and FPGA is the *parallel synchronization*, where extra cells are added to every gate in order to overcome EPE. Let us consider logics using any n-input functions. To ensure minimum leakage from the circuit and avoid glitches, synchronization should be performed according to the following rules:

- **Rule 1:** Evaluation should always be late *i.e.*, evaluation phase starts after all the input signals are valid.
- **Rule 2:** Precharge phase starts :
  1. only after all the inputs becomes NULL and the evaluation outputs are memorized.
  2. before the first input becomes NULL (which do not need any memorization).

For instance, DRSL [39] ensures synchronization before evaluation but not before precharge, therefore it has glitches when precharge phase starts as shown in Figure 4.9. Notice that DRSL+, as could be named DRSL with positive gates, would not glitch. STTL [164] performs synchronization before evaluation and it uses the first method to synchronize before precharge.

Technological Bias

Masked DPL logic styles like DRSL [39] and MDPL [135] aim at fixing all technological biases by randomly switching between the complementary parts. Routing imbalance, which is likely the major source of such vulnerability, can be removed by "fat wire" [177] and "back-end duplication" [72]. Another possibility would be to find a way of ensuring a perfectly balanced placement and routing through constraints during the CAD flow.

4.2 Optimized Placing and Routing for DPL Countermeasures

As stated in section 4.1.4 one of the most critical DPL vulnerabilities results from an imbalance between dual nets. Ways of imposing specifically balanced placement and/or routing on these nets can therefore be studied in order to enhance the robustness of those countermeasures.

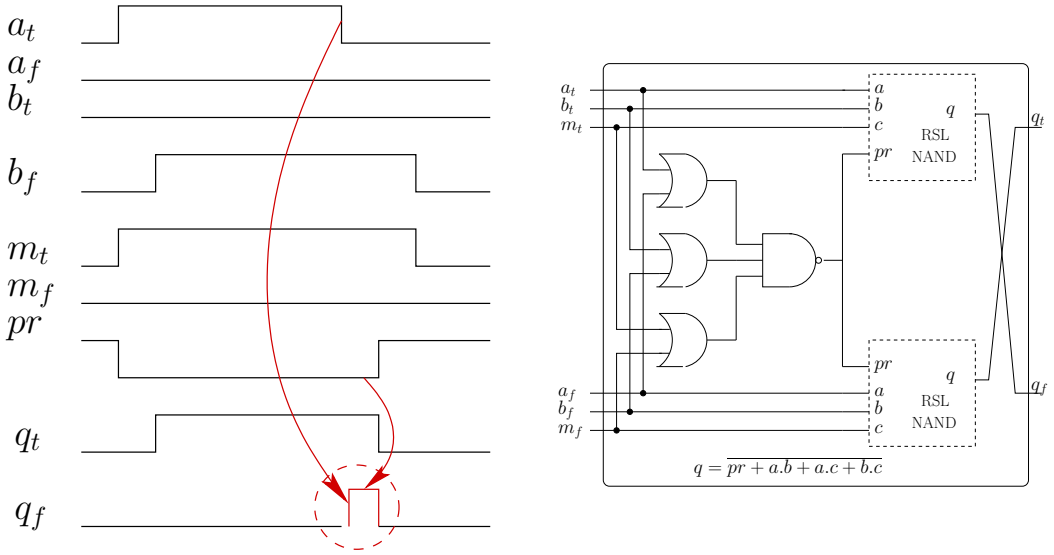


Figure 4.9: Possible glitch in DRSL due to lack of synchronization before the precharge.

4.2.1 Balanced Placement on Altera Stratix

The main goal is to force a balanced placement of all dual "True" and "False" cells, by systematically putting them close to each other. This can be achieved in Altera FPGAs by using *LogicLocks* constraints provided by QuartusII.

A *LogicLock* expresses the geometrical *lock*ing of specified *logic* resources within a rectangular region of Logic Array Blocs (LABs). As all QuartusII constraints and settings, they can be written as TCL scripts and automatically integrated in the design flow via the input Quartus Setting File (*.qsf*). To plug a given resource in a LogicLock the following TCL command is used:

```
set_instance_assignment -name LL_MEMBER_OF <logiclock_name> -to
"<instance_name>" -section_id <logiclock_name>
```

Several options also allows the user to customize those LogicLocks, they can all be described with the same syntax:

```
set_global_assignment -name LL_<option> <value> -section_id <logiclock_name>
```

The most relevant options are listed below:

1. LL_ORIGIN : coordinates for the left-bottom LAB of the LogicLock (for instance LAB_X1_Y2).

2. LL_WIDTH : width (in number of LABs).

3. LL_HEIGHT : height (in number of LABs).

4. LL_AUTO_SIZE (ON/OFF): when set to "ON" the LogicLock size is automatically adjusted.

5. LL_STATE : when set to "FLOATING", the LogicLock position is automatically defined (else set to "LOCKED").

6. LL_RESERVED (ON/OFF): allows to ensure that only the chosen resources are placed in the given LogicLock.

7. LL_PARENT : defines a hierarchical placement of LogicLocks ⇒ each "children" is placed within its "parent".

The first idea was to place each dual couple in a separate LogicLock of the smallest size i.e. a LAB. However, as LogicLock regions cannot overlap, the density would be limited to two LUT4 per LAB. For obvious area consumption reasons, this scheme was not sustainable on a real implementation.

Secondly, several couples were packed in the same LAB in order to optimise the resource utilisation. Keeping in mind that Stratix LABs are composed of 10 LUT4, this resulted in a maximum of 5 couple per LAB. However, this method induces congestion issues (especially within the S-Boxes) for the routing software and is therefore not directly applicable. In order to solve this problem, the S-Boxes were treated separately from the rest of the design.

Eventually, it was put to light that 5 couples (within the S-boxes) could be stacked in the same LAB, provided they have roughly the same logical depth. Moreover, as stated in Section 4.1.3, the S-Boxes are generated by an ad-hoc software, which gives the possibility to automatically create a corresponding TCL constraint file (*.tcl*) placing each dual couple in a separate LogicLock. A parser written as a Perl script (*.pl*) was then developed, in order to sort them by logical depth (which is available in the instance name) and force five couples in the same LAB. Taking this into account, new constraints were generated and resulted in a successful placement and routing for the maximal density of 5 pairs per LAB. An illustration is given in Figure 4.10.

The validity of this placement was thoroughly verified by using the post place-and-route location assignments generated by QuartusII with the "export assignments" option. Indeed, by calling this command, a new *.qsf* file can be created, with LUT-level placement information on the entire design, written with the following syntax:

set_global_assignment -name LL_NODE_LOCATION LAB_Xm_Yn_Nk -to <instance_name> -section_id <logiclock_name>

This is in fact a LogicLock constraint, allowing the user to attain a LUT-level precision: N0-9 ↔ first to tenth LUT.

Finally, this constraint was exploited in order to perform a last placement strategy on this design:"True" and "False" instances were all positioned next to each other. As a matter of fact, previous timing analyses showed that the worst-case switching time

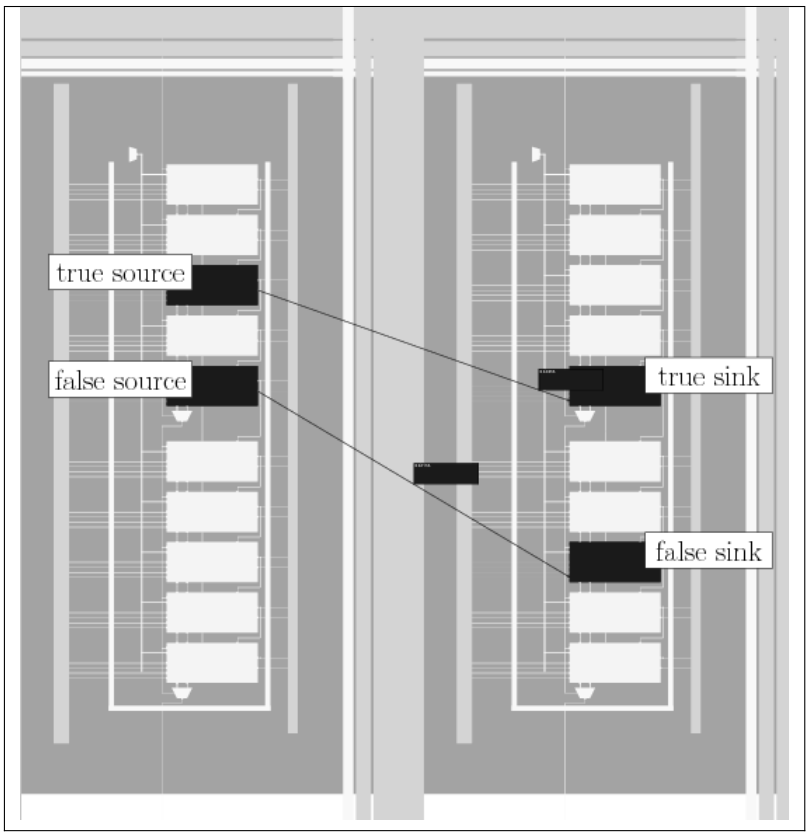


Figure 4.10: Constrained dual-placed DES S-Box # 5 in a Stratix (zoom on two adjacent LABs).

delays, for the former placement, usually took place when the "True" and "False" parts of the same net were distant from one another (within a LAB) therefore possibly not able to access the same type of routing resources.

However, these constraints must be automatically generated. Thus, the conception flow was altered in the following way, as illustrated in Figure 4.11: the fitter is run a first time, placing five couple per LAB as before, then the *.qsf* file is exported. A new parser is developed in order to put the dual instances next to each other (by modifying the "_N(0-9)" index within a LAB). Finally, this modified *.qsf* file is used as the input constraint file for a new whole compilation.

A drawback of such technique is to significantly increase the compilation time, due on one hand to the additional constraints needed to be taken into account by the CAD software and on the other hand by the required second compilation.

This placement was validated the same way as before, by checking the final post place-and-route assignments.

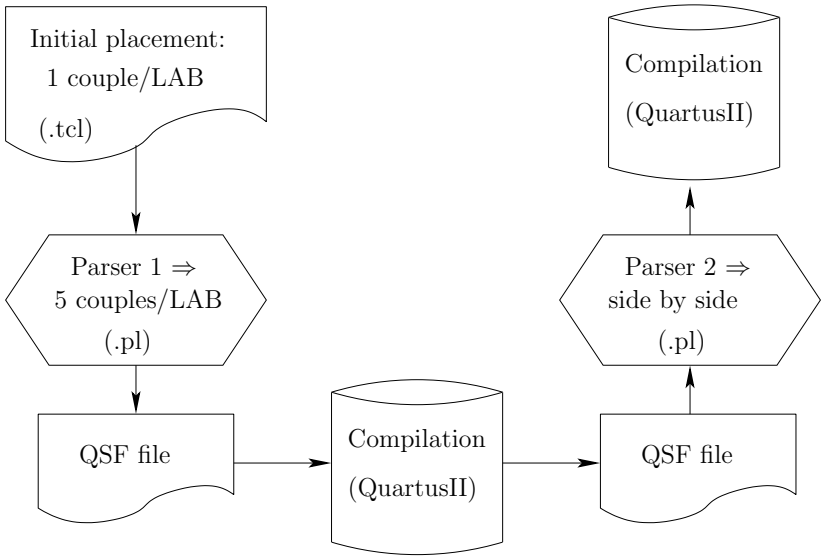


Figure 4.11: Design Flow with Balanced placement on Stratix.

Worst-case switching delay between "True" and "False" nets, as well as their mean and standard deviation are given in Table 4.6. It is noteworthy that no actual attack results are available using these placement strategies, as the successful DPA on WDDL 3DES, described in Section 4.1.3 had not yet been performed at this time.

Table 4.6: Timing Analysis for Different Placement Strategies of WDDL 3DES on Stratix.

Strategy	**Worst-Case** (ps)	**Mean** (ps)	**Std Dev** (ps)
reference	1578	342	347
1/LAB	1685	90	97
5/LAB	1933	135	143
side-by-side	1201	106	112

On one hand, those results show that the best placement in terms of mean delay is the basic strategy where each dual couple is put in a different LAB. However, for obvious area consumption reasons, this approach is unfit for industrial applications. On the other hand, the side-by-side strategy proves to be almost as efficient as the latter regarding the mean switching delay, while being optimized in terms of resource utilisation. Moreover, with this approach, the worst-case delay is clearly smaller than with one couple per LAB, which should increase the robustness against SCAs, as those attacks only require one leaking bit to be successful.

*

To conclude, it has been shown that the differences in switching delays of dual nets, which are a source of side-channel leakage, can be reduced by automatized placement strategies. The "side-by-side" approach, can be used to force five dual couples in each LAB (on a Stratix FPGA) and visibly decrease this delays, by allowing the true and false parts to access similar hardware resources. However, this methodology increases the compilation time, which can be an issue when considering very large designs. Eventually these placement strategies are hardware-dedicated, namely they can only be used on Stratix or similar FPGAs, making them hardly fitted for present industrial applications. Nevertheless, these experiments should be seen as a proof of concept, showing that constraint placement is a viable method to significantly reduce the side-channel leakage on DPLs FPGA implementations and that it can be deployed in an automatized way. Therefore, alternate methods should be studied, aiming at achieving a greater level of genericity and targeting more recent FPGA technology.

4.2.2 Placement-induced Routing with *LogicLocks* on ALM-based FPGAs

As the FPGA technology improves, countermeasures should be designed to exploit these new architectures, in order to optimise performances, area consumption and robustness. Taking that into account, all further experiments are performed on a "Side-channel Attack Standard Evaluation BOard" (SASEBO-B) [86], developed in 2007 by the Tohoku University and the Japanese Research Center for Information Security (RCIS). The SASEBO-B, incorporates two Altera [10] Stratix II FPGA: one *EP2S30F672C5N* supposed to embed all control modules and one *EP2S15F484C5N* for the cryptographic modules. Having two FPGAs enhances the accuracy of the measurements as it uncouples the power consumption of the cryptographic parts from the power consumption of the others. Measurements are acquired using the original 1 Ω spying shunt resistor in the positive rail of the cryptographic FPGA core power supply. Power consumption measurements, are collected using a *1132A* differential probe and a *54855 Infiniium* oscilloscope from Agilent Technologies. The final setup has a 6 GHz bandwidth and a 40 GSa/s maximal sample rate.

Moreover, unlike the Stratix used in the experiments of the previous Sections, StratixII FPGAs are based on the same base logic elements, called "Adaptive Logic Modules" (ALMs), as all the following families up to the oncoming StratixVI. The high-level block diagram of the StratixII ALM is shown in Figure 4.12.

On the left, eight inputs drive a combinatorial logic block programmable as either one 6-bit Look-up-Table (6-LUT) or two Adaptive LUT (ALuTs), both having their own register, named respectively *reg0* and *reg1*, on the right.

On another topic, after performing several attacks on this new platform, targeting different internal states of of the DES algorithm, namely the outputs of the register (R), the key-dependent *XOR* (XOR_K) and different layers of the S-Boxes, it was put to light that the registers displayed a far stronger leakage, making the attacks clearly

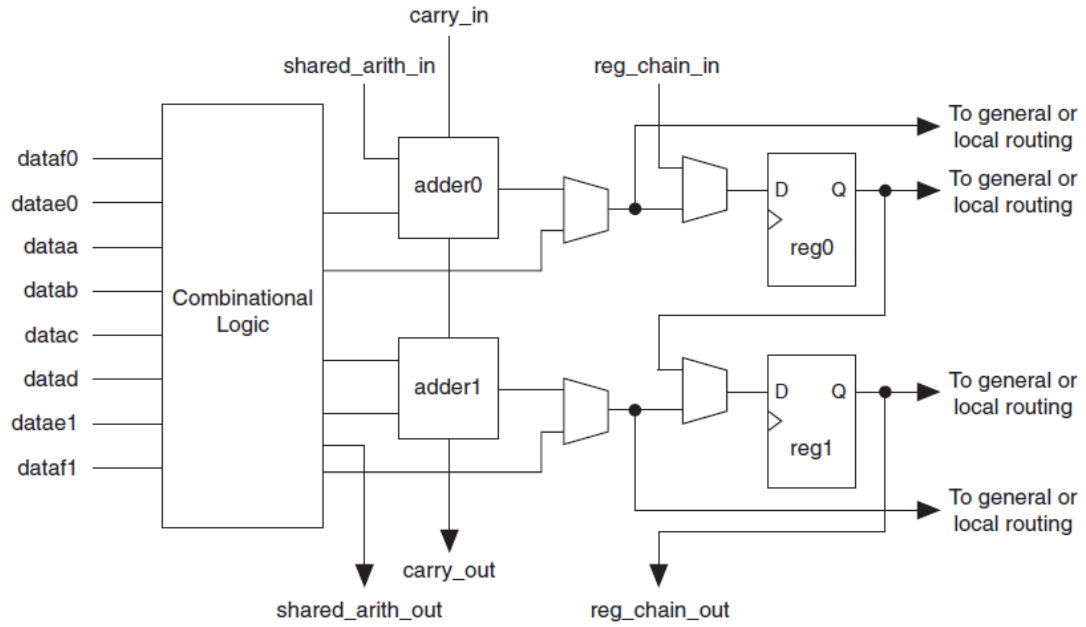


Figure 4.12: High-Level block diagram of an ALM, extracted from "Stratix II Device Family Data Sheet, volume 1" [10].

more efficient than on the combinatorial part, in terms of Measurements To Disclose (MTD). As a matter of fact, this can be explained by considering that the switching of a flip-flop generates a far more powerful power consumption than a combinatorial gate, making the information leakage way more visible. The register switching also happens in a synchronized manner, optimizing the efficiency of statistical attacks like DPA or CPA.

Moreover, the deeper in the combinatorial cone was the target internal state, the more MTD were required to retrieve the sub-keys, i.e. attacks guessing the S-Boxes output were less efficient than those targeting the XOR_K. Therefore, rather than imposing a precise constraint placement on the whole algorithm, which would induce prohibitive compilation time, special efforts were made to reduce the technological bias, near the registers. Namely the idea was to enforce the most balanced placement and routing on the zone of highest leakage, i.e. both master and slave registers (R_M and R_S) and the adjacent XOR_K, while keeping a generic packing of the dual couples for the rest of the algorithm (i.e. one couple per ALM).

Making the hypothesis that on one hand routing lines within an ALM are fast and similar for the two registers and on another hand that all ALMs are roughly equivalent, two Placement And Routing (PAR) strategies are designed, aiming at exploiting those characteristics:

1. The "Vertical" strategy, places "True" and "False" networks as close as possible, by assembling each dual couple in the same ALM (see top of Figure 4.13).

2. The "Horizontal" strategy, allows routing resources to be identical for the "True" and "False" networks, by packing R_M, R_S and XOR_K in the same ALM and its dual part in the adjacent ALM (see bottom of Figure 4.13).

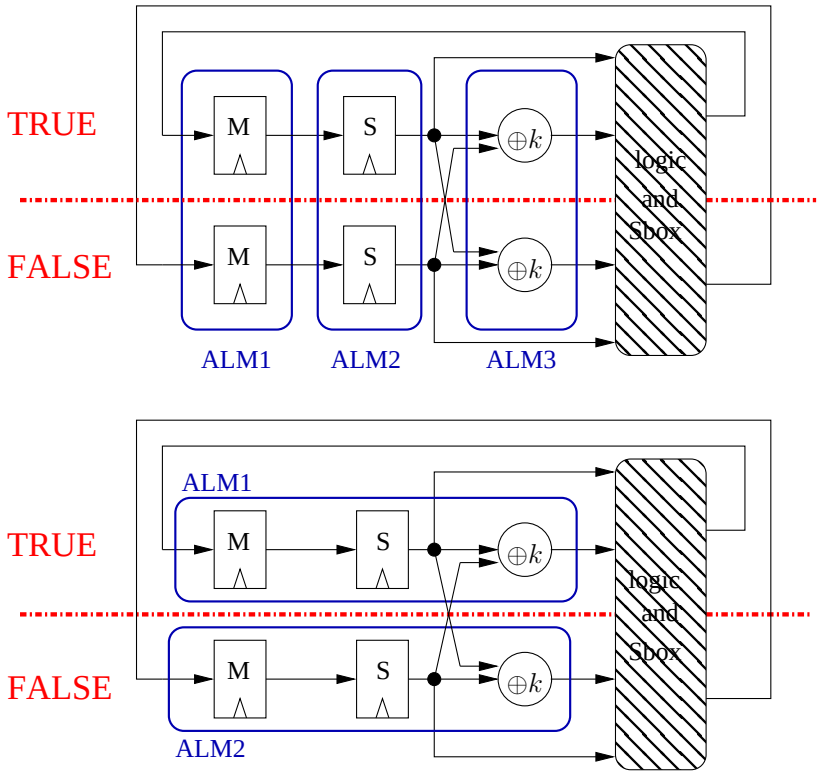


Figure 4.13: Vertical and Horizontal PAR strategies.

Those placements were carried out in a similar way as in Section 4.2.1. As a matter of fact, the *LogicLocks* constraints for the StratixII and following families are roughly the same as for the Stratix, with the addition of a few options, which are irrelevant in this context.

The major difference with previous placement methods, is that the "Vertical" and "Horizontal" strategies require absolute placement, meaning that the constraints must include their exact location, including the LAB coordinates, from the start. Indeed, each instance must be precisely placed in a designated container. In order to reduce the final design complexity and the CAD software compilation time, all bundles containing the true and false part related to one bit of both R_M, R_S and XOR_K are orderly positioned on top of one another. This way, most inputs and outputs are treated in a symmetrical manner for all bundles. Moreover, the chances that all bits of the register

have access to the same routing resources and generate similar power consumptions are also increased.

The design flow is then modified, as illustrated in Figure 4.14 and a new software is developed, to parse the initial TCL file (containing the constraints to put one couple per LAB) and perform the following alterations:

- Extract all constraints relative to either the XOR_K or the registers.
- Generate the LogicLocks assignments to achieve the required placement on these instances. As precise location information (LAB coordinates) are required for those constraints, the coordinates of the topmost LAB are defined beforehand and all the remaining LABs are placed beneath one another, as a column.
- Generate assignments for the rest of the datapath, placing four couples per LAB in a similar way as in Section 4.2.1 (eight 4-input LUTs are available in a LAB of the considered StratixII).
- Output the resulting *.qsf* file to be used for the compilation.

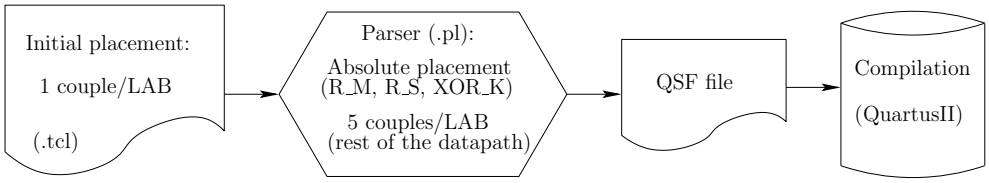


Figure 4.14: Design Flow for Vertical and Horizontal Strategies.

It is noteworthy than those PAR strategies induces a significant overhead in terms of compilation time. As a matter of fact, the more precise placement constraints are deployed, the longer the compilation, as the CAD software must take them into account in addition to basic constraints of speed, area optimization ... For instance, regarding the Vertical and Horizontal strategies, compilation time went from roughly 5 minutes, without them, to 30 minutes. Even if in this case, the 25 minutes increase is not critical, when compiling a larger and more complex design, this could prove to be a significant drawback.

An example of the constraints employed to force the placement of the true part for both *XOR* and registers with the "Horizontal" approach is given below:

```
set_instance_assignment -name LL_MEMBER_OF <Region> -to XOR_K -
section_id <Region> set_instance_assignment -name LL_NODE_LOCATION LC-
COMB_X10_Y24_N4 -to XOR_K -section_id <Region>
set_instance_assignment -name LL_MEMBER_OF <Region> -to R_M -section_id
<Region> set_instance_assignment -name LL_NODE_LOCATION LCFF_X10_Y24_N1
-to R_M -section_id <Region>
set_instance_assignment -name LL_MEMBER_OF <Region> -to R_S -section_id
<Region> set_instance_assignment -name LL_NODE_LOCATION LCFF_X10_Y24_N3
-to R_S -section_id <Region>
```

On one hand, the *LL_NODE_LOCATION LCCOMB* constraint can be used to pinpoint the two available 4-input LUTs of an ALM, designated by $[\_Nm]$ with $m = 2n$, $n < N_{ALM}$ (N_{ALM} being the number of ALM per LAB). On the other hand, the *LL_NODE_LOCATION LCFF* target the registers, with indexes $[\_Nm]$ such as $m = 2n + 1, n < N_{ALM}$.

The floor-plan of the Vertical strategy is shown in the layout of Figure 4.15. The four ALMs on the top left correspond to four bits of R_M. Two dual wires output of each of them, go to R_S in the middle, then to XOR_K on the right and finally reach the S-Box and XOR_L (both outside of the figure). The wires on the middle top realize the DES expansion function (E) towards the next S-Box, while that on the middle bottom comes from the previous one. These of XOR_K are clearly visible on the top right of the figure. This schematic is reproduced for each S-Box, thus eight times. Post PAR timing annotations give a first idea of the balance between dual wires: for example, the bit of R_M on the top has an imbalance of $289 - 290 = -1$ ps.

Figure 4.16 illustrates the floor-plan for the Horizontal strategy. The upper and lower ALMs respectively contain the true and false networks. The *reg1* register from Figure 4.12 is assigned to the master register, while the slave register is implemented in *reg0* and XOR_K in the combinatorial logic block, on the left. The gain in balance is optimum as timing annotations have exactly the same value.

As in Section 4.2.1, static timing analysis is achieved by exploiting the Standard Delay Format (SDF) file. Mean and standard deviation statistics on timing differences between the true and false network are given in Table 4.7 in picoseconds. First column corresponds to the imbalance without specific PAR constraints. It serves as reference to stand out improvement generated by the two PAR strategies. In average, timing imbalances of the XOR_K are divided by 2. Regarding the registers, both strategies have significant impact on the routing of dual nets. In both cases the imbalance decrease is greater for the master than the slave. Indeed, on one hand, the average delay for the master is divided by a factor 10, respectively 100 by the vertical and horizontal strategies. And on the other hand, those of the slaves are divided by factors 4, respectively 10.

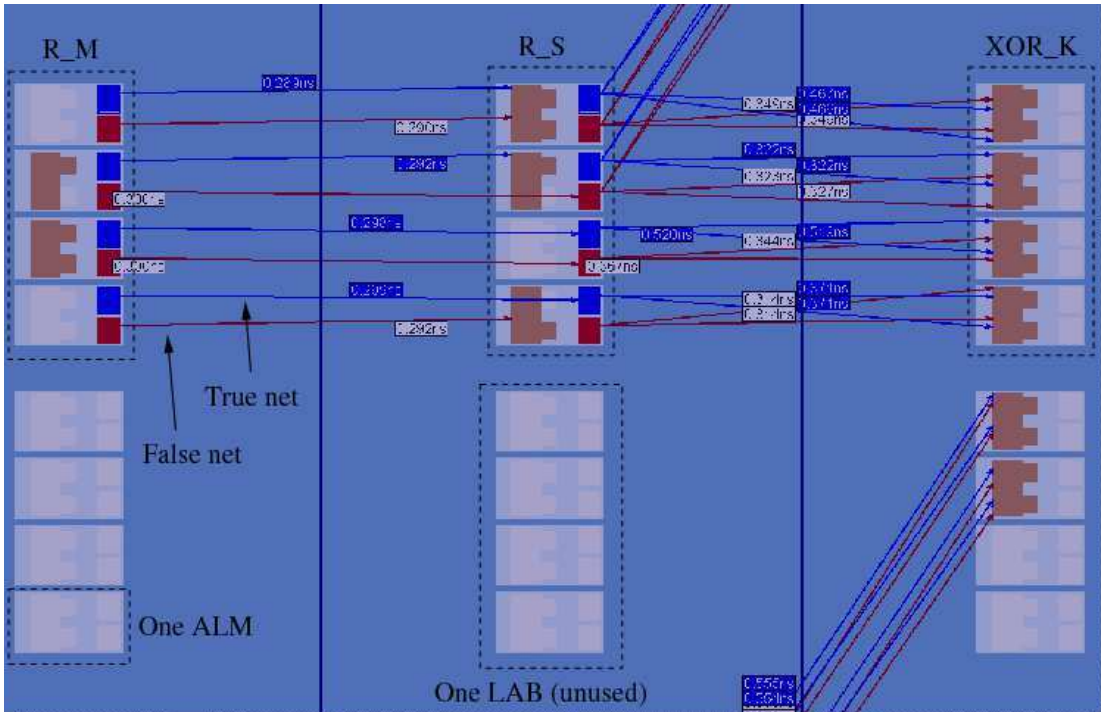


Figure 4.15: Vertical strategy.

Table 4.7: Timing Unbalance for Vertical and Horizontal strategies (in ps).

PAR Strategy	None		Vertical		Horizontal	
Element	**Mean**	**Std Dev**	**Mean**	**Std Dev**	**Mean**	**Std Dev**
LR Master Register	251	322	24	23	3	3
LR Slave Register	566	327	137	88	25	28
XOR_K function	501	298	272	203	290	227

Actual attacks on those three modules were performed using the experimental setup described at the beginning of this Section. In a view to cover a large SCA threat range, 6,400,000 power measurements were acquired and several analyses were considered: DPA and CPA, with both Hamming Weight (HW) and Hamming Distance (HD) models, by guessing one to four bits of the register. The results are summarized in Table 4.8. They are purely comparable as traces were acquired using the same experimental setup and the same pseudo-random messages, generated from the same seed. Upper 4.8(a) part concerns the unconstrained WDDL cryptoprocessor, serving as reference to estimate the security gain provided by WDDL and to study the impact of the differential PAR. Middle 4.8(b) and lower 4.8(c) parts deal respectively with the results of Vertical and Horizontal PAR strategies.

Experiments confirm that CPA is more powerful than DPA in presence of noise

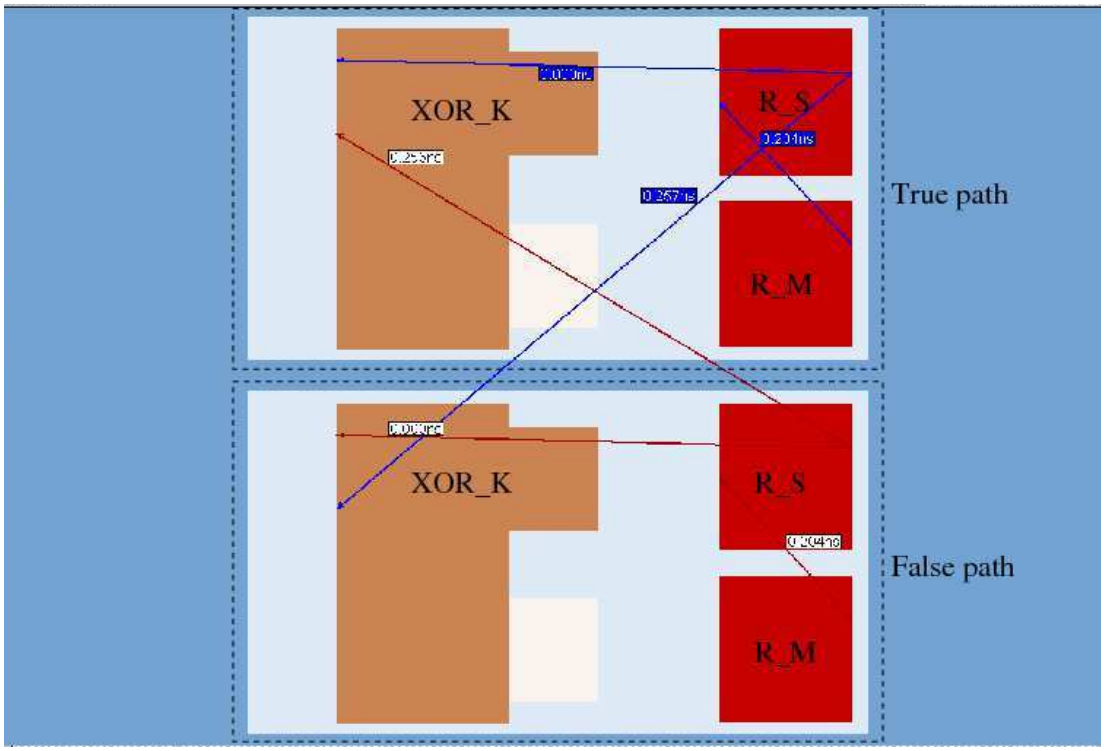


Figure 4.16: Horizontal strategy.

(observations were not averaged). Best results on unprotected implementations are obtained with the HD model, as it matches the physical phenomenon responsible for power consumption (commutations) and by targeting four bits at the same time. For WDDL, because of the zero value precharge, best analyses are done by guessing the HW as it equals the number of commutation: $\text{HD}(0, x) = \text{HW}(x)$. As stated in Section 4.1.3, targeting a single bit is more powerful than four bits as the leakages in WDDL could be opposite for different bits and therefore counterbalance themselves.

For each part of Table 4.8(a)(b)(c), bits getting out from the same S-Box have been grouped together and their position in the R register after permutation of 3DES is recalled by the second and sixth lines. For example, bit 1 of S-Box 1 becomes bit 9 of R. Fourth and eighth lines provide the ratio between MTD of unprotected and protected modules, so-called Security Gain (SG).

In the best cases SG is increased by a factor 22 when using WDDL without specific efforts of PAR and 364 when using the Horizontal strategy. The Vertical one seems to be the most robust overall, as bit 31 and 18, marked with black shaded cells, do not allow disclosure with up to 6,400,000 observations. However, a reliable security assessment considers the worst case scenario. With 3DES, the proper security gain of one S-Box may be considered as the minimal SG amongst those of its four output bits,

Table 4.8: Statistics for Bits of R_S Register.

(a) WDDL 3DES Module without PAR Constraints																
S-Box	1				2				3				4			
Bit of R	9	17	23	31	13	28	2	18	24	16	30	6	26	20	10	1
Sec. Gain	3	1	5	1	1	6	3	1	2	11	10	6	2	4	2	1
S-Box	5				6				7				8			
Bit of R	8	14	25	3	4	29	11	19	32	12	22	7	5	27	15	21
Sec. Gain	1	2	5	7	4	2	8	1	3	22	2	4	1	1	3	2

(c) WDDL 3DES Module with Vertical Strategy																
S-Box	1				2				3				4			
Bit of R	9	17	23	31	13	28	2	18	24	16	30	6	26	20	10	1
Sec. Gain	50	10	9	5	1	8	10	4	12	20	9	31	6	21	23	21
S-Box	5				6				7				8			
Bit of R	8	14	25	3	4	29	11	19	32	12	22	7	5	27	15	21
Sec. Gain	2	19	13		214	5	15		19	352	11	31	36	1	8	31

(b) WDDL 3DES Module with Horizontal Strategy																
S-Box	1				2				3				4			
Bit of R	9	17	23	31	13	28	2	18	24	16	30	6	26	20	10	1
Sec. Gain	9	8	7	2	3	10	4	1	18	15	25	11	2	5	20	5
S-Box	5				6				7				8			
Bit of R	8	14	25	3	4	29	11	19	32	12	22	7	5	27	15	21
Sec. Gain	2	4	2	15	110	8	11	15	23	364	18	23	32	4	9	14

noted SG_{min}.

Thus, a more accurate conclusion is that the WDDL implementation without specific efforts of PAR brings but a small increase in terms of robustness, with a SG_{min} of 1 (i.e. no gain) for six S-Boxes out of eight and 2 for the others, resulting in an average SG_{min} of 1.125. On another hand, the global SG for WDDL is 3.5 in average.

The Horizontal PAR strategy shows a SG_{min} between 1 and 18, with a mean of 6.5 and a global mean of 49. Finally, with the Vertical strategy, SG_{min} is situated between 1 and 11, for an average of 5 and a global mean of 71. Moreover, two bits of the Vertical Strategy present a high robustness, as they do not allow a disclosure over 6,400,000 observations.

*

These results are very promising, as they show a clear improvement in terms of overall robustness for both PAR strategies, with regards to an unprotected DES and its

basic WDDL implementation Moreover two bits are still not attackable with 6,400,000 SCA measurements, which implies that, in the right conditions, this type of countermeasure can lead to an adequate level of security. However, considering that on one hand, given enough observations and when performing analyses on all bits, every sub-key can yet be retrieved. And on the other hand, the overhead in terms of compilation time is significant, this countermeasure, as it stands, is not yet sufficient to be the sole protection for sensitive cryptographic designs. Nevertheless, additional research should be undertaken to increase the robustness level, for instance different ways to place the Horizontal or Vertical bundles with regards to each other, or additional PAR on the rest of the algorithm.

4.3 BCDL: a new DPL logic

While the reduction of technological bias by balancing the placement and routing of DPLs like WDDL shows interesting results in terms of security gain, it proves not to be sufficient yet to properly thwart SCAs. Indeed, even when the "True" and "False" parts are properly balanced, there remains the matter of the early propagation effect. Therefore, another approach is considered which consists in developing a new DPL logic, with the purpose of counteracting such weaknesses, so-called Balanced Cell-based Dual-rail Logic (BCDL).

4.3.1 BCDL Principle

The main goal of BCDL is to avoid most of the vulnerabilities in current DPL. It aims at removing EPE and reducing the technological bias, while keeping area and performances fit for industrial applications. It is therefore based on two principles:

1. A specific **synchronization** scheme based on a global precharge signal (to meet the rules explained in section 4.1.4) is added to all logic gates, before the actual precharge or evaluation.

2. The synchronization is performed on **Bundle Data** (which is well adapted to FPGA LUT structure).

Basic Synchronization Scheme

Synchronization in asynchronous logic is usually performed between two signals with a rendezvous cell "RV", also called "C-element". RV is a memory that only changes its state when there is unanimity (to 0 or 1) on its inputs. With BCDL, the synchronization is done on Bundle Data, without any memory element, using specific cells: U_0 and U_1 (Unanimity to 0 and 1).

- U_1 is the signal authorizing the evaluation. It raises up to 1 when all signals have left the precharge state. More precisely $U_1(x, y, \dots)$ is defined by Equation (4.1):

$$U_1(x, y, \dots) \doteq \begin{cases} 1 & \text{if } x \neq (0,0) \text{ and } y \neq (0,0), \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

- $U_0 = 1$ when all the inputs are in the precharge state, as shown in Equation (4.2):

$$U_0(x, y, \dots) \doteq \begin{cases} 1 & \text{if } x = y = \dots = (0,0), \\ 0 & \text{otherwise.} \end{cases} \tag{4.2}$$

Actual calculation only starts if there is unanimity (U_1 or U_0 valid) and is frozen otherwise.

Figure 4.17 shows a schematic diagram for synchronization of bundle data.

