# An Overview of Power Analysis Attacks Against Field Programmable Gate Arrays

F.-X. Standaert, E. Peeters, G. Rouvroy, J.-J. Quisquater

***Abstract*— Since their introduction by Kocher in 1998, power analysis attacks have attracted significant attention within the cryptographic community. While early works in the field mainly threatened the security of smart cards and simple processors, several recent publications have shown the vulnerability of hardware implementations as well. In particular, Field Programmable Gate Arrays are attractive options for hardware implementation of encryption algorithms, but their security against power analysis is a serious concern, as we discuss in this article. For this purpose, we present recent results of attacks attempted against standard encryption algorithms, provide a theoretical estimation of these attacks based on simple statistical parameters and evaluate the cost and security of different possible countermeasures.**

## I. Introduction

Recent developments in information technologies made the secure transmission of digital data a critical design point. Large data flows have to be exchanged securely and involve encryption rates that sometimes may require hardware implementations. In this context, reprogrammable devices such as Field Programmable Gate Arrays (FPGAs) are highly attractive solutions for hardware implementations of encryption algorithms and numerous papers underline their growing performances and flexibility for any digital processing application.

Although cryptosystem designers assumed a long time that secret parameters will be manipulated in closed, reliable computing environments, Kocher *et al.* stressed in 1998 (see [20]) that actual computers and microchips leak information correlated with the data handled in physical devices. Consequently, side-channel attacks based on time, power and electromagnetic measurements were successfully applied to the smart card technology. Because of their intrinsic opportunities to perform parallel computing, hardware and FPGA implementations were initially believed to provide practical security against side-channel opponents. This was then denied by a number of works.

The first successful power analysis attack against an FPGA was carried out by Örs *et al.* in 2003 [30]. They mounted an attack against an elliptic curve cryptographic processor and were able to retrieve the secret key by simple visual inspection of the leakage traces. Various publications followed this first result and confirmed the possibility to apply power analysis to FPGAs [31], [37], [38]. Almost at the same time an attack exploiting the electromagnetic leakage of FPGAs was proposed in [9]. Further investigations on the electromagnetic behavior of FPGAs have recently been conducted in [8]. However, most of these results remained practice-oriented and the security of hardware devices was not considered in a general perspective. This latter concern is therefore investigated in this article.

During the last years, a lot of research has been conducted on power analysis attacks and their countermeasures. These investigations have led to theoretical and practical improvements of the technique, following different trends (*e.g.* in [1], [4], [6], [13], [27]). The first method in use was the Differential Power Analysis (DPA), originally introduced by Kocher. However, recent works on which we mainly focus in this paper, merely suggest a statistical analysis based on the use of correlation measurements. This approach allows a better use of the information leakage, but also seems to be a natural way to proceed, allowing simple analysis, based on well-known statistical tools. We note that different solutions could be considered to mount power analysis attacks and the use of the correlation coefficient is not optimal. For example, maximum likelihood techniques [11] may yield better results. However, with the simple power consumption models considered within this survey, correlation attacks provide good results and are extremely easy to manipulate (*e.g.* they do not require any estimation of the noise in the target devices).

Several proposals have also been introduced to protect actual implementations (*e.g.* in [2], [10], [12], [15], [22], [26], [41]). These countermeasures may be inserted at different levels of a cryptographic design, *e.g.* algorithmic or physical, but in general, they only reduce the side-channel leakage and do not fundamentally prevent the attacks. As a consequence, the correct evaluation of their cost and efficiency is of primary interest. Some interesting work about the theoretical predictions of power analysis attacks and countermeasures can be found in [12], [22], [27] for smart cards and processors, but although a growing interest of the cryptographic community, a similar treatment of power analysis attacks is missing for FPGAs.

In this article, we review various aspects of recent attacks performed against FPGA implementation of encryption algorithms, with a strong focus on symmetric-key block ciphers. In particular, we define the different steps of a Correlation Power Analysis (CPA) and evaluate their practical impact in the final attack probability of success, allowing an intuitive understanding of its relevant parameters. We also discuss certain possible countermeasures to protect an implementation from these information leakages and evaluate their cost with respect to the additional security obtained. All the presented results are supported by practical experiments carried out against commercial FPGAs. Finally, we suggest certain directions for further research in the field.