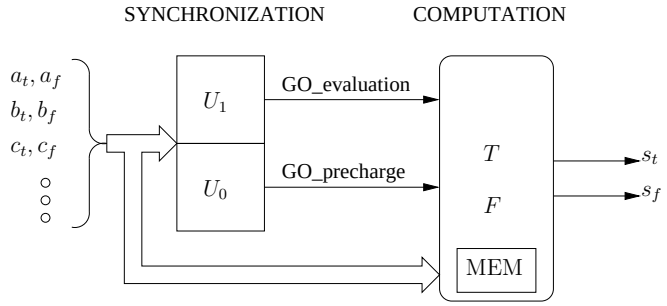


Figure 4.17: Synchronization and "bundled" data in BCDL.

Optimized Synchronization Principle with Global Precharge

Calculation of the precharge is quite a simple operation (compared with evaluation), as it only requires that all cell outputs be forced to 0, while the latter is an actual computation of signals carrying information. Based on this property, the model can optimized by using a simplified rendezvous scheme, coupled with a **global precharge signal**, PRE. It is used to induce the precharge state globally, in a very short amount of time. Thus, it allows the designer to reduce the complexity (and increase the performances) of the BCDL rendez-vous cell, by replacing the "unanimity to 0" (of Figure 4.17) by a logical "AND" between the PRE signal and the output of the "unanimity to 1" (see Figure 4.18).

The actual computation is then synchronized by the $U/\overline{PRE}$ signal as follows:

- When $U/\overline{PRE}$ falls to 0 (just after the signal PRE), the precharge is forced, independently from the inputs. Using a global signal in FPGAs ensures that the precharge signal will always be faster than any input. As a matter of fact PRE

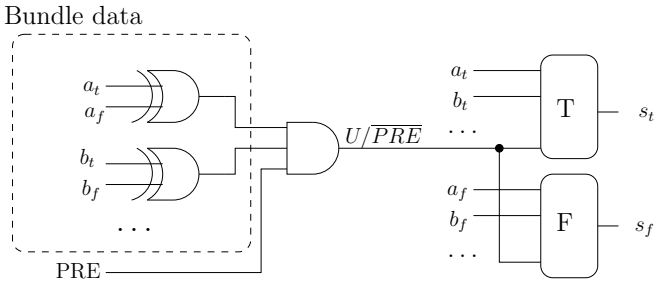


Figure 4.18: BCDL n-input cell.

takes advantage of the FPGA global lines which are specific, fast and sized to broadcast heavy loaded signals. Moreover the frequency of PRE is half that of the clock signal which can be generated from a FPGA PLL without any skew with respect to the clock.

- When $U/\overline{PRE}$ raises to 1, indicating that, on one hand, the signal PRE is valid and on the other hand, that the "rendez-vous" of inputs is over, the evaluation phase begins.

Precise temporal relationships between signals of a 2-input OR gate (where (a_t, a_f), (b_t, b_f) are the inputs and (s_t, s_f) the output) are shown in Figure 4.19.

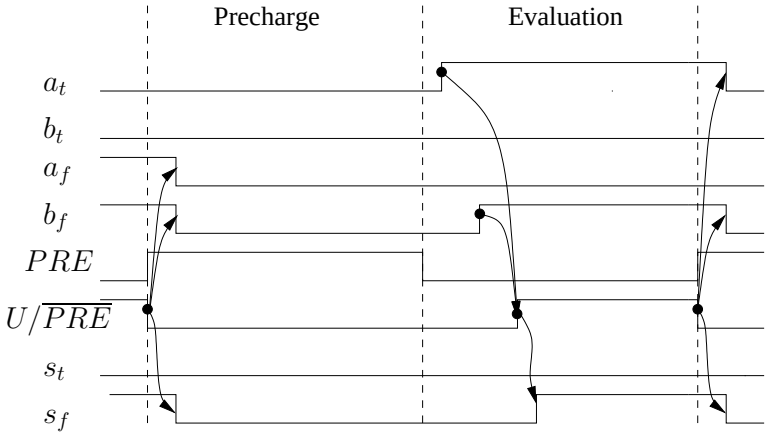


Figure 4.19: Temporal relationships of a 2-input BCDL OR gate signals.

LUT-Level Optimization

Based on the above properties, this logic is able to overcome early propagation on a global scale (between each cell of the circuit). However, it may also be synchronized at LUT-level, in order to avoid local early evaluation and technological imbalance.

An analysis of FPGA cells shows that their LUT structure is a tree of multiplexers as shown in Figure 4.20.

Local synchronization is achieved by applying the two following constraints:

- The $U/\overline{PRE}$ signal is assigned to the first column of this tree.
- The inputs e_t and e_f are plugged on the same pin respectively on "True" and "False" cells.

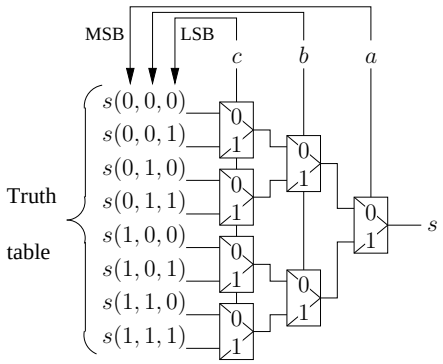


Figure 4.20: Structure of a LUT.

With these constraints, significant properties can be obtained regarding local robustness:

- **No glitches:** As the $U/\overline{PRE}$ signal is the first to switch before the precharge state, the internal nets are all forced to '0' without any glitch, regardless of the inputs. Likewise, it is the last one to switch prior to the evaluation after the other inputs are already positioned.
- **Reduction of the technological bias:** The total number of commutations for "True" and "False" equipotentials does not change according to the inputs. It is a constant $= (2^n - 1)$ for a n-input LUT. It is therefore difficult to discriminate the activity of "True" from that of "False" as the consumption profile is identical regarding the couple "True", "False". This is illustrated in Figure 4.21, which describes all the combinations of a 2-input XOR, when $U/\overline{PRE}$ switches. Bold nets correspond to multiplexers outputs that are switching. There is, thereby, an overall balance in terms of switching time as well as energy consumption (number of simultaneous switching).
- **No local early evaluation or precharge:** Indeed, $U/\overline{PRE}$ is always delaying the evaluation (switching to '1' last) and forcing the precharge (falling to '0' before any other signal). In other words, the evaluation is always **delayed** and the precharge always **anticipated**, regardless of the data.

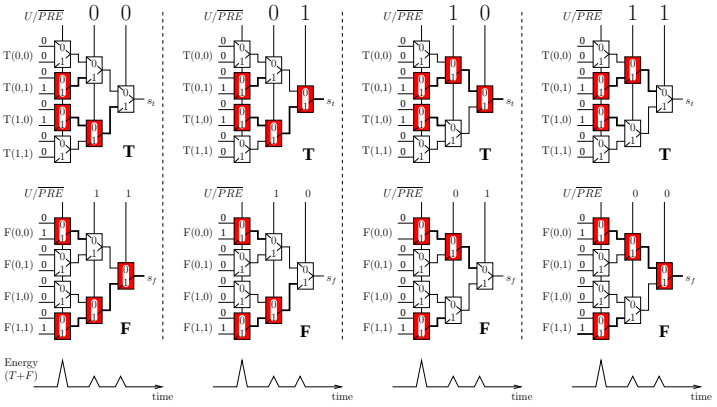


Figure 4.21: Local switching balance in BCDL: LUT3 example.

Reduced Area

Area-consumption is a limitation for most DPL counter-measures on FPGA. Even when the main goal is to achieve the best robustness, it proves useless if it is actually too complex to implement on a real device. Thereby, one of the main objectives of BCDL is to keep a reasonable complexity. Thanks to the synchronization schemes three significant properties are obtained:

- **Reduced S-Box area:** As stated in Section 4.1, in various DPL style logic, one basic 8-input S-Box (2^8 byte) could be merely dualized into two "True" and "False" 16-input S-Boxes (512×2^8 byte). This huge size can be reduced by building a local precharge signal but it might induce glitches due to the lack of synchronization. BCDL takes advantage of its global precharge signal to reduce the RAM size to only **4 times** the basic one without any glitch risk. Indeed there are "True" and "False" S-Boxes, which only have one more input than the basic implementation, that is the $U/\overline{PRE}$ signal. This way, during the precharge, when $U/\overline{PRE}$ is low, the RAM output is always null. Such S-boxes are illustrated in Figure 4.22.

- **Reduced complexity:** Due to the absence of glitches within LUTs (see section 4.3.1), BCDL is not limited to positive functions and can use all 2^{2^n} existing functions (for a n-input LUT), which provides many optimization opportunities.

- **Integrated rendez-vous:** Recent FPGA technologies can be exploited to make this optimization. In fact, for a 2-input function, if the target FPGA is made of 5-input LUTs ($LUT5$) or more, the synchronization scheme can directly be integrated in the "True" and "False" cells. Indeed, if the $true$ and $false$ signals as well as $U/\overline{PRE}$ are inputs of the same LUT, the rendez-vous function and the actual cell function can be computed at once, within the same truth table.

Being non-positive, a classic *XOR* gate is subject to glitches and do not propagate the precharge in DPLs like WDDL, therefore it is not used in such countermeasures. However, by adding the BCDL synchronisation scheme and $U/\overline{PRE}$ signal, it can be made glitch-free and able to properly precharge. As a matter of fact, by observing Table 4.9 it is clear that during the precharge state (when $U/\overline{PRE} = 0$), the output is always null. Moreover, the output is non null only when both inputs are in a valid state.

This way, any 2-input BCDL function can be implemented with only 2 $LUT5$ as shown in Figure 4.23.

Table 4.9: Truth table of a 2-input BCDL *XOR* gate.

$U/\overline{PRE}$	a_t	a_f	b_t	b_f	s_t	s_f
0	X	X	X	X	0	0
1	0	0	X	X	0	0
1	1	1	X	X	0	0
1	X	X	0	0	0	0
1	X	X	1	1	0	0
1	0	1	0	1	0	1
1	0	1	1	0	1	0
1	1	0	0	1	1	0
1	1	0	1	0	0	1

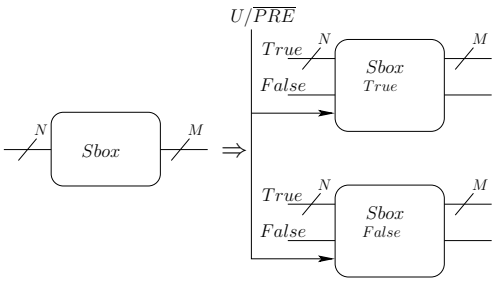


Figure 4.22: BCDL S-Boxes.

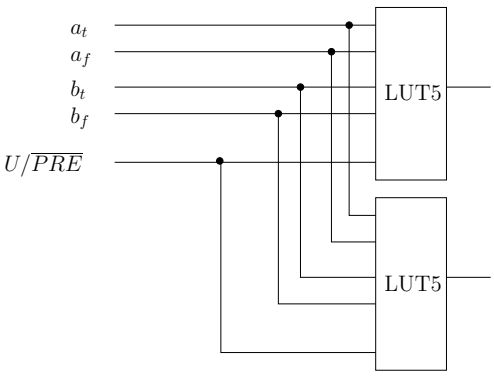


Figure 4.23: Optimized -BCDL 2-input gate.

High Performances

As of now, all DPL-based counter-measures have about the same performances and that is a speed at least two times slower than the unprotected architecture. This is mainly the result of the typical 2-phase functioning (*precharge*, *evaluation*) which is common to all DPLs. Most of the times, the precharge must have roughly the same duration as the evaluation, in WDDL for example, it must last long enough for the 0 to go through all the logic. On the other hand, thanks to the global precharge signal, BCDL can be optimized to be faster than any other DPL. As a matter of fact, the global signal being extremely fast and homogeneously distributed throughout the device, the duration of the precharge state can be reduced significantly. More time is then given to the evaluation, which dictates the speed of the design. This can be achieved by using a non-regular clock, as shown in Figure 4.24. Using this scheme, the speed can be raised up to $\sim 1.3 - 1.5$ times the basic one.

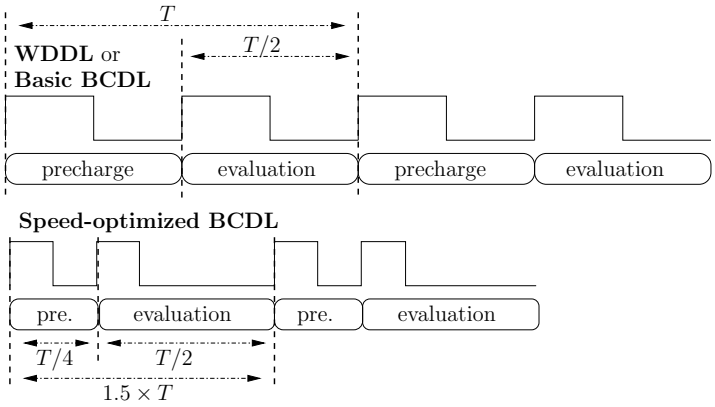


Figure 4.24: Basic BCDL *versus* speed-optimized BCDL timings.

Built-In Robustness against Fault Attacks

BCDL, as any DPL style without early propagation, is fully immune against setup violation attacks as demonstrated by Selmane and al. in [156] . Moreover BCDL is, by construction, resilient to simple faults ($1 \rightarrow 0$ and $0 \rightarrow 1$) and mostly immune against multiple faults. As a matter of fact, as was previously illustrated in Table 4.9, the truth table of BCDL cells is such that if a single net is stuck at 0 or 1 (i.e. $true$ and $false$ nets of the same variable have the same value), both true and false cells will output 0, an "error state". Due to the propagation properties of BCDL cells, this error state is then automatically propagated till the output, quickly spreading through all the following logic layers and thus erasing the faulty output. Therefore, the only fault model that can lead to systematic successful attacks is the dual bit-flip, namely a simultaneous $0 \overset{*}{\rightarrow} 1$ and $1 \overset{*}{\rightarrow} 0$ on the same dual-rail couple. Then when considering

multiple faults, a large quantity of simple faults will be randomly generated, with regards to a dual net couple (inducing error states), along with some more unlikely but devastating bit-flips. However, as the error states are systematically propagated, they will likely replace those before the output, hence making them unexploitable. This absorption property is all the more efficient as the number of error states generated by the multiple faults is high. Therefore, the only way to inject a poisonous fault is to stress the circuit sufficiently to have multiple faults, without nonetheless creating too many so as to leave a chance for them not to be absorbed during their percolation towards the outputs. For more detailed demonstration, the reader is referred to [22].

4.3.2 Implementation on Stratix II

Regarding the implementation of BCDL, two orthogonal methodologies were considered:

- A "Top-Down" approach, to automatically generate the BCDL netlist from a single-rail one.
- A "Bottom-up" strategy, consisting in building basic BCDL blocs and using them to construct the whole design by hand.

Top-Down

The goal of this strategy is to be able to automatically generate the BCDL version of most algorithms, from a single-rail netlist. The design flow, illustrated in Figure 4.25 can be described as follows:

1. The RTL description of the coprocessor is written in Hardware Description Language (HDL), with clearly separate control and datapath parts.
2. The datapath is fed into an ASIC Synthesizer (Cadence RC compiler), which uses a custom library of FPGA cells. It outputs a post synthesis *.vm* file, containing the single-rail design described in LUT functions.
3. This file is then inputted into an ad-hoc software, written in *perl*, for dualization. The *.vm* netlist is parsed and single-rail LUT functions are transformed into BCDL cells, written as Altera specific LUT primitives. The conversion from single-rail to BCDL cells requires the following transformations:
   - Every signal is dualized into a "True" and "False" couple.
   - Each n-input function f is replaced by pair of $(n+1)$-input functions $(f, \overline{f})$. The additional input being plugged to the $U/\overline{PRE}$ signal.
   - Synchronisation logic, as described in Section 4.3.1 is added before every cell.

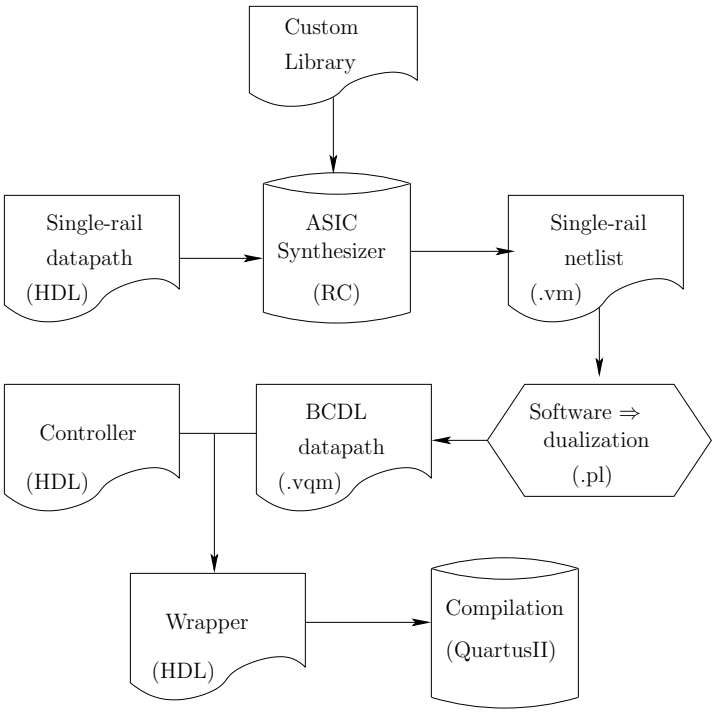


Figure 4.25: BCDL Top-Down Compilation Flow.

- Registers are replaced by two pair of master-slave registers driven on the same clock, similar to WDDL registers.

Eventually, this software outputs a Verilog Quartus Mapping File (*.vqm*) .

4. A single- to dual-rail wrapper is designed to link the control part to the BCDL datapath and the whole design is then compatible with the QuartusII software for compiling and generating the bitstream.

This method has the advantages of being fully automatized and able to convert most single-rail datapaths to BCDL style. However it also comes with a few drawbacks, indeed the design must be clearly separate between control and datapath and it requires the possession of a commercial ASIC Synthesizer.

Bottom-Up

The second approach is the complete opposite. The netlist is built "by hand", with only a few optimised BCDL blocs. The design flow in proceeds as follows:

1. As before, the control and datapath parts are designed separately.

2. The datapath description is done in a structural way, using only the basic operators (or primitives) of this algorithm. For instance, the AES can be implemented with only *XORs*, registers, multiplexers and RAM blocs for the S-Boxes.
3. Then the single-rail datapath is dualized as follows:
   - All primitives are transformed into BCDL compliant primitives, including synchronisation logic and the additional $U/\overline{PRE}$ signal.
   - All signals are dualized into a dual couple.
   - RAM blocs are duplicated to fit the "True" and "False" paths and the $U/\overline{PRE}$ signal is added, as an extra address bit, to create the BCDL S-Boxes depicted in Section 4.3.1.
4. Finally a single- to dual-rail wrapper is devised as in the "Top-Down" strategy and the resulting design can be fed to the Altera CAD tools.

This method proves to be more time consuming to deploy that the "Top-Down" one, however it is entirely ad-hoc and leave place for hand-made design optimisations.

BCDL AES 128 implementation cost for both architectures

AES was implemented with both "Top-Down" and "Bottom-Up" approaches, however an interesting optimisation can be performed for the latter, by taking advantage of recent FPGA technology, especially ALM-based FPGA in the case of the Altera family, that allows to implement two 5-input functions in the same ALM, provided that they share at least three common inputs [10]. This way, a complete BCDL *XOR* gate, comprised of the dual couple (*XOR*,*XNOR*), can be fitted in the combinatorial part of one single ALM.

Area consumption, in terms of ALM, RAM blocs and registers, for both BCDL implementations, are compared with those of the unprotected AES, as well as the maximal clock frequency. Results are given in Table 4.10.

Table 4.10: BCDL Implementation Cost.

Architecture	Unprotected AES	Top-Down BCDL	Bottom-Up BCDL
ALUT	819	9272	3662
Registers	271	1041	1041
RAM blocks (M4K)	20	—	40
Frequency (MHz)	80.1	37.8	39.7

For the Bottom-Up approach, the number of registers and Aluts are respectively quadrupled and multiplied by factord 4.5. Indeed each 2-input function requires two 5-input LUTs and each register is dualized in two master-slave couples. As the the

M4K memory blocks are under-exploited by the single-rail implementation (a sole 256-byte S-Box is stored in one M4K), the number of blocks is only doubled for this BCDL design, as one BCDL 512-byte S-Box can be stored in one block.

With the Top-Down approach, the number of ALUTs is roughly multiplied by a factor 11.5, however the S-Boxes are implemented in ALUTs, which make the comparison difficult.

4.3.3 Experimental Results

Given the implementation results and the constraining necessity of using a commercial ASIC synthesizer in order to deploy the Top-Down strategy, experimental evaluation are focused on the Bottom-Up approach.

First the "Mutual Information as a Metric" (MIM) is computed in order to evaluate the information leakage, as described in Section 2.1.3.

Figure 4.26 and Figure 4.27 respectively depict the MIM, in number of bits, on all sixteen S-Boxes for the reference unprotected AES 128 and the BCDL AES implementations.

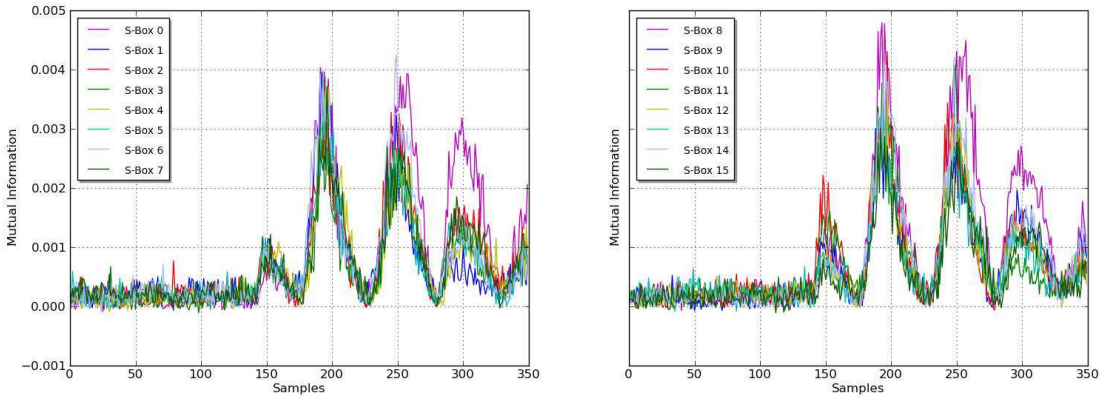


Figure 4.26: MIM on the unprotected AES.

On one hand, the reference AES displays peaks of information leakage from 0.002 to roughly 0.005 bits depending on the S-Box. On the other hand, the overall mutual information of BCDL is clearly smaller, with several bytes displaying a very low information leakage (less than 0.0003 bits) and others presenting a relatively high one, between 0.0006 and 0.0014 bits. The latter should therefore be more vulnerable to classical SCAs. However multi-bit analysis on DPLs is not optimized as different imbalances may compensate one another. Therefore, to precisely assess this weakness, mono-bit analysis is performed. The example of S-Box0 and S-Box8 are depicted in Figure 4.28. It is noteworthy that only one bit over eight displays a significantly high

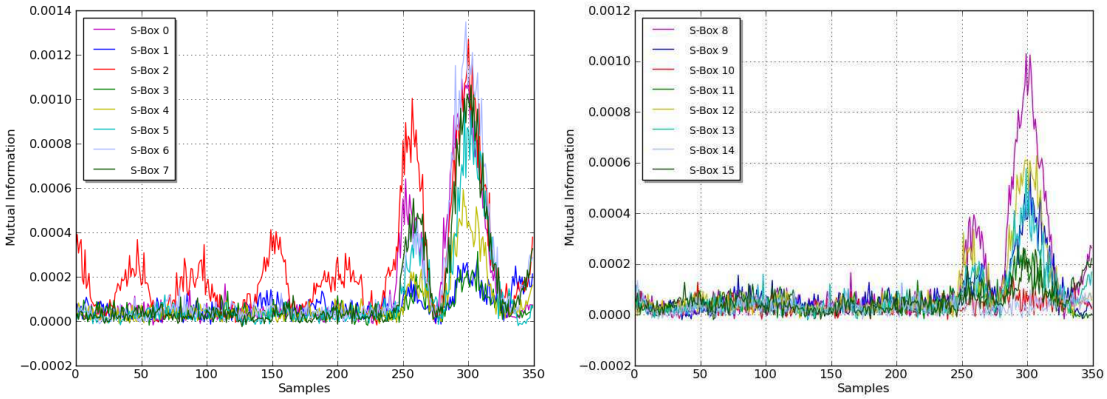


Figure 4.27: MIM on the BCDL AES.

leakage. This is most probably due to routing imbalances on this particular couple of dual nets.

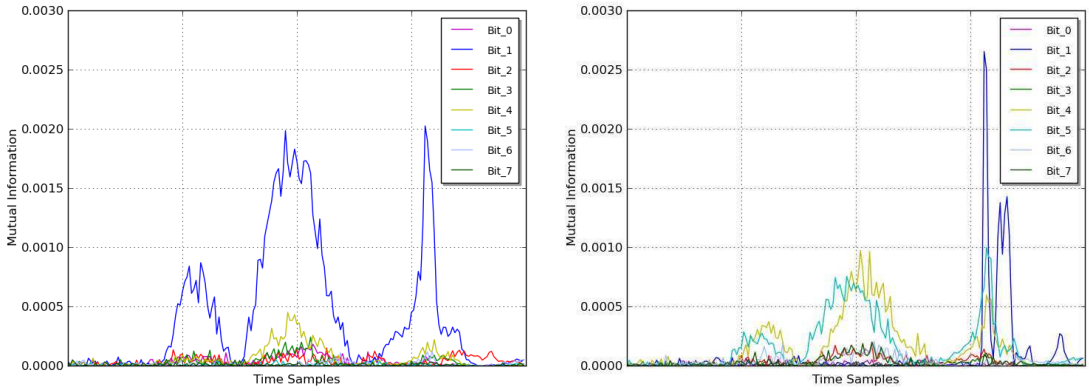


Figure 4.28: Mono-bit MIM on BCDL AES S-Box0 (left) and S-Box8 (right).

Other behaviours can be observed for instance on S-Box4 and S-Box7 (Figure 4.29). In those cases, there are no such gaps as before, although a few bits can be singled out.

To validate those analyses, actual CPA is performed on both the reference unprotected and the Bottom-Up BCDL implementation. If the leakage represented by the MIM is proportional to the security level of the countermeasure, the most leaking bits should be the less robust. Using the same setup as in Section 4.2.2, respectively 160000 and 30000 traces are recorded for the BCDL and unprotected AES designs. Note that, in order to reduce the noise, each trace is in fact an average over 256 power measurements. In the following the number of traces are displayed for the sake of simplifying the notations. CPA is performed on the latter with the classical Hamming Distance

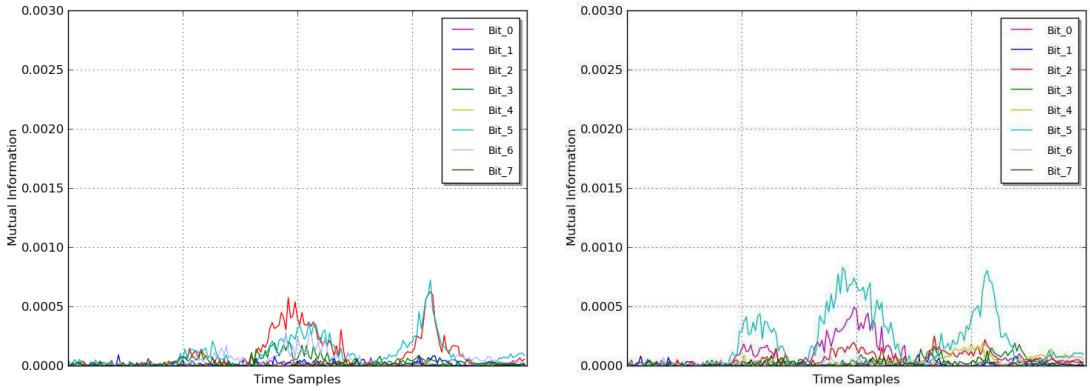


Figure 4.29: Mono-bit MIM on BCDL AES S-Box4 (left) and S-Box7 (right).

(HD) model on 8 bit for each S-Box. Results are displayed in Table 4.11. As can be expected, all sub-keys are easily recovered, most of them in less that 4500 traces.

Table 4.11: CPA on the reference AES 128.

S-Box	0	1	2	3	4	5	6	7
MTD	2382	3123	3741	4188	3838	3402	1925	3516
S-Box	8	9	10	11	12	13	14	15
MTD	2240	3315	8778	4482	3299	1488	1892	2524

On the other hand, analyses on DPLs are best done on one bit rather than several and using the Hamming weight model, as stated previously. Thus such methodology is applied on the BCDL implementation and corresponding results are given in Table 4.12. In order to conduct a thorough study, CPA is performed on every bit of all S-Boxes. Whenever the attack is unsuccessful, meaning that the actual secret key cannot be singled out during the analysis of the 160000 side-channel measurements, a minus ("-") character is displayed in Table 4.12. In other cases, four values are given:

- The number of traces corresponding to the beginning of the first stability run for the considered sub-key, denoted by "S_{idx}" (the lowest values are shaded with gray). Generally this would correspond to the number of MTD, indeed once an actual secret sub-key is stable, it usually remains that way with increasing number of processed traces. However in this case, most keys are not perfectly stable from S_{idx} to the end of the 160000 traces (this could result from several factors, including acquisition issues). In this context, labeling those attacks as successes or failures is not straightforward. Unfortunately, due to the time consuming nature of this analysis, the first-order success rate could not be computed, although

is is clear that such behaviour directly affect this metric. For instance, an adversary that would only consider the result after 160000 processed traces could end up finding a false key, although the actual secret key was stable for most of the analysis.

- The stability length (in percentage of 160000) is displayed in the row following S_{idx} and denoted by "S_L (%)". Moreover, when the key hypothesis found after all traces are processed is actually the secret key the table contains a "✓".
- The security gain with regards to the unprotected implementation is showed in the last row , when considering the worst-case scenario, namely that S_{idx} equals the number of MTD.

As can be observed, S-Box6 remains unbroken over the 160000 measurements and globally the CPA is unsuccessful on most bits. Nevertheless, all other S-Boxes appear to be vulnerable to first-order CPA on at least one and at most four bits out of eight. The S_{idx} value varies from 2773 for bit 1 of S-Box9 to 70206 for bit 7 of S-Box16, leading to respective security gain factors of 1.2 and 28. As always, a proper robustness evaluation must be made be considering the worst-case scenarios, namely the lowest security gain for each S-Box. Even then several S-Boxes show significant improvement, for instance S-Box7, 9 and 13, with gain factors of respectively 12.8, 16.6 and 29.3. Unfortunately a few subkeys are recovered with a relatively small number of traces, with regards to the unprotected implementation. That is the case for the gray shaded cells of Table 4.12, namely S-Box0, 8 and 11 whose security gain is less than 3. It is noteworthy that only the attack on bit 1 of S-Box0 is stable from $S_{idx} = 5332$ till the end of the 160000 traces. On other S-Boxes the right key is systematically lost before the end, which would affect the success rate of this CPA. To summarize, to global security gain of BCDL is very promising, all the more when compared with the previous results on DPLs with and without constraint placement displayed in Table 4.8. Moreover, even though a sub-key is considered broken when the CPA is successful on at least one bit, the fact that most bits of this implementation show high robustness implies that an adversary would have to conduct an analysis, on all bits of all S-Boxes, in order to obtain similar results, which by essence add to the overall security of this countermeasure.

Now when confronting our previous MIM observations with those results, a definite correspondence can be observed. As a matter of fact, the theoretical leakage of S-Box0 was the strongest for bit 1 with a clear gap and the second most leaking bit was the fourth. Then as depicted in Table 4.12, bit 1 is the most vulnerable of S-Box1, with $S_{idx} = 5332$ and $S_L = 97\%$. Moreover, the only other weak bit is the fourth, with $S_{idx} = 7455$ and $S_L = 73\%$, which implies a notably less efficient attack. The same conclusions can be drawn when comparing the MIM and experimental result for S-Box8 and 7. Indeed, bit 1 of S-Box8 show considerable leakage, followed by bit 4

Table 4.12: CPA on the BCDL AES 128.

S-Box	**0**								**1**							
Bit	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
S_{idx}	-	5332	-	-	7455	-	-	-	-	14851	-	-	-	40328	-	32318
S_L **(%)**	-	97 ✗	-	-	73 ✗	-	-	-	-	91 ✓	-	-	-	73 ✗	-	79
Gain	-	2.2	-	-	3	-	-	-	-	4.8	-	-	-	12.9	-	10.3
S-Box	**2**								**3**							
Bit	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
S_{idx}	-	57994	-	-	-	-	-	-	-	-	-	-	51898	45215	57289	-
S_L **(%)**	-	60 ✗	-	-	-	-	-	-	-	-	-	-	65 ✗	70 ✗	62 ✗	-
Gain	-	15.6	-	-	-	-	-	-	-	-	-	-	12.4	10.7	13.5	-
S-Box	**4**								**5**							
Bit	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
S_{idx}	-	-	21350	-	-	13374	53798	-	-	-	26820	-	22861	-	-	33774
S_L **(%)**	-	-	86 ✗	-	-	92 ✗	65 ✗	-	-	-	83 ✗	-	85 ✗	-	-	77 ✗
Gain	-	-	5.5	-	-	3.4	14.2	-	-	-	7.9	-	6.7	-	-	9.7
S-Box	**6**								**7**							
Bit	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
S_{idx}	-	-	-	-	-	-	-	-	-	-	-	-	-	44904	-	-
S_L **(%)**	-	-	-	-	-	-	-	-	-	-	-	-	-	70 ✗	-	-
Gain	-	-	-	-	-	-	-	-	-	-	-	-	-	12.8	-	-
S-Box	**8**								**9**							
Bit	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
S_{idx}	-	2773	-	-	14908	11718	-	-	54932	-	-	58685	-	-	-	-
S_L **(%)**	-	99 ✗	-	-	91 ✗	93 ✗	-	-	62 ✗	-	-	60 ✗	-	-	-	-
Gain	-	1.2	-	-	6.8	5.0	-	-	16.6	-	-	17.6	-	-	-	-
S-Box	**10**								**11**							
Bit	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
S_{idx}	33584	-	-	-	-	-	-	-	19263	-	-	-	-	12666	-	-
S_L **(%)**	78 ✗	-	-	-	-	-	-	-	88 ✗	-	-	-	-	92 ✗	-	-
Gain	3.8	-	-	-	-	-	-	-	4.2	-	-	-	-	2.7	-	-
S-Box	**12**								**13**							
Bit	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
S_{idx}	33569	-	-	-	-	-	43185	-	43886	-	-	-	-	-	-	-
S_L **(%)**	78 ✗	-	-	-	-	-	72 ✗	-	59 ✗	-	-	-	-	-	-	-
Gain	10.1	-	-	-	-	-	13.0	-	29.3	-	-	-	-	-	-	-
S-Box	**14**								**15**							
Bit	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
S_{idx}	-	-	35695	20705	6380	17261	-	-	15706	-	-	-	-	-	-	70206
S_L **(%)**	-	-	77 ✗	87 ✗	96✗	89 ✗	-	-	90 ✗	-	-	-	-	-	-	53 ✗
Gain	-	-	18.9	10.5	3.3	8.9	-	-	6.4	-	-	-	-	-	-	28.0

and 5, which clearly concurs with the CPA. Nonetheless, MIM on S-Box4 singled out bit 5 and 3 as the most obvious targets, however the CPA was not successful on the latter. Moreover, the corresponding sub-key was recovered by attacking bit 2 and 6

which didn't stand out in the MIM. With the hypothesis that the measures and computations were properly performed, this suggest that the leakage evaluation by MIM, as it was computed here, is not sufficient to perfectly describe the robustness level against SCAs, but remains an interesting metric to evaluate the global security of such countermeasures.

4.4 Comparative DPL Overview and Conclusion

Table 4.13 draws up a comparison of the main DPL styles, in terms of principle, design constraints and performance, highlighting most of the known advantages (synchronization,...) and drawbacks (primitives, back-end constraints,...) of such countermeasures.

Masking allows to greatly reduce the technological bias, but also results in a significant increase of area and requires the addition of a Random Number Generator (RNG). As a matter of fact, it involves at least a transformation of 2-input operations into 3-input majority function (MDPL) or into a 4-input RSL gate (DRSL). Moreover, as discussed in Section 3.2.2 successful attacks have recently been presented on those countermeasures.

Synchronization on both precharge and evaluation is mandatory to avoid glitches and early propagation effects, as described in Section 4.1.4.

Primitive constraints induce a higher complexity, by reducing the panel of usable functions (like in WDDL where only positive gates are allowed), or by binding the designer to use specific functions that can be more area-consuming or slower than basic ones (MDPL, DRSL).

Back-end constraints generate extra compilation time and design work as the P/R stage has to properly balanced the "True" and "False" networks. It can also cause a loss of performance, like in STTL where the synchronisation signal must be manually made slower than the others, by adding delay elements between each gates, in order to ensure that it always switches last.

Technological bias is a significant source of information leakage and must therefore be as low as possible to ensure a perfectly secure counter-measure, as stated in Section 4.1.2.

Performances and Area are most significant when considering the industrial needs and designing architectures including numerous IPs on the same FPGA. In Table 4.13, performances are described by a factor with regards to the unprotected implementation, while the area consumption is represented by a number of asterisks ($*$), between one to six, six being the lowest consumption.

$*$

When considering the case of industrial applications, the issues of low resource consumption and high security are both mandatory. Unfortunately no countermeasure can be both perfectly secure and extremely low-cost, hence customized trade-offs must be found for each designs. In this chapter, after an evaluation of the existing DPL vulnerability, two countermeasures are described.

First, two constraint placement strategies are described to enhance the security of classical DPLs like WDDL, by reducing the technological bias due to routing imbalance. Results of a CPA carried out on all bits show clear improvements in terms of

Table 4.13: DPL performance and security features overview.

Logic	Mask (+RNG)	Synchro		Constraints		Tech Bias	Perfs.	Area
		Pre	Eval	Primitives	Back-end			
Unprotected	no	✗	✗	no	no	-	1	∗∗∗∗∗∗
WDDL[176, 178]	no	✗	✗	positive gates only	balanced place&route	high	$< 1/2$	∗∗∗
WDDL [14] **with DBD**	no	✗	✗	positive gates only	copy&paste	low	$<< 1/2$	∗∗
IWDDL [117]	no	✓	✓	no	netlist post-processing	low	$< \frac{1}{2 \cdot n_i}$ ‡	∗∗
SDDL [176]	no	✗	✗	no	copy&paste	low	$< 1/2$	∗∗∗
SDDL [184] **with P&R**	no	✗	✗	no	copy&paste + P&R	no	$<< 1/2$	∗∗∗
Partial DDL [92]	no	✗	✗	no	copy&paste	low	$< 1/2$	∗∗∗∗∗
DWDDL [192]	no	✗	✗	positive gates only	copy&paste	no	$< 1/2$	∗
MDPL [135]	yes	✗	✗	MAJ †	no	no	$< 1/2$	∗∗
iMDPL [134]	yes	✓	✓	MAJ †	no	no	$< 1/2$	∗
DRSL [39]	yes	✓	✗	no	no	no	$< 1/2$	∗∗
STTL [164]	no	✓	✓	no	delay on sync signal	very low	$< 1/5$	∗
BCDL	no	✓	✓	no	balanced place&route	low	$> 1/2$ ⋆ $< 1/2$	∗∗∗

† MAJ stands for the majority gate: $MAJ(a, b, c) \doteq a \cdot b + b \cdot c + c \cdot a$.
‡ n_i is the maximum number of inverters amongst all combinatorial paths.
⋆ when using the speed optimisation of Section 4.3.1.

overall robustness with two bits still robust with 6,400,000 measurements. Nevertheless, the global security level as well as the overhead in compilation time make this scheme not sufficient, as it stands, to be the sole countermeasure against SCA for an industrial application.