• F.-X. Standaert, E. Peeters, G. Rouvroy and J.-J. Quisquater are with UCL Crypto Group, Laboratoire de Microélectronique, Université catholique de Louvain, Place du Levant, 3, B-1348 Louvain-la-Neuve, Belgium

The rest of the paper is structured as follows. Section II describes the correlation power analysis that we investigate in the article. Section III illustrates the attack principles on a very simple encryption network implemented on a FPGA. Attacks against standard algorithms are discussed in Section IV and a theoretical treatment of the correlation technique is given in Section V. Section VI evaluates the cost and efficiency of certain countermeasures. Conclusions are in Section VII.

## II. Correlation Power Analysis

### A. Description of the target device

Power analysis attacks (and more generally side-channel attacks) present a very practical threat for the security of cryptographic algorithm implementations. However, these attacks are also less general than classical cryptanalysis (*e.g.* linear [24], differential [5]) and usually target one specific circuit. For this reason, a first step in power analysis is to identify the device and implementation under attack.

In the context of this article, we investigated the specific case of FPGA implementations of block ciphers. In particular, the Data Encryption Standard (DES, [28]) and Advanced Encryption Standard Rijndael (AES, [29]) will be studied in Section IV. For clarity purposes, our theoretical predictions will also be discussed with a simple Substitution Permutation Network. Most important, the devices targeted in this report are Xilinx Virtex® and Spartan® FPGAs for which a detailed information can be found in the data sheets [47], [48].

### B. Selection of a power consumption model

In power analysis attacks, an attacker uses a hypothetical model of the device under attack to predict its power consumption. These predictions are then compared to the real measured power consumption in order to recover secret information (*i.e.* secret key bits). The quality of the model has a strong impact on the effectiveness of the attack and it is therefore of primary importance.

For example, in CMOS circuits, it is reasonable to assume that the main component of the power consumption is due to the switching activity. For a single CMOS gate, we can express it as follows [36]:

$$P_D = C_L V_{DD}^2 P_{0\to 1} f \tag{1}$$

where $C_L$ is the gate load capacitance, $V_{DD}$ the supply voltage, $P_{0\to 1}$ the probability of a $0 \to 1$ output transition and $f$ the clock frequency. Equation (1) specifies that the power consumption of CMOS circuits is data-dependent and therefore allows to mount practical attacks. However, more complex and accurate power consumption models could be considered and would consequently improve the efficiency of such attacks. Note also that the dependence of the power consumption on the data handled strongly depends on the technology considered. The power consumption in CMOS devices is proportional to the switching activity. In the case of dynamic circuits [36], power is only dissipated when the output is set to zero, which results in a different power consumption model. In more advanced technologies (*e.g.* dynamic and differential logic families [41]), the power consumption is even nearly independent of the input signals, providing a protection against power analysis attacks.

As this paper discusses the security of FPGA implementations of block ciphers and most present FPGAs are build from CMOS gates, the remaining sections are based on the following **hypothesis**: "An estimation of the FPGA power consumption at time $t$ is given by the number of bit transitions in the device registers at this time". This hypothesis was successfully used in, *e.g.* [30], [31], [37], [38]. Nevertheless, it is important to have in mind that the objective of this paper is mainly to analyze the behavior of correlation power analysis attacks from a rather theoretical point of view. Therefore, our survey pays only little attention to the measurement process in side-channel attacks. As will be emphasized later in the paper, improved power consumption models and measurement techniques could be considered and consequently increase the actual efficiency of the resulting power analysis attacks.

### C. Prediction of the device power consumption

Based on the previous hypothesis, an attacker may estimate the power consumption of a cryptographic implementation by simply predicting the number of bit transitions in the device registers. This can be done using a selection function $D$ that we define as follows. Let $X_i$ and $X_{i+1}$ be two consecutive values inside a target register (*i.e.* the register values during two consecutive clock cycles). An estimation of the target register power consumption at the time of the transition between these values is given by the function $D = H(X_i \oplus X_{i+1})$, where $H(x)$ is the Hamming weight of a bit vector $x$. An attacker who has to predict the transitions inside the registers of an implementation therefore needs to answer two basic questions:

1) Which register transitions can we predict?
2) Which register transitions do leak information?

Answering these questions determine which registers will be targeted during the attack. We formalized these questions with two definitions that we illustrate on the simple block cipher of Figure 1. Our target encryption network is a reduced version of the Khazad block cipher [3], where the S blocks represent small $4 \times 4$ non-linear substitution boxes, the P blocks represent 8-bit permutations (*i.e.* wire crossings), the D layer is a linear diffusion layer and $\oplus$ is a bitwise key addition. In addition, the grey boxes represent the registers inserted in order to pipeline the design. Remark that due to the pipeline structure, one encryption of this block cipher is performed in 9 clock cycles. The definitions are as follows:

$i$. The $predictability$ of a register is related to the number of key bits one must know to predict its transitions. For block ciphers, this depends on the size of the S-boxes and the diffusion layer. In practice, it is assumed that it is possible to guess up to 16 or 32 key bits, and the diffusion layer usually prevents guessing of more than one block cipher round. For example, the dark grey registers in Figure 1 are *predictable* (as all the other registers before the diffusion layer).

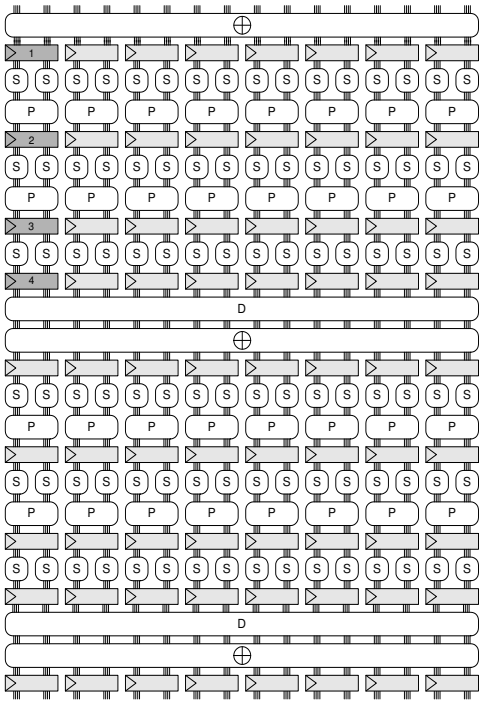


Fig. 1. Target encryption network.

$ii$. We denote a register as a $full$ (*resp.* $empty$) register if its transitions leak (*resp.* do not leak) secret information. For example, it is obvious that an input (*resp.* output) register does not leak any secret information as it only contains the plaintext (*resp.* ciphertext). However, a consequence of our prediction model is that the registers following an initial (*resp.* final) key addition do not leak information either. Indeed, the register transitions after an initial key addition can be expressed as:

$$H(input_1 \oplus key \oplus input_2 \oplus key) = H(input_1 \oplus input_2)$$

Therefore, the transitions in register 1 (see Figure 1) do not depend on the key and this register is *empty* (as all the registers before the first layer of S-boxes). We note that this observation strongly depends on the power consumption model in use and is not true in general.

Based on these definitions, the prediction of a device power consumption takes place as follows.

Let $N$ be the number of plaintext/ciphertext pairs for which the power consumption measurements are accessible. Let $K$ be the secret encryption key. During the prediction phase, the attacker selects the target registers and clock cycle for the previously defined selection function $D$. Then, he predicts the value of $D$ (*i.e.* the number of bit switches inside the target registers in the targeted clock cycle) for the $g$ possible key guesses and $N$ different plaintexts. The result of this prediction phase is an $N \times g$ **selected prediction matrix**.

In our example, the grey registers 2, 3 and 4 are *predictable* and *full*. As these registers are 8-bit long, the matrix contains numbers between 0 and $3 \times 8 = 24$ and the number of key guesses necessary to predict these transitions is $g = 2^8 = 256$. Remark that we selected these registers for illustration purposes and any set of *predictable* and *full* registers can be used to mount an attack. In addition, targeting registers 2,3 and 4 only allows to obtain eight key bits and a complete key recovery involves to repeat the predictions for the other key bits. In Figure 1, there are eight parallel S-boxes and therefore eight prediction steps will be necessary.

For theoretical purposes, it is finally interesting to define the $N \times 1$ **global prediction vector** that contains the number of bit switches inside all the device registers, in the targeted clock cycle for $N$ different plaintexts. This is only feasible if the key is known (*i.e.* when simulating the attacks).