Second, BCDL proves to be an interesting trade-off between robustness and complexity, when compared to other existing DPLs. As a matter of fact most of them are still vulnerable to classical or specific SCAs, while being more consuming in terms of resources and/or performances than BCDL. Moreover, its level of resilience against simple as well as multiple fault attacks allows this scheme to be viewed as a global countermeasure against both passive and active SCAs. In that regard, BCDL could become an alternative to masking for relatively low-cost devices. Although masking schemes generally display better performances than DPLs in hardware architectures,

the need for additional fault detection mechanism, like redundancy, put them on equal terms with several DPLs.

Nevertheless, as shown by our experimental results, there are yet flaws in the security of this countermeasure, as a few leaking bits still show vulnerability to classical CPA, given that a large number of side-channel measurements are available. It is noteworthy that an alternate implementation of BCDL using T-Box was proposed by Bhasin and al. [23] in order to remove this leakage and resulted in a diminution of the global leakage. However in a similar way as was presented in the previous Section, a few highly leaking bits remained on most S-Boxes, allowing a quasi full key recovery.

Therefore, as a perspective, additional research should be undertaken to pinpoint those leaking bits and properly assess the cause of their vulnerability, which is most probably local routing imbalances between dual nets. In this case the addition of customized placement constraints could be an appropriate solution to reduce such leakage and greatly improve the security level of BCDL.

Chapter 5

New Masking Scheme for AES

5.1 Masking Vulnerability

As stated in Section 3.2.1, state of the art first-order masking schemes are still vulnerable to several SCAs, including the so-called Variance-based Power Analysis (VPA [167, §4.3]) and high-order SCAs. This can be explained by studying the Probability Density Functions (*pdf*) of the registers activity. As a matter of fact, classical masking architectures are composed of two parallel paths: one for the masked sensitive data and the other for the mask alone. The global activity corresponds to the sum of those two, which results in an imbalance of the corresponding *pdf*s during an attack.

Let's take the example of AES 128. Hardware implementations of this algorithm are preferably attacked on the last round. Indeed, it is possible to guess one byte, noted y, of round 9 from one byte of the cipher-text x simply by guessing one 8-bit portion of the last round key, because there is no $MixColumns()$ operation in this round. The activity $\mathcal{A}$ of the last round can then be described as the sum of the Hamming Distance between the last two registers of both the masked data and the mask paths such as:

$$\begin{aligned} \mathcal{A} &= HW((y \oplus m) \oplus (x \oplus m')) + HW(m \oplus m') \\ \mathcal{A} &= HW(z \oplus \Delta(m)) + HW(\Delta(m)) \ , \end{aligned}$$

where:

- m and m' are respectively the values of two last mask registers.
- $\Delta(m) = m \oplus m'$.
- $z = x \oplus y$.

When performing a classical first-order analysis like DPA or CPA, as well as a VPA, the considered activity model is: $\mathcal{A}' = HW(z)$, thus side-channel measurements are sorted and classified in partitions depending on this value, that varies from 0 to 8 (for 8-bit registers). Therefore, given that $\Delta(m)$ can take any 8-bit value, there are nine

possible *pdf*s of $\mathcal{A}$, depending on $HW(z)$. These theoretical distributions are depicted in Figure 5.1.

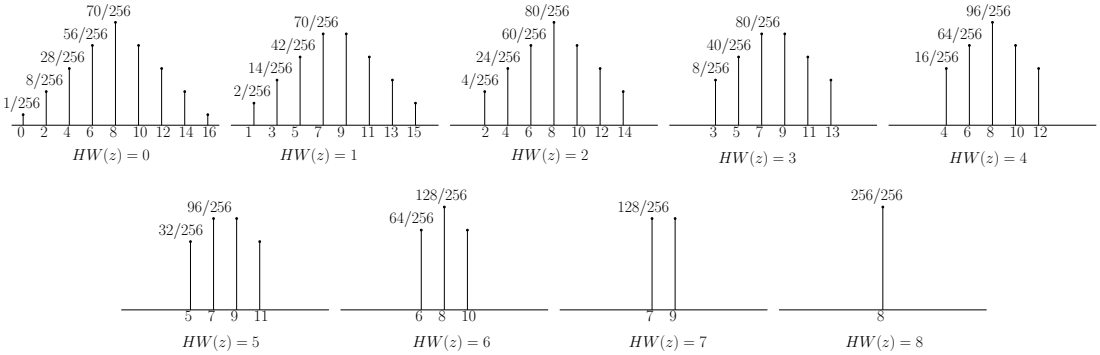


Figure 5.1: *pdf*s of an AES 8-bit register activity with state of the art masking.

As can be observed, all distribution present the exact same mean, namely 8, which induces robustness against DPA and CPA. However, their variances are visibly decreasing when the value of $HW(z)$ increases. This property results in a side-channel leakage, which is the source of the vulnerability against VPA and similar attacks.

In this section, simulations are performed in order to illustrate these differences during a VPA or second order zero-offset CPA. Two types of target architectures are simulated:

1. A reference unprotected AES 128.
2. A protected AES 128 using Boolean masking.

The boolean masking is chosen considering that it is the most vastly spread variant, with proven security against first-order attacks.

The leakage is a function of the distance between x and y, *i.e.* $x \oplus y$ [169]. This simulation considers the "Hamming Distance" consumption model, as it is most commonly used in SCAs [33],especially on FPGAs [169]. The sensitive variable is the value $z = x \oplus y$.

Observations ($\mathcal{O}$) were simulated as single values, equal to the theoretical leakage L corresponding to one byte of the last sub-key k. Simulations were performed both in a "perfect" scenario: without any noise (5.1); and with the addition of a Gaussian random variable $R_g(m, \sigma)$, of mean $m = 0$ and increasing variance σ, representing possible algorithmic noises (5.2).

$$\mathcal{O} = L. \tag{5.1}$$

$$\mathcal{O} = L + R_g(0, \sigma), \sigma \in 1, 2, 3. \tag{5.2}$$

Depending on the architecture, the simulated leakages take different values:

- *Unprotected reference*: Hamming Distance between the last state register and the cipher-text, such as:

$$L_{ref} = HW(x \oplus y)$$
$$L_{ref} = HW(z)$$

- *State-of-the-art*: sum of Hamming Distance for the masked data part and Hamming Distance for the mask part, such as:

$$L_{mask} = HW(z \oplus \Delta(m)) + HW(\Delta(m)) \ ,$$

The following analyses are performed:

- Differential Power Attack (DPA), on both architectures;
- Correlation Power Attack (CPA), on both architectures;
- Variance-based Power Attack (VPA), on the state of the art;
- Mutual Information as a Metric (MIM).

Results are evaluated using the first-order success rate and guessing entropy metrics, proposed by Standaert in [168]. An attack is said to be successful when a success rate of 90% is reached.

As can be expected in this environment, DPA and CPA are, on one hand, clearly efficient on the unprotected version, as depicted in Figure 5.2. Indeed, with the noise variance increasing, the 90% success rate threshold is reach between 10 and 200 observation. On the other hand, those attacks remain unsuccessful on the masked version for up to 200000 observations, even in the absence of noise.

Nevertheless, the VPA simulation proves to be successful on the protected architecture in 1000 to 15000 observations depending on the noise, as depicted in Figure 5.4. As stated before, this is due to the disymmetry in the *pdfs* for the different partitions. To illustrate this phenomena, the *pdfs* generated by one simulated attack, without any noise and over 200000 observations, are displayed in Figure 5.3. It is noteworthy that the theoretical distribution properties are validated, hence leading to the successful attack.

Mutual information is used to evaluate the information leakage, as described in Section 2.1.3 and referred to as "Mutual Information as a Metric" (MIM). In this context, simulation results in a leakage of $\approx$ 1.0 bit on the *state-of-the-art* masking and

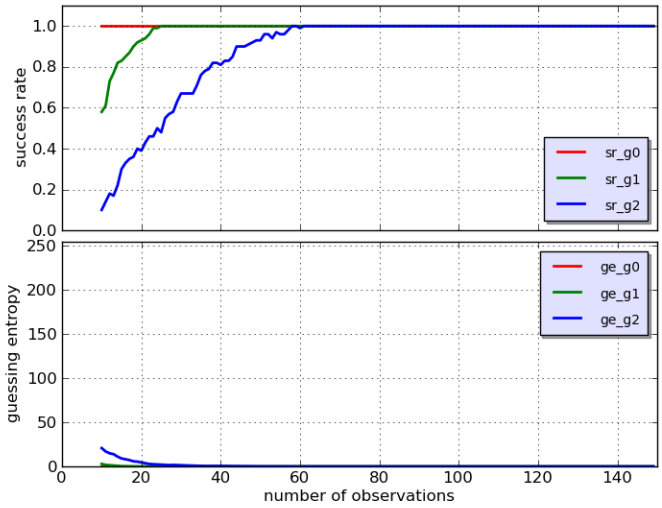


Figure 5.2: Success Rate and Guessing Entropy for 100 CPA Simulations on an unprotected AES.

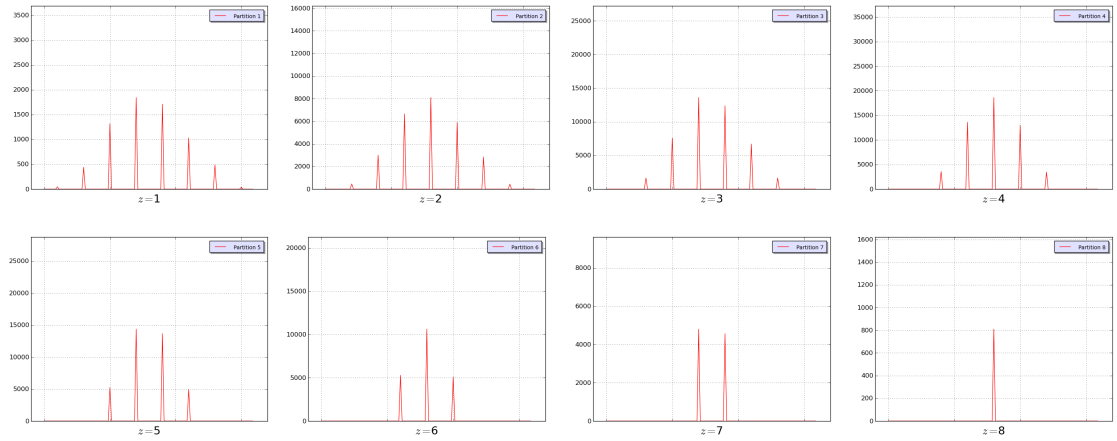


Figure 5.3: "State-of-the-Art" Masking *pdfs* during Simulation on 200000 observations.

$\approx$ 2.5 bit for the unprotected version. It is noteworthy to recall that a leakage metric points out vulnerabilities, that could in practice not be exploitable by an adversary.

To conclude, in order to be resistant to VPA and similar attacks, it is mandatory to either integrate high order masking schemes or use a different architecture, not based on the usual 2-path structure.

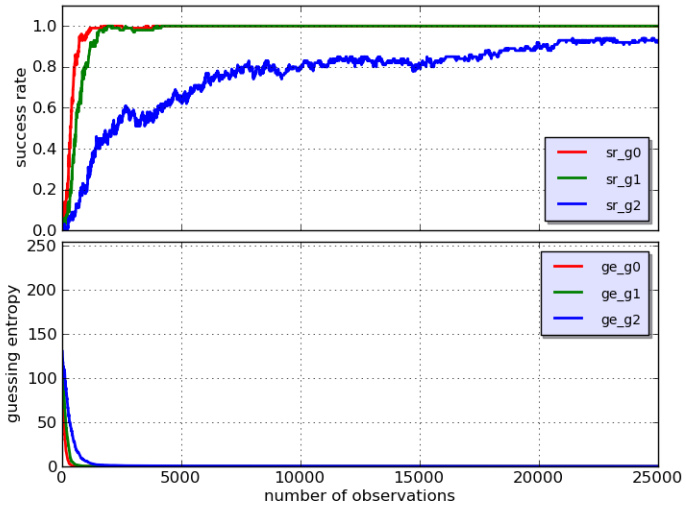


Figure 5.4: Success Rate and Guessing Entropy for 100 VPA Simulations on "State-of-the-Art" Masking.

5.2 New masking scheme: Rotating S-Box Masking (RSM)

Taking into account the existing masked architectures vulnerabilities and area overheads a new FPGA-targeting scheme is devised, with the goal of removing the weakness against variance-based attacks, while keeping the area and performance overhead as low as possible.

In the following are given a detailed description of the proposed countermeasure, namely RSM (Rotating S-Box Masking), in terms of rationale and architecture, as well simulation and experimental results. The considered architecture is a straightforward implementation of *AES 256* on FPGA, without pipelining and with 16 S-Boxes implemented in ROM. The following notations are employed: *S* for S-Box, *SB* for the whole SubBytes operation, *SR* for ShiftRows, *MC* for MixColumns.

5.2.1 RSM: Principle and Implementation

To our best knowledge, the state-of-the-art masking in hardware (*e.g.* [142, 171]) are based on the *Global Look-up Table* scheme: as stated in Section 3.2.1 the S-Boxes are addressed by the masked data and the mask, thereby inducing a side-channel leakage. Instead, the RSM countermeasure adheres to the *Re-computation Method* described in the same Section. The S-Boxes are addressed only by the masked data: RSM has a mono-path structure. This feature grants to RSM an immunity to variance-based attacks (VPA) and makes it considerably resistant to MIA [18]. Nonetheless, RSM is

based on using precomputed sets of constant masks rather than random ones and specific customized S-Boxes with built-in input and output unmasking/masking operations.

Rotating S-Boxes

As stated in Section 3, for most countermeasures on symmetrical cryptoprocessors, like AES or DES, the critical part in terms of area and computation time is the non-linear operation (*i.e.* the S-Boxes). Therefore, the main improvement brought by our design lies within the definition of low-cost, high performance S-Boxes.

RSM uses the same number of S-Boxes (16) as an unprotected implementation for the entire computation of the AES algorithm. But unlike any previous masking scheme, all those S-Boxes are different. They all contain a mechanism to unmask the input data, perform the basic $S(x)$ (where x is an 8-bit unmasked data) and re-mask it with another constant. However, these new S-Boxes would clearly be a source of first-order information leakage if implemented in logic gates, as the unmasked variable would be associated to an actual net. Therefore those S-Boxes shall be stored in RAM/ROM for either FPGAs or ASICs [157], after being precomputed as follows:

- Before programming the device, sixteen 8-bit constants m_{0-15} are randomly chosen once and for all. Those will be the base masks for the rest of this countermeasure.
- The 16 rotating S-Boxes $S'_{0-15}(x')$ (x' being an 8-bit masked data) are then designed to verify: $S'_j(x') = S(m_j \oplus x') \oplus m_{j+1 \pmod{16}}$, with $j \in \{0-15\}$.
- At each round of the AES algorithm, the S-Boxes are rotated by one position in direction $\mathcal{D}$, in order to successively compute all 16 possible SB'_j such as:

$$SB'_j = SB(M_j \oplus X') \oplus M_{j+1 \pmod{16}}, \forall j \in \{0-15\}\,, \tag{5.3}$$

where *SB* denotes the whole operation on 128-bit data, X' is the 128-bit masked state and $M_j = \{m_j, m_{j+1 \pmod{16}}, \ldots, m_{j+15 \pmod{16}}\}$.

Thus, considering that the 128-bit masked state X' is such as $X'_i = X_i \oplus M_j$ at round i, SB'_{0-15} will unmask it using the first *Xor* with M_j, perform the usual $SB(X)$ and re-mask it with $M_{j+1 \pmod{16}}$. This way, during the next round, thanks to the rotation, the same process will take place, using the next constant: unmasking with $M_{j+1 \pmod{16}}$ and re-masking with $M_{j+2 \pmod{16}}$. The order in which the constants are used is always the same, as it is fixed by the rotation direction $\mathcal{D}$, but depending on the one chosen for the first round: thus 16 different scenarios are possible, which induces a masking entropy of 4 bits.

The rotating S-Boxes can be implemented in hardware by adding barrel shifters on both sides of the SB operation, as shown in Figure 5.5. This way, at each round before

the S-Boxes, the state register can be shifted in direction $\mathcal{D}$, by an amount of bytes equal to the number j of Equation (5.3). Afterwards, the inverse process is performed, in order to rotate the state back to its original position.

Hence our new optimized SB'_{0-15} operator is only composed of 16 customized S-boxes S'_{0-15} (the same size as a basic one) and two barrel shifters, which induce but a small increase in terms of complexity and computation time, with regards to an unprotected AES implementation.

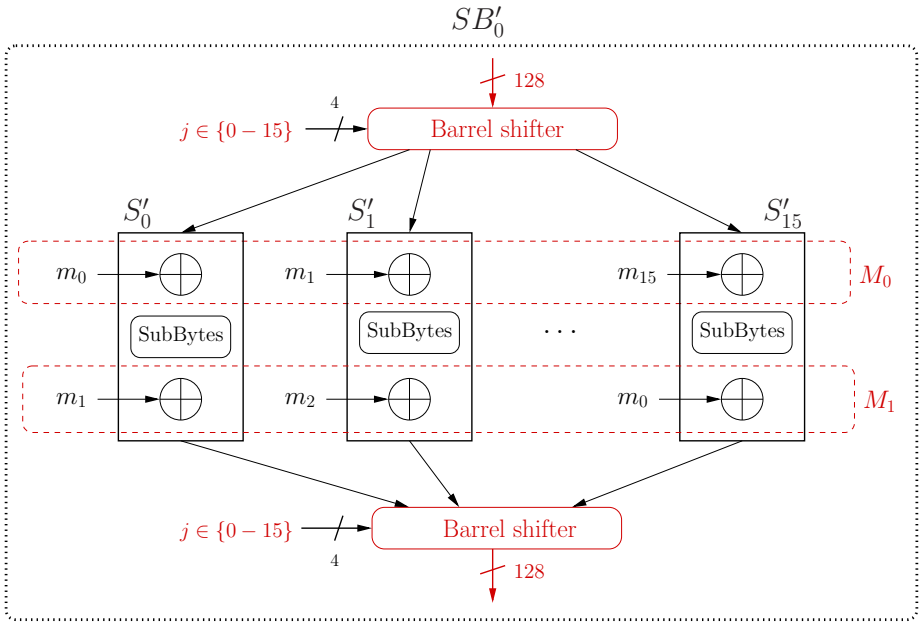


Figure 5.5: Revolving S-Boxes.

Masking the linear operations

From the 16 8-bit base masks, m_{0-15}, chosen to create the rotating S-Boxes, described in the previous Section, 5 sets of 16 128-bit constants are deduced and will be used to mask the linear part of the algorithm, while matching the required inputs and outputs of the SB'_{0-15} operator:

1. The first set consists of basic constants denoted by M_{0-15}, with $M_0 = \{m_0, m_1, \ldots, m_{15}\}$ and M_{1-15} are the successive rotations of one byte of M_0 in direction $\mathcal{D}$, such that:

$$M_j = \{m_j, m_{j+1 \pmod{16}}, \ldots, m_{j+15 \pmod{16}}\}$$
$$\forall j \in \{1-15\}.$$

2. The second set, MMS_{0-15} is defined as:

$$MMS_j = MC \circ SR(M_j) \oplus M_j, \quad \forall j \in \{1 - 15\}.$$

3. Constants of the third set, denoted by MS_{0-15}, verify:

$$MS_j = SR(M_j), \quad \forall j \in \{1 - 15\}.$$

4. Finally the last two sets, namely $IMMS_{0-15}$ and IMS_{0-15} are respectively identical to MMS_{0-15} and MS_{0-15} but with the inverse functions.

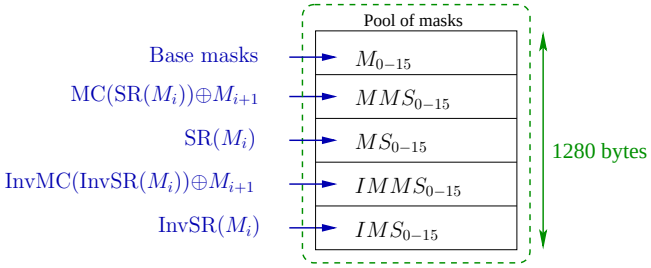


Figure 5.6: Storing masks in ROM/RAM.

These constants are precomputed and stored in RAM/ROM, for a total of 1280 bytes, as depicted in Figure 5.6.

Figure 5.7 depicts the linear part of the datapath.

During the first round, a constant (M_j) is randomly chosen from the first set and *Xor*-ed with the initial plain-text (X) (this way, as in the state-of-the-art masking, the power consumption is decorrelated from the actual data). The resulting masked state, $X'_{state1} = X \oplus M_j$, is the input of SB'_j. Then, as described in Section 5.2.1, its output is $X'_{sbox1} = SB(X) \oplus M_{j+1}$.

Thanks to their linear properties, masking the *SR*, *MC* and *AddRoundKey* functions only requires a simple *Xor* operation. Keeping that in mind, the second set is used to simultaneously unmask the data at the end of each round and re-mask it with the next constant. Hence, the result of the linear operations is:

$$\begin{aligned} X'_{state2} &= MC \circ SR(SB(X) \oplus M_{j+1}) \oplus K_{round} \\ &= MC \circ SR(SB(X)) \oplus MC \circ SR(M_{j+1}) \oplus K_{round}. \end{aligned}$$

Thereby, *Xor*-ing this value with MMS_j, removes the current mask: $MC \circ SR(M_{j+1})$ and re-masks it with M_{j+1}. Thus ensuring that the state register of the next round is indeed the expected masked value $X \oplus M_{j+1}$.

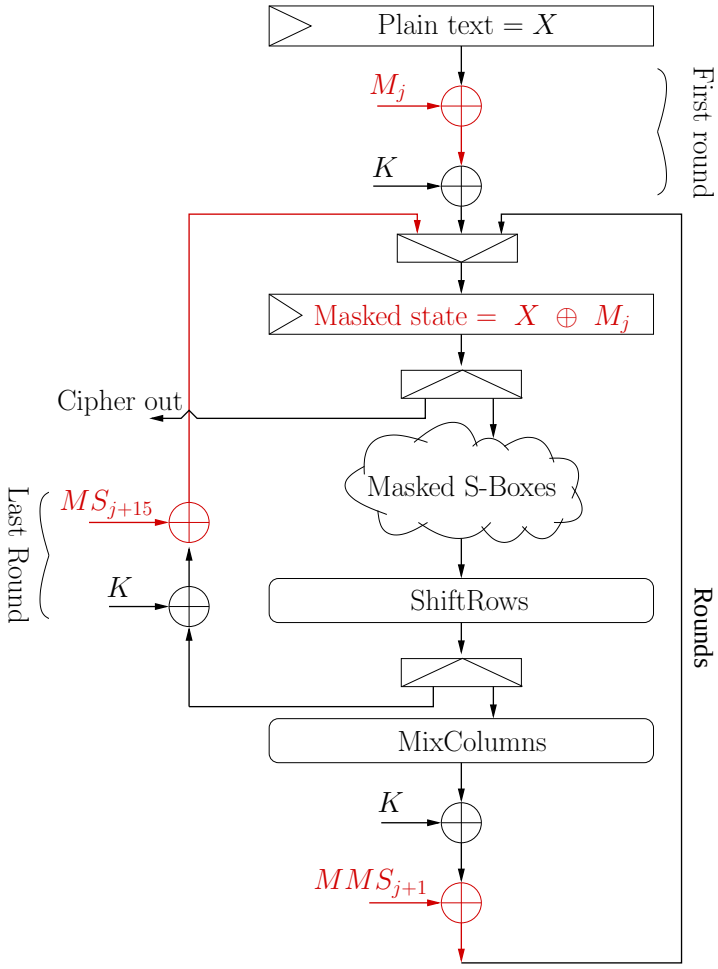


Figure 5.7: Linear part of the RSM datapath.

Finally during the last round, due to the absence of *MC*, the masked ciphered value is:

$$SR(SB(X) \oplus M_{j+14 \pmod{16}}) \oplus K_{round}, \quad j \in \{0-15\}\,.$$

Therefore, it can directly be unmasked with the constants of the third set, *i.e.* MS_j.

5.2.2 Practical Robustness Evaluation

Regarding faults injections, it is noteworthy that RSM, as every other masking techniques, is not intrinsically protected against such attacks. However, classical fault detection schemes for symmetrical algorithms described in Section 3.4.1 are perfectly compatible with this countermeasure and should be implemented, in addition to RSM, in order to ensure a high level of security.

Nonetheless, in order to validate RSM robustness against first- an second-order zero-offset SCAs, both simulation and real-life attacks are performed. Then, in Section 5.3, a complete theoretical security evaluation is given.

Simulation

Under the same conditions as in Section 5.1, simulation is performed on the RSM countermeasure. As before, the leakage model is the "Hamming Distance" and observations are firstly generated without noise, in order to simulate an attack carried out in optimal conditions.

The leakage observations correspond to the Hamming Distance between the last masked state and the unmasked cipher-text, such as:

$$L_{RSM} = HW((x \oplus y \oplus m_{0-15}))$$
$$L_{RSM} = HW(z \oplus m_{0-15}) \ ,$$

where m_{0-15} denotes the 16 8-bit base masks described in Section 5.2.1.

DPA, CPA and VPA prove to be unsuccessful against RSM, with a null success rate for 100 attacks on 200000 observations, as depicted in Figure 5.8. Moreover, the guessing entropy of RSM is roughly stable at 175, which implies that the attack is not likely to succeed even with an increasing number of observations.

The information leakage evaluation with MIM gives only ≈ 0.015 bit for RSM, which is much less than the previous ≈ 1.0 bit for the *state-of-the-art* masking.

Implementation on Altera StratixII

To perform actual attacks and precisely measure the overheads in terms of area consumption and performances, an RSM-protected AES 128, as well as a reference unprotected one were implemented on an Altera StratixII, soldered on a SASEBO-B board provided by the RCIS [86]. It is noteworthy that two different AES architectures were used for this experiments and those of Section 4.3.3. Both bitstreams were generated using version 11.0 of the QuartusII software, with default synthesis and fitter options.

Area occupation and performance results are shown in Table 5.1.

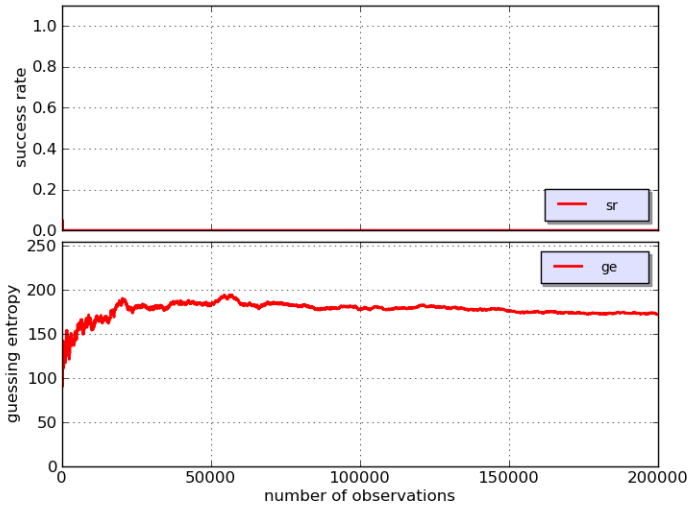


Figure 5.8: Success Rate and Guessing Entropy for 100 VPA Simulations on RSM.

Table 5.1: Implementation results for reference and protected AES

	Unprotected	**RSM**	**Overhead**
Number of ALUTs (%)	1451 (5%)	2049 (7.5%)	48%
Number of M4K ROM Blocs (%)	20 (14%)	28 (19%)	40%
Frequency (MHz)	133.8	88.5	34%

The number of ALUTs and M4K are given both in absolute value and in percentage of the total FPGA resources. As can be observed, the overheads in terms of logical cells, ROM blocks and clock frequency are all within reasonable ranges, even for industrial applications where several IPs are included in the same FPGA.

As stated in Section 3.2.1 few papers have dealt with an actual implementation of a complete masked AES design on FPGA. In [118], Mentens and al. proposed such an implementation, combining Boolean and multiplicative masking. However, this type of countermeasure has been shown to be susceptible to so-called zero-value attacks, that exploit the absence of masking on the `0x00` byte value. More recently, Regazzoni and al. [142] have developed a full Boolean masking scheme on a Xilinx Virtex5 FPGA, obtaining an area consumption of roughly three times the unprotected one and a performance penalty of 50%.

In this context, our implementation brings a significant improvement in terms of area overhead, while keeping a reasonable performance degradation. It is however noteworthy that a precise and fair comparison between FPGA designs is quite difficult

and depends on many factors [53] such as technology, vendor and synthesis options.

Experimental Results

To corroborate the simulation results, the same attacks were performed on the StratixII implementation of the RSM countermeasure. Power consumption measurements were acquired, using a *differential probe* plugged to the positive rail of the FPGA core power supply through a $1\ \Omega$ shunt resistor, coupled with a *54855 Infiniium oscilloscope* from Agilent Technologies [4].

DPA, CPA and VPA were all unsuccessful on 150000 power consumption measurements. Moreover they all displayed a success rate of roughly 0%. As a reference, the results of a CPA, in Measurement To Disclose (MTD), on the unprotected implementation are displayed in Table 5.2. This attack is able to retrieve the entire secret key in less than 12000 power measurements.

Table 5.2: CPA on the reference unprotected AES 128.

Sub-key	1	2	3	4	5	6	7	8
MTD	2821	10215	2627	10372	6333	3046	5194	6841
Sub-key	9	10	11	12	13	14	15	16
MTD	11683	9510	11743	11857	8368	8822	11770	3681

MIM was performed on the same 150000 power traces, in order to experimentally evaluate the information leakage of RSM. Results, displayed in Table 5.3, show that, for all sub-keys, the leakage is included between 0.001 and 0.012 bit, which corroborates the simulation performed in Section 5.2.2 and should hardly be exploitable for a conclusive attack.

Table 5.3: Mutual Information as a Metric on the AES protected by RSM.

Sub-key	0	1	2	3	4	5	6	7
MIM	0.012	0.006	0.008	0.006	0.010	0.007	0.006	0.005
Sub-key	8	9	10	11	12	13	14	15
MIM	0.004	0.011	0.001	0.008	0.004	0.012	0.009	0.002

*

These results empirically demonstrate the robustness of RSM against common SCAs, as well as VPA, which is known to be efficient against state of the art first-order masking schemes. However, in order to properly evaluate the security level brought

by this new countermeasure, a thorough theoretical study is undertaken in the next Section 5.3.

5.3 RSM Theoretical Security Proof

Using the notations introduced in Section 5.1, the leakage of AES, when considering an attack in the last round, is defined as a function of $x \oplus y$, x being a byte of the ciphertext and y the corresponding byte of the state register of round 9. Now, when the RSM countermeasure is applied, the value y is actually replaced by $y \oplus m$, where m is one of the 16 mask values described in Section 5.2.1. $z = x \oplus y$ still describes the sensitive variable. In a view to introduce statistical notions, let's denote by capital letters (Z and M) the random variables and by small letters (z and m) their realizations. The leakage function thus has the form:

$$\mathcal{L}(Z, M) = \mathscr{L}(Z \oplus M) \ . \tag{5.4}$$

In this expression, Z and M are n-bit vectors, *i.e.* live in $\mathbb{F}_2^n$. The leakage function $\mathscr{L} : \mathbb{F}_2^n \to \mathbb{R}$ depends on the hardware. In a conservative perspective, $\mathscr{L}$ is assumed to be bijective. This choice is the most favorable to the adversary and is thus considered in the leakage estimation. Now, in practice, the leakage functions are not bijective. The canonical example is that of the Hamming weight leakage, where each bit of $Z \oplus M$ dissipate the same. Let us denote by x_i the component $i \in [\![1, n]\!]$ of $x \in \mathbb{F}_2^n$. The Hamming weight of x is expressed as $\mathsf{HW}[x] = \sum_{i=1}^{n} x_i$.

The leakage function of Equation (5.4) will now be studied formally, as per the guidelines presented in [168]. More precisely, the following metrics are considered:

- The mutual information between the $\mathcal{L}(Z, M)$ and the sensitive variable Z with $\mathscr{L}$ bijective as a *leakage metric*. This quantity is noted $\mathsf{I}[\mathcal{L}(Z, M); Z]$ and still referred to as MIM.

- *Security metrics* to quantify the easiness to actually turn a leakage into a successful attack. In this case, the focus will be on $\mathscr{L} = \mathsf{HW}$. First of all, the optimal correlation between $\mathsf{HW}[Z \oplus M]$ and Z is considered a metric. It is traditionally called the (first-order) correlation power analysis, or CPA [33]. But CPA can be defeated easily with only two mask values. Therefore it is important to consider higher-order CPA (HO-CPA) and notably the second-order CPA, also abridged 2O-CPA [187]. However, CPA and 2O-CPA exploit only the first two moments of the distribution of $\mathcal{L}(Z, M)$. Therefore, a second security metric is also used, namely the mutual information. It is known in the literature as MIA [18]. Security-wise, our goal is to minimize the first and second-order correlation coefficients and the MIA.

5.3.1 Information Theoretic Evaluation of the Countermeasure

The specificity of this study is to consider masks M that are not completely entropic. Thus, the probability $\mathsf{P}[M = m]$ depends on m. Our target is to restrict to a relevant subset of the masks uniformly, that is every mask is used with the same probability. Let's $J \subseteq \mathbb{F}_2^n$ be the set of masks actually used. Thus:

$$\mathsf{P}[M = m] = \begin{cases} 1/\mathsf{Card}[J] & \text{if } m \in J \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

This probability law is also written $M \sim \mathcal{U}(J)$. From an information theoretic point of view, the entropy of M can be characterized. By definition, $\mathsf{H}[M] = -\sum_{m \in J} \frac{1}{\mathsf{Card}[J]} \log_2 \frac{1}{\mathsf{Card}[J]} = \log_2 \mathsf{Card}[J]$ bit. The minimal number of masks is 1, which corresponds to the absence of countermeasure (take $M = 0$ in Equation (5.4)). At the opposite, when all the 2^n masks are used, the countermeasure is optimal.

Eventually, it is assumed that the adversary does not conduct a chosen message attack, *i.e.* $Z \sim \mathcal{U}(\mathbb{F}_2^n)$. Note that even if the adversary cannot actually choose the messages, he has nonetheless the possibility to discard some messages so as to artificially bias the side-channel attack. But a priori, the adversary does not know which plaintext Z to favor. A biased side-channel attack has been detailed in [102, 186]. However, this attack is adaptive and thus requires that a breach be already found. Nonetheless, in this context, the target is the protection of the secret at the early stages of the attack, namely the adversary still does not have any clue about the most likely hypotheses for the secret. This hypothesis is called the *non-adaptive known plain-text model* in [168].

Whatever the actual leakage function $\mathscr{L}$, $\mathsf{I}[\mathscr{L}(Z \oplus M); Z] = 0$ if $\mathsf{H}[M] = n$ bit (or equivalently, if $M \sim \mathcal{U}(\mathbb{F}_2^n)$). So with all the masks, the countermeasure is perfect.

If $\mathscr{L}$ is bijective (*e.g.* $\mathscr{L} = \mathsf{Id}$), then $\mathsf{I}[\mathscr{L}(Z \oplus M); Z] = n - \mathsf{H}[M]$. This results directly from the observation that:

- $\mathsf{H}[\mathscr{L}(Z \oplus M)] = \mathsf{H}[\mathscr{L}(Z)] = n$ bit, since $Z \sim \mathcal{U}(\mathbb{F}_2^n)$ and
- $\mathsf{H}[\mathscr{L}(Z \oplus M) \mid Z] = \mathsf{H}[M]$ bit because Z and M are independent.

It can be noticed that this quantity is independent of the exact J, provided $\mathsf{Card}[J]$ is fixed. This means that degrading the countermeasure (*i.e.* choosing $\mathsf{Card}[J] < 2^n$) introduces a vulnerability, while decreasing the cost.

Now, it can be checked to which extent this vulnerability is exploitable, considering a realistic leakage function. Specifically, it can be shown that if $\mathscr{L}$ is not injective, then the MIA metric $\mathsf{I}[\mathscr{L}(Z \oplus M); Z]$ depends on J. More precisely, when J as two (complementary) elements, then the MIA is independent of J. But when J is made up of strictly more than two masks, the MIA depends on J. For example, on $n = 8$ bits,

- $\mathsf{I}[\mathscr{L}(Z \oplus M); Z] = 1.42701$ bit if $J = \{\texttt{0x00}, \texttt{0x0f}, \texttt{0xf0}, \texttt{0xff}\}$, but
- $\mathsf{I}[\mathscr{L}(Z \oplus M); Z] = 0.73733$ bit if $J = \{\texttt{0x00}, \texttt{0x01}, \texttt{0xfe}, \texttt{0xff}\}$.

Thus, it is relevant to search for mask sets, at a constant budget (*i.e.* for a given $\mathsf{Card}[J]$), that minimize the mutual information $\mathsf{I}[\mathsf{HW}[Z \oplus M]; Z]$. Nonetheless, without a method, it is not obvious to conduct a reasoned search. Indeed, the default solution is to draw at random one mask set J and to compute $\mathsf{I}[\mathsf{HW}[Z \oplus M]; Z]$. It is immediate to see that such method will indeed provide solutions harder to attack using MIA than the others, but that will maybe fail in front of other less sophisticated attacks. Typically, J sets only constrained by their cardinality are likely to yield functions trivially attackable by CPA. Therefore the following method is proposed:

- First mask sets J that resist first and second order correlation attacks (*i.e.* CPA and 2O-CPA, the easiest attacks against single-masked countermeasures) are found. This is the topic of Section 5.3.2.
- Then, amongst these solutions, those minimizing the risk of MIA are selected. Section 5.3.3 specifically analyses this point (already quickly discussed in Section 5.3.2).

Another argument to focus primarily on CPA and 2O-CPA is that they require in practice less side-channel measurements to succeed the attack than MIA. Indeed, MIA, as well all other information theoretic-based attacks (*e.g.* template attacks [38] and stochastic attacks [152]), need to estimate conditional probability functions, which requires many measurements [62]. Also, from the certification standpoint, the common criteria [1] demands that the implemented countermeasures resist "state-of-the-art" attacks [49]. Now, CPA and 2O-CPA are much more studied in the information technology security evaluation facilities (ITSEFs) than information theoretic attacks.

5.3.2 Security against CPA and 2O-CPA

The average of the leakage function given in Equation (5.4) depends on $\mathscr{L} : \mathbb{F}_2^n \to \mathbb{R}$. As already mentioned, the Hamming weight ($\mathscr{L} = \mathsf{HW}$) is chosen, to conduct exact computations and match with realistic leakage functions observed in practice . Thus the average of leakage function, noted $\mathsf{E}\mathcal{L}(Z, M)$, is equal to:

$$\mathsf{E}\ \mathsf{HW}[Z \oplus M] = \frac{1}{\mathsf{Card}[J]} \sum_{m \in J} \frac{1}{2^n} \sum_{z \in \mathbb{F}_2^n} \mathsf{HW}[z \oplus m] = \frac{1}{\mathsf{Card}[J]} \sum_{m \in J} \frac{n}{2} = \frac{n}{2}\,. \tag{5.5}$$

Against HO-CPA of order $d \geqslant 1$, the most powerful adversary correlates her guesses about the sensitive variable with the optimal function [139] defined as:

$$\begin{aligned} f_{\mathrm{opt}}^{(d)}(z) &\doteq \mathsf{E}\left((\mathcal{L}(Z, M) - \mathsf{E}\mathcal{L}(Z, M))^d \mid Z = z \right) \\ &= \mathsf{E}\left(\left(\mathsf{HW}[Z \oplus M] - \frac{n}{2} \right)^d \mid Z = z \right) \\ &= \frac{1}{\mathsf{Card}[J]} \sum_{m \in J} \left(\frac{-1}{2} \sum_{i=1}^{n} (-1)^{(z \oplus m)_i} \right)^d , \end{aligned} \tag{5.6}$$

because if $b \in \{0,1\}$, then $b - \frac{1}{2} = -\frac{1}{2}(-1)^b$. Recall that the RSM countermeasure uses only one mask variable M and thus leaks at only one date (*i.e.* for a given timing sample). In this context, HO-CPA consists in studying the linear dependency between the d-th moments of the leakage classes and the optimal function $f_{\text{opt}}^{(d)}(z)$ of the sensitive variable z.

For the designer of the countermeasure, the objective is to make Equation (5.6) independent of z. There is always a solution that consists in choosing $J = \mathbb{F}_2^n$. Nonetheless, with $\mathsf{Card}[J] < 2^n$, the existence of solutions is a priori not trivial. In this case, if is impossible to find masks that keep $f_{\text{opt}}^{(d)}(z)$ (defined in Equation (5.6)) independent from z, the secondary goal is to minimize the correlation coefficient:

$$\rho_{\text{opt}}^{(d)} \doteq \frac{\mathsf{Var}\left(f_{\text{opt}}^{(d)}(Z)\right)}{\mathsf{Var}\left(\left(\mathcal{L}(Z,M) - \mathsf{E}\mathcal{L}(Z,M)\right)^d\right)} = \frac{\mathsf{Var}\left(\mathsf{E}\left(\left(\mathsf{HW}[Z \oplus M] - \frac{n}{2}\right)^d \mid Z\right)\right)}{\mathsf{Var}\left(\left(\mathsf{HW}[Z \oplus M] - \frac{n}{2}\right)^d\right)}. \quad (5.7)$$

In this equation, Var represents the variance, defined on a random variable X as $\mathsf{Var}(X) \doteq \mathsf{E}\left(X - \mathsf{E}X\right)^2$.

In the two next subsections 5.3.2 and 5.3.2, the analytical expression of Equation (5.7) is derived. Then these expressions are unified in subsection 5.3.2 by replacing the notion of subset J by an indicator function f. The sets of masks that completely allow to deny CPA and 2O-CPA are given exhaustively in subsection 5.3.2 for $n = 4$ and in subsection 5.3.2 for $n = 5$.

Resistance against First-Order Correlation Attacks

When $d = 1$, Equation (5.7) is equal to:

$$\rho_{\text{opt}}^{(1)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{\mathsf{Card}[J]}\sum_{m \in J}(-1)^{m_i}\right)^2. \quad (5.8)$$

This correlation $\rho_{\text{opt}}^{(1)}$ can be equal to zero if and only if (iff), for all $i \in [\![1,n]\!]$, $\mathsf{E}M_i = 1/2$. This means that the masks are balanced. It is possible to find such masks iff $\mathsf{Card}[J]$ is a multiple of two. A construction consists in building a set of masks by adding a new mask and its complement. Conversely, in a set containing an odd number of different masks, it is impossible to as many ones as zeros for any component. For instance, Table 5.4 illustrates how to generate balanced sets of masks in the case $n = 4$.

A trivial example consists in taking two masks, m and $\neg m$ (such as `0x00` and `0xff` on $n = 8$ bits). This is sufficient to thwart first-order attacks. At the opposite, without mask (J is equal to the singleton $\{$`0x00`$\}$) or with a single mask ($J = \{m\}$, whatever $m \in \mathbb{F}_2^n$), the correlation coefficient reaches its maximum (*i.e.* $+1$, because Equation (5.7) considers a correlation in absolute value).

Table 5.4: Mask sets J that make the masking countermeasure immune to first order CPA. The masks go by pair, symmetrically with the middle of the table.

	$\mathsf{Card}[J] = 2^4$	$\mathsf{Card}[J] = 2^3$	$\mathsf{Card}[J] = 2^2$	$\mathsf{Card}[J] = 2^1$
J	0000	0000	0000	0000
	0001			
	0010			
	0011	0011	0011	
	0100	0100		
	0101			
	0110			
	0111	0111		
	1000	1000		
	1001			
	1010			
	1011	1011		
	1100	1100	1100	
	1101			
	1110			
	1111	1111	1111	1111

Table 5.5: Security metrics for the masks sets of Table 5.4 and the singleton.

$\mathsf{Card}[J]$	$\mathsf{H}[M]$	$\rho^{(1)}_{\text{opt}}$	$\rho^{(2)}_{\text{opt}}$	$\mathsf{I}[\mathsf{HW}[Z \oplus M]; Z]$	$\mathsf{I}[Z \oplus M; Z]$
2^4	4	0	0	0	0
2^3	3	0	0.166667	0.15564	1
2^2	2	0	0.333333	1.15564	2
2^1	1	0	1	1.40564	3
2^0	0	1	1	2.03064	4

Resistance against Second-Order Correlation Attacks

When $d = 2$, Equation (5.7) is equal to:

$$\rho^{(2)}_{\text{opt}} = \frac{1}{n(n-1)} \left(\frac{1}{\mathsf{Card}[J]^2} \sum_{(m,m') \in J^2} \left(\sum_{i=1}^{n} (-1)^{(m \oplus m')_i} \right)^2 - n \right). \tag{5.9}$$

As an illustration, Table 5.5 shows the optimal correlation coefficients of order 1 and 2 for the masks sets of Table 5.4 ($n = 4$ bit). The last column has been added, for $\mathscr{L} = \mathsf{Id}$ and the last row corresponds to a constant masking (unprotected implementation), which serves as a reference.

Expression of $\rho^{(1,2)}_{\text{opt}}$ as a Function of an Indicator f

The expressions of $\rho^{(1)}_{\text{opt}}$ and $\rho^{(2)}_{\text{opt}}$ (altogether referred to as $\rho^{(1,2)}_{\text{opt}}$) defined in Equation (5.8) and (5.9) lay a mathematical ground to search for suitable J. Nonetheless, these equations remain at the set-theory level. To simplify the problem, let's introduce the Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$, defined as: $\forall m \in \mathbb{F}_2^n, f(m) = 1 \Leftrightarrow m \in J$. Then, "$\sum_{m \in J}$" can simply be replaced by "$\sum_{m \in \mathbb{F}_2^n} f(m)$" in the previously established equations .

The Fourier transform $\hat{f} : \mathbb{F}_2^n \to \mathbb{Z}$ of the Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is defined as $\forall a \in \mathbb{F}_2^n, \hat{f}(a) \doteq \sum_{m \in \mathbb{F}_2^n} f(m)(-1)^{a \cdot m}$. It allows for instance to write $\mathsf{Card}[J] = \sum_{m \in J} 1 = \sum_{m \in \mathbb{F}_2^n} f(m) = \hat{f}(0)$. Recall $\mathsf{Card}[J] \in [\![1, 2^n]\!]$, hence $\hat{f}(0) > 0$.

Then Equation (5.8) rewrites:

$$\rho^{(1)}_{\text{opt}} = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{\hat{f}(e_i)}{\hat{f}(0)} \right)^2, \tag{5.10}$$

where e_i are the canonical basis vectors $(0, \cdots, 0, 1, 0, \cdots, 0)$, the unique 1 laying at position i.

Also, Equation (5.9) rewrites:

$$\begin{aligned}\rho_{\text{opt}}^{(2)} &= \frac{1}{n(n-1)} \sum_{(i,i')\in[\![1,n]\!]^2} \left(\left(\frac{\hat{f}(e_i \oplus e_{i'})}{\hat{f}(0)} \right)^2 - n \right) \\ &= \frac{1}{n(n-1)} \sum_{\substack{(i,i')\in[\![1,n]\!]^2 \\ i\neq i'}} \left(\frac{\hat{f}(e_i \oplus e_{i'})}{\hat{f}(0)} \right)^2 . \end{aligned} \tag{5.11}$$

Thus, the RSM countermeasure resists:

1. first-order attacks iff $\forall a$, $\mathsf{HW}[a] = 1 \Rightarrow \hat{f}(a) = 0$;
2. first and second-order attacks iff $\forall a$, $1 \leqslant \mathsf{HW}[a] \leqslant 2 \Rightarrow \hat{f}(a) = 0$.

As a sanity check, it can be verified that these properties hold when all the 2^n masks are used, *i.e.* when f is constant (and furthermore equal to 1). Indeed, in this case, $\hat{f}(a) = \sum_m f(m)(-1)^{a\cdot m} = \sum_m (-1)^{a\cdot m} = 2^n \delta(a)$, where δ is the Kronecker symbol.

Now, it can be noticed that for Boolean functions, the notions of Fourier and Walsh transforms are very alike. Indeed:

$$\forall a \neq 0, \hat{f}(a) = \sum_m f(a)(-1)^{a\cdot m} = \sum_m (-1)^{a\cdot m} \tfrac{1}{2} \left(1 - (-1)^{f(m)}\right) = -\tfrac{1}{2}\widehat{(-1)^f}(a)\,.$$

Therefore, the previous conditions are equivalent to the following statement: the countermeasure resists $d \in \{1, 2\}$ order CPA iff $\forall a$, $\mathsf{HW}[a] \leqslant d \Rightarrow \widehat{(-1)^f}(a) = 0$.

We insist that this characterization is not equivalent to saying that f is d-resilient (defined in [36, page 45]). Indeed, a resilient function is balanced, which is explicitly not the case of f. Therefore, in the sequel a new kind of Boolean functions is studied, that have everything in common with resilient functions but the balanceness of the plain function. The corollary is that, to the authors' best knowledge, no known construction method exists for this type of functions. Nonetheless, it is interesting to get an intuition about what characterizes a good resilient function. In [36, §7.1, page 95], it is explained that the highest degree of resiliency of a $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is $n - 2$. This maximum is reached by affine functions (functions of unitary algebraic degree). Nonetheless, in our case, affine functions are not the best choice, because they are balanced. This means that the cardinality of their support (*i.e.* $\mathsf{Card}[J]$) is 2^{n-1}, which is large. Therefore, non-affine functions f of algebraic degree strictly greater than one (noted $d_{\text{alg}}^{\circ}(f) > 1$) will be considered, whenever possible.

Functions $f : \mathbb{F}_2^4 \to \mathbb{F}_2$ that Cancel $\rho_{\text{opt}}^{(1,2)}$

For $n = 4$, all the sets J can be tested. The table 5.6 reports all the functions f that cancel $\rho_{\text{opt}}^{(1)}$ and $\rho_{\text{opt}}^{(2)}$. In this table, the truth-table of f, given in the first column, is

Table 5.6: All the functions $f : \mathbb{F}_2^4 \to \mathbb{F}_2$ that cancel $\rho_{\text{opt}}^{(1,2)}$.

f	$\mathsf{HW}[f]$	$\mathsf{H}[M]$	$\rho_{\text{opt}}^{(1)}$	$\rho_{\text{opt}}^{(2)}$	$\mathsf{I}[\mathsf{HW}[Z \oplus M]; Z]$	$\mathsf{I}[Z \oplus M; Z]$	$d_{\text{alg}}^{\circ}(f)$
`0x3cc3`	8	3	0	0	0.219361	1	1
`0x5aa5`	8	3	0	0	0.219361	1	1
`0x6699`	8	3	0	0	0.219361	1	1
`0x6969`	8	3	0	0	0.219361	1	1
`0x6996`	8	3	0	0	1	1	1
`0x9669`	8	3	0	0	1	1	1
`0x9696`	8	3	0	0	0.219361	1	1
`0x9966`	8	3	0	0	0.219361	1	1
`0xa55a`	8	3	0	0	0.219361	1	1
`0xc33c`	8	3	0	0	0.219361	1	1
`0xffff`	16	4	0	0	0	0	0

encoded in hexadecimal. Let's note $\mathsf{HW}[f]$ the number of ones in the truth-table and recall that $\mathsf{HW}[f] = \mathsf{Card}[J]$. Columns 4, 5 and 6 are security metrics, whereas column 7 is the leakage metric (MIM). There are non-trivial solutions only for $\mathsf{Card}[J]$ equal to half of the complete mask set cardinal. The MIA (column 6) shows two values: 0.219361 and 1 bit. Those values shall be contrasted with the MIA:

- without countermeasure ($\mathsf{Card}[J] = 1$): MIA = 2.19819 bit and
- with two complementary masks ($\mathsf{Card}[J] = 2$, which thwarts CPA but not 2O-CPA): MIA = 1.1981 bit.

Thus the countermeasure resists better correlation and information theoretic attacks, at the expense of more masks. Indeed, apart from $f = 1$, all the solutions are affine ($d_{\text{alg}}^{\circ}(f) = 1$) and thus have a Hamming weight of $2^{n-1} = 8 \gg 2$.

In this table, some functions belong to equivalent classes. Namely, two of them can be identified:

- the permutations of the bits (because the summations over i in Equation (5.10) or i, i' in Equation (5.11) is invariant in any change of the bits order) and
- the complementation. Indeed, $\widehat{\neg f}(a) = \sum_{m \in \mathbb{F}_2^n} \neg f(m)(-1)^{a \cdot m} = \sum_{m \in \mathbb{F}_2^n} (1 - f(m))(-1)^{a \cdot m} = 2^n \delta(a) - \hat{f}(a)$. Now, in Equation (5.10) and (5.11), $a \neq 0$ and $\hat{f}$ is involved squared. Thus $\rho_{\text{opt}}^{(1,2)}(\neg f) = \rho_{\text{opt}}^{(1,2)}(f)$.

The same can be said for the mutual information. This lemma is useful:

Lemma 5.3.1 *Let A and B be two random variables and ϕ a bijection; then* $\mathsf{I}[A;\phi(B)] = \mathsf{I}[A;B]$.

This equality is obtained simply by writing the definition of the mutual information as a function of the probabilities, and by doing a variable change. Then:

- Let us call σ a permutation of $[\![1,n]\!]$. This function is a bijection and its inverse is also a permutation. The Hamming weight is invariant if σ is applied on its input (*i.e.* $\mathsf{HW} = \mathsf{HW}\circ\sigma$). Hence $\mathsf{HW}[Z\oplus\sigma(M)] = \mathsf{HW}[\sigma^{-1}(Z\oplus\sigma(M))] = \mathsf{HW}[\sigma^{-1}(Z)\oplus M]$ (because σ is furthermore linear with respect to the addition). Let's note $Z' = \sigma^{-1}(Z)$, a random variable that is also uniform. Thus, $\mathsf{I}[\mathsf{HW}[Z\oplus\sigma(M)];Z] = \mathsf{I}[\mathsf{HW}[Z'\oplus M];\sigma(Z')]$. By considering $\phi = \sigma$, we prove that $\mathsf{I}[\mathsf{HW}[Z\oplus\sigma(M)];Z] = \mathsf{I}[\mathsf{HW}[Z'\oplus M];Z'] = \mathsf{I}[\mathsf{HW}[Z\oplus M];Z]$, because Z and Z' have the same probability density function.

- Regarding the complementation, it is straightforward to note that $\mathsf{HW}[Z\oplus\neg M] = \mathsf{HW}[\neg(Z\oplus M)] = n - \mathsf{HW}[Z\oplus M]$. By considering $\phi : x \mapsto n - x$, the invariance of the mutual information is also attained by the complementation of the mask.

So, there are eventually only three classes of functions listed in Table 5.6, modulo the two aforementioned equivalence classes. They are summarized below:

1. $f(x_1,x_2,x_3,x_4) = \bigoplus_{\substack{i\in I\subseteq[\![1,4]\!]\\ \mathsf{Card}[I]=3}} x_i$, (*aka* `0x3cc3`, `0x5aa5`, `0x6699`, `0x6969`) or complemented (*aka* `0x9696`, `0x9966`, `0xa55a`, `0xc33c`); According to the criteria stated at the end of Section 5.3.1, those functions are the best solutions for $n = 4$.

2. $f(x_1,x_2,x_3,x_4) = \bigoplus_{i=1}^{4} x_i$ (*aka* `0x6996`) or $f(x_1,x_2,x_3,x_4) = 1 \oplus \bigoplus_{i=1}^{4} x_i$ (*aka* `0x9669`), that has no advantage of the previous solutions;

3. the constant function $f = 1$ (*aka* `0xffff`).

To resist first-order attacks, the masks set can be partitioned in two complementary sets; this means that there exists $\tilde{J}$, a subset of J, such that: $J = \tilde{J} \cup \neg\tilde{J}$, where $\neg\tilde{J} \doteq \{\neg m, m \in J\}$. Incidentally, it is noteworthy that this is not a mandatory property. Typically, this property is not verified any longer at order 2. For instance, in the solution $f =$ `0x3cc3`, $\texttt{0x0} \in J$ but $\neg\texttt{0x0} = \texttt{0xf} \notin J$.

In conclusion, when $n = 4$ and the designer cannot afford using all the 16 masks, then with 8 masks, the RSM countermeasure is able to resist CPA, 2O-CPA and leak the minimal value of 0.219361 bit (about ten times less than the unprotected implementation, for which the MIA is 2.19819 bit).

Table 5.7: Summary of the security metrics of $f : \mathbb{F}_2^5 \to \mathbb{F}_2$ that cancel $\rho_{\text{opt}}^{(1,2)}$.

Nb. classes	$\mathsf{HW}[f]$	$\mathsf{H}[M]$	$\rho_{\text{opt}}^{(1)}$	$\rho_{\text{opt}}^{(2)}$	$\mathsf{I}[\mathsf{HW}[Z \oplus M]; Z]$	$\mathsf{I}[Z \oplus M; Z]$	$d_{\text{alg}}^{\circ}(f)$
3	8	3	0	0	0.32319	2	2
4	12	3.58496	0	0	0.18595	1.41504	3
2	16	4	0	0	0.08973	1	1
2	16	4	0	0	0.08973	1	2
4	16	4	0	0	0.12864	1	2
2	16	4	0	0	0.16755	1	1
4	16	4	0	0	0.26855	1	2
6	16	4	0	0	0.32495	1	2
1	16	4	0	0	1	1	1
4	20	4.32193	0	0	0.07349	0.67807	3
3	24	4.58496	0	0	0.04300	0.41504	2
1	32	5	0	0	0	0	0

Functions $f : \mathbb{F}_2^5 \to \mathbb{F}_2$ that Cancel $\rho_{\text{opt}}^{(1,2)}$

For $n = 5$, all the subsets J of $\mathbb{F}_2^5$ (2^{32} of them, it is the maximum achievable on a personal computer, as precised in [36, page 6]) have been tested. There are 1057 functions that cancel $\rho_{\text{opt}}^{(1,2)}$. The lowest value for $\mathsf{HW}[f]$ is 8. There are 60 functions of weight 8, but only three classes modulo the invariants. The functions, sorted regarding their properties, are shown in Table 5.7. As opposed to the case $n = 4$, there are non-affine solutions. In this table, only the number of equivalent classes is given. For a list of all functions, the reader is referred to [127].

The greater $\mathsf{H}[M]$, the smaller the mutual information with $\mathscr{L} = \mathsf{HW}$ in general, but for some remarkable solutions (*e.g.* the one $\mathsf{MIA} = \mathsf{I}[\mathsf{HW}[Z \oplus M]; Z] = 1$ of algebraic degree 1 for $\mathsf{HW}[f] = 16$). Also, it is worth noting that for a given budget (*e.g.* 16 masks) and security requirement (resistance against CPA and 2O-CPA), some solutions are better than the others against MIA. Indeed, the leaked information in Hamming weight model spans from 0.0897338 bit to 1 bit.

5.3.3 Exploring More Solutions Using SAT-Solvers

In order to explore problems of greater complexity, SAT-solver are indicated tools. f is modeled as a set of 2^n Boolean unknowns. The problem consists in finding f such that $\forall a, 1 \leqslant \mathsf{HW}[a] \leqslant 2, \hat{f}(a) = 0$, for a given $\mathsf{Card}[J] = \hat{f}(0)$. A SAT-solver either proves that there is no solution, or that a solution exists and provides (at least) one. However, it may not terminate on certain instances of large exploration space (which

has not been an issue in this work). In this section, the issue of feeding our problem into a SAT-solver is first explained, then, the SAT-solver is used in the case of AES (with $n = 8$). The idea is to look for low $\mathsf{Card}[J]$ solutions and for a given $\mathsf{Card}[J]$, the solutions of minimal MIA.

Mapping of the Problem into a SAT-Solver

Knowing that $\mathsf{Card}[J] = \hat{f}(0)$, the problem $\rho_{\text{opt}}^{(1,2)}(f) = 0$ rewrites:

$$\begin{aligned} &\forall a, 1 \leqslant \mathsf{HW}[a] \leqslant 2, \quad \sum_x f(x)(-1)^{a\cdot x} = 0 \quad \Leftrightarrow \\ &\forall a, 1 \leqslant \mathsf{HW}[a] \leqslant 2, \quad \sum_x f(x) \wedge (a \cdot x) = \frac{1}{2}\sum_x f(x) = \frac{1}{2}\mathsf{Card}[J] \ . \end{aligned} \tag{5.12}$$

A SAT-solver verifies the validity of clauses, usually expressed in conjunctive normal form (CNF). It is known that cardinality constraints can be formulated compactly thanks to Boolean clauses. More precisely, any condition "$\leqslant k(x_1, \cdots, x_n)$", for $0 \leqslant k \leqslant n$, can be expressed in terms of CNF clauses [159]. It can be noted that:

$$\mathsf{HW}[x] \leqslant k \quad \Leftrightarrow \quad n - \mathsf{HW}[\neg x] \leqslant k \quad \Leftrightarrow \quad \mathsf{HW}[\neg x] \geqslant n - k\,.$$

Hence, satisfying $\geqslant k(x_1, \cdots, x_n)$ is equivalent to satisfying $\leqslant n - k(\neg x_1, \cdots, \neg x_n)$. Thus, testing the equality of a Hamming to $\frac{1}{2}\mathsf{Card}[J]$ can be achieved by the conjunction of two clauses: $\leqslant \frac{1}{2}\mathsf{Card}[J](x_1, \cdots, x_n)$ and $\leqslant n - \frac{1}{2}\mathsf{Card}[J](\neg x_1, \cdots, \neg x_n)$.

The $n = 8$, the number of useful literals, $\{f(x), x \in \mathbb{F}_2^n\}$, is 2^8. However, the constraints $\mathsf{Card}[J] = \hat{f}(0)$ and $\rho_{\text{opt}}^{(1,2)}(f) = 0$ (see Equation (5.12)) introduce 1,105,664 auxiliary variables and translate into 2,219,646 clauses, irrespective of $\mathsf{Card}[J] \in \mathbb{N}^*$.

Existence of Low Hamming Weight Solutions for $n = 8$

The software `cryptominisat` [165, 166] is used to search for solutions. The problem is tested for all the $\mathsf{Card}[J]$ from 2 to 2^n, by steps of 2, as independent problems. Each problem requires a few hours to be solved. Impressively low Hamming weight solutions are found. The table 5.8 represents some of them. There are solutions only for $\mathsf{Card}[J] \in \{4 \times \kappa, \kappa \in [\![3, 61]\!] \cup \{64\}\}$. Also, the mutual information with a Hamming weight leakage as a function of $\mathsf{H}[M]$ is plotted in Figure 5.9. These values are low when compared to:

- MIA $= 2.5442$ bit without masking ($\mathsf{Card}[J] = 1$) and
- MIA $= 1.8176$ bit with a mask that takes two complementary values ($\mathsf{Card}[J] = 2$).

Those MIA figures concern countermeasures that do not protect against 2O-DPA. The table 5.8 basically indicates that the margin gain in MIA resistance decreases when the cost of the countermeasures, proportional to $\mathsf{HW}[f]$, increases.

Table 5.8: Metrics for one $f : \mathbb{F}_2^8 \to \mathbb{F}_2$ that cancel $\rho_{\text{opt}}^{(1,2)}$, found by a SAT-solver.

$\mathsf{HW}[f]$	$\mathsf{H}[M]$	$\rho_{\text{opt}}^{(1)}$	$\rho_{\text{opt}}^{(2)}$	$\mathsf{I}[\mathsf{HW}[Z \oplus M]; Z]$	$\mathsf{I}[Z \oplus M; Z]$	$d_{\text{alg}}^{\circ}(f)$
12	3.58496	0	0	0.387582	4.41504	6
16	4	0	0	0.219567	4	5
20	4.32193	0	0	0.228925	3.67807	6
24	4.58496	0	0	0.235559	3.41504	5
28	4.80735	0	0	0.144147	3.19265	6
32	5	0	0	0.135458	3	5
36	5.16993	0	0	0.090575	2.83007	6
40	5.32193	0	0	0.078709	2.67807	5
44	5.45943	0	0	0.067960	2.54057	6
48	5.58496	0	0	0.060515	2.41504	5
52	5.70044	0	0	0.092676	2.29956	6
56	5.80735	0	0	0.054936	2.19265	5
60	5.90689	0	0	0.049069	2.09311	6
64	6	0	0	0.035394	2	2
68	6.08746	0	0	0.042374	1.91254	6
72	6.16993	0	0	0.036133	1.83007	5
76	6.24793	0	0	0.034194	1.75207	6
80	6.32193	0	0	0.031568	1.67807	5
84	6.39232	0	0	0.030072	1.60768	6
88	6.45943	0	0	0.026941	1.54057	5
92	6.52356	0	0	0.027042	1.47644	6
96	6.58496	0	0	0.022992	1.41504	5
100	6.64386	0	0	0.024316	1.35614	6
104	6.70044	0	0	0.022257	1.29956	5
108	6.75489	0	0	0.021458	1.24511	6
112	6.80735	0	0	0.019972	1.19265	4
116	6.85798	0	0	0.020481	1.14202	6
120	6.90689	0	0	0.018051	1.09311	5
124	6.9542	0	0	0.018397	1.0458	6
128	7	0	0	0.015095	1	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮

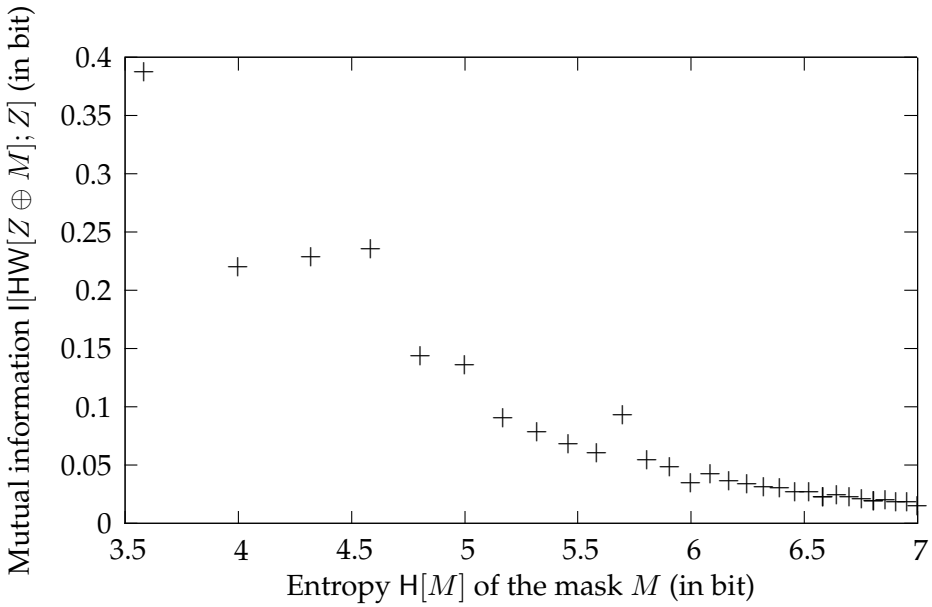


Figure 5.9: Mutual information of the leakage in Hamming weight with the sensitive variable Z, for one solution that cancels $\rho_{\text{opt}}^{(1,2)}$ found by the SAT-solver.

Exploration of Solutions for $n = 8$ and a Fixed Card$[J]$

There are nonequivalent solutions for a same Card$[J]$. Various seeds of the SAT-solver are needed to discover these solutions. All the solutions found by the SAT-solver for Card$[J] = 12$ have the same MIA value: 0.387582 bit. It can also be shown that for Card$[J] = 16$, various MIA values exist. The SAT-solver has notably came across, from best to worst: 0.181675, 0.213996, 0.215616, 0.216782, 0.219567, 0.220733, 0.246318, 0.249556, 0.251888, 0.253508, 0.254674, 0.257459, 0.388196, 0.434113, 1.074880 and 1.074950.

It is noteworthy that with the SAT-solver, some solutions are found, however they cannot be easily classified. Thus it is unsure the best one has actually been found. Nonetheless, it is already of great practical importance to exhibit some solutions.

As can be observed, those results are slightly different than those obtained via actual experiments in Section 5.2.2. Namely the latter values were smaller than those displayed here. This can be explain by the presence of noise during the side-channel measurements, which degrades the accuracy of the MIA. Indeed, given a greater number of observations, experimental values should approach the theoretical ones. Then again, this difference shows that even when a theoretical leakage exists, it not necessarily a simple matter to actually exploit it in order to perform an attack.

*

To conclude, this study shows that the masks set of the RSM countermeasure can be selected in order to ensure robustness to CPA and 2O-CPA and minimize the infor-

mation leakage. The criteria for masks selection has been formalized as a condition on the Walsh transform of an indicator function and used heuristically in a SAT-solver. The best solutions for $n = 4$ and $n = 5$ are exhibited and the existence of varied values of the mutual information for some masks cardinality for $n = 8$ is proven, thanks to the SAT-solver. It is notably shown that amongst the masks subsets that allow a resistance at orders 1 and 2 against CPA, some are less sensitive to MIA than others, especially for $\mathsf{Card}[J] = 16$. Obviously, at first sight, it can seem very audacious to mask an eight bit sensitive data with only four bits of mask. But it is indeed possible due to the high non-injectivity of the HW function, that maps 256 values into only 9. Nonetheless, in order to achieve an even greater level of security, methods to remove or at least lessen the information leakage could be considered.

5.4 RSM: Optimisations

As stated in Section 5.2.1 and 5.3, although RSM proves to be robust against first- and second-order CPA, the security evaluation shows a possible information leakage due to the difference in entropy between the inputs and the masks. Thus, in order to ensure an optimal security and remove all possible leakage, three new versions of this countermeasure are introduced: a time-security, a surface-security trade-off and one using partial reconfiguration.

In all cases, the idea is to regularly generate new pools of masks and customized S-Boxes, in order to approach the full 8-bit masking entropy.

5.4.1 Surface-security trade-off

The first idea is to compute new sets of constants and S-Boxes while using the old ones and store them in additional memory, therefore not altering the performances. For this purpose, it is mandatory to use RAM for storing both S-Boxes and constants and the required size is doubled. Some logic operators also need to be added:

- A 128-bit random number generator (RNG), to produce the 16 new base masks m_{0-15}.
- A barrel shifter.
- A *MC* and *InvMC* operator.
- A *SR* and *InvSR* operator (no actual logic gates).
- One basic AES S-Box and its inverse.
- 4/8 8-input *Xor* gates.

This way, each time a given set of S-Boxes and constants is used for the countermeasure, another is being computed and stored in the additional memory. With the additional barrel shifter, *MC*, *SR* and their inverse operators, the 5 sets of 16 constants can easily be generated from a 128-bit random number in 80 clock cycles (one per constant).

As for the S-Boxes, if the target device includes Dual-Port RAM, one S-Box can be used for two parallel computations, hence 8 *Xor* gates instead of 4. As a matter of fact, they can be created within $256 \times 8 = 2048$ or $256 \times 16 = 4096$ clock cycles, for respectively 8 and 4 *Xor* gates, considering the basic S-Boxes and their inverse are generated in parallel.

Eventually the area consumption of this countermeasure would be roughly twice the regular one (for both logical gates and memory blocks), for the same performances, but with a masking entropy of almost 8 bits.

Moreover, considering that a straightforward implementation of AES 256 takes 14 clock cycles to process, it follows that a potential adversary would be able to take at most 150 to 300 side-channel measurements for a given S-Box/constants set, which should hardly be sufficient to perform a meaningful analysis.

5.4.2 Time-security trade-off

The second idea is to pause the encryptions every once in a while, in order to compute a new sets of S-Boxes and constants, using the existing AES operators, as well as an additional basic AES S-Box/InvS-Box, 4/8 8-input *Xor* gates and a 128-bit RNG. As stated in the previous section, it would take between $4096 + 80 = 5076$ and 2538 clock cycles, that is respectively about $50\ \mu s$ or $25\ \mu s$ at 100 MHz.

5.4.3 Using partial reconfiguration

Finally it should be possible to take advantage of recent FPGA technologies, like the Xilinx Virtex5 family, which allows partial reconfiguration of the device. As a matter of fact, the mask recomputation could be performed by an embedded processor, which would regularly reconfigure the RAM blocks containing both constants and S-Boxes. This way, the countermeasure size and performances would be almost unchanged, except for the reconfiguration times, during which the computations must be paused. We insist that knowing the masks is of no use for an adversary (since only the direction $\mathcal{D}$ is sensitive): therefore, the mask fresh process needs not be protected against SCA.

5.5 Conclusion

As discussed in Section 3.2.1, masking is maybe the most commonly used countermeasure against first-order SCAs, specially in software. However when the target device

is an FPGA, implementing a fully-fledged masking on AES induces significant overheads in terms of resources and performances. Moreover, classical first-order masking schemes have been proven sensitive to VPA as well as high-order SCAs. Several papers proposed high-order masking techniques for software, which show implementation costs not fitted for complex industrial applications.

In this context, we presented RSM, a new type of masking scheme for AES, based on the idea of using a set of 16 predefined masks rather than fully random ones. Taking advantage of this feature, RSM S-Boxes are build with intrinsic masking, allowing them to remain the size of regular unprotected ones, whereas in other schemes, masking the S-Boxes is usually the most area and/or time consuming process. Moreover, RSM is designed with a mono-path structure, which induces both a surface reduction and an increased robustness against variance- and mutual information-based attacks, with regards to a state of the art architecture.

To validate the concept, RSM was implemented on an Altera StratixII FPGA. Results clearly show a significant decrease in resource consumption, as well as similar performances, compared to the state of the art. Both simulations and real-life attacks were performed and empirically demonstrated the robustness of this countermeasure against DPA, CPA and VPA.

Moreover, a thorough theoretical evaluation is given, proving that a full 8-bit mask entropy is not necessary to resist first-order SCAs and that, provided the set of masks is properly chosen, RSM is secure against first- and second-order zero-offset CPA. As a matter of fact, Section 5.3 showed that a set of only 12 masks is sufficient to thwart those attacks. Moreover it is shown that depending on the mask set, the information leakage varies, allowing a designer to chose mask set inducing minimal leakage.

Nevertheless, as most masking schemes, RSM does not possess any fault detection mechanism, hence the addition of a classical technique (for instance encryption-decryption) may be required to attain a high level of security against all types of SCAs. Obviously this would directly impact either the performances, or the area of the global countermeasure, increasing either one by a factor two. However, even when divided by two, the performances of RSM are similar, if not better than the relatively low-cost DPLs schemes while its security level is higher than most.

Chapter 6

Experimental Evaluation of Countermeasures for Asymmetrical Algorithms

Existing countermeasures for asymmetrical algorithms already offer a robustness level and implementation cost suitable for industrial applications. It is therefore not necessary to design new ones, but rather to carefully chose the most appropriate association of countermeasures, depending on the target device, in order to provide protection against all kinds of SCAs, while keeping acceptable speed and area consumption.

6.1 Simple ECC design

As stated before, although a countermeasure is sound in theory and even when embedded in an ASIC (e.g. WDDL), its implementation on FPGA could by itself induce newly found vulnerability or unexpected resource consumption, due to the intrinsic specificity of such device. The pragmatic goal of this Section is therefore not to develop an optimized ECC coprocessor in terms of either performances or area consumption, but rather to rapidly allow, on one hand the implementation on FPGA of several classical countermeasures in order to properly evaluate their actual overhead and robustness. And on the other hand, the possibility to perform basic SCAs and assess their actual cost and feasibility on ad-hoc FPGA implementations. The *doubling attack* is specially studied, as it is said to be efficient against usual SPA protections, as well as several low-cost DPA countermeasures, like a variant of the message blinding (see Section 3.3.2 and the randomization of the secret described in Section 3.3.2.

First of all, a simple ECC architecture of the *NIST p192* curve was implemented, based on the modular operators described in [60]. As in Section 4.2.2, the target device is an Altera StratixII FPGA, soldered on a SASEBO-B board.

This section will describe the main components of the design which are the modular operators (especially modular multiplication and division) used to compute the two parts of the ECC scalar multiplication: **point addition** and **point doubling**. The input point of these two operations can be represented in several coordinates systems (affine, Jacobian, ...). This architecture uses the basic affine coordinates, which are not optimal in terms of performances, but sufficient to fulfill our goal and experimentally assess that no additional cost or vulnerability results from a FPGA implementation.

6.1.1 Point doubling

Let $Q = (x_3, y_3)$ be the double of $A = (x_1, y_1)$, in affine coordinates Q is computed as follows:

$s = 3{x_1}^2 + a/2y_1$
$x_3 = s^2 - 2x_1$
$y_3 = s(x_1 - x_3) - y_1$

Let $\mathcal{O}$ be the point at infinity, if $A = \mathcal{O}$ then $Q = \mathcal{O}$.
If $A = -B$ i.e. $A = (x_2, -y_2)$ then $A + B = O$.

6.1.2 Point addition

Let $Q = (x_3, y_3)$ be the sum of $A = (x_1, y_1)$ and $B = (x_2, y_2)$, Q is computed as follows:

$s = y_2 - y_1/x_2 - x_1$
$x_3 = s^2 - x_1 - x_2$
$y_3 = s(x_1 - x_3) - y_1$

If $A = \mathcal{O}$ then $Q = B$.
If $y_1 = 0$ then $Q = 2A = \mathcal{O}$.