In our example, the design contains $8 \times 9 = 72$ 8-bit registers, and the global prediction vector values are between 0 and $8 \times 72 = 576$.

### D. Measurement of the device power consumption

During the measurement phase, the attacker lets the device encrypt the same $N$ plaintexts as during the prediction phase, with one secret key. While the chip is operating, he measures the power consumption for the different encryptions and stores the power consumption value for the targeted clock cycle[1]. As a result of the measurement phase, the attacker obtains an $N \times 1$ **global consumption vector** with the values of the power consumption during the targeted clock cycle, for $N$ different plaintexts.

### E. Correlation analysis

In the final phase of a power analysis attack, the attacker compares the theoretical predictions of the power consumption with its real measurements. For this purpose, a practical solution, used in several papers and intensively discussed in [6], is to compute the correlation coefficient between the global consumption vector and all the columns of the selected prediction matrix (corresponding to all the $g$ possible key guesses). If the attack is successful, it is expected that only the correct key guess leads to a correct prediction of the power leakage and thus to a high correlation value.

An efficient way to perform the correlation between theoretical predictions and real measurements is to use the Pearson coefficient (see [18]). Let $M(i)$ denote the $i$th measurement data (*i.e.* the $ith$ trace) and $M$ the set of traces. Let $P(i)$ denote the prediction of the model for the $i$th trace and $P$ the set of such predictions. Then we calculate:

$$C(M,P) = \frac{\mu_{M.P} - \mu_M . \mu_P}{\sigma_M . \sigma_P} \qquad (2)$$

where $\mu_M$ denotes the mean of the set of traces $M$ and $\sigma_M^2$ its variance. If this correlation is high, it is usually assumed that the prediction of the model, and thus the key hypothesis, is correct.

[1]Measurement setups for power analysis attacks have already been intensively described in the open literature. A usual method is to observe the voltage variations over a small resistor inserted in the supply circuit of the cryptographic device. Those setups are out of the scope of this survey.

Finally, theoretical predictions of the attack can be performed by using the global prediction matrix instead of the global consumption matrix. As the global prediction matrix contains the number of bit switches inside all the registers, it represents a theoretical noise free measurement and may help to determine the minimum number of texts needed to mount a successful attack. This scenario is referred to as an attack using simulated data in the following sections.

## III. An illustrative attack

This section illustrates our descriptions with some experiments performed against an FPGA implementation of the block cipher represented in Figure 1.

### A. An attack using simulated data

In the attack using simulated data, we chose $N = 1000$ random plaintexts and one secret key and we produced the selected prediction matrix and global prediction vector, as defined in the previous section. Thereafter, we performed the correlation phase between these two matrixes. As the relevant information to determine is the minimum number of plaintexts necessary to extract the correct key, we calculated the correlation coefficient for different values of $N$: $1 \leq N \leq 1000$. In order to underline the importance of clearly setting the attacker capabilities, we also considered two experiments. A first one where the selected prediction matrix contained the transitions in register 4 only (in Figure 2) and a second one where it

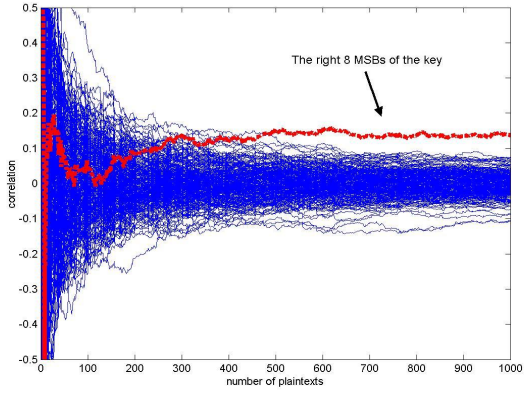


Fig. 2. A simulated attack using predictions for register 4 only.

contained the transitions in registers 2, 3 and 4 (in Figure 3). We can observe in the figures that both attacks are successful, but the second experiment is significantly faster. In practice, the required number of plaintexts is about respectively 600 and 300, confirming that different attacker capabilities (*i.e.* different knowledge of the design) may yield different threats.