6.1.3 Modular operators

Modular addition, subtraction, doubling

As shown in Figure 6.1, the modular addition either computes $A + B$ or $A + B - P$ (where A,B are the inputs and P the ECC prime number) depending on the output carry of $A + B$. In the same way, subtraction either computes $A - B$ or $A - B + P$ and doubling either $2A$ or $2A - P$, as respectively illustrated in Figure 6.2 and Figure 6.3.

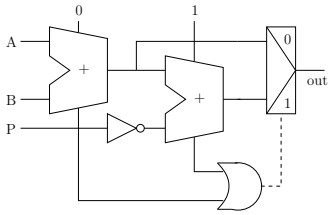


Figure 6.1: Modular Addition.

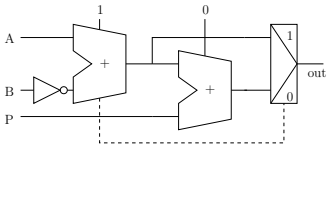


Figure 6.2: Modular Subtraction.

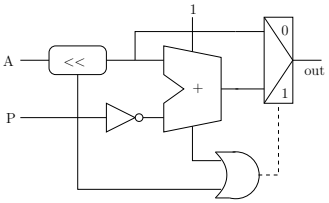


Figure 6.3: Modular Doubling.

Modular Multiplication

Modular multiplications are performed using Algorithm 16, which is designed to have a time complexity of exactly n clock cycles, where n is the bit length of P. Its datapath, described in Figure 6.4 is mainly composed of a n-bit register, a modular adder, a doubling unit and a simple $n\ downto\ 0$ counter making office of controller.

Algorithm 16 Modular Multiplication Algorithm.

Input: $P\ and\ A, B \in (0, P-1)$
Output: $A * B\ mod\ P$
$T = 0$
for $i = 0\ to\ n-1$ **do**
  $T \leftarrow 2T + A\dot{B}_{k-1-i}$
  $T \leftarrow T\ mod\ P$
Return T

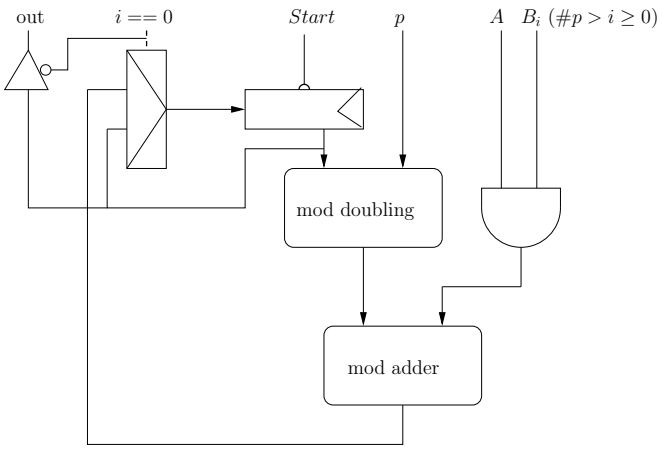


Figure 6.4: Modular Multiplication.

Modular division/inversion

Modular divisions are performed using the "Binary Inversion Algorithm" described in Algorithm 17. It can compute either an inversion or a division, depending on the value of the $d/\bar{i}$ bit.

Algorithm 17 Binary Inversion Algorithm.

Input: $P\ and\ A, B \in (0, P-1)$
Output: $B \div A\ mod\ P$
$u = a, v = p, x1 = B, x2 = 0$
while $u = 1$ and $v = 1$ **do**
  while u even **do**
    $u \leftarrow u/2$
    if $x1$ even **then**
      $x1 \leftarrow x1/2$
    else
      $x1 \leftarrow (x1 + P)/2$
  while v even **do**
    $v \leftarrow v/2$
    if $x2$ even **then**
      $x2 \leftarrow x2/2$
    else
      $x2 \leftarrow (x2 + P)/2$
  if $u \geq v$ **then**
    $u \leftarrow u - v$, $x1 \leftarrow x1 - x2$
  else
    $v \leftarrow v - u$, $x2 \leftarrow x2 - x1$
if $u = 1$ **then**
  Return $x1$, **else** Return $x2$

At every iteration either u or v is reduced by at least one bit length. It follows that the total number of iterations of the main *while* loop is at most $2n$, where n is the bit length of P.

On one hand, the loop condition of the main *while* loop is implemented by two comparators with '1', as shown in Figure 6.5 and those of the inner *while* loops by an *AND* gate between u_0 and v_0. On the other hand, the *Step1*u/v blocks depicted in Figure 6.6 perform the operations within the two inner *while* loops and the *Step2* block of Figure 6.7 those within the main loop.

6.1.4 Unprotected Datapath

The first implementation, depicted in Figure 6.8, is a simple datapath composed of one instance of each modular operators, two operand registers with their respective

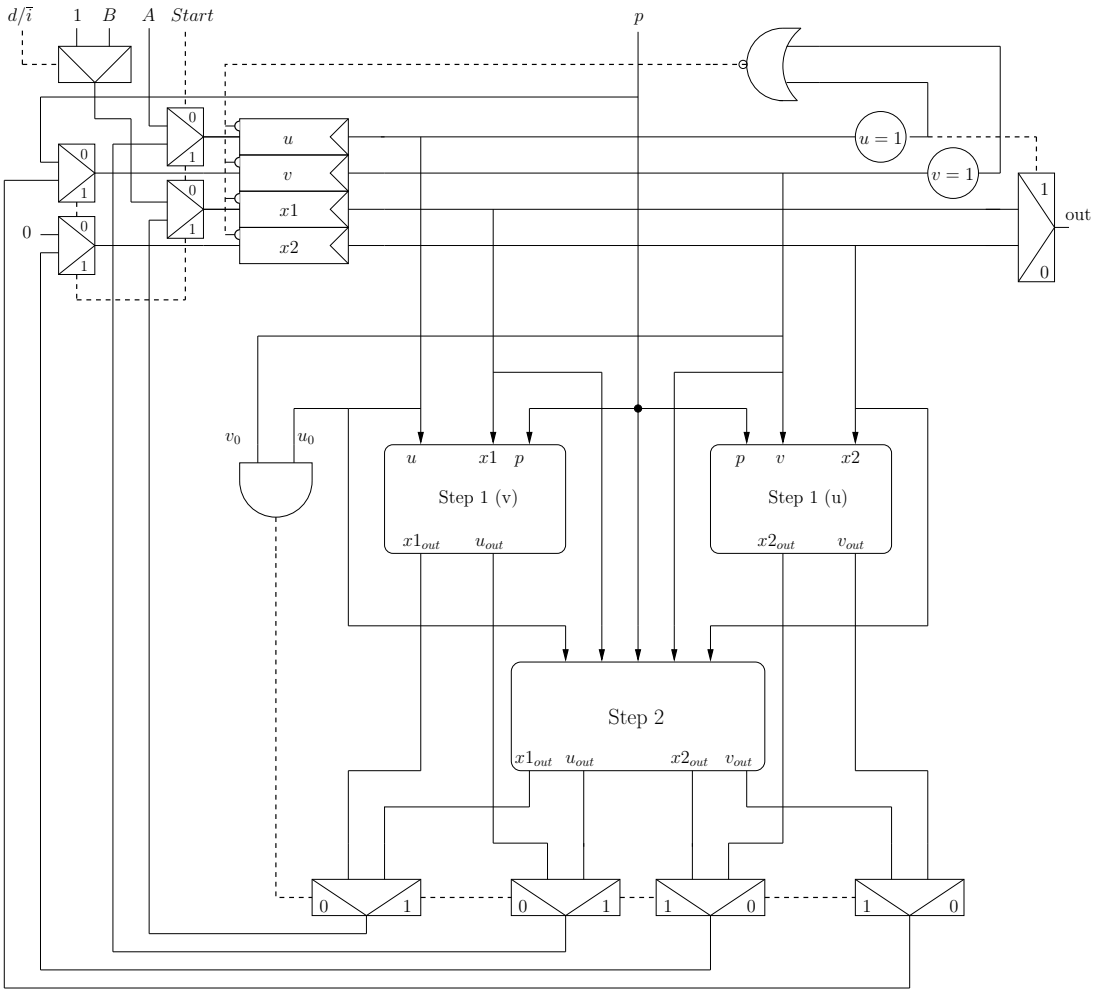


Figure 6.5: Modular Division/Inversion.

8-input multiplexers, two registers to store the intermediate x and y coordinates and one register to store the value of s which is used twice in the computation of both point addition and doubling. The inputs and outputs are transmitted, from and to an 8-bit large bus, through shifting registers.

The controller sets the selection signals for multiplexers and enable signals for modular operators and registers, in order to compute one operation at each clock cycle and store the result in the appropriate register. As a matter of fact, most operations are dependent on the previous results and cannot be computed in parallel, except at the beginning of the point addition where both a modular addition and subtraction are performed simultaneously.

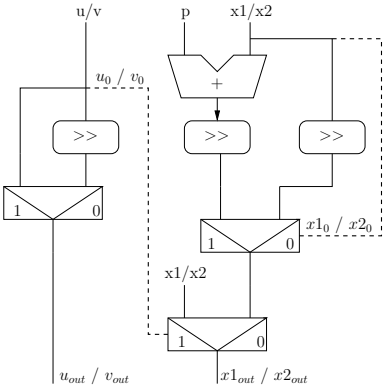


Figure 6.6: Modular Division: Step1.

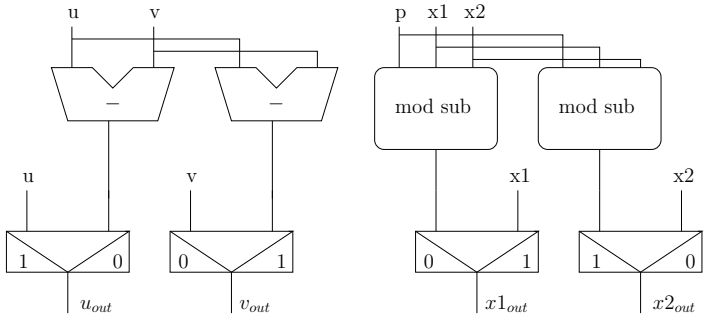


Figure 6.7: Modular Division: Step2.

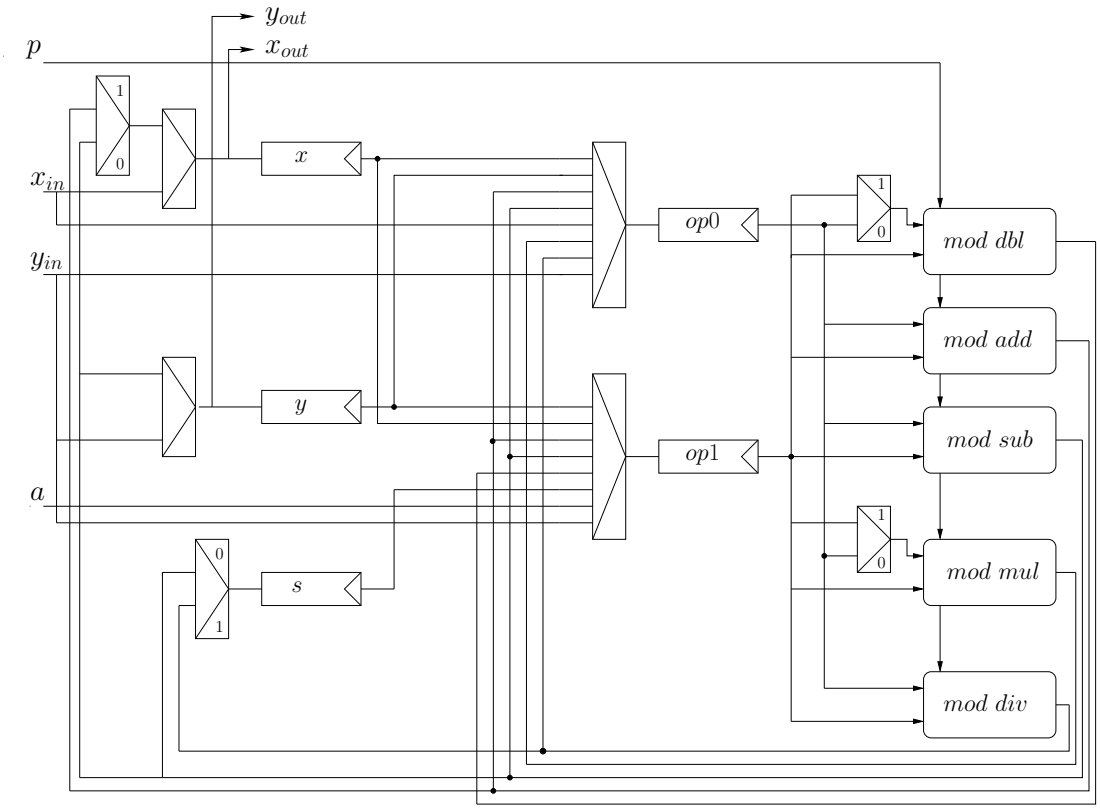


Figure 6.8: ECC unprotected datapath.

6.1.5 *Double an Add Always* Implementation

As one of the most commonly used SPA countermeasure, the *Double an Add Always* algorithm, described in Algorithm 4 is first evaluated. To implement this countermeasure, only two new registers and multiplexers are needed as depicted in Figure 6.9, as well as slight modifications of the controller part.

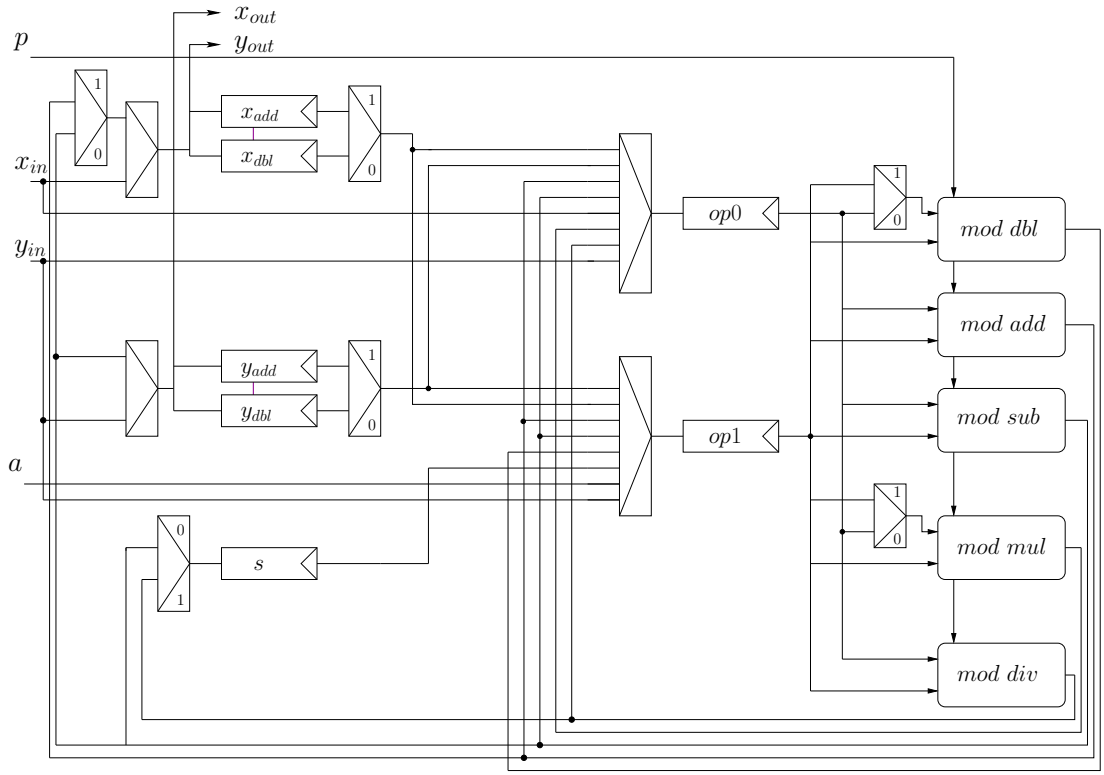


Figure 6.9: ECC protected datapath (*Double an Add Always*).

6.1.6 Protected implementation with "Random Splitting of Scalar"

The other evaluated countermeasure is the random splitting of scalar. As a matter of fact,as stated in Section 3.3.2 it is vastly regarded as one of the most robust against differential attacks, RPA/ZPA and even some fault attacks. Indeed, several papers view it as a reference in terms of SCA protection, but argue that the area/performance overheads are prohibitive. However, as discussed in Section 3.3.2, the only other way of providing robustness against all statistical attacks, as well as chosen-input simple analyses, is to perform input blinding, alone or within a modified scalar multiplication and to draw a new random point at each iteration. In this case, it has been shown

in Section 3.3.2 that blinding is even more costly than splitting.

In terms of implementation, this countermeasure is relatively simple to deploy. The previous datapath, including the *Double an Add Always*, is duplicated, as both parts of the key must be treated in parallel for the protection to be effective. Then the controller is modified to include the initial subtraction, parallel execution and final point addition. Obviously these two additional operations have a negligible impact on the performances compared to the scalar multiplication.

Precise implementation results are given in Table 6.1 for the three different architectures.

Table 6.1: Implementation results for ECC

╲	**Unprotected**	**Double an Add Always**	**Random Splitting**
Number of ALUTs	7405	8050	14673
Number of Registers	3437	3828	6281
Frequency (MHz)	39,5	39,3	38,9

As can be expected, the *double and add always* requires two additional 192-bit registers with regards to the unprotected implementation (as well as a few register bits for the controller part) and some ALUTs for the multiplexers and the controller. One another hand, both logic and memory elements are roughly doubled for the random splitting countermeasure. Regarding the output frequencies, it is noteworthy that they are almost equal, with a downgrade of 0.01 % between the unprotected and the random splitting. As a matter of fact, the critical path is not affected by those countermeasures, but for the multiplexers placed before the x and y registers.

6.2 Experimental Results

Actual attacks were performed on those three architectures, in order verify their theoretical robustness and assess the feasibility of simple attacks, namely SPA and *doubling attack*.

6.2.1 SPA on the Unprotected Implementation

The first conducted attack is a SPA on the basic design. Although being rather simple to deploy, the SPA has been described as extremely efficient in recovering the entire secret key of unprotected asymmetrical algorithms such as RSA and ECC, as discussed in Section 2.1.1.

With the same experimental setup as in Section 4.2.2, power consumption measurements are taken during the computation of the ECC scalar multiplication. In order

to lessen the noise, the same input point is used 256 times and the mean of all traces is used for the analysis. The resulting curve is shown in fig 6.10.

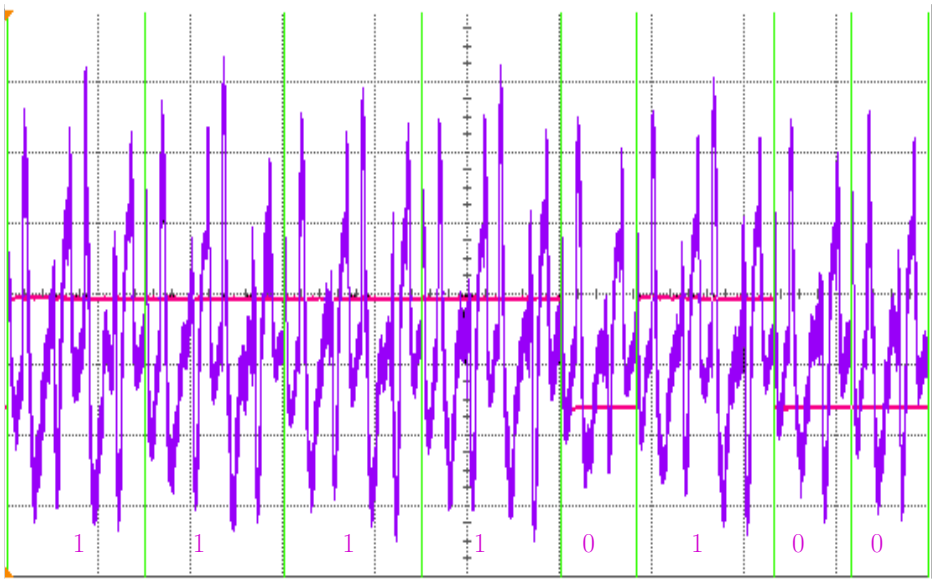


Figure 6.10: SPA on unprotected ECC implementation.

The purple curve shows the power consumption, while the pink one determines the value of the key for each iteration of the scalar multiplication. The green curve is only displayed to clearly mark the boundaries of those iterations.

It can be observed that the patterns of the power consumption for the turns with doubling and addition ($key\ bit\ = 1$) are clearly different than those with only doubling ($key\ bit\ = 0$). As a matter of fact, in the first case, a turn is composed of four power spikes, with a very short gap between the 2^{nd} and 3^{rd}, the 3^{rd} spike being the highest. On another hand, when only a doubling operation is performed, only 2 spikes are visible.

Eventually, as can be expected, one mean power measurement is enough to get the entire value of the secret key on such an unprotected design.

6.2.2 SPA on the Double an Add Always

In order to illustrate the behaviour of the *Double an Add Always* algorithm, SPA was conducted on this implementation, in the same conditions as Section 6.2.1. Figure 6.11 shows the resulting SCA curve, after performing the average over 256 power measurements.

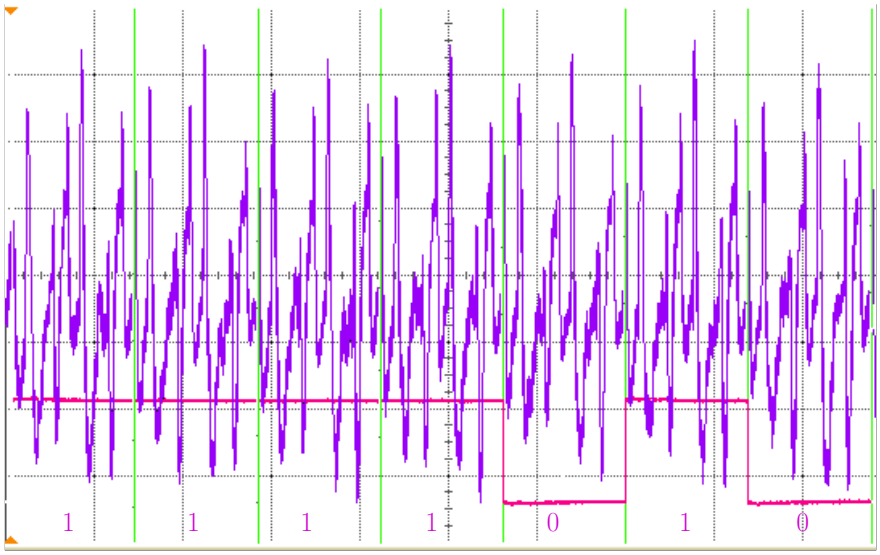


Figure 6.11: SPA on *Double an Add Always* ECC implementation.

As anticipated, all iterations of the scalar multiplication display similar behaviours regarding power consumption. Indeed, this sole curve would not allows an adversary to directly find any bit of the secret key.

However algorithms like *Double an Add Always* and *Montgomery Ladder*, as well as other countermeasures against statistical attacks like the *randomisation of the secret* are said to be vulnerable against chosen-input elaborate SPAs, namely *doubling attack* and *Comparative Power Analysis*. Therefore, in order to experimentally illustrate this weakness and assess the difficulty and feasibility of such techniques, when targeting ad-hoc FPGA implementations, a *Doubling Attack* was performed on this architecture.

6.2.3 "Doubling Attack" on the Double an Add Always

First a point $P : (x_p, y_p)$ of the curve is randomly chosen and power measurements are recorded during the execution of $Q_1 = [k].P$. To reduce the computational load and allow a correct precision in terms of number of samples by iteration of the scalar multiplication, only a few rounds are considered. Moreover, for more clarity, 256 observations are taken and the average resulting curve is stored as a *.csv* file in order to be studied later on.

Then the coordinates of point $P_2 : (x_{p2}, y_{p2})$ such as $P_2 = 2P$ are computed by software and another round of power signatures are acquired, corresponding to the computation of $Q_2 = [k].P_2 = 2.[k].P$. As before, the average over 256 measurements for the same number of iterations is generated and stored as a *.csv* file.

Finally the software part of the attack is undertaken. Thanks to a trigger pointing out the beginning of the scalar multiplication, the number N of samples for a single iteration is recovered. Then the N first samples are removed from the *.csv* file of Q_1, in order to simulate a left shif of one iteration and a point-by-point subtraction is performed between this file and the one of Q_2. For this attack to be successful, the difference should be nearly null when the key bit k_i, related to Q_2^i (denoting Q_2 at iteration i), is equal to 0. Indeed the doubling operation of Q_2^i should, in this case, be the same as the one of Q_1^{i-1}.

Results are displayed in Figure 6.12. The topmost curve represent the shifted computation of $Q_1 = [k].P$, while the second and third curves are respectively the measurement corresponding to $Q_2 = [k].2.P$ and the difference between the previous two. Moreover the key bit value is displayed by the green line (1 when up and 0 when down).

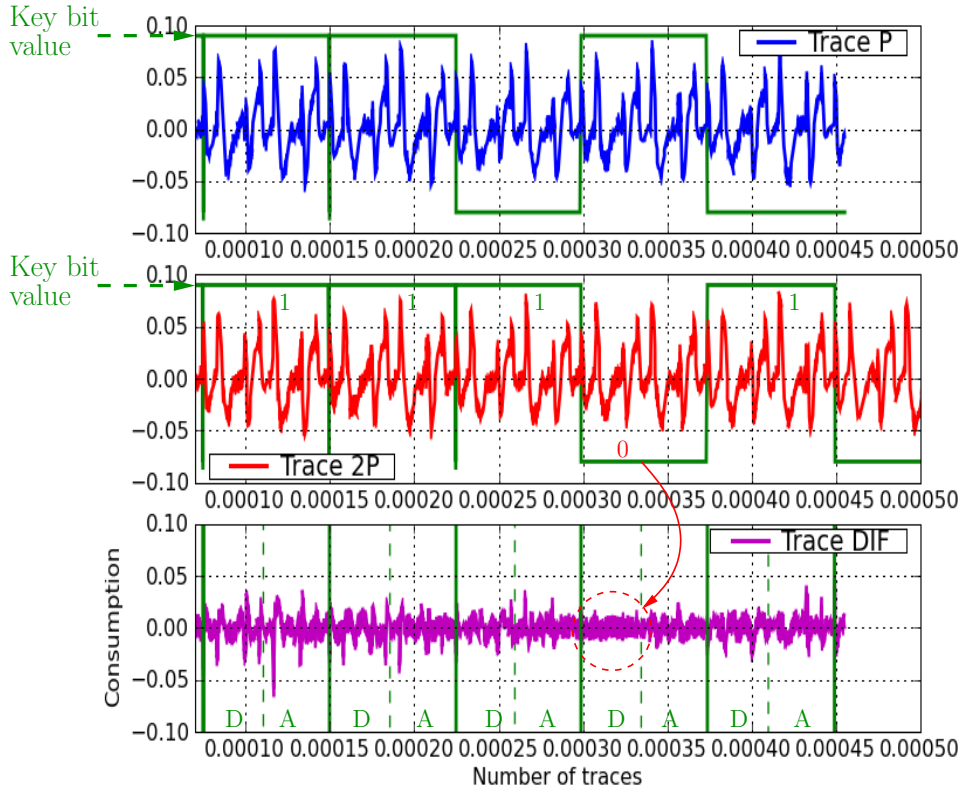


Figure 6.12: Doubling Attack on *Double an Add Always* ECC implementation.

As expected, it can be observed that when $k_i = 0$ for Q_2^i, the difference is minimal

in the first part of each iteration (left of the green vertical dashed line), which corresponds to the doubling operation. This is illustrated by dashed circle.

In the end, this experiment shows that chosen-message SPAs are indeed easily mounted in practice and efficient. Thus, care must be taken by designers to include appropriate countermeasures against such attacks.

6.2.4 "Doubling Attack" on the Random Splitting of Scalar

When coupled with a SPA resistant algorithm, namely *Double an Add Always* or *Montgomery Ladder*, the random splitting should prove resistant to most existing passive SCAs, including chosen message attacks. However, no paper has yet dealt with such an attack on this countermeasure. Therefore, in the perspective of verifying its theoretical robustness, a *doubling attack* is performed on this new architecture, in the same conditions as in Section 6.2.1.

Results are displayed in Figure 6.13. As expected, this implementation proves to be robust against the *doubling attack*. As a matter of fact, the real secret key is not used during the computation, as discussed in Section 3.3.2, hence no real correlation can be found between this key and the doubling operations.

6.3 Conclusion

In this section, we presented simple ECC architectures with and without two classical countermeasures against SCA. Rather than seeking optimizations, the pragmatic goal of those designs was to experimentally verify the overheads in term of performances and resource consumption of such schemes when embedded in FPGA, as well as their theoretical security against several SCAs. This was realized by actual implementations on an Altera StratixII FPGA followed by SPAs and *Doubling Attacks* on both schemes. The results showed that choosing an FPGA as the target device does not induce unexpected overhead or vulnerability against the performed SCAs.

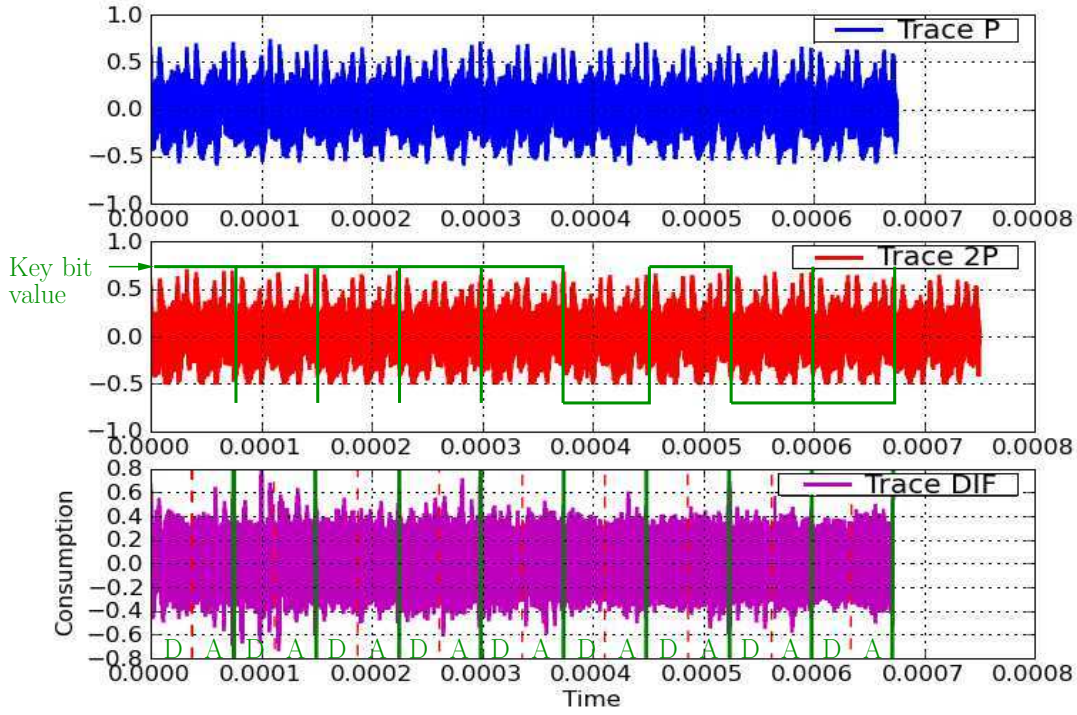


Figure 6.13: Doubling Attack on ECC implementation with *Double an Add Always* and Scalar Splitting.

Chapter 7

Design of New Attacks

Recent study shows that most distinguishers are equivalent asymptotically [116], *i.e.* they are sound. However, in real-life attacks, the scarce resource is usually the number of available side-channel measurements. In that regard, three novel SCA scheme are presented hereafter which aim at deceasing the number of MTD required to perform successful SCAs. First, the Principal Component Analysis (PCA) is used as a distinguisher, in order to exploit multiple time samples within each observation. The second idea is to combine either measurement techniques or distinguisher in order create new powerful SCAs. Finally the behaviour of the secret key with regards to its rank during classical SCAs is studied and taken advantage of to propose a generic SCA enhancement algorithm.

7.1 First Principal Component Analysis (FPCA)

Principal Component Analysis (PCA) is a multivariate data analytic technique [146, 158] with applications in several fields such as computer vision [104, 163], robotics [193], sociology and economics [160]. PCA can be employed to identify patterns in multidimensional data set and visualise them in a lower dimensional space, in order to highlight their similarities and differences. Regarding SCAs, PCA has already revealed its efficiency as a pre-processing tool for template attacks [141].

The idea behind the First Principal Component Analysis (FPCA), is to perform an attack, using the projection on the first principal components as an SCA distinguisher, to sort and rank key-dependent partitionings, in order to select the most relevant key hypothesis.

7.1.1 FPCA: Principle

The global process of FPCA is similar to most SCA like DPA or CPA. First a number of leakage observations are recorded, generally power or EM measurements. Those are

sorted into partitions, for all key hypotheses, depending on a leakage model chosen for the attack. And finally a distinguisher decides on the most relevant key hypothesis.

As before, the considered leakage model is the Hamming Distance (HD), as it is commonly used, specially for FPGA implementations [33, 169]. The number of partitions, noted N_p, is inferior or equal to the number of possible values taken by the model, $\#HD$. The trivial partitioning associates each value to one partition, hence $N_p = \#HD$. Another possibility, inspired from [120], is to build less partitions, storing observations with close values of the model together. For instance, let's consider an attack on the DES algorithm, the target is one 4-bit sub-key, thus a HD such as: $0 \leq \#HD \leq 4$. In this case, a possible partitioning would consist of three groups: the first for observations with a $HD > 2$, the second for $HD = 2$ and the third for $HD < 2$.

7.1.2 Reference Statistic

Once observations are sorted in N_p partitions, a statistic entity denoted S_{ref} (mean, variance, ...) is chosen and a corresponding statistical trace is computed for each partitions and referred to as *reference*. For instance, with the mean as S_{ref}, the reference is the average trace of all observations within a given partition. Thus a set of N_p references, denoted by R_k, is created for each key hypothesis k. Those references will be used by the PCA as criterion to highlight differences between the partitions. It is noteworthy that only one S_{ref} must be used during an given attack. The principle of FPCA is to exploit the analysis of the different R_k to discriminate the behavior of the secret key with regards other hypotheses

7.1.3 FPCA distinguisher

For a given k, the dependencies between references are made more eligible by PCA when those references are projected to the new axes system of the principal components. PCA can analyze these dependencies by measuring their dispersion in this new coordinate space. As a matter of fact, the larger the eigenvalue (denoted by λ) corresponding to one eigenvector is, the greater the references dispersion on this eigenvector. As described in equation (7.1), the total variance V_{tot} of one R_k is equal to the sum of all eigenvalues corresponding to all principal components:

$$V_{tot} = \sum_{j=1}^{N_p} \lambda_j \,. \tag{7.1}$$

Then, given a valid power model, the reference statistic S_{ref} is sound when its values are correlated to the activity of the actual secret key. In this case, the V_{tot} related to this key will increase with the number of recorded observations, as this partitioning should display the most visible differences between the values of S_{ref}. By contrast, the

V_{tot} of other hypotheses should approaches zero with an increasing number of measurements, due to the fact that their random behaviours will induce asymptotically similar distribution within the partitions, hence indistinguishable S_{ref}.

It is noteworthy that the first and second principal components are the most significant, given that they cover most of the total variance. This quantity, noted V_m for the $m\text{-}th$ principal component P_m, is defined as follows:

$$V_m(P_m) = \lambda_m / V_{tot} \,,$$

where the eigenvalue λ_m corresponds to P_m. The cumulative V_m, denoted by CV, is introduced for m' principal components and defined by:

$$CV(P_1, \ldots, P_{m'}) = (\sum_{i=1}^{m'} \lambda_i) / V_{tot} \,.$$

In practice, the last principal components are generally viewed as related to the noise and only a few m' components are considered for analysis.

The main idea of FPCA, is to reduce the dimensionality of power consumption measurements, by using PCA, in order to take into account the information of different time samples and thus to thoroughly exploit the leakage. In this perspective, the CV criterion is taken advantage of, to extract the significant components. For instance, only the m' first components, which cover more than 95% of the total variance, are considered for each key hypothesis k.

Therefore, let's define the indicator F_k such as:

$$F_k = \sum_{m=1}^{m'} (\lambda_m) \,.$$

where m' is the number of considered components and λ_m the eigenvalue for P_m. This indicator corresponds to the factor of dispersion and represents a global description of the leakage without the need of more detailed knowledge about the encryption process. It can thus be used on most architectures including Dual rail Precharge Logic (DPL), which aim at making the activity of the cryptographic process constant and independent from the manipulated data. Indeed as stated before, an perfect DPL implementation is hardly feasible in real FPGA applications, and F_k can exploit any leaked information within the side-channel measurements.

Eventually, the best key guess corresponds to the highest value of F_k for all key hypotheses i.e. $argmax(F_k)$.

7.1.4 FPCA vs DES and masked DES

To validate the FPCA concept, SCAs were performed on two different DES architectures: an unprotected ASIC implementation, freely available on line, in the context of the first version of DPA CONTEST competition [73, 174] and an FPGA implementation of DES protected with a classical boolean masking countermeasure. The experimental setup for the latter is the same as in Section 4.2.2. However, in order to reduce the computational overhead, the observations are truncated and only the two first rounds of the algorithm are considered.

On one hand, for the unprotected implementation the mean is chosen for S_{ref}, as its leakage for the secret key hypothesis is linearly correlated to the Hamming distance model. On the other hand, masking techniques are designed to produce a constant average consumption, hence the mean should not be a sound reference when dealing with such countermeasure. Nevertheless, as stated in Section 5.1, masking has been proven to be vulnerable against variance-based attacks like VPA, thus the variance is chosen as S_{ref}.

With the purpose of drawing a thorough comparison, several attacks are considered, namely DoM, DPA and CPA for the unprotected DES and VPA for the masked one. As before, their efficiency is evaluated with the success rate and guessing entropy metrics [168]. Results regarding the unprotected implementation are displayed in Figure 7.1 and Figure 7.2. As can be observed, FPCA outperforms the other attacks. As a matter of fact, the 90% success rate is reached in about 110 observations, versus 190 for CPA and DPA. The guessing entropy also behave accordingly, as FPCA is the first to reach the best rank (o). Nevertheless, it is noteworthy that DPA shows an almost similar guessing entropy as FPCA, although it proven to be clearly less efficient in terms of success rate.

Regarding the masked architecture, results depicted in Figure 7.3 and Figure 7.4 shows that although FPCA and VPA are asymptotically equivalent, FPCA is clearly faster to reach a significant success rate, for instance it takes roughly 11000 observations to reach 80% versus 15000 for the VPA. In the same way, the guessing entropy shows a much faster decrease for the FPCA.

*

To summarize FPCA is a new type of statistical SCA, based on using the PCA as distinguisher. Its efficiency has been put to light against both unprotected and masked DES FPGA implementations. Moreover empirical results showed superior performance of FPCA with regards to classical SCA, namely DoM, DPA, CPA and VPA, when evaluated with the success rate and guessing entropy metrics.

This work opens he door for several researches, for instance extending this scheme to new applications based on other multivariate data analytic tools such as the Linear

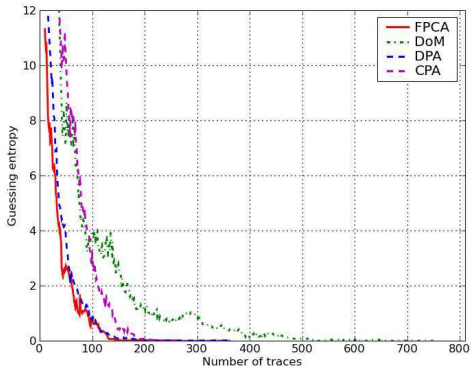


Figure 7.1: Unprotected DES guessing entropy metric.

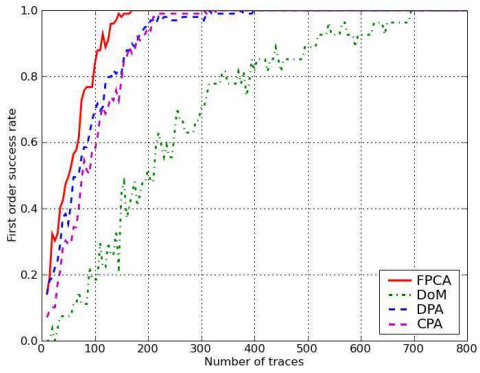


Figure 7.2: Unprotected DES 1st-order success rate metric.

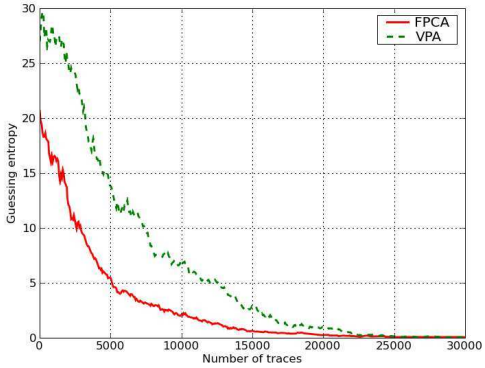


Figure 7.3: Masked-ROM guessing entropy metric.

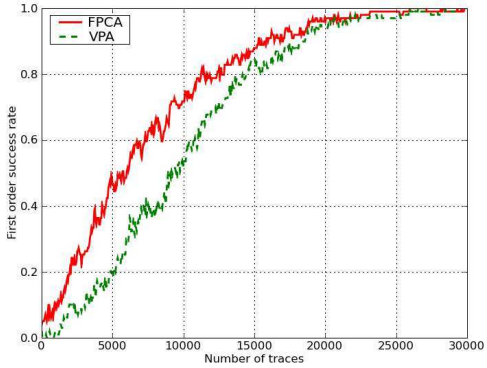


Figure 7.4: Masked-ROM 1st-order success rate metric.

Discriminant Analysis (LDA), PCA based Spearman correlation, Kernel PCA or Independent Component Analysis (ICA), which have been proposed as alternatives to the basic PCA. A theoretical study of optimized *reference statistic* should also be undertaken in order to make more eligible the description of the leakage related to the secret information.

7.2 Combined Attacks: Measurements Combination

Like FPCA, numerous attack schemes have been proposed since the introduction of DPA by Kocher [99], generally describing new distinguishers, for instance the correlation [33, 75, 100, 105], mutual information [13, 62], or variance [167].

However, few papers have tackled the idea of devising combined methods to im-

prove existing attacks. A combination of power and timing attacks is deployed in [151] to attack RSA and combinations of both leakage models and different time samples are proposed in [3].

Nonetheless, the Section hereafter and the following, deal with alternate combination methods, namely combining different measurements and different distinguishers.

To mount a successful SCA, the acquisition process is a critical step. When targeting an FPGA implementation, common acquisition techniques use a differential probe plugged into the core power supply through a shunt resistor (in the case of power measurements), or an EM antennae , placed either on the top of the FPGA or on the bottom decoupling capacitors (for EM measurements). These capacitors have the interesting property to be linked to different power banks of the FPGA Another method is the EM cartography [147] which can construct a dynamic image of the device, showing several leakage points of different intensity. As a matter of fact, EM radiations are not necessarily located at the exact spot of the process they are correlated to. When dealing with complex cryptographic circuits, that can be spread throughout the device, or at least positioned across different power banks, sensitive information leakage is deemed to occur at several different locations. Some of these leakage points or decoupling capacitors can be used to perform a successful attack, however the number of Measurement To Disclose (MTD) will vary.

Rather than to chose a single attack target, leakages from several sources can be combined, in order to accelerate the attack, namely decrease the number of MTD.

7.2.1 Theoretical Background

Information gain of a single attribute X with respect to class C, also known as mutual information between X and C, measured in bits is:

$$Gain_c(X) = I(X;C) = \sum_x \sum_c P(x,c) \log \frac{P(x,c)}{P(x)P(c)} \quad . \tag{7.2}$$

Equivalently:

$$I(X;C) = H(X) - H(X|C) \quad . \tag{7.3}$$

Where $H(X)$ is the entropy of X and $H(X|C)$ the conditional entropy of X knowing C. Information gain can be viewed as a measure of the strength of a 2-way interaction between an attribute X and the class C. This can be extended to a 3-way interaction by introducing the "interaction gain" [85]. It is also measured in bits and represent the difference between the actual decrease in entropy achieved by the joint attribute XY and the expected decrease in entropy with the assumption of independence between X and Y. Interaction gain can be considered equivalent to multivariate mutual information [61]. To simplify the calculation of entropy let's consider that the distribution of X is Gaussian. In this case the entropy can be calculated as a function of standard deviation σ_x of X as:

$$H(X) = \sum_i p(x_i) \log_2(p(x_i)) = \log_2(\sigma_x \sqrt{(2\pi e)})$$

Estimating entropy using Gaussian parametric method might not be very accurate but it works well in practice [137]. Nevertheless, other methods of estimating entropy can be equally applied.

The Venn diagram representation of the interaction gain is displayed in Figure 7.5.

Equation (7.4) is defined using the notations of Figure 7.5, where Z is a third random variable of representation z. As per this Equation (7.4), interaction gain is equal to $-G$. If X and Y are independent, then $I(X,Y;Z) = I(X;Z) + I(Y;Z)$ hence the interaction gain $I(X;Y;Z)$ is zero. From Figure 7.5(a) and (b) it derives that combination is possible when the information equal to D is added to $I(X;Z)$ with introduction of Y. This makes $I(X,Y;Z) = D + G + F$. If D is null, then introduction of Y is not providing any extra information.

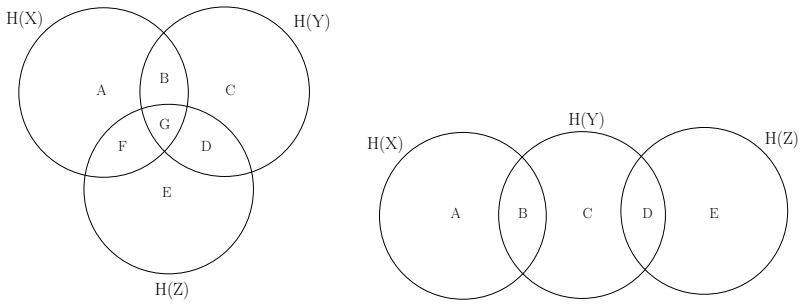


Figure 7.5: Venn diagram representation of a case when combination is (a) possible, (b) not possible.

$$\begin{aligned} I(X;Y;Z) &= I(X,Y;Z) - I(X;Z) - I(Y;Z) \\ I(X;Y;Z) &= (D+F+G) - (F+G) - (D+G) = \textbf{-G} \end{aligned} \tag{7.4}$$

This condition can be validated by a simple test: the possibility of combination noted PC can be computed as a ratio:

$$PC = \frac{Max(I(X;Z), I(Y;Z))}{I(X,Y;Z)}.$$

For a combination to exist, the value of PC should lie between 0.5 and 1, where PC=1 will suggest no combination is possible. In the context of combined attacks, interaction gain can be directly applied. This is a profiling step because the knowledge of the secret key is required to calculate PC. Alternatively, further studies could consider using PC as a distinguisher.

7.2.2 Experimental Results

The target is an unprotected DES cryptoprocessor implemented on an Altera Stratix-II FPGA. attacks are performed using the same experimental platform as in Section 4.2.2 and antennas of the HZ–15 kit from Rohde & Schwarz in order to measure the electromagnetic fields.

The targeted leakage points are the decoupling capacitors on the backside of the FPGA. As a matter of fact, the analysis of the post fitting floor-plan validated the assumption that the design was spread across different power banks. Moreover, due to the small number of capacitors, a trial-and-error method was efficient to choose two leaking capacitance. Indeed, complex cryptographic designs generally consumes more than other operations on a chip which can be observed on an EM trace. Moreover a crypto-processor is a bulky design and is likely spread over different power banks in an FPGA which are terminated by different capacitors. Therefore, different capacitors leak more information about a certain part of the circuit. Capacitors where these cryptographic operation are obvious or clearly distinguishable on the EM trace were chosen for the attack and antennas were placed close to each of them, as illustrated in Figure 7.6. Two sets of 5000 EM measurements were then collected from those capacitors for the same dataset.

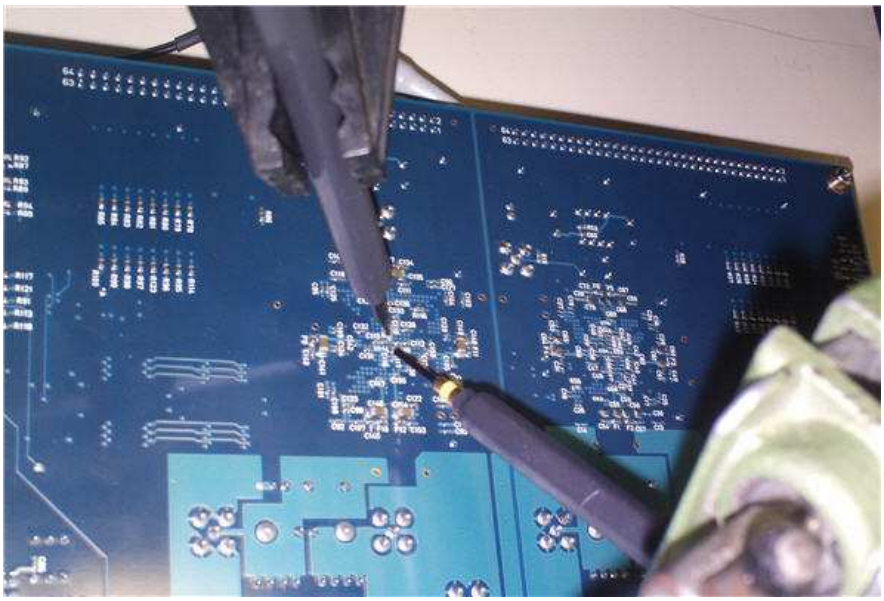


Figure 7.6: Placement of antennae for a combined EMA based on combination of measurements.

The first step is to evaluate the possibility of combination. Figure 7.7 shows the PC values for the first Sbox. Figure 7.7 (a) considers traces from two capacitance called

C_1 and C_2 which are leaking relevant information. It can be seen that the value of PC is close to 0.5 when the value of mutual information is relevant. Figure 7.7 (b) considers traces from a leaking capacitance and another point which is not leaking. Here the value of PC is close to 1. In other parts of the trace there is noise and the value of PC is changing randomly. Unfortunately, no set of traces for which PC may take values between 0.5 and 1 was available. It can be inferred from this experiment that combination is possible for the traces in Figure 7.7 (a).

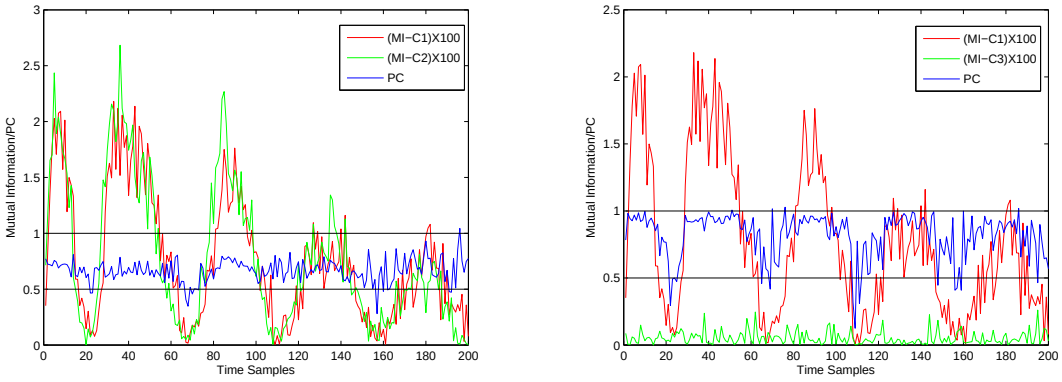


Figure 7.7: Calculation of PC for two cases when combination is (a) possible, (b) not possible (mutual information of the two measurements is multiplied by 100 to visualize on the same scale as PC.)

Next step is to experimentally validate the efficiency of measurement combination, using a classical SCA, for instance CPA. This attack was performed on the measurements collected from the two capacitors $capa_1$ and $capa_2$ independently. Table 7.1 summarises the averaged number of MTD to recover each sub-key, over 30 realisations on both set of traces. It can be observed that a CPA on $capa_1$ gives better results on S-Box 0,1,3 and 7 and inversely $capa_2$ proves to be better suited for the other S-Boxes.

The combination of attacks on both sets of measurements is performed via an aggregate function[1] Ψ. In this case a simple $\Psi = Sum()$ is chosen, but further research could be undertaken to study the relevance of other aggregate functions. The attack used to apply combination is CPA. Note that other distinguishers like Spearman or Gini could also be used. The results for the combined SCA, displayed in the third row of Table 7.1, clearly show the improvement brought by this method, with regards to the classical mono-source CPA. As a matter of fact, the combination allows to retrieve all sub-keys in less MTD than the other two CPAs, with a security gain of 4.16 to 44.86%.

[1] In Computer Science, an aggregate function is a special type of operator that returns a single value based on multiple rows of data.

This also complies with the condition on PC. However the scale of PC and the gain in number of MTD cannot be compared as each quantity is computed in a different manner.

Table 7.1: Number of MTD for a CPA with and without measurement combination.

S-Box No.	0	1	2	3	4	5	6	7
$capa_1$	350	943	733	400	410	320	548	592
$capa_2$	432	1073	720	980	176	281	551	192
$Combination$	212	750	397	251	165	270	448	184
Gain (%)	39.42	20.46	44.86	37.25	6.25	3.96	18.24	4.16

*

To conclude, it has been demonstrated in this Section that multiple measurements can be exploited to enhance existing SCAs on FPGA implementations. As a matter of fact, when cryptographic designs are spread over several power banks, different leakages can be observed when measuring the electro-magnetic radiations of the corresponding decoupling capacitance, hence providing multiple sources of side-channel leakage, that can be efficiently combined. Theoretical background based on information theory metrics was provided to test combination by computing possibility of combination PC and actual experiments were performed that resulted in a gain of about 45%. Obviously such methodology aims at improving SCAs in general, but there may be cases which are better off using the classical techniques.

Further work include a study of optimized aggregate functions and alternate was of performing measurement combination, for instance combining power and EM observations.

7.3 Combined Attacks: Distinguisher Combination

This Section present a second technique which consists in combining different existing distinguishers in order to decrease the required number of MTD to perform successful attacks. This methodology is demonstrated by combining Pearson and Spearman correlation coefficients, as they are commonly used during SCAs on cryptographic algorithms implemented in FPGA. Nevertheless, any type and number of SCA distinguishers could be combined, depending on the application.

First, the Gini correlation is introduced. It combines advantages of the Pearson correlation ρ and the Spearman correlation r. Therefore its application to side-channel analysis is studied as the first combination technique.Second, some practical methods to combine results of different distinguishers in order to create a more powerful SCA are discussed.

7.3.1 Mathematical Background

Let X and Y be two random variables with probability density function (*pdf*) $P_X(x)$ and $P_Y(y)$, respectively. Suppose we want to best approximate Y with another variable X based only on the knowledge of their joint distribution $P_{X,Y}(x, y)$. The problem is to find a function $\phi(.)$ of X that best fits Y among all possible forms of ϕ. In our study, the variable X is deterministic since it is theoretically predicted from a known cryptographic process. Whereas, the variable Y is a real measure acquired by an oscilloscope. Thus, for sake of clarity, the variable X is called *the prediction* and Y *the measurement*. Depending on the causal connections between X and Y, their true relationship may be linear or non linear. The independence of X and Y implies that they are uncorrelated. The converse is true only under the Gaussian assumption. In fact, this assumption states that the joint distribution of X and Y is bivariate normal. In this case, X and Y are said to be *jointly Gaussian* variables.
However, regardless of the true nature of the relation, a linear model can be used for an initial approximation when X and Y are scalar:

$$Y = \phi(X) + \epsilon = (\alpha + \beta X) + \epsilon, \tag{7.5}$$

where α is the *intercept*, β is the *slope* of the line and ϵ is the error of the approximation.

Definition 7.3.1 *A set of n random variables $X_1, X_2, \ldots, X_n$ are **jointly Gaussian** if $\sum_{i=1}^{n}(a_i X_i)$ is a Gaussian random variable $\forall$ real a_i, with $i \in [1..n]$.*

A common pitfall about the validity of the Gaussian assumption is to check only that X and Y are drawn from normal distributions. If X and Y are each individually Gaussian then this does not imply that they are jointly Gaussian. Generally, a joint distribution $P_{X,Y}(x, y)$ is said to be bivariate normal if the four conditions **normal conditional distribution**, **linearity**, **homoscedasticity** and **normal marginal distribution** are satisfied [12]. Homoscedasticity means that the conditional distribution of Y given $X = x$ has finite variance for each x. Moreover, under these conditions, ϵ must be drawn from a zero mean normal distribution. In other words, ϵ is a random variable strictly independent from X and a linear function ϕ characterizes the dependence between X and Y entirely. Estimation theory shows that under the Gaussian assumption, ρ is the best tool to totally characterize the association (purely linear) between X and Y [50, 91, 183]. However, in real situations, it is hard to get a perfect binormal joint distribution. In such situations, the higher the deviation from the Gaussian assumption is, the lower the efficiency of ρ. In this case, other correlation coefficients have been developed to be more robust[1] than the Pearson correlation. Some examples include the Spearman coefficient r and Kendall's tau r_τ (rank correlations), biserial and

[1] A statistical criterion that does not make any assumption about the joint distribution is said to be robust or distribution free.

tetrachoric [125]. Spearman correlation measures both the linear and the non-linear relationship between the two variables, as it does not requires that the observations are drawn from a Gaussian distribution. It is a *non-parametric* coefficient that was first applied in side-channel context in [17].

In the literature of correlation analysis, there is no rule to determine whether the ρ will outperform its competitors or not, provided the deviation from Gaussianity is not excessive. In this insight, statisticians have recently started to investigate actual combinations between existing correlation coefficients, which bridge the gap between Pearson coefficient and its competitors.

7.3.2 Gini Correlation: A Mixture of Pearson and Spearman Coefficients

Pearson correlation, ρ, might perform poorly when the data is attenuated by non-linear transformations, in contrast to Spearman correlation, r. However, r is not as efficient as ρ under the Gaussianity. This robust alternative to ρ might lose its efficiency especially when the data involves different types of variables (*e.g.* discrete/continuous). Moreover, when the number of different values taken by either variables is small, then this might create a *problem of ties* (*i.e.* there is a tie while ranking the data. This affects considerably the quality of r [69]). In such cases, the loss of efficiency might not be compensated by the robustness in practice. For this purpose, statisticians have recently came with an interesting combination between Pearson and Spearman coefficients, namely Gini correlation (ξ), which has been proposed in [149]. Spearman correlation r , which is just Pearson correlation ρ applied on already ranked data, can be defined using the notion of cumulative distributions as:

$$r_{(X,Y)} = \rho_{(F_X,F_Y)} = \frac{1}{\sigma_{F_X}\sigma_{F_Y}} Cov(F_X(X), F_Y(Y)) \tag{7.6}$$

where F_X and F_Y are the cumulative distribution of X and Y, respectively. The Gini correlation coefficient is given by:

$$\xi_{X,Y} = \frac{Cov(X, F_Y(Y))}{Cov(X, F_X(X))} \tag{7.7}$$

Note that in general ξ is not symmetric i.e. $\xi_{X,Y} \neq \xi_{Y,X}$. In practice, the choice between the two forms, depends on the type of variables X and Y. In statistics, it has been reported that if X is discrete and Y is continuous, then $\xi_{Y,X}$ would be a good choice.

Practical Computation of Gini Correlation and Properties

Consider n couples (X_i, Y_i) with $i \in [1..n]$ of independent variables drawn from a bivariate distribution. If these couples of variables are ordered (sorted from low values to high values) with respect to the X_i, new couples of variables $(X_{(i)}, Y_{[i]})$ can be

generated, where $X_{(1)} < \cdots < X_{(n)}$. $Y_{[1]}, \cdots, Y_{[n]}$ are the related concomitants [126], which depend on the ordering of the X_i. As proposed in [149], $\xi_{Y,X}$ is computed as:

$$\xi_{Y,X} = \frac{\sum_{i=1}^{n}(2i-1-n)Y_{[i]}}{\sum_{i=1}^{n}(2i-1-n)Y_{(i)}} \,. \tag{7.8}$$

Note that, $\xi_{X,Y}$ is computed in the same way as $\xi_{Y,X}$, by just reversing the roles of X and Y. For more details about the Gini correlation coefficient, the reader is referred to [191].

The three correlation functions (Pearson, Spearman and Gini) are compared using simulated traces. The leakage function of a FPGA can be modelled as $\mathcal{L}(x) = \text{HW}(x) + \alpha \cdot \delta(x)$, where $\delta(\cdot)$ is the Kronecker symbol. Here $\mathcal{L}$ is the leakage function and HW is the Hamming weight function. This leakage model was verified on public AES traces of DPA contest v2 [175] as shown in Figure 7.8. The attack targets 8 bits at the output of an AES Sbox.Thus, the HW function can take nine possible values (HW=0,1,2,3,4,5,6,7,8). The Figure 7.9 (a) shows the comparison of three correlation coefficients as a function of α. A proper approximation for the Gaussian case is seen when α is 0 and all three correlation coefficients are equivalent empirically. When α is negative, Pearson correlation is not optimal. Spearman and Gini still perform very well as they are not sensitive to monotonic transformation. For positive α, Pearson is not suitable and Spearman also becomes less optimal as the function $\mathcal{L}$ is no more monotonic. Since the transformation is not drastic the Spearman correlation tries to stabilise itself (Figure 7.9 (b)). However, Gini does show some improvement over Pearson and Spearman. As stated earlier, Gini is a combination of Pearson and Spearman, it can therefore be argued that combination helps in non-ideal cases. Next, empirical approaches to combine Pearson and Spearman are proposed.

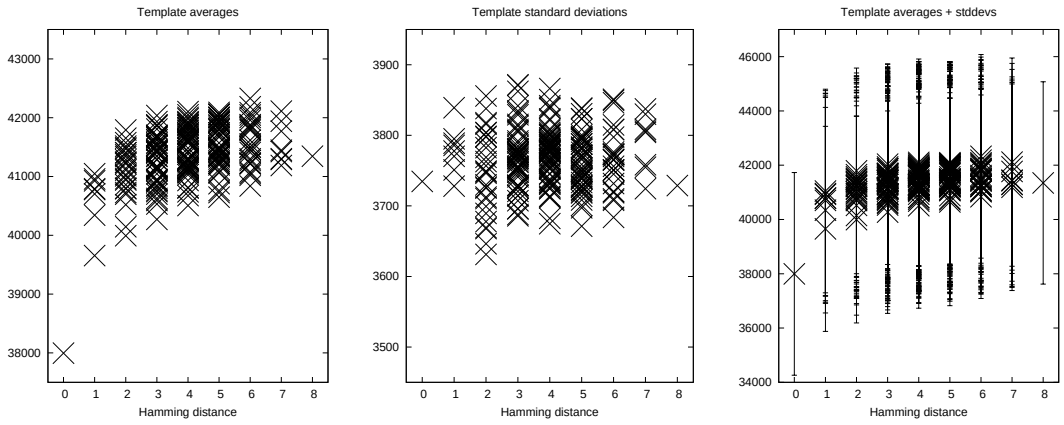
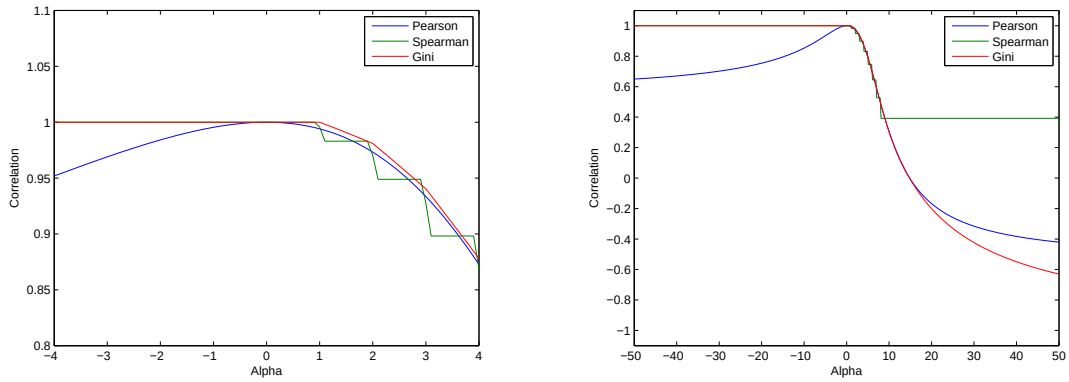


Figure 7.8: Leakage function of Sbox 0 (DPA contest v2).



(a) (b)

Figure 7.9: (a) Three correlation coefficients on the leakage function $\mathcal{L}$, extended in (b) to higher values of α.

7.3.3 Pearson-Spearman Combination: An Empirical Approach

Why the Combination Works

As stated previously, most side channel attacks differ in the measurements partitioning process and the used distinguisher. Otherwise, they usually run iteratively and a new ranking of all secret hypotheses is created at each iteration. Our starting argument to combine two different distinguishers, noted δ_1 and δ_2 involves four observations. The first is that the two distinguishers are equivalent (*i.e.* display similar evolutions), in terms of *success rate* and *guessing entropy* security metrics [167] when performed in parallel, on the same set of side-channel measurements. In addition, we observed that the *secret key* mostly keeps the same temporal position for both distinguishers unlike the false key hypotheses. Let's define the *predicted key*[1] PK as the key hypothesis that has the best rank at the current iteration. Its value is updated for each processed measurement. It can be observed that the two distinguishers often display different PK: $PK_{\delta_1} \neq PK_{\delta_2}$. But more importantly, this emphasizes the fact that δ_1 and δ_2 are statistically different, even if they are exploiting the same dependency. Eventually, the last observation is that *secret key* is always ranked among the best key hypotheses for both distinguishers. In fact, when the attack succeeds, the *predicted key* is the actual *secret key*, as the partitioning of traces is sound. The *secret key* achieves a Guessing entropy of zero when the attack succeeds. This is not the case for *false keys* which should have an unstable (random) rank.

[1]The *predicted key* is also known as *the best key*.

Combination Formula

Consider two side-channel attacks, sca_1 and sca_2, that verify the empirical observations mentioned before. Those attacks are combined by taking into account the scores given by both distinguishers for the same key hypothesis, denoted by $\Delta_i(k)$, $i \in 1, 2$ for a given hypothesis k. Aggregate functions noted Ψ are used for the combination. In this case, both the the $Max()$ and $Sum()$ functions are considered. Similarly, Gini correlation can be imagined to use the *ratio* as an aggregate function.

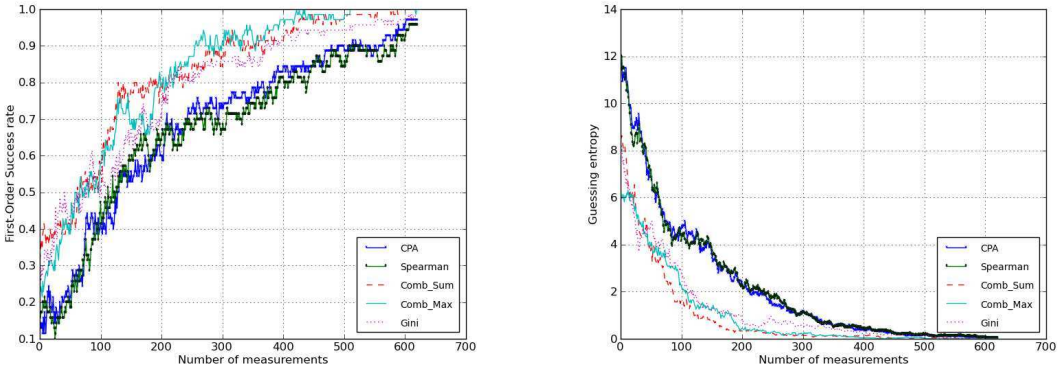
For each key hypothesis k, a new score is generated by computing $\Psi(\Delta_1(k), \Delta_2(k))$.

7.3.4 Experimental Results and Discussion

The measurement setup is similar to the one of Section 7.2.2. The attacks are performed on the same unprotected DES implementation, by recording 5000 side-channel measurements (averaged 256 times).

The analysis of the marginal distributions of both the prediction X and the measurement Y, revealed that their joint distribution is not perfectly Gaussian. In fact, there is some deviation from the bivariate normal assumption; and therefore the Pearson correlation coefficient ρ might not be optimal. Moreover, Spearman correlation coefficient r might not be optimal too, because X takes a small number of different values which does not allow a reliable approximation by normal distribution. As stated before, this might create a problem of ties, which considerably affects the quality of r. The conducted experiment involves five side-channel attacks evaluated in terms of first-order success rate (SR) and Guessing entropy (GE) security metrics: correlation power analysis (CPA), Spearman rank correlation, Gini correlation and two empirical combination attacks. The combined attacks using the $Sum()$ and $Max()$ as aggregate functions are denoted by $Comb_{Sum}$ and $Comb_{Max}$ respectively.

According to Figure 7.10 (a) and Figure 7.10 (b), CPA and Spearman attacks have similar behaviours. This confirms our previous empirical statements. Clearly, the combined attacks (Gini Correlation, $Comb_{Max}$ and $Comb_{Sum}$) outperform CPA and Spearman attacks. As a matter of fact, for a SR threshold fixed at 80%, the number of traces needed to succeed the combined attacks is around *200 traces*. CPA and Spearman attacks need much more traces to do so (*400 traces*) and thus the gain is about 50%. Unsurprisingly, the GE metric shows a superior efficiency for the combined attacks as the rank of the *secret key* converges more rapidly toward the best rank than CPA and Spearman attacks. Besides, for both metrics, Gini correlation is slightly less efficient than the empirical combinations, $Comb_{Sum}$ and $Comb_{Max}$. Let S_1, S_2 be two inputs of aggregate function with respective noise of standard deviation σ_1, σ_2. The signal-to-noise ratio (SNR) of the $Comb_{Sum}$ is $(S1 + S2)/\sqrt{{\sigma_1}^2 + {\sigma_2}^2}$. When S_1 and S_2 are equal, the SNR of combination using $Comb_{Sum}$ is increased by $\sqrt{2}$. Similarly, the increase in SNR when two distinguishers are combined using $Comb_{Max}$ can be computed. However, Gini Correlation is more generic and might be more suitable in other empirical circumstances.



(a) (b)

Figure 7.10: CPA, Spearman vs Combination: (a) Success Rate and (b) Guessing Entropy.

*

In recent cryptographic applications, the scarce resource when performing an SCA is often the number of side-channel measurements. Therefore this Section presented a methodology to combined classical side-channel distinguishers, in order to improve those attacks in terms of MTD. Both theoretical (Gini) and empirical (aggregate functions) approaches were considered and practical attacks on an FPGA implementation of DES resulted in a gain of roughly 50% MTD.

Depending on the target, different distinguishers can be combined using appropriate aggregate functions. Therefore future research could focus on studying new combinations of different distinguishers (potentially more than two), customised for the target and taking into account possible presence of SCA countermeasures.

7.4 Rank Corrector

As stated in Section 2.1, statistical attacks mainly consist of two independent steps:

1. A partitioning of the side-channel observations based on key hypotheses.
2. A sorting and ranking of those partitionings by a distinguisher, that selects the most relevant amongst all the secret hypotheses.

Since the introduction of DPA by Kocher, numerous distinguishers have been developed, some being more or less efficient depending on several criterion like the target algorithm, architecture and implementation. Studies by Gierlichs and al. [63] even show that some distinguishers are better for the first order success rate and others are better for the guessing entropy. Nonetheless, few papers have tried to devise generic methods to improve known SCAs.

The *Rank Corrector* is such a scheme, an algorithm which aims at enhancing existing attacks, independently of distinguishers or implementations.

7.4.1 Background

As stated in Section 2.1, the main difference regarding most SCAs relies on the measurements partitioning process and the used distinguisher. Otherwise they usually run iteratively and a new ranking of all secret hypotheses is created at each iteration. When a key hypothesis is ranked first for a certain amount of iterations, it is returned by the SCA software and the attack is considered successful when it is, indeed, the actual secret.

Notations

The following notations are used in the remainder of this Section:

- RC is the *rank corrector*.
- SK is the secret key.
- PK, the predicted key, is the key hypothesis which has the best rank for the current iteration. The value of PK is updated for each new observation.
- PK_i denotes the predicted key at iteration i.
- FK represents a false key hypothesis (all but the secret key).
- R_k, $R_{k,i}$ are respectively the ranks of key hypothesis k for the current iteration and iteration i.
- S_{init} is the iteration number corresponding to the beginning of stability for a given PK.
- $Trace$ denotes a power or electro-magnetic measurement.
- MTD, or Measurement To Disclosure, is the total number of traces needed to successfully perform the attack.

Key rank behaviours

In theory, for a great number of observations, SK should always be ranked first, when the partitioning of traces is sound. This is not the case for FKs which should have an unstable (random) rank. However, actual attacks are usually performed with a limited number of measurements or aim, at least, to be successful using as few of them as possible. Therefore, the behaviours of the ranks of both the secret key and false key hypotheses were studied, by performing numerous DPAs and CPAs on four different architectures of DES and three of AES, implemented Altera Stratix-II and Xilinx

Virtex-II FPGAs. The goal of this study was to find an empirical method taking advantage of the distinctive behaviours between SK and the FKs.

First of all, we observed that R_{SK} is always roughly decreasing until it reaches first position, considering that the best rank is 0. Figure 7.11 shows examples of such behaviour during a Differential Power Analysis (DPA) on 6 bits of the first S-Box of a DES coprocessor (a) and a Correlation Power Analysis (CPA) on 8 bits of the first S-Box of an AES 256 (b), both implemented on FPGA. While in Figure 7.11(a) R_{SK} decreases almost monotonically, in Figure 7.11(b) it oscillates much more while doing so. Then, in both cases, R_{SK} clearly fluctuates within a short range of the first positions before definitely stabilising. This behaviour is observed most of the time, however, in rare cases, R_{SK} can stabilise as soon as it reaches the first position, without fluctuating.

Regarding false keys, we observed that, as the number of processed traces increases, they clearly tend to display more random behaviours. Figure 7.12 shows two examples of false key rank evolution during the same DPA as Figure 7.11(a). On the one hand, the leftmost one ($FK1$) is almost random and never ranks first, thus will not be treated as a potential secret key. This type of behaviour is easily differentiable from SK. On the another hand, the rank of the rightmost key ($FK2$) does reach the first position at some point and could therefore be concurrent to SK. However, with the increasing iteration number, R_{FK2} clearly raises, which would not be the case for SK.

In conclusion, our study shows that it should indeed be possible to differentiate between SK and the FKs based on the observation of their ranking. As a matter of fact, R_{SK} roughly decreases and then usually fluctuates between a few positions before stabilising, whereas the R_{FKs}, when they reach the first rank, usually become random a few iterations later.

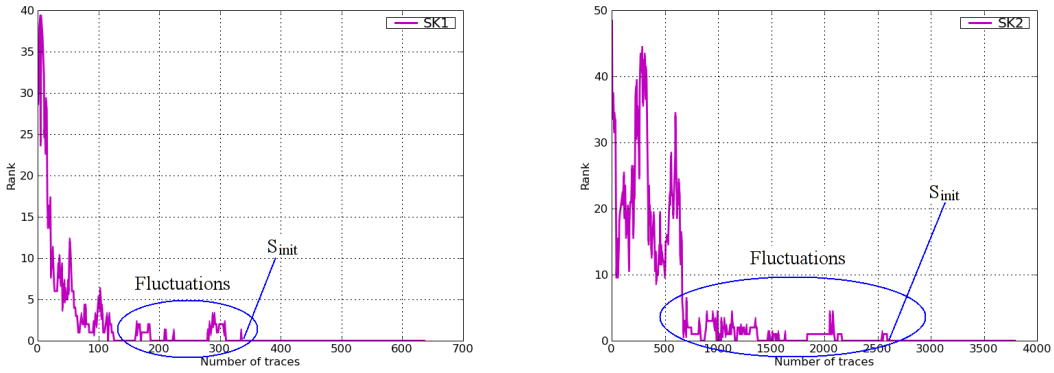
7.4.2 Application field

The main feature of the rank corrector is to be as generic as possible. It can therefore be applied to a wide range of attacks. Indeed, an attack scheme only needs to meet three requirements to be compatible, with this software:

1. It has to be iterative (for instance DPA iterates on a number of power consumption measurements).

2. A ranking of all key hypotheses must be produced at each iteration.

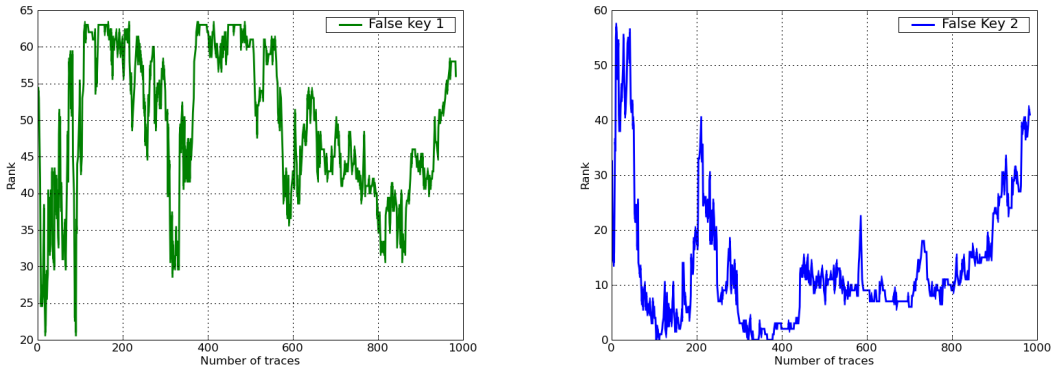
3. The SCA software must decide on the secret key by observing the stability of the first ranked hypothesis.