### B. An attack using measured data

When attacking a device practically, the selected prediction matrix stays unchanged (we predicted transitions in registers 2, 3 and 4, as in Figure 3) while we replace the global prediction vector by the global consumption vector. Therefore, we let the

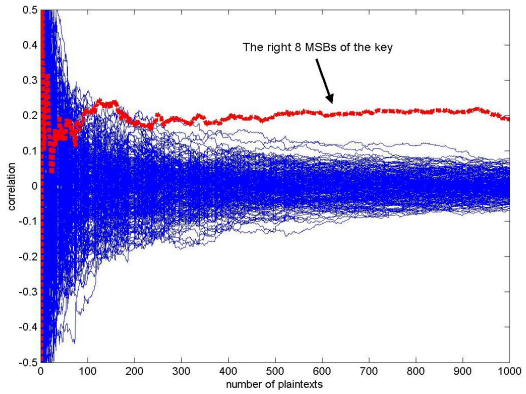


Fig. 3. A simulated attack using predictions for register 2,3,4.

FPGA encrypt 2000 plaintexts with the same key as we did in the previous section and produced the matrix as described in Section II-C.

To evaluate the quality of our measurements, we made a preliminary experiment and computed the correlation coefficient between the global prediction vector and the global consumption vector, for different number of measurements: $1 \leq N \leq 2000$. As illustrated in Figure 4, the correlation between both vectors is approximately 0.45, confirming our hypothesis to provide a reasonable estimation of the device power consumption. Also, the correlation is not perfect (*i.e.* equal to one), confirming that the power consumption model is not perfect. As already suggested in Section II-B, improved models and measurement tools could be considered, *e.g.* using simple signal processing techniques to improve the quality of the results. As an illustration, in [40], the use of averaging and filtering is investigated and some more specific power consumption models are proposed.

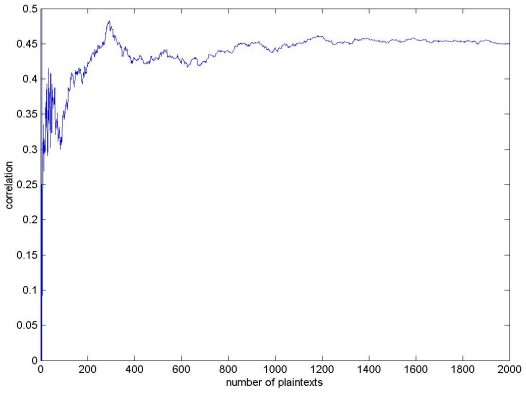


Fig. 4. Preliminary experiment.

In order to identify the correct key guess, we used the correlation coefficient again. As it is shown in Figure 5, the correct key guess is distinguishable after about 1200 traces. As a consequence, the attack is practically successful, *i.e.* the selected prediction matrix is sufficiently correlated with the real measurements and we can extract key information.

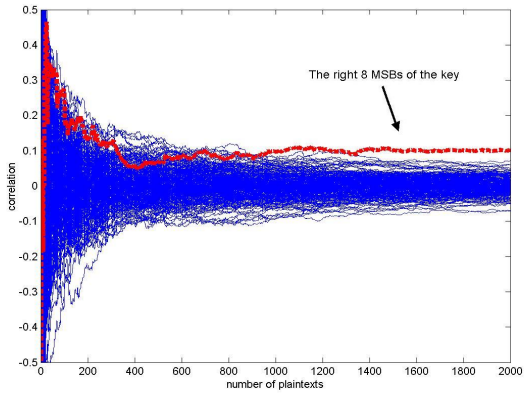


Fig. 5. An attack using real measurements.

## IV. Attacks targeting standard algorithms

The techniques described in the previous sections have been successfully applied to a variety of cryptographic algorithms, including the DES in [37] and AES Rijndael in [31], [38]. In particular, reference [38] relates the security of an implementation to efficiency considerations and evaluates the effect of pipelining and unrolling techniques in this context. It is notably demonstrated that pipelining a loop implementation does not provide an effective countermeasure if an attacker has access to the design details because most of the registers in the pipeline remain predictable. On the other hand, the combination of pipelining and unrolling techniques may counteract power analysis attacks as a random noise generator, because only the outer rounds of such an implementation can then be predicted.