(a) Rank of SK during a DPA on DES

(b) Rank of SK during a CPA on AES

Figure 7.11: Examples of rank behaviours for the secret key.



(a)

(b)

Figure 7.12: Examples of rank behaviours for false keys.

Up to now, the usual criterion employed to decide on SK is, indeed, the stability of the rank. For example, in the first edition of DPA contest [174], a stability of 100 traces had to be achieved in order to validate the attacks, performed on an unprotected DES cryptoprocessor. Moreover, most of the passive SCAs, usually based on the exploitation of power consumption or electromagnetic measurements, present an iterative behaviour as, at some point, they process those traces one after the other.

Thereby, the rank corrector can be used to enhance a very large number of attacks, like DPA, CPA or MIA.

7.4.3 Basic Principle

The rank corrector (RC) is a **generic**[1] custom-made algorithm, which aims at exploiting the key behaviours described in Section 7.4.1 in order to significantly reduce the number of traces needed to achieve a successful SCA.

As a matter of fact, it studies in real-time, the evolution of an iterative ranking (for instance the one produced by a DPA software), in order to virtually reassign previous rank positions to the current PK, depending on past and current rankings. The detailed algorithm is described in Section 7.4.5. It is totally independent of the attack, given that it verifies the requirements described in section 7.4.2. Indeed, it only modifies, on the fly, the stability of the target SCA.

Eventually, RC can be seen as a plug-in, designed to enhance most existing SCAs.

Let's uppose that a DPA is performed on one sub-key of a cryptographic device implementing an algorithm like DES or AES and that it will be successful, meaning that SK will eventually be ranked first and reach the given stability. Before stabilizing, R_{SK} should be roughly decreasing (as stated in Section 7.4.1). Then most of the time, after reaching the first position ($rank = 0$), it will fluctuate within a short range and then stabilize, from S_{init} to MTD.

In this case, RC will detect those fluctuations in the proximity of the stabilisation, remove them and increase the stability counter by an equal amount (G) of traces. Thereby, G represents the gain of RC with regards to a simple DPA. Figure 7.13 illustrates this scenario, by showing a zoom of the evolution of the R_{SK} displayed in Figure 7.11(a), with and without the *rank corrector*. As we can be observed in this example, without RC the stability starts after 340 traces, whereas with RC, it does after only 240 traces. Thus inducing a gain of 100 traces.

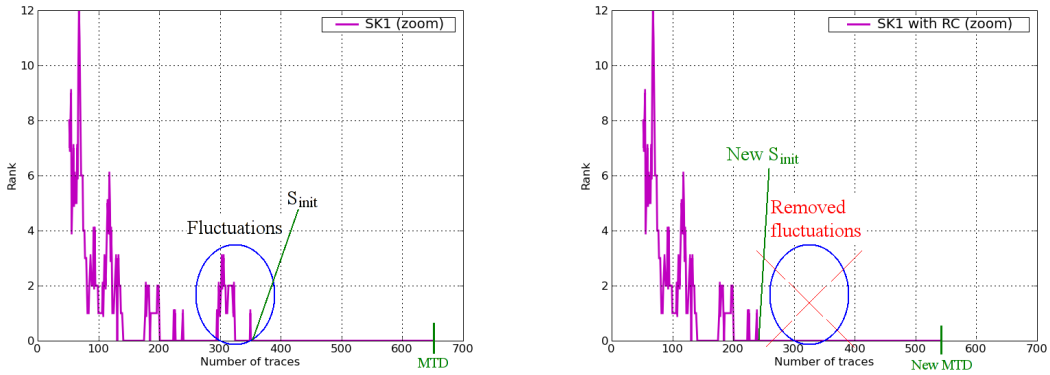
The main idea of RC is to locate and disqualify FKs that are ranked first during the fluctuations of SK. Therefore, it proceeds as follows:

1. When a PK has been stable for certain amount of traces STH (stability threshold), starting at S_{init}, it is considered as a potential SK, as shown in Figure 7.14(a).

2. Then RC scans a small range of traces before S_{init} (called *correction range*) searching for fluctuations of the current PK (let's call it CPK). If they exceed a certain limit R_{max}, CPK is disqualified and will no longer be a candidate for SK.

3. In the other case, RC will check the ranks, at the current iteration, of all other PKs present in the correction range and discard all those that show a rank exceeding R_{max}. Then the rank of SK, within the correction range, is modified by

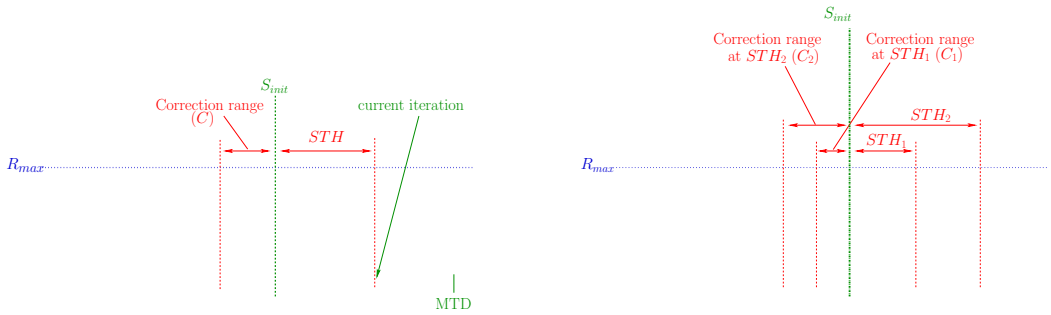
[1] We insist that our methodology does not consist in trying to tune an attack on a given acquisition campaign so that it retrieves the key as fast as possible, as in [108]. Instead, we attempt to pre-characterize a set of parameters from a training campaign, and to use this prior knowledge subsequently in a positive view to speed up forthcoming attacks.



(a) Basic DPA. (b) DPA with RC.

Figure 7.13: Rank of SK during a DPA, with and without RC.

removing the discarded keys.



(a) RC principle. (b) RC threshold mechanism.

Figure 7.14: Illustration of RC principle.

Moreover, RC operates by using increasing values of STH. Each time the stability of PK reaches a given STH_n (with $n \in \mathbb{N}^*$), RC is launched. This threshold mechanism was chosen for two reasons. On one hand it allows RC to easily discard any PK that isn't stable for at least STH_1 and only take into consideration the potential SKs. Moreover, for each new threshold, the correction range (i.e. the potential gain) increases, as shown in Figure 7.14(b) which is coherent with the fact that the more stable a PK is, the more likely it is to be SK. On another hand, it keeps the computation time of the attack close to the original one as RC is only called a few times.

7.4.4 RC Parameters and their evaluation

In order to be as generic as possible, RC was designed as a **parametric** algorithm. It is thereby based on two main parameters, that allow it to adapt to almost any attack, independently of any bias introduced by either the architecture, the implementation, or the acquisition process:

1. S is the minimum stability required to ensure that the attack will always decide on the actual secret key (SK).

2. R_{max} is the maximum fluctuation range of SK before the stabilization (i.e. between the first time SK attains the first position and its stabilization).

These two parameters must be correctly evaluated for RC to work properly. For instance choosing S too small, could easily lead any SCA to decide on a false key and using RC in this context would clearly increase the probability of doing so.

For this purpose, we use a clone device and undergo a profiling phase, computing multiple attacks for different SK. For each key and each attack, the evolution of the ranks of SK (R_{SK}) and every FK is recorded, in order to determine the best R_{max} and S. Indeed, the more representative these two parameters are of SK (and not of any FK), the more efficient RC will be, as it will be able to disregard more FKs and almost only consider the actual SK as a correction target.

Eventually, this profiling phase ensures that RC will not lead to finding a false key.

Aside from R_{max} and S, RC takes into account a few other secondary parameters: while these parameters do influence the maximum gain of RC, they do not have a major impact on the overall results. Therefore and in order to simplify the notations they will be fixed, in the following, to the values used during our experiments.

1. n: the number of thresholds, $1 \leq n \leq 3$.

2. STH_n: the n^{th} stability threshold, $STH_n = (n * S)/4$.

3. C_n: the n^{th} correction range, $C_n = STH_n/2$.

Those empirical values were deduced from thorough studies on several cryptographic devices. Naturally, they may not be optimal for all SCAs and all implementations and a finer study should be carried out, using the clone device, before every specific attack.

7.4.5 Description of the algorithm

Algorithm 18 gives a detailed description of RC. When launched, it starts by searching for an occurrence of the current PK (CPK), in the first rank of the C_n iterations before S_{init} (the current iteration number being $CIT = S_{init} + STH_n$). The search starts at $S_{init} - C_n$ down to $S_{init} - 1$ (step 2 and 3 of our algorithm). Finding CPK at iteration

IT, means that R_{CPK} did actually reach first position and fluctuate before stabilizing, meaning it is, as such, open to correction by RC (step 4).

All PKs between S_{init} and IT are then checked in the reverse order (from S_{init} to IT) and reassigned to CPK when possible (step 5). This way, whenever a rank that should not be corrected is found, RC is stopped. Several scenarios can then occur: the trivial one is when $PK_j = CPK$, with $j \in S_{init}\ to\ IT$ (though it is never true for $j = S_{init} - 1$). In this case, RC directly increases the stability by one iteration. When $PK_j \neq CPK$ (step 9), RC will look at $R_{CPK,j}$. If $R_{CPK,j} < R_{max}$ (i.e. $R_{CPK,j}$ is near the first position), that means CPK is a possible candidate to be SK (step 10). RC then checks if $R_{PK_j,CIT} \geq R_{max}$ and if this second condition is verified, PK_j is disqualified as a potential SK and removed from the ranking (step 12 and 13). This check is mandatory, as, for the first thresholds, CPK could be a false key, in which case the real SK is likely to be one of the PK_j, with $j < S_{init}$. This step is repeated until $CPK = PK_j$ or a PK_j that cannot be disqualified ($R_{PK_j,CIT} < R_{max}$) is found. Indeed, when $R_{PK_j,CIT} < R_{max}$ our algorithm considers PK_j to be a possible SK and thus no correction is made. Then, if $CPK = PK_j$, the stability is once again increased (step 15). As a matter of fact we suppose, based on the observations of Section 7.4.1 and the profiled value of R_{max} that R_{SK} will never go past R_{max} once it has been ranked first. Thus any PK that goes past R_{max} is definitively discarded.

These steps are repeated for each threshold and each time the number of traces that might be corrected increases. As a matter of fact, the more stable a PK is, the more likely it is to be SK. Moreover, a FK that was not corrected at the first threshold, (for instance because it was ranked second), will usually not be in the same position at the second threshold and will then be replaced by PK.

Consequently, the maximum gain of RC can be computed as shown in Equation 7.9:

$$GAIN_{max} = \sum_{n=1}^{3} \frac{STH_n}{2} \equiv \sum_{n=1}^{3} \frac{n * S}{8} \tag{7.9}$$

7.4.6 Example

Figure 7.15 illustrates the evolution of key ranks during an SCA using RC, when the stability of a given key reaches the first threshold. K represents our secret key and S_{init} the iteration number marking the beginning of its stability, while K_0, K_1 and K_2 are three false keys. The process of RC can be described in three steps:

1. The rank of K (R_K) reaches the first threshold (i.e. a stability of $STH_1 = S/4$ traces), thus RC searches for K in the PKs of the $S/8$ prior traces. It is found at iteration IT, implying that R_K did actually reach rank 0 and fluctuate before stabilizing.

Algorithm 18 RC detailed algorithm.

```
for each threshold STH_n (n ∈ 1,2,3) do
  for iteration in S_init − C_n to S_init do
    Search for an occurrence of the current CPK.
    if CPK is found at iteration = IT then
      for j in S_init to IT do
        Check the value of PK_j
        if PK_j = CPK then
          increase stability by 1
        else
          if R_CPK,j < R_max then
            while CPK ≠ PK_j and R_PKj,CIT ≥ R_max do
              Remove PK_j from the ranking
            if CPK = PK_j then
              increase stability by 1
          else
            Exit.
    else
      Exit.
return stability
```

2. Two different PKs, K_1 and K_2 are found respectively at iteration $S_{init} - 1$ and $S_{init} - 3$. For those iterations R_K is compared to R_{max}. As $R_K < R_{max}$ is true in both cases, RC enters the next step of the algorithm and checks the ranks of K_1 and K_2 at the current iteration $CIT = S_{init} + S/4$.

3. $R_{K_2,CIT}$ is greater than R_{max}, so K_2 is discarded. Then, K which was ranked second, becomes the new PK of iteration $S_{init} - 1$ and the stability is increased. K is already ranked first at iteration $S_{init} - 2$ so the stability is once again increased. $R_{K_1,CIT}$, on another hand, does not verify the condition, meaning it is a possible candidate for SK and RC is therefore stopped.

In this example RC produced a gain of 2 traces after the first threshold and S_{init} is thereby updated as shown in Figure 7.15.

7.4.7 Optimization

Although most of the time SK does fluctuate before stabilizing, there are rare cases where it permanently stabilizes as soon as it reaches the first position (this occurred in less than 4% of all the attacks performed during our study). In this case, the gain should be null. In order to decrease the probability of having such a gain, we take advantage of the fact that R_{SK} is likely to be $< R_{max}$ just before stabilizing. Thus RC will search for occurrences of SK in those ranks, before the stabilization and try

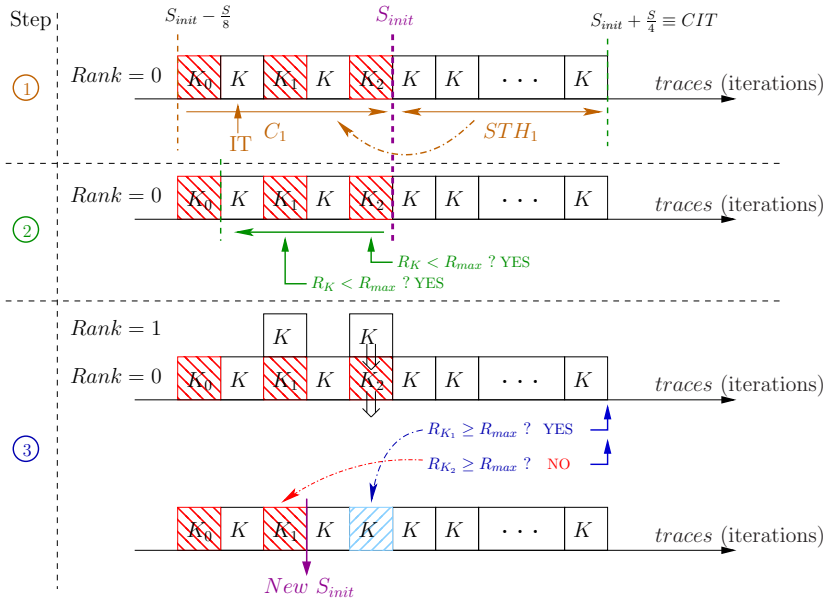


Figure 7.15: Illustration of an SCA using RC, at the first threshold.

to disqualify the corresponding PKs, in order to reassign SK to the first rank. This new search range, called $C2_n$, is another secondary parameter (like n and C_n) and will also be fixed, in the following, to our experimental value: $C2_n = STH_n/4$ (see Section 7.4.4).

Algorithm 18 is then complemented as follows: step 18 is replaced by Algorithm 19. Then when CPK is not present in the first search range (step 17 of Algorithm 18) RC will check if $R_{CPK} < R_{max}$ for a smaller range of iterations: from $S_{init} - C2_n$ to S_{init}. The following process is similar to the former one, if all PKs can be disqualified ($R_{PK_k,CIT} \geq R_{max}$), CPK becomes PK_k and the stability is increased.

Algorithm 19 RC optimization.

```
1: if CPK is not found then
2:   for k in S_init to S_init − I2_n do
3:     if R_CPK < R_max then
4:       while CPK ≠ PK_j and R_PKj,CIT ≥ R_max do
5:         Remove PK_j from the ranking
6:       if CPK = PK_j then
7:         increase stability by 1
8:     else
9:       Exit
```

Obviously SK will not always display such a behaviour and there will thus be

cases where the gain is null. A more thorough study of these situations could be undertaken in the future, in order to improve the efficiency of this algorithm.

7.4.8 Experimental Results

Our experiments were once again conducted on StratixII FPGAs, soldered on two SASEBO-B boards (one for the actual attack an one for the clone device), with the same experimental setup as in Section 4.2.2. The target cryptoprocessor implemented in those devices is an unprotected DES.

First of all, the parameters (S and R_{max}) were estimated with a profiling phase on the first FPGA. Using three different keys, 100 attacks were performed for each one. Eventually, $S = 110$ and $R_{max} = 5$ were deduced as the optimal values for these parameters.

Then, 50000 traces were acquired in order to perform several DPAs on the real device, with and without using the *rank corrector*.

In order to better assess the efficiency of this algorithm, the first-order success rate [168] and guessing entropy are computed and results are respectively displayed in Figure 7.16 and Figure 7.17.

The improvement brought by this scheme is clearly visible on the curve of the success rate. For instance a success rate of 80% is reached with less than 90 traces with RC, when the basic DPA needs more than 110 traces to do so.

While the guessing entropy is also always lower with RC than without, the gap between the two is definitely thinner than for the success rate. This is explained by the fact that the correction takes place when the rank of SK is lower than 5, thus when computing the mean of ranks on a large number of attacks, it does not have a great impact on the final results.

Moreover, after 300 complete attacks on random pools of side-channel measurements, a mean gain of 43.7 traces is obtained. Considering that, as shown in Figure 7.16, the basic DPA requires 250 to 300 traces to complete the attack, using RC evetually results in a gain of $\sim 15\%$ in terms of MTD.

*

To conclude, the *Rank Corrector* algorithm presented in this Section, is not an actual Side-Channel Attack, but rather a method to enhance most SCAs, namely reducing their number of MTDs and improving their first-order success rate. RC is based on the principle that the ranking evolution of the secret key during an SCA, is distinguishable from those of the false key hypotheses and that it can be exploited in order to accelerate the attack. This specific behaviour has been empirically demonstrated on several different SCAs and target cryptographic devices.

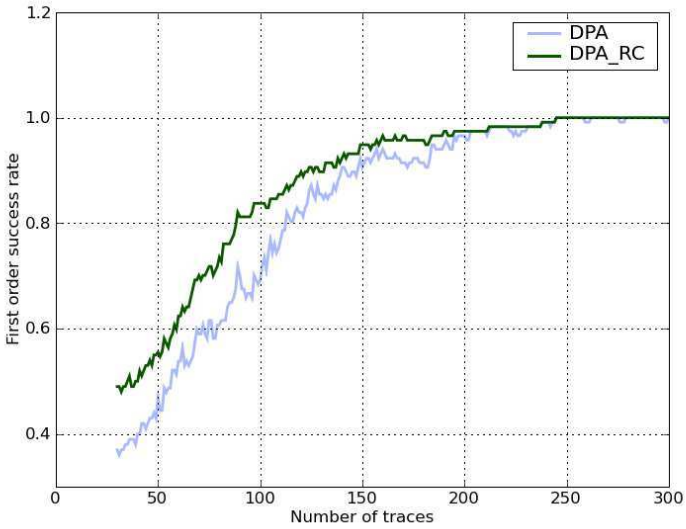


Figure 7.16: First-order success rate for DPA with and without RC.

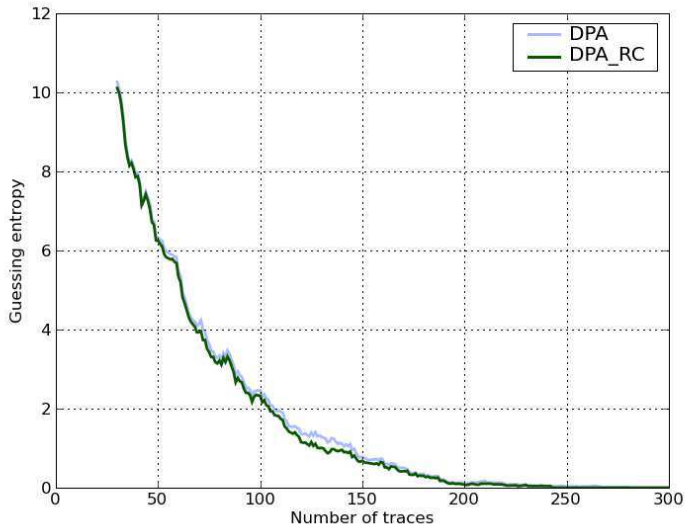


Figure 7.17: Guessing entropy for DPA with and without RC.

Two parameters which are the threshold of stability and the maximum fluctuation range of the secret key are mandatory for RC to work efficiently. When these parameters are well profiled a gain of at least 15% in terms of MTD can be expected with regards to the classical SCA.

As perspectives to enhance the RC it should be interesting to refine the rank evo-

lution analysis, by considering the evolution of the values obtained by the SCA, in addition to the rankings. A theoretical validation based on a study of the probability distribution evolution is also foreseen to prove and improve the efficiency of this algorithm.

7.5 Conclusion

Since the introduction of the power analysis concept, a wealth of SCA have been developed, usually in the goal of evaluating the robustness of cryptographic algorithm with and without countermeasures. Measuring the strength of those attacks is not a trivial task. As of now such assessment is generally performed using the success rate and guessing entropy metrics or more classically the number of MTD (which is well illustrated by the annual DPA CONTEST [174]). Moreover, in real attack situations, the critical resource is likely to be the available number of side-channel measurements. In this context, we described three novel SCA ideas aiming at reducing the number of MTD. First, using the PCA as a new distinguisher gave rise to the FPCA. Actual experiments showed its efficiency on both a regular and masked implementation of DES, with better results in terms of success rate and guessing entropy than classical DPA, CPA and VPA. Second, combination techniques of both measurements and distinguishers were presented and resulted in clear improvements with regards to the corresponding single schemes. Finally a SCA enhancement algorithm so-called *rank corrector* (RC) was proposed. By detecting the particular behaviour of the secret key during most classical SCAs, it is able to improve the success rate of the attack and decrease the number of MTD.

Chapter 8

Conclusion and Perspectives

8.1 Concluding Remarks

This work focuses on the investigation and development of SCA countermeasures for industrial applications, presenting optimized trade-offs between security level and resource consumption. Two types of countermeasures are commonly deployed to protect cryptographic algorithms against such schemes, namely information hiding (i.e. DPLs) or masking. A study on the advantages and weaknesses of both methods is given, focusing on the particular issue of FPGA implementation. Concerning DPLs, two major vulnerability are put to light, namely the *early propagation effect* and *technological bias*. A practical evaluation of those weaknesses is performed on a 3DES by exploiting post place and route timing simulations, resulting in the first published attack on WDDL implemented on FPGA. This evaluation is then taken advantage of to present three novel countermeasures in the context of symmetrical algorithms. First, specific placement and routing strategies are deployed in the goal of raising the security level of classical DPLs, like WDDL, when implemented on FPGA. Although such technique allows a visible increase in the number of MTD by reducing the *technological bias*, it seems insufficient by itself to provide a satisfying level of security against statistical SCAs. Second, a novel DPL style so-called BCDL is presented, which make use of a particular synchronization scheme to remove the *early propagation effect* and reduce the implementation cost of the S-Boxes. MIM as well as a full CPA on all bits are performed to experimentally evaluate its robustness and a global comparison is drawn between most existing DPLs. Results show that although a few bits of BCDL still show relatively high leakage levels, this countermeasure remains a very interesting trade-off between resource consumption and security against both statistical SCAs and perturbation attacks. Finally a new masking scheme so-called RSM is proposed. By considering a fixed set of masks rather than fully random ones, RSM can be implemented with significantly fewer resources than any other masking schemes, all the more on FPGA. Moreover its high security level is demonstrated by thorough experimental and theoretical studies. Given that the masks are properly chosen, RSM proves

to be robust against classical first-order SCAs, as well as second-order zero-offset CPA and VPA. By contrast, for asymmetrical algorithms, reasonable trade-offs between security and implementation cost are possible with existing countermeasures. Therefore an ECC coprocessor as well as two classical countermeasures are implemented and confronted to SPA and *doubling attacks*. Results show that no unexpected vulnerability or resource consumption are induced by their implementation on FPGA. Eventually three new SCA schemes are presented. The FPCA exploits PCA as a SCA distinguisher, leading to improved results, in terms of success rate, than classical attacks (DPA, CPA, VPA). Moreover it is proven efficient against both unprotected and masked implementations of DES. Combination methods are also described as a way to create powerful SCAs and reduce the required number of MTD. Techniques to combine both measurements and distinguishers are dealt with, resulting in significant improvements in terms of success rate and MTD. The *rank corrector* algorithm is finally proposed as a tool to enhance existing SCA. Its parametric property allows the rank corrector to be compatible with most classical SCA like DPA, CPA or MIA. Experiments conducted with DPAs on DES show a gain of 15% in terms of MTD as well as an improvement of the success rate.

8.2 Perspectives

The first perspective of this work is to enhance the place and route strategy of BCDL. In that regard a systematic protocol to pinpoint the few highly leaking bits should be developed. Then specific placement and routing assignments could be deployed to remove or at least lessen those leakages. Such back-end constraints may be added in a similar way of those presented in Section 4.2.2. Another possibility would be to focus on Xilinx FPGAs which seem to provide the designer with more accurate tools to perform such tasks.

Additional research may also be undertaken concerning the RSM, on one hand to tune its implementation for different devices. For instance recent FPGA technologies, embedding large RAM block should provide to possibility to include several masks sets for almost no additional cost. On the other hand the security evaluation could be further deepened, to ensure its robustness against newly developed SCAs. Another interesting issue would be to seek customized fault detection mechanisms for this countermeasure.

The attack field eventually offers many perspectives. Further work on PCA and other multivariate data analytic tools could be undertaken to create new distinguishers. A deeper theoretical and experimental analysis of the *rank corrector* may allow optimizations of this scheme in terms of efficiency and genericity. Finally the concept of combined SCA opens a great number of doors to future work, as a wide range of SCA combinations (in both measurement technique, distinguishers and SCA types) may potentially lead to powerful new attacks.

List of Publications

[A] Sylvain Guilley, Sumanta Chaudhuri, Laurent Sauvage, Tarik Graba, Jean-Luc Danger, Philippe Hoogvorst, Ving-Nga Vong, Maxime Nassar, Florent Flament **Shall we trust WDDL?.** In Future of Trust in Computing 2008, Berlin, Germany.

[B] Sylvain Guilley, Sumanta Chaudhuri, Laurent Sauvage, Tarik Graba, Jean-Luc Danger, Philippe Hoogvorst, Vinh-Nga Vong, Maxime Nassar **Place-and-Route Impact on the Security of DPL Designs in FPGAs.** In HOST 2008, Anaheim, USA.

[C] Laurent Sauvage, Maxime Nassar, Sylvain Guilley, Florent Flament, Jean-Luc Danger and Yves Mathieu **DPL on Stratix II FPGA: What to Expect?.** In ReConFig 2009, IEEE Computer Society, Cancún, Quintana Roo, México.

[D] Laurent Sauvage and Sylvain Guilley and Jean-Luc Danger and Yves Mathieu and Maxime Nassar **Successful Attack on an FPGA-based Automatically Placed and Routed WDDL+ Crypto Processor.** In DATE 2009, April 20–24, Nice, France.

[E] Shivam Bhasin, Jean-luc Danger, Florent Flament, Tarik Graba, Sylvain Guilley, Yves Mathieu, Maxime Nassar, Laurent Sauvage and Nidhal Selmane. **Combined SCA and DFA Countermeasures Integrable in a FPGA Design Flow.** In ReConFig, pages 213-218. IEEE Computer Society, December 9-11 2009. Cancun, Mexico.

[F] Maxime Nassar, Shivam Bhasin, Jean-luc Danger, Guillaume Duc, and Sylvain Guilley. **BCDL: A high performance balanced DPL with global pre–charge and without early-evaluation.** In DATE 10, pages 849-854. IEEE Computer Society, March 8-12 2010. Dresden, Germany.

[G] Youssef Souissi, Maxime Nassar, Sylvain Guilley, Jean-Luc Danger et Flament Florent **First Principal Components Analysis: A New Side Channel Distinguisher.** In ICISC 2010 LNCS 14th Annual International Conference on Information Security and Cryptology.

[H] Youssef Souissi and Jean-Luc Danger and Sami Mekki and Sylvain Guilley and Maxime Nassar **Techniques for electromagnetic attacks enhancement.** In DTIS (Design & Technologies of Integrated Systems) 2010, March 23-25.

[I] Laurent Sauvage and Maxime Nassar and Sylvain Guilley and Florent Flament and Jean-Luc Danger and Yves Mathieu **Exploiting Dual-Output Programmable Blocks to Balance Secure Dual-Rail Logics.** In International Journal of Reconfigurable Computing, Hindawi, 2010.

[J] Youssef Souissi, Maxime Nassar, Sylvain Guilley, Shivam Bhasin and Jean-luc Danger. **Embedded Systems Security: An Evaluation Methodology Against Side Channel Attacks.** In DASIP. IEEE Computer Society, Nov 2-4 2011. Tampere, Finland.

[K] Nicolas Debande, Youssef Souissi, Sylvain Guilley, Jean-Luc Danger, Maxime Nassar et Thanh-Ha Le. **"Re-Synchronization by Moments": An Efficient Solution to Align Side-Channel Traces.** In WIFS 2011 IEEE International Workshop on Information Forensics and Security.

[L] Maxime Nassar, Youssef Souissi, Sylvain Guilley et Jean-Luc Danger. **"Rank Correction": A New Side-Channel Approach For Secret Key Recovery**. In Info Sec HiComNet 2011 International Conference.

[M] Youssef Souissi, Sylvain Guilley, Maxime Nassar, Shivam Bhasin et Jean-Luc Danger. **"Time-Success rate" as a new security metric for Side-Channel Analysis.** In Poster Session of CHES 2011.

[N] Youssef Souissi, Shivam Bhasin, Maxime Nassar, Sylvain Guilley et Jean-Luc Danger. **Combination of Measurements to accelerate side channel attacks.** In Poster Session of CHES 2011.

[O] Sylvain Guilley, Guillaume Duc, Ph. Hoogvorst, Moulay aziz Elaabid, Shivam Bhasin, Youssef Souissi, Nicolas Debande, Laurent Sauvage et Jean-Luc Danger. **Vade Mecum on Side-Channels Attacks and Countermeasures for the designer and the Evaluator.** In DTIS 2011 Design & Technology of Integrated Systems in nanoscale era.

[P] Nicolas Debande, Youssef Souissi, Maxime Nassar, Sylvain Guilley, Thanh-Ha Le et Jean-Luc Danger. **Side Channel Analysis enhancement: A proposition for measurements resynchronisation.** In CryptArchi Workshop 2011.

[Q] Maxime Nassar and 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, December 11-14, pages 22-39.

[R] Youssef Souissi, Shivam Bhasin, Maxime Nassar, Sylvain Guilley and Jean-luc Danger. **Towards Different Flavors of Combined Side-Channel Attacks.** In RSA Cryptographers Track, CT-RSA, LNCS, To Appear. Springer, Feb 27-March 2 2012. San Francisco, CA, USA.

[S] Maxime Nassar and Sylvain Guilley and Jean-Luc Danger and Youssef Souissi **RSM: a Small and Fast Countermeasure for AES, Secure against First- and Second-order Zero-Offset SCAs.** In DATE 2012, March 12-16, Dresden, Germany.

Bibliography

[1] Common Criteria (*aka* CC) for Information Technology Security Evaluation (ISO/IEC 15408).
Website: http://www.commoncriteriaportal.org/. 105

[2] Moulay Abdelaziz El Aabid, Sylvain Guilley, and Philippe Hoogvorst. Template Attacks with a Power Model. Cryptology ePrint Archive, Report 2007/443, December 2007. http://eprint.iacr.org/2007/443/. 9

[3] Moulay Abdelaziz El Aabid, Oliver Meynard, Sylvain Guilley, and Jean-Luc Danger. Combined Side-Channel Attacks. In *WISA*, volume 6513 of *LNCS*, pages 175–190. Springer, August 24-26 2010. Jeju Island, Korea. DOI: 10.1007/978-3-642-17955-6_13. 138

[4] Agilent Technologies: http://www.agilent.com/. 56, 102

[5] Toru Akishita and Tsuyoshi Takagi. Zero-Value Point Attacks on Elliptic Curve Cryptosystem. *In Proc. Information Security - ISC, LNCS*, page 218233, 2003. 14

[6] Mehdi-Laurent Akkar, Régis Bevan, and Louis Goubin. Two power analysis attacks against one-mask methods. In *Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers*, pages 332–347, 2004. 22

[7] Mehdi-Laurent Akkar and Christophe Giraud. An Implementation of DES and AES Secure against Some Attacks. In LNCS, editor, *Proceedings of CHES'01*, volume 2162 of *LNCS*, pages 309–318. Springer, May 2001. Paris, France. 19, 20

[8] Mehdi-Laurent Akkar and Louis Goubin. A generic protection against High-order differential Power Analysis. In LNCS, editor, *Proceedings of FSE'03*, volume 2887 of *LNCS*, pages 192–205. Springer, 2003. Berlin, Germany. 22

[9] Pierre alain Fouque and Frederic Valette. The doubling attack why upwards is better than. In *Downwards, Workshop on Cryptographic Hardware and Embedded Systems 2003 (CHES 2003), LNCS 2779*, pages 269–280. Springer-Verlag, 2003. 39

[10] Altera FPGA designer. http://www.altera.com/. xi, 63, 64, 80

[11] Cédric Archambeau, Éric Peeters, François-Xavier Standaert, and Jean-Jacques Quisquater. Template Attacks in Principal Subspaces. In *CHES*, volume 4249 of *LNCS*, pages 1–14. Springer, October 10-13 2006. Yokohama, Japan. 9

[12] B.C. Arnold, E. Castillo, and J.M. Sarabia. *Conditional specification of statistical models*. Springer series in statistics. Springer, 1999. 143

[13] Sébastien Aumonier. Generalized Correlation Power Analysis. In *ECRYPT Workshop "Tools for Cryptanalysis"*, 24-25 September 2007. 137

[14] Karthik Baddam and Mark Zwolinski. Divided Backend Duplication Methodology for Balanced Dual Rail Routing. In *CHES*, volume 5154 of *LNCS*, pages 396–410, Washington, DC, USA, aug 2008. Springer. DOI: 10.1007/978-3-540-85053-3_25. 25, 88

[15] Benoît Badrignans, Jean-Luc Danger, Viktor Fischer, Guy Gogniat, and Lionel Torres. *Security Trends for FPGAS – From Secured to Secure Reconfigurable Systems*. Springer, June 20 2011. DOI: 10.1007/978-94-007-1338-3. 18

[16] Hagai Bar-El, Hamid Choukri, David Naccache, Michael Tunstall, and Claire Whelan. The Sorcerer's Apprentice Guide to Fault Attacks. *Proceedings of the IEEE*, 94(2):370 –382, February 2006. 14

[17] Lejla Batina, Benedikt Gierlichs, and Kerstin Lemke-Rust. Comparative Evaluation of Rank Correlation Based DPA on an AES Prototype Chip. In *ISC*, volume 5222 of *Lecture Notes in Computer Science*, pages 341–354. Springer, September 15-18 2008. Taipei, Taiwan. 9, 144

[18] 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. 8, 95, 103

[19] G. Bertoni, L. Breveglieri, I. Koren, and P. Maistri. An efficient hardware-based fault diagnosis scheme for aes: performances and cost. In *Defect and Fault Tolerance in VLSI Systems, 2004. DFT 2004. Proceedings. 19th IEEE International Symposium on*, pages 130 – 138, oct. 2004. 43

[20] Guido Bertoni, Luca Breveglieri, Israel Koren, Paolo Maistri, and Vincenzo Piuri. Error Analysis and Detection Procedures for a Hardware Implementation of the Advanced Encryption Standard. *IEEE Trans. Computers*, 52(4):492–505, 2003. 43

[21] Régis Bevan and Erik Knudsen. Ways to Enhance Differential Power Analysis. In *ICISC*, volume 2587 of *Lecture Notes in Computer Science*, pages 327–342. Springer, November 28-29 2002. Seoul, Korea. 8

[22] Shivam Bhasin, Jean-Luc Danger, Florent Flament, Tarik Graba, Sylvain Guilley, Yves Mathieu, Maxime Nassar, Laurent Sauvage, and Nidhal Selmane. Combined SCA and DFA Countermeasures Integrable in a FPGA Design Flow. In *ReConFig*, pages 213–218. IEEE Computer Society, December 9–11 2009. Cancún, Quintana Roo, México, DOI: 10.1109/ReConFig.2009.50, http://hal.archives-ouvertes.fr/hal-00411843/en/. 78

[23] Shivam Bhasin, Jean-Luc Danger, Tarik Graba, and Sylvain Guilley. How to design BCDL Logic with the best Trade-off between Complexity and Robustness. In *CryptArchi*, Bochum, Germany, June 15–18 2011. Bochum, Germany; (abstract). 89

[24] Shivam Bhasin, Sylvain Guilley, Youssef Souissi, and Jean-Luc Danger. Efficient FPGA Implementation of dual-rail countermeasures using Stochastic Models, September 26-27 2011. Non-Invasive Attack Testing Workshop (NIAT 2011), co-organized by NIST & AIST. Nara, Japan. (PDF). 10

[25] Ingrid Biehl, Bernd Meyer, and Volker Müller. Differential fault attacks on elliptic curve cryptosystems. In *CRYPTO '00: Proceedings of the 20th Annual International Cryptology Conference on Advances in Cryptology*, pages 131–146, London, UK, 2000. Springer-Verlag. 15, 16

[26] Ingrid Biehl, Bernd Meyer, and Volker Müller. Differential fault attacks on elliptic curve cryptosystems. volume 1880 of *Lecture Notes in Computer Science*, pages 131–146. Springer Berlin / Heidelberg, 2000. 15

[27] Eli Biham and Adi Shamir. Differential Fault Analysis of Secret Key Cryptosystems. In *CRYPTO*, volume 1294 of *LNCS*, pages 513–525. Springer, August 1997. Santa Barbara, California, USA. DOI: 10.1007/BFb0052259. 14

[28] Johannes Blömer, Jorge Guajardo, and Volker Krummel. Provably Secure Masking of AES. In LNCS, editor, *Proceedings of SAC'04*, volume 3357, pages 69–83. Springer, August 2004. Waterloo, Canada. 21

[29] Johannes Blömer and Jean-Pierre Seifert. Fault Based Cryptanalysis of the Advanced Encryption Standard (AES) Financial Cryptography. In Rebecca Wright, editor, *Financial Cryptography*, volume 2742 of *Lecture Notes in Computer Science*, pages 162–181. Springer Berlin / Heidelberg, 2003. 14, 15

[30] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the importance of checking cryptographic protocols for faults. In *Proceedings of the 16th annual international conference on Theory and application of cryptographic techniques*, EUROCRYPT'97, pages 37–51, Berlin, Heidelberg, 1997. Springer-Verlag. 15