A particular advantage of the correlation power analysis used in these references is the possibility to obtain "theoretical predictions" of the attacks, using simulated data. However, in practice, these predictions require the computation of a fastidious amount of correlation values (typically $g \times N$, where $g$ is the number of key guesses considered) and are specific to one single implementation, device, secret key and selection of plaintexts. As a consequence, a statistical approach to evaluate a circuit security would be relevant.

An interesting work about the theoretical evaluation of power analysis attacks and countermeasures was proposed in [22]. In the following section we apply and extend this statistical analysis to FPGA implementations.

## V. Further theoretical predictions

### A. Definitions

We start with a few classical definitions for which we assume that the block cipher rounds behave like a random number generator. In practice, this is only true after a few rounds, when the diffusion is complete. Based on this hypothesis, the number of bit switches in the cryptographic design registers are distributed as a binomial that we approximated with a normal distribution.

Let $G$ be a normal random variable, with parameters $\mu_G$ and $\sigma_G^2$, representing the global transitions in the cryptographic design registers. If the device contains $n$ 1-bit registers, we have $\mu_G = n/2$ and $\sigma_G^2 = n/4$.

Let $P_i$ be a normal random variable, with parameters $\mu_{P_i}$ and $\sigma_{P_i}^2$, representing the *predictable* transitions in the cryptographic device *full* registers, for a fixed key guess $i$, $i \in [0,\ g-1]$.

Let $U$ be a normal random variable, with parameters $\mu_U$ and $\sigma_U^2$, representing the unknown (or empty[2]) transitions in the cryptographic device registers.

Let $M$ be a normal random variable, with parameters $\mu_M$ and $\sigma_M^2$, representing the measured power consumption of the cryptographic device.

From these definitions, we consider $P$ and $U$ as independent normal random variables such that:

$$G = P + U$$
$$\mu_G = \mu_P + \mu_U$$
$$\sigma_G^2 = \sigma_P^2 + \sigma_U^2$$

Finally, we remember the correlation coefficient definition:

$$r_{X,Y} = \frac{\mu_{X.Y} - \mu_X.\mu_Y}{\sigma_X.\sigma_Y}$$

### B. Evaluation of the correlation coefficient $r_{P,G}$

In order to evaluate the success rate of the correlation analysis, we used the following theorem, demonstrated in [7]:

***Theorem:*** *The average correlation coefficient between the sum of $n$ arbitrary independent identically distributed random variables and the sum of the first $m < n$ of these equals $\sqrt{m/n}$.*

Therefore, if a cryptographic design contains $n$ 1-bit registers, from which $m$ are *predictable* and *full*, we approximate the correlation coefficient value between variables $G$ and $P$ by:

$$r_{P,G} \simeq \sqrt{m/n}$$

As an illustration, in Figure 2, we predict 8 bits out of 576 and the correlation coefficient value is $r_{P,G} \simeq \sqrt{8/576} = 0.12$. Similarly, in Figure 3, we predict 24 bits out of 576 and the correlation coefficient value is $r_{P,G} \simeq \sqrt{24/576} = 0.20$. Remark again that these predictions implicitly assume that the block cipher rounds generate random intermediate values. In many applications, this hypothesis leads to practical attacks.

### C. Distribution of the correlation coefficient

The sampling distribution of a Pearson correlation coefficient $r$ is best described by transforming $r$ to a variable $z$ such that:

$$z = \frac{1}{2} \log \frac{1+r}{1-r}$$

[2]For simplicity, our attacks did not take advantage of *predictable* and *empty* register transitions and those where consequently included in $U$. However, in practice, those transitions could be removed from $U$ in order to decrease the algorithmic noise, see Section V.G. In the latter case, the correlation we would be interested in is $r_{P,P+U}$ rather than $r_{P,G}$.

This Fisher-transformed correlation coefficient is normally distributed with good approximation, even for small values of $N$, with standard deviation (see [18]):

$$\sigma_z(N) = \frac{1}{\sqrt{N-3}} \tag{3}$$

### D. Success rate of the attack using simulated data

Let $C$ be a normal random variable with parameters $\mu_C$ and $\sigma_C^2$, representing the Fisher-transformed correlation coefficient between the global transitions $G$ and the correct partial prediction $P_i$ of these transitions (*i.e.* corresponding to the correct key guess). We approximated $\mu_C$ with the previously computed correlation coefficient value, $\mu_C = r_{P,G}$. The variance $\sigma_C^2$ is estimated according to Equation (3).

Let $W$ be a normal random variable with parameters $\mu_W$ and $\sigma_W^2$, representing the Fisher-transformed correlation coefficient between the global transitions $G$ and a wrong partial prediction $P_i$ of these transitions (*i.e.* corresponding to a wrong key guess). For such a wrong key candidate, we have $\mu_W = r_{P,G} = 0$ and $\sigma_W^2 = \sigma_C^2$.

The success rate of a correlation analysis using simulated data depends on the probability that we can distinguish the correlation coefficient of a correct key guess $C$ from the correlation coefficient of a wrong key guess $W$. In practice, if there are $g$ possible key guesses to compare and assuming that these are independent experiments[3], the success rate is approximated by:

$$SR \simeq \mathrm{P}[C > W]^{g-1}$$

For evaluating *SR*, we assume again that $C$ and $W$ are independent random variables. Therefore, we can define a new normal random variable $\Delta r = C - W$, with parameters $\mu_{\Delta r} = \mu_C - \mu_W$ and $\sigma_{\Delta r}^2 = \sigma_C^2 + \sigma_W^2$. It is clear that:

$$P[C > W] = P[\Delta r > 0]$$

And therefore we have:

$$SR \simeq \Big( \int_0^\infty \frac{1}{\sigma_{\Delta r}\sqrt{2\pi}} \exp - \frac{(x-\mu_{\Delta r})^2}{2\,\sigma_{\Delta r}^2}\, dx \Big)^{g-1}$$

### E. Success rate of the attack using measured data

As far as measured data are concerned, the attacker only has to replace the global prediction matrix by the global consumption matrix. According to the definitions of Section V-A, it means that he has to compute $r_{P,M}$ rather than $r_{P,G}$. Because we have the conditional independence between $P$ and $M$ (*i.e.* knowing the global prediction $G$, there is nothing to gain in knowing the global consumption $M$), a simple expression for this coefficient can be derived (demonstrated in the Appendix):

$$r_{P,M} = r_{P,G} \times r_{G,M} \tag{4}$$

[3]This is clearly not the case in reality and some wrong key guesses may generate transitions correlated with the correct key guess transitions. However, it is a commonly used assumption in cryptanalysis, *e.g.* in linear/differential cryptanalysis. Moreover, these correlated key guesses could be taken into account by simply using $\mu_W \neq 0$.

As an illustration, in Figure 3, we observe that $r_{P,G} \simeq 0.20$ and in Figure 4, we observe that $r_{G,M} \simeq 0.45$. According to Equation (4), we should find $r_{P,M} \simeq 0.20 \times 0.45 = 0.09$, as it is confirmed in Figure 5. In this equation, the coefficient $r_{G,M}$ only relates to the quality of the measurement and is independent of the attack considered. It is an intrinsic characteristic of the measurement setup that has to be estimated once. On the contrary, the coefficient $r_{P,G}$ is specifically related to the implementation under attack and depends on the number of register transitions that can actually be predicted. Using this expression for the correlation coefficient, the success rate of any correlation attack can be estimated with Algorithm 1.

To confirm this analysis, we evaluated the success rate of the correlation attack using real measurements that is represented in Figure 5, for different number of measurements: $1 \leq N \leq 3000$. This predicted success rate is shown in Figure 6, where we clearly observe that the attack is successful after approximately 1200 plaintexts.

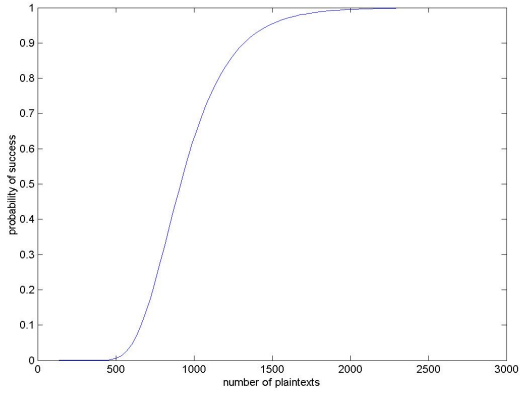


Fig. 6. Theoretical prediction of the success rate with $r_{P,M} = 0.09$.