[31] Arnaud Boscher, Helena Handschuh, and Elena Trichina. Blinded fault resistant exponentiation revisited. *Fault Diagnosis and Tolerance in Cryptography, Workshop on*, 0:3–9, 2009. 38

[32] E. Brier and M. Joye. Weierstrass Elliptic Curves and Side-Channel Attacks. *Public Key Cryptography*, 2274:335, 2002. 34

[33] Éric Brier, Christophe Clavier, and Francis Olivier. Correlation Power Analysis with a Leakage Model. In *CHES*, volume 3156 of *LNCS*, pages 16–29. Springer, August 11–13 2004. Cambridge, MA, USA. 8, 92, 103, 134, 137

[34] Martin Bär, Hermann Drexler, and Jürgen Pulkus. Improved Template Attacks. In *COSADE*, pages 81–89, February 4-5 2010. Darmstadt, Germany. http://cosade2010.cased.de/files/proceedings/cosade2010_paper_14.pdf. 9

[35] D. Canright and Lejla Batina. A very compact "perfectly masked" s-box for aes. In *Proceedings of the 6th international conference on Applied cryptography and network security*, ACNS'08, pages 446–459, Berlin, Heidelberg, 2008. Springer-Verlag. 21

[36] Claude Carlet. Boolean Functions for Cryptography and Error Correcting Codes: Chapter of the monography Boolean Models and Methods in Mathematics, Computer Science, and Engineering. pages 257–397. Cambridge University Press, Y. Crama and P. Hammer eds, 2010. Preliminary version available at (http://www.math.univ-paris13.fr/ carlet/chap-fcts-Bool-corr.pdf). 109, 112

[37] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards Sound Approaches to Counteract Power-Analysis Attacks. In *CRYPTO*, volume 1666 of *LNCS*. Springer, August 15-19 1999. Santa Barbara, CA, USA. ISBN: 3-540-66347-9. 18

[38] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template Attacks. In *CHES*, volume 2523 of *LNCS*, pages 13–28. Springer, August 2002. San Francisco Bay (Redwood City), USA. 8, 9, 105

[39] Zhimin Chen and Yujie Zhou. Dual-Rail Random Switching Logic: A Countermeasure to Reduce Side Channel Leakage. In *CHES*, volume 4249 of *LNCS*, pages 242–254. Springer, October 10-13 2006. Yokohama, Japan, http://dx.doi.org/10.1007/11894063_20. 29, 58, 88

[40] Benoit Chevallier-Mames, Mathieu Ciet, and Marc Joye. Low-cost solutions for preventing simple side-channel analysis: Side-channel atomicity, 2003. 33

[41] Mathieu Ciet. *Aspects of Fast and Secure Arithmetics for Elliptic Curve Cryptography*. PhD thesis, Universite Catholique de Louvain, 2003. 39

[42] Mathieu Ciet and Marc Joye. (Virtually) Free randomization techniques for elliptic curve cryptography. In *Information and Communications Security (ICICS 2003)*, volume 2836 of *Lecture Notes in Computer Science*. Springer-Verlag, 10 2003. 39

[43] Mathieu Ciet and Marc Joye. Elliptic curve cryptosystems in the presence of permanent and transient faults. *Des. Codes Cryptography*, 36(1):33–43, 2005. 16

[44] Christophe Clavier and Marc Joye. Universal exponentiation algorithm - a first step towards provable spa-resistance. In *Cryptographic Hardware and Embedded Systems - CHES 2001, volume 2162 of Lecture Notes in Computer Science*, pages 300–308. Springer-Verlag, 2001. 32

[45] J.-S. Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In *In Proceedings of CHES'99*, pages 292–302. Spring73-Verlag, 1999. 41

[46] Jean-Sébastien Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In *Proceedings of the First International Workshop on Cryptographic Hardware and Embedded Systems*, CHES '99, pages 292–302, London, UK, UK, 1999. Springer-Verlag. 31, 35, 36, 38, 41

[47] Jean-Sébastien Coron, Emmanuel Prouff, and Matthieu Rivain. Side Channel Cryptanalysis of a Higher Order Masking Scheme. In *CHES*, volume 4727 of *LNCS*, pages 28–44. Springer, September 10-13 2007. Vienna, Austria. 22

[48] Nicolas Courtois and Louis Goubin. An Algebraic Masking Method to Protect AES Against Power Attacks. In Dongho Won and Seungjoo Kim, editors, *ICISC*, volume 3935 of *Lecture Notes in Computer Science*, pages 199–209. Springer, 2005. 21

[49] Common Criteria. Application of Attack Potential to Smartcards, Mandatory Technical Document, Version 2.7, Revision 1, CCDB-2009-03-001, March 2009. http://www.commoncriteriaportal.org/files/supdocs/CCDB-2009-03-001.pdf. 105

[50] Pierre Dagnelie. *Statistique théorique et appliquée. Tome 2, Inférence statistique à une et à deux dimensions*. De Boeck, 2006. 143

[51] Nicolas Debande, Youssef Souissi, Sylvain Guilley, Jean-Luc Danger, and Maxime Nassar. "Re-synchronization by Moments": an efficient solution to align Side-Channel traces. In *WIFS, IEEE Intl. Workshop on Information Forensics and Security*, pages 1–6, November 29th – December 2nd 2011. Foz do Iguaçu, Brazili. DOI: 10.1109/WIFS.2011.6123143. 17

[52] Jean-F. Dhem, F. Koeune, P.-A. Leroux, P. Mestre, J.-J. Quisquater, and J.-L. Willems. A Practical Implementation of the Timing Attack. In *CARDIS*, pages 167–182, 1998. http://citeseer.nj.nec.com/dhem98practical.html. 6, 7

[53] Saar Drimer. Security for Volatile FPGAs (university of Cambrige technical report number 763), November 2009. 102

[54] H. M. Edwards. A Normal Form for Elliptic Curves. *Bulletin of the American Mathematical Society*, page 393422, 2007. 34

[55] Pierre-Alain Fouque, Reynald Lercier, Denis Réal, and Frédéric Valette. Fault attack on elliptic curve montgomery ladder implementation. In *FDTC '08: Proceedings of the 2008 5th Workshop on Fault Diagnosis and Tolerance in Cryptography*, pages 92–98, Washington, DC, USA, 2008. IEEE Computer Society. 16

[56] Pierre-Alain Fouque, Denis Réal, Frédéric Valette, and M'hamed Drissi. The carry leakage on the randomized exponent countermeasure. In *Cryptographic Hardware and Embedded Systems - CHES 2008, 10th International Workshop, Washington, D.C., USA, August 10-13, 2008. Proceedings*, pages 198–213, 2008. 12, 13, 39, 40

[57] Pierre-Alain Fouque and Frederic Valette. The Doubling Attack, Why Upwards Is Better than Downwards. pages 269–280, 2003. ISBN: 978-3-540-40833-8. 10

[58] Guillaume Fumaroli, Ange Martinelli, Emmanuel Prouff, and Matthieu Rivain. Affine Masking against Higher-Order Side Channel Analysis. In Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors, *Selected Areas in Cryptography*, volume 6544 of *LNCS*, pages 262–280. Springer, 2010. 19

[59] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. Electromagnetic Analysis: Concrete Results. In *CHES*, volume 2162 of *LNCS*, pages 251–261. Springer, May 14-16 2001. Paris, France. 3

[60] Santosh Ghosh, Monjur Alam, Indranil Sen Gupta, and Dipanwita Roy Chowdhury. A robust gf(p) parallel arithmetic unit for public key cryptography. In *Proceedings of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools*, pages 109–115, Washington, DC, USA, 2007. IEEE Computer Society. 119

[61] Benedikt Gierlichs, Lejla Batina, Bart Preneel, and Ingrid Verbauwhede. Revisiting Higher-Order DPA Attacks: Multivariate Mutual Information Analysis. In *CT-RSA*, volume 5985 of *LNCS*, pages 221–234. Springer, March 1-5 2010. San Francisco, CA, USA. 138

[62] Benedikt Gierlichs, Lejla Batina, Pim Tuyls, and Bart Preneel. Mutual information analysis. In *CHES, 10th International Workshop*, volume 5154 of *Lecture Notes in Computer Science*, pages 426–442. Springer, August 10-13 2008. Washington, D.C., USA. 105, 137

[63] Benedikt Gierlichs, Elke De Mulder, Bart Preneel, and Ingrid Verbauwhede. Empirical comparison of side channel analysis distinguishers on DES in hardware. In IEEE, editor, *ECCTD. European Conference on Circuit Theory and Design*, pages 391–394, August 23-27 2009. Antalya, Turkey. 148

[64] Benedikt Gierlichs, Kerstin Lemke-Rust, and Christof Paar. Templates vs. Stochastic Methods. In *CHES*, volume 4249 of *LNCS*, pages 15–29. Springer, October 10-13 2006. Yokohama, Japan. 9

[65] Jovan Dj. Golic and Christophe Tymen. Multiplicative Masking and Power Analysis of AES. In *CHES*, volume 2523 of *Lecture Notes in Computer Science*, pages 198–212. Springer, August 13-15 2002. San Francisco, USA. 20, 21

[66] Louis Goubin. A Refined Power-Analysis Attack on Elliptic Curve Cryptosystems. *Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography: Public Key Cryptography*, pages 199 – 210, 2003. ISBN: 978-3-540-00324-3. 13

[67] Louis Goubin. A refined power-analysis attack on elliptic curve cryptosystems. In *Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography: Public Key Cryptography*, PKC '03, pages 199–210, London, UK, UK, 2003. Springer-Verlag. 41, 42

[68] Louis Goubin and Jacques Patarin. DES and Differential Power Analysis. The "Duplication" Method. In *CHES*, LNCS, pages 158–172. Springer, Aug 1999. Worcester, MA, USA. 18

[69] F.J. Gravetter and L.B. Wallnau. *Essentials of statistics for the behavioral sciences*. Thomson/Wadsworth, 2008. 144

[70] Sylvain Guilley, Sumanta Chaudhuri, Laurent Sauvage, Tarik Graba, Jean-Luc Danger, Philippe Hoogvorst, Vinh-Nga Vong, and Maxime Nassar. Place-and-Route Impact on the Security of DPL Designs in FPGAs. In *HOST*, IEEE, pages 29–35, June 9 2008. Anaheim, USA. ISBN = 978-1-4244-2401-6. 49

[71] Sylvain Guilley, Florent Flament, Renaud Pacalet, Philippe Hoogvorst, and Yves Mathieu. Security Evaluation of a Balanced Quasi-Delay Insensitive Library. In *DCIS*, Grenoble, France, nov 2008. IEEE. 6 pages, Session 5D – Reliable and Secure Architectures, ISBN: 978-2-84813-124-5, full text in HAL: `http://hal.archives-ouvertes.fr/hal-00283405/en/`. 58

[72] Sylvain Guilley, Philippe Hoogvorst, Yves Mathieu, and Renaud Pacalet. The "Backend Duplication" Method. In *CHES*, volume 3659 of *LNCS*, pages 383–397. Springer, 2005. August 29th – September 1st, Edinburgh, Scotland, UK. 58

[73] Sylvain Guilley, Philippe Hoogvorst, and Renaud Pacalet. A Fast Pipelined Multi-Mode DES Architecture Operating in IP Representation. *Integration, The VLSI Journal*, 40(4):479–489, July 2007. DOI: 10.1016/j.vlsi.2006.06.004. 136

[74] Sylvain Guilley, Karim Khalfallah, Victor Lomne, and Jean-Luc Danger. Formal Framework for the Evaluation of Waveform Resynchronization Algorithms. In LNCS, editor, *WISTP: Information Security Theory and Practices. Smart Cards, Mobile and Ubiquitous Computing*, volume 6633 of *LNCS*, pages 100–115. Springer, June 1-3 2011. Heraklion, Greece. DOI: 10.1007/978-3-642-21040-2_7. 17

[75] Neil Hanley, Robert McEvoy, Michael Tunstall, Claire Whelan, Colin Murphy, and William P. Marnane. Correlation Power Analysis of Large Word Sizes. In *ISSC (Irish Signals and System Conference)*, pages 145–150. IET, 13-14 Sept 2007. Edinburgh, Scotland, UK. 137

[76] Neil Hanley, Michael Tunstall, and William P. Marnane. Unknown Plaintext Template Attacks. In *WISA*, volume 5932 of *Lecture Notes in Computer Science*, pages 148–162. Springer, August 25-27 2009. Busan, Korea. 9

[77] Naofumi Homma, Atsushi Miyamoto, Takafumi Aoki, Akashi Satoh, and Adi Shamir. Comparative Power Analysis of Modular Exponentiation Algorithms. *IEEE Trans. Computers*, 59(6):795–807, 2010. 11

[78] Philippe Hoogvorst. The Variance Power Attack. In *COSADE*, pages 4–9, February 4-5 2010. 8

[79] E. Brier I. Dechene and M. Joye. Unified Point Addition Formulae for Elliptic Curve Cryptosystems. *In Embedded Cryptographic Hardware : Methodologies and Architectures*, page 24725, 2004. 34

[80] Yuval Ishai, Amit Sahai, and David Wagner. Private Circuits: Securing Hardware against Probing Attacks. In *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 463–481. Springer, August 17–21 2003. Santa Barbara, California, USA. 22

[81] Kouichi Itoh, Tetsuya Izu, and Masahiko Takenaka. Address-bit differential power analysis of cryptographic schemes ok-ecdh and ok-ecdsa. In *Cryptographic Hardware and Embedded Systems - CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 399–412. Springer Berlin / Heidelberg, 2003. 12

[82] Kouichi Itoh, Tetsuya Izu, and Masahiko Takenaka. A practical countermeasure against address-bit differential power analysis. In Colin Walter, Çetin Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2003*, volume 2779 of *Lecture Notes in Computer Science*, pages 382–396. Springer Berlin / Heidelberg, 2003. 40

[83] Tetsuya Izu and Tsuyoshi Takagi. Exceptional procedure attack on elliptic curve cryptosystems. *PKC 2003, LNCS*, 2567:224–239, 2003. 34

[84] M. Izumi, J. Ikegami, K. Sakiyama, and K. Ohta. Improved countermeasure against address-bit dpa for ecc scalar multiplication. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, pages 981 –984, march 2010. 40

[85] Aleks Jakulin and Ivan Bratko. Analyzing Attribute Dependencies. In *PKDD 2003, volume 2838 of LNAI*, pages 229–240. Springer-Verlag, 2003. 138

[86] Japanese RCIS-AIST, SASEBO development board: `http://www.rcis.aist.go.jp/special/SASEBO/index-en.html`. 63, 100

[87] M. Joye. *Defences against Side-Channel Analysis*. Cambridge University Press, 2005. Chapter V. 39

[88] M. Joye and J.-J. Quisquater. Hessian Elliptic Curves and Side-Channel Attacks. *Cryptographic Hardware and Embedded Systems*, 2162:402410, 2001. 34

[89] Marc Joye and Christophe Tymen. Protections against Differential Analysis for Elliptic Curve Cryptography. In *CHES*, volume 2162 of *Lecture Notes in Computer Science*, pages 377–390. Springer, May 14-16 2001. Paris, France. 41, 42

[90] Marc Joye and Sung-Ming Yen. The Montgomery Powering Ladder. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *CHES*, volume 2523 of *Lecture Notes in Computer Science*, pages 291–302. Springer, 2002. 15

[91] Edward W. Kamen and Jonathan Su. *Introduction to optimal estimation, Advanced textbooks in control and signal processing, Control and Signal Processing Series*. Springer, 1999. 143

[92] Jens-Peter Kaps and Rajesh Velegalati. DPA Resistant AES on FPGA Using Partial DDL. In *FCCM: 18th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines*, pages 273–280. IEEE Computer Society, May 02–May 04 2010. Charlotte, North Carolina, USA. DOI: 10.1109/FCCM.2010.49. 27, 88

[93] M. Karpovsky, K.J. Kulikowski, and A. Taubin. Robust protection against fault-injection attacks on smart cards implementing the advanced encryption standard. In *Dependable Systems and Networks, 2004 International Conference on*, pages 93 – 101, june-1 july 2004. 14

[94] R. Karri, K. Wu, P. Mishra, and Yongkook Kim. Concurrent error detection schemes for fault-based side-channel cryptanalysis of symmetric block ciphers. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 21(12):1509 – 1517, dec 2002. 43

[95] Michael Kasper, Werner Schindler, and Marc Stöttinger. A stochastic method for security evaluation of cryptographic FPGA implementations. In Jinian Bian, Qiang Zhou, Peter Athanas, Yajun Ha, and Kang Zhao, editors, *FPT*, pages 146–153. IEEE, 2010. 10

[96] Chong Kim, Jong Shin, Jean-Jacques Quisquater, and Pil Lee. Safe-error attack on spa-fa resistant exponentiations using a hw modular multiplier. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *Information Security and Cryptology - ICISC 2007*, pages 273–281. Springer Berlin / Heidelberg, 2007. 31

[97] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *Proceedings of CRYPTO'96*, volume 1109 of *LNCS*, pages 104–113. Springer-Verlag, 1996. (PDF). 3, 4, 6, 35, 36, 38

[98] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In *CRYPTO*, volume 1666 of *LNCS*, pages pp 388–397. Springer, 1999. 3, 4, 8, 17, 35

[99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In *Proceedings of CRYPTO'99*, volume 1666 of *LNCS*, pages 388–397. Springer-Verlag, 1999. 137

[100] Yuichi Komano, Hideo Shimizu, and Shinichi Kawamura. Built-in determined subkey correlation power analysis. Cryptology ePrint Archive, Report 2009/161, 2009. http://eprint.iacr.org/2009/161. 137

[101] Oliver Kömmerling and Markus G. Kuhn. Design principles for tamper-resistant smartcard processors. In *Proceedings of the USENIX Workshop on Smartcard Technology on USENIX Workshop on Smartcard Technology*, pages 2–2, Berkeley, CA, USA, 1999. USENIX Association. 14

[102] Boris Köpf and David Basin. An information-theoretic model for adaptive side-channel attacks. In *CCS'07: Proceedings of the 14th ACM conference on Computer and communications security*, pages 286–296, New York, NY, USA, 2007. ACM. 104

[103] Konrad J. Kulikowski, Mark G. Karpovsky, and Alexander Taubin. Power Attacks on Secure Hardware Based on Early Propagation of Data. In *IOLTS*, pages 131–138. IEEE Computer Society, 2006. Como, Italy. 48

[104] USA Kyungnam Kim Department of Computer Science University of Maryland. Face recognition using principal component analysis. Available online on 26 february 2002. 133

[105] Thanh-Ha Le, Jessy Clédière, Cécile Canovas, Bruno Robisson, Christine Servière, and Jean-Louis Lacoume. A Proposition for Correlation Power Analysis Enhancement. In *CHES*, volume 4249 of *LNCS*, pages 174–186. Springer, 2006. Yokohama, Japan. 137

[106] Yang Li, Kazuo Sakiyama, Lejla Batina, D. Nakatsu, and Kazuo Ohta. Power Variance Analysis breaks a masked ASIC implementation of AES. In *DATE*, pages 1059–1064. IEEE, March 8-12 2010. Dresden, Germany. 8

[107] P.-Y. Liardet and N. P. Smart. Preventing SPA/DPA in ECC Systems Using the Jacobi Form. *Cryptographic Hardware and Embedded Systems*, 2162:391401, 2001. 34

[108] Victor Lomné, Amine Dehbaoui, Philippe Maurine, Lionel Torres, and Michel Robert. Differential Power Analysis enhancement with statistical preprocessing. In IEEE, editor, *DATE*, March 8-12 2010. 152

[109] Yingxi Lu, M.P. O'Neill, and J.V. McCanny. Fpga implementation and analysis of random delay insertion countermeasure against dpa. In *ICECE Technology, 2008. FPT 2008. International Conference on*, pages 201 –208, dec. 2008. 17

[110] Jiqiang Lv and Yongfei Han. Enhanced DES implementation secure against differential power analysis in smart-cards. In *Information Security and Privacy, 10th Australasian Conference*, volume 3574 of *LNCS*, pages 195–206, Brisbane, Australia, July 2005. Springer-Verlag. 22

[111] Houssem Maghrebi, Jean-Luc Danger, Florent Flament, and Sylvain Guilley. Evaluation of Countermeasures Implementation Based on Boolean Masking to Thwart First and Second Order Side-Channel Attacks. In *SCS*, IEEE, pages 1–6, November 6–8 2009. Jerba, Tunisia. DOI: 10.1109/ICSCS.2009.5412597. 8, 19

[112] Houssem Maghrebi, Sylvain Guilley, Jean-Luc Danger, and Florent Flament. Entropy-based Power Attack. In *HOST*, IEEE Computer Society, pages 1–6, June 13-14 2010. Anaheim Convention Center, Anaheim, CA, USA. DOI: 10.1109/HST.2010.5513124. 8

[113] P. Maistri and R. Leveugle. Double-data-rate computation as a countermeasure against fault analysis. *Computers, IEEE Transactions on*, 57(11):1528 –1539, nov. 2008. 43

[114] Hideyo Mamiya, Atsuko Miyaji, and Hiroaki Morimoto. Secure elliptic curve exponentiation against rpa, zra, dpa, and spa. *IEICE Transactions*, 89-A(8):2207–2215, 2006. 37

[115] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, December 2006. ISBN 0-387-30857-1, http://www.dpabook.org/. 3, 18

[116] Stefan Mangard, Elisabeth Oswald, and François-Xavier Standaert. One for All - All for One: Unifying Standard DPA Attacks. Cryptology ePrint Archive, Report 2009/449, 2009. 133

[117] Robert P. McEvoy, Colin C. Murphy, William P. Marnane, and Michael Tunstall. Isolated WDDL: A Hiding Countermeasure for Differential Power Analysis on FPGAs. *ACM Trans. Reconfigurable Technol. Syst.*, 2(1):1–23, 2009. 26, 88

[118] Nele Mentens, Lejla Batina, Bart Preneel, and Ingrid Verbauwhede. An fpga implementation of rijndael: Trade-offs for side-channel security, 2004. 21, 101

[119] Thomas S. Messerges. Securing the AES Finalists Against Power Analysis Attacks. In *Fast Software Encryption'00*, pages 150–164. Springer-Verlag, April 2000. New York. 20, 22

[120] Thomas S. Messerges, Ezzy A. Dabbish, and Robert H. Sloan. Investigations of Power Analysis Attacks on Smartcards. In *USENIX — Smartcard'99*, pages 151–162, May 10–11 1999. Chicago, Illinois, USA (Online PDF). 134

[121] Thomas S. Messerges, Ezzy A. Dabbish, and Robert H. Sloan. Examining Smart-Card Security under the Threat of Power Analysis Attacks. *IEEE Trans. Computers*, 51(5):541–552, 2002. 12

[122] Peter L. Montgomery. Speeding the Pollard and Elliptic Curve Methods of Factorization. *Mathematics of Computation*, 48(177):243–264, 1987. 31

[123] Elke De Mulder, Benedikt Gierlichs, Bart Preneel, and Ingrid Verbauwhede. Practical DPA Attacks on MDPL. In *First International Workshop on Information Forensics and Security (WIFS)*. IEEE Signal Processing Society, December 6-9 2009. London, United Kingdom. Also http://eprint.iacr.org/2009/231. 28, 29

[124] Frédéric Muller and Frédéric Valette. High-order attacks against the exponent splitting protection. In *Public Key Cryptography - PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography*, pages 315–329, 2006. 40

[125] J.L. Myers and A.D. Well. *Research design and statistical analysis*. L. Erlbaum Associates, 1995. 144

[126] H. N. Nagaraja. Functions of concomitants of order statistics. *Journal of the Indian Society for Probability and Statistics*, 7:15–32, 2003. 145

[127] 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 — Complete version. Cryptology ePrint Archive, Report 2011/534, September 2011. http://eprint.iacr.org/2011/534. 112

[128] Elisabeth Oswald, Stefan Mangard, and Norbert Pramstaller. Secure and efficient masking of aes - a mission impossible? Cryptology ePrint Archive, Report 2004/134, 2004. http://eprint.iacr.org/. 20

[129] Elisabeth Oswald, Stefan Mangard, Norbert Pramstaller, and Vincent Rijmen. A Side-Channel Analysis Resistant Description of the AES S-box. In LNCS, editor, *Proceedings of FSE'05*, volume 3557 of *LNCS*, pages 413–423. Springer, February 2005. Paris, France. 21

[130] Elisabeth Oswald and Kai Schramm. An Efficient Masking Scheme for AES Software Implementations. In *WISA'05*, pages 292–305, 2005. 21

[131] A.D. Oviedo, University of Waterloo. Dept. of Electrical, and Computer Engineering. *On fault-based attacks and countermeasures for elliptic curve cryptosystems*. University of Waterloo, 2008. 44

[132] Gilles Piret and Jean-Jacques Quisquater. A Differential Fault Attack Technique against SPN Structures, with Application to the AES and KHAZAD. In *CHES*, volume 2779 of *LNCS*, pages 77–88. Springer, September 2003. Cologne, Germany. 14

[133] Gilles Piret and François-Xavier Standaert. Security Analysis of Higher-Order Boolean Masking Schemes for Block Ciphers (with Conditions of Perfect Masking). *IET Information Security*, 2(1):1–11, 2008. DOI: 10.1049/iet-ifs:20070066. 21

[134] Thomas Popp, Mario Kirschbaum, Thomas Zefferer, and Stefan Mangard. Evaluation of the Masked Logic Style MDPL on a Prototype Chip. In *CHES*, volume 4727 of *LNCS*, pages 81–94. Springer, Sept 2007. Vienna, Austria. 28, 88

[135] Thomas Popp and Stefan Mangard. Masked Dual-Rail Pre-charge Logic: DPA-Resistance Without Routing Constraints. In *Proceedings of CHES'05*, volume 3659 of *LNCS*, pages 172–186. Springer, August 29 – September 1 2005. Edinburgh, Scotland, UK. 27, 28, 58, 88

[136] Emmanuel Prouff and Matthieu Rivain. A Generic Method for Secure SBox Implementation. In Sehun Kim, Moti Yung, and Hyung-Woo Lee, editors, *WISA*, volume 4867 of *Lecture Notes in Computer Science*, pages 227–244. Springer, 2007. 19, 20, 21

[137] Emmanuel Prouff and Matthieu Rivain. Theoretical and Practical Aspects of Mutual Information Based Side Channel Analysis. In Springer, editor, *ACNS*, volume 5536 of *LNCS*, pages 499–518, June 2-5 2009. Paris-Rocquencourt, France. 139

[138] Emmanuel Prouff and Matthieu Rivain. Theoretical and Practical Aspects of Mutual Information Based Side Channel Analysis. In *IJACT*, 2010. 9

[139] Emmanuel Prouff, Matthieu Rivain, and Régis Bevan. Statistical Analysis of Second Order Differential Power Analysis. *IEEE Trans. Computers*, 58(6):799–811, 2009. 105

[140] Jean-Jacques Quisquater and David Samyde. *Eddy current for Magnetic Analysis with Active Sensor*. Springer, 2002. 14

[141] Christian Rechberger and Elisabeth Oswald. Practical Template Attacks. In *WISA*, volume 3325 of *LNCS*, pages 443–457. Springer, August 23-25 2004. Jeju Island, Korea. 9, 133

[142] Francesco Regazzoni, Yi Wang, and Francois-Xavier Standaert. FPGA Implementations of the AES Masked Against Power Analysis Attacks. In *COSADE*, pages 56–66, February 2011. Darmstadt, Germany. 21, 23, 95, 101

[143] Matthieu Rivain, Emmanuelle Dottax, and Emmanuel Prouff. Block ciphers implementations provably secure against second order side channel analysis. *Fast Software Encryption FSE 2008*, 5086:127–143, 2008. 22

[144] Matthieu Rivain and Emmanuel Prouff. Provably Secure Higher-Order Masking of AES. In Stefan Mangard and François-Xavier Standaert, editors, *CHES*, volume 6225 of *LNCS*, pages 413–427. Springer, 2010. 22

[145] Minoru Saeki and Daisuke Suzuki. Security Evaluations of MRSL and DRSL Considering Signal Delays. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E91-A(1):176–183, 2008. DOI: 10.1093/ietfec/e91-a.1.176. 28, 29

[146] Gilbert SAPORTA. *Probabilités analyse des données et statistiques*. 2008. 133

[147] Laurent Sauvage, Sylvain Guilley, and Yves Mathieu. ElectroMagnetic Radiations of FPGAs: High Spatial Resolution Cartography and Attack of a Cryptographic Module. *ACM Trans. Reconfigurable Technol. Syst.*, 2(1):1–24, March 2009. Full text in `http://hal.archives-ouvertes.fr/hal-00319164/en/`. 138

[148] Patrick Schaumont and Kris Tiri. Masking and Dual Rail Logic Don't Add Up. In *CHES*, volume 4727 of *LNCS*, pages 95–106. Springer, September 10-13 2007. Vienna, Austria. 27, 28, 29

[149] Edna Schechtman and Shlomo Yitzhaki. A measure of association base on Gini's Mean difference. *Communications in statistics. Theory and methods*, 16:207 – 231, 1987. 144, 145

[150] Werner Schindler. A Timing Attack against RSA with the Chinese Remainder Theorem. In *CHES*, volume 1965 of *LNCS*, pages 109–124. Springer, 2000. 6

[151] Werner Schindler. A Combined Timing and Power Attack. In David Naccache and Pascal Paillier, editors, *Public Key Cryptography*, volume 2274 of *Lecture Notes in Computer Science*, pages 263–279. Springer, 2002. 138

[152] Werner Schindler. Advanced stochastic methods in side channel analysis on block ciphers in the presence of masking. *Journal of Mathematical Cryptology*, 2(3):291–310, October 2008. ISSN (Online) 1862-2984, ISSN (Print) 1862-2976, DOI: 10.1515/JMC.2008.013. 9, 105

[153] Werner Schindler, François Koeune, and Jean-Jacques Quisquater. Improving divide and conquer attacks against cryptosystems by better error detection / correction strategies. In *Cryptography and Coding*, volume 2260 of *Lecture Notes in Computer Science*, pages 245–267. Springer Berlin / Heidelberg, 2001. 6

[154] Werner Schindler, Kerstin Lemke, and Christof Paar. A Stochastic Model for Differential Side Channel Cryptanalysis. In LNCS, editor, *CHES*, volume 3659 of *LNCS*, pages 30–46. Springer, Sept 2005. Edinburgh, Scotland, UK. 8, 9

[155] Kai Schramm and Christof Paar. Higher Order Masking of the AES. In David Pointcheval, editor, *CT-RSA*, volume 3860 of *LNCS*, pages 208–225. Springer, 2006. 22

[156] Nidhal Selmane, Shivam Bhasin, Sylvain Guilley, Tarik Graba, and Jean-Luc Danger. WDDL is Protected Against Setup Time Violation Attacks. In *FDTC*, pages 73–83. IEEE Computer Society, September 6th 2009. In conjunction with CHES'09, Lausanne, Switzerland. DOI: 10.1109/FDTC.2009.40; Online version: http://hal.archives-ouvertes.fr/hal-00410135/en/. 77

[157] Shaunak Shah, Rajesh Velegalati, Jens-Peter Kaps, and David Hwang. Investigation of DPA Resistance of Block RAMs in Cryptographic Implementations on FPGAs. In Viktor K. Prasanna, Jürgen Becker, and René Cumplido, editors, *ReConFig*, pages 274–279. IEEE Computer Society, 2010. 96

[158] Jonathon Shlens. A tutorial in Principal Component Analysis. Available online on 10 decembre 2005. 133

[159] Carsten Sinz. Towards an Optimal CNF Encoding of Boolean Cardinality Constraints. In Peter van Beek, editor, *CP*, volume 3709 of *Lecture Notes in Computer Science*, pages 827–831. Springer, 2005. 113

[160] S.Kolenikov and G.Angeles. The use of discrete data in PCA for socio-economic status evaluation. Available online on 2 february 2005. 133

[161] Sergei Skorobogatov. Synchronization method for SCA and fault attacks. *Journal of Cryptographic Engineering*, 1:71–77, 2011. 10.1007/s13389-011-0004-0. 17

[162] Sergei Skorobogatov and Ross Anderson. Optical fault induction attacks. In Burton Kaliski, çetin Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 31–48. Springer Berlin / Heidelberg, 2003. 14

[163] Lindsay I Smith. A tutorial in Principal Component Analysis. Available online on 26 february 2002. 133

[164] Rafael Soares, Ney Calazans, Victor Lomné, Philippe Maurine, Lionel Torres, and Michel Robert. Evaluating the robustness of secure triple track logic through prototyping. In *SBCCI'08: Proceedings of the 21st annual symposium on Integrated circuits and system design*, pages 193–198, New York, NY, USA, September 1-4 2008. ACM. 29, 58, 88

[165] Mate Soos. SAT-solver "`cryptominisat`", Version 2.9.0, January 20 2011. https://gforge.inria.fr/projects/cryptominisat. 113

[166] Mate Soos, Karsten Nohl, and Claude Castelluccia. Extending SAT Solvers to Cryptographic Problems. In Oliver Kullmann, editor, *SAT*, volume 5584 of *Lecture Notes in Computer Science*, pages 244–257. Springer, 2009. 113

[167] François-Xavier Standaert, Benedikt Gierlichs, and Ingrid Verbauwhede. Partition vs. Comparison Side-Channel Distinguishers: An Empirical Evaluation of Statistical Tests for Univariate Side-Channel Attacks against Two Unprotected CMOS Devices. In *ICISC*, volume 5461 of *LNCS*, pages 253–267. Springer, December 3-5 2008. Seoul, Korea. 8, 91, 137, 146

[168] François-Xavier Standaert, Tal Malkin, and Moti Yung. A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks. In *EUROCRYPT*, volume 5479 of *LNCS*, pages 443–461. Springer, April 26-30 2009. Cologne, Germany. 93, 103, 104, 136, 158

[169] François-Xavier Standaert, Éric Peeters, François Macé, and Jean-Jacques Quisquater. Updates on the Security of FPGAs Against Power Analysis Attacks. In *ARC*, volume 3985 of *LNCS*, pages 335–346. Springer-Verlag, March 2006. Delft, The Netherlands. 8, 92, 134

[170] François-Xavier Standaert, François Koeune, and Werner Schindler. How to compare profiled side-channel attacks? In *Applied Cryptography and Network Security*, volume 5536 of *Lecture Notes in Computer Science*, pages 485–498. Springer Berlin / Heidelberg, 2009. 9

[171] François-Xavier Standaert, Gaël Rouvroy, and Jean-Jacques Quisquater. FPGA Implementations of the DES and Triple-DES Masked Against Power Analysis Attacks. In *FPL*. IEEE, August 2006. Madrid, Spain. 95

[172] Douglas Stebila and Nicolas Thériault. Unified point addition formulæ and side-channel attacks. In *CHES, LNCS, volume 4249*, 2006. 34

[173] Daisuke Suzuki and Minoru Saeki. Security Evaluation of DPA Countermeasures Using Dual-Rail Pre-charge Logic Style. In *CHES*, volume 4249 of *LNCS*, pages 255–269. Springer, October 10-13 2006. Yokohama, Japan. http://dx.doi.org/10.1007/11894063_21. 48

[174] TELECOM ParisTech SEN research group. DPA Contest (1st edition), 2008–2009. http://www.DPAcontest.org/. 136, 151, 160

[175] TELECOM ParisTech SEN research group. DPA Contest (2nd edition), 2009–2010. http://www.DPAcontest.org/v2/. 145

[176] Kris Tiri and Ingrid Verbauwhede. A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation. In *DATE'04*, pages 246–251. IEEE Computer Society, February 2004. Paris, France. DOI: 10.1109/DATE.2004.1268856. 24, 26, 88

[177] Kris Tiri and Ingrid Verbauwhede. Place and Route for Secure Standard Cell Design. In Kluwer, editor, *Proceedings of WCC / CARDIS*, pages 143–158, Aug 2004. Toulouse, France. 49, 58

[178] Kris Tiri and Ingrid Verbauwhede. A digital design flow for secure integrated circuits. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 25(7):1197–1208, 2006. 24, 88

[179] R. Toth, Z. Faigl, M. Szalay, and S. Imre. An advanced timing attack scheme on rsa. In *Telecommunications Network Strategy and Planning Symposium, 2008. Networks 2008. The 13th International*, pages 1 –24, 2008. 6

[180] Elena Trichina and Antonio Bellezza. Implementation of elliptic curve cryptography with built-in counter measures against side channel attacks. In Burton Kaliski, çetin Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 297–312. Springer Berlin / Heidelberg, 2003. 39

[181] Elena Trichina, Domenico De Seta, and Lucia Germani. Simplified adaptive multiplicative masking for aes. In *Cryptographic Hardware and Embedded Systems - CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 71–85. Springer Berlin / Heidelberg, 2003. 21

[182] Elena Trichina and Tymur Korkishko. *Secure AES Hardware Module for Resource Constrained Devices*, pages 215–229. 2005. 21

[183] Stéphane Tufféry and Gilbert Saporta. *Data mining et statistique décisionnelle. L'intelligence des données*. Technip, 2010. ISBN: 978271080946-3. 9, 143

[184] R. Velegalati and J.-P. Kaps. Improving security of sddl designs through interleaved placement on xilinx fpgas. In *Field Programmable Logic and Applications (FPL), 2011 International Conference on*, pages 506 –511, sept. 2011. 26, 88

[185] Nicolas Veyrat-Charvillon and François-Xavier Standaert. Mutual Information Analysis: How, When and Why? In *CHES*, volume 5747 of *LNCS*, pages 429–443. Springer, September 6-9 2009. Lausanne, Switzerland. 9

[186] Nicolas Veyrat-Charvillon and François-Xavier Standaert. Adaptive Chosen-Message Side-Channel Attacks. In Jianying Zhou and Moti Yung, editors, *ACNS*, volume 6123 of *Lecture Notes in Computer Science*, pages 186–199, 2010. 104

[187] Jason Waddle and David Wagner. Towards Efficient Second-Order Power Analysis. In *CHES*, volume 3156 of *LNCS*, pages 1–15. Springer, 2004. Cambridge, MA, USA. 20, 103

[188] Colin D. Walter. Simple power analysis of unified code for ECC double and add. pages 86–115. 2004. 34

[189] Chih-Hsu Yen and Bing-Fei Wu. Simple error detection methods for hardware implementation of advanced encryption standard. *IEEE Trans. Comput.*, 55:720–731, June 2006. 43

[190] Sung-Ming Yen and Marc Joye. Checking before output may not be enough against fault-based cryptanalysis. *IEEE Trans. Comput.*, 49(9):967–970, 2000. 15

[191] Shlomo Yitzhaki. Gini's mean difference: a superior measure of variability for non-normal distributions. *International Journal of Statistics*, 2:285 – 316, 2003. 145

[192] Pengyuan Yu and Patrick Schaumont. Secure FPGA circuits using controlled placement and routing. In *CODES+ISSS'07: Proceedings of the 5th IEEE/ACM international conference on Hardware/software codesign and system synthesis*, pages 45–50, New York, NY, USA, 2007. ACM. 27, 88

[193] Zeng Guang Hou. PCA for data fusion and navigation of mobile robots. volume 3495 of *LNCS*, pages 610–611. Springer, 2005. 133