### F. A simple model

The previous considerations (summarized in Algorithm 1), finally allowed us to compute the number of plaintexts necessary to have a successful attack (*i.e.* an attack for which $SR = 0.9$), in function of the correlation coefficient value. It is represented in Figure 7. From this final experiment, we observed that the number of plaintexts $N_{0.9}$ required to mount a correlation analysis attack can simply be estimated with:

$$N_{0.9} \simeq c \times \frac{1}{r_{P,M}^2} \tag{5}$$

where $c$ is a constant depending on the number of key guesses considered and the required success rate. In our example, $g = 256$, the required success rate is 0.9 and a practical value for $c$ is approximately 10.

As an illustration, the attack of Figure 2 has a correlation coefficient of $r_{P,G} \simeq \sqrt{8/576} = 0.12$ and is successful after about 600 measurements. The attack of Figure 3 has a correlation coefficient of $r_{P,G} \simeq \sqrt{24/576} = 0.20$ and is successful after approximately 300 measurements. Finally, the attack of Figure

**Algorithm 1 Theoretical prediction of the correlation power analysis**

1. Determine the total number of 1-bit registers in the FPGA design, $n$.
2. Determine the number of *predictable* and *full* such registers, $m$.
3. Determine the correlation coefficient value, $r_{P,G} \simeq \sqrt{m/n}$.
4. Estimate the measurement quality $r_{G,M}$. A typical value is 0.5. Use $r_{G,M} = 1$ in case of an attack using simulated data.
5. Determine $r_{P,M} = r_{P,G} \times r_{G,M}$.
6. Determine the number of key guesses $g$.
7. Determine the number of measurements (*i.e.* plaintexts) available, $N$.
8. The success rate is approximated by:

$$SR \simeq \left( \int_0^{\infty} \frac{1}{\frac{1}{\sqrt{N-3}}\sqrt{2\pi}} \exp - \frac{(x - r_{P,M})^2}{\frac{2}{N-3}} \, dx \right)^{g-1}$$

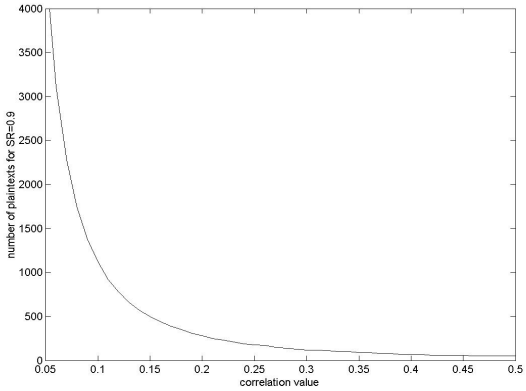


Fig. 7. Theoretical prediction of the CPA with $SR = 0.9$ and $g = 256$.

5 has a correlation coefficient of $r_{P,M} \simeq 0.20 \times 0.45 = 0.09$ and is successful after about 1200 measurements. These results confirm that our theoretical analysis allows a good prediction of the attack success probability.

Remark that these theoretical predictions allow us to clearly relate power analysis attacks with classical cryptanalysis. In particular, the final estimation of Equation (5) is very similar to the final estimation for linear cryptanalysis, where $r_{P,M}$ is replaced by the probability of a linear approximation [24]. This suggests that some problems could be commonly solved for classical and side-channel cryptanalysis, *e.g.* the problem of correlated key guesses in linear/differential cryptanalysis and the problem of "ghost peaks" in side-channel attacks, as explained in [6].

### G. Links with previous works

In different published works attempting to describe the behavior of circuits with respect to power analysis attacks (*e.g.* [12], [22], [27]), a useful measurement is the signal-to-noise ratio (SNR) of the attack. In order to relate this SNR with our previous definitions, we first use the context of an attack using simulated data in which we have $G = S + N_a$. Here, $G$ represents a noise-free measurement of the power consumption (*i.e.* the previously defined global consumption matrix), $S = P$ is the signal (*i.e.* the *predictable* and *full* transitions) and $N_a = U$ is the algorithmic noise (*i.e.* the noise produced by unpredictable transitions in the target design). *Since the DC components of $P$ and $U$ are not relevant for the calculation of the correlation coefficient, only the AC components (i.e. the variances) of the signals are considered in this equation [22]*:

$$SNR = \frac{\sigma_S^2}{\sigma_{N_a}^2} \tag{6}$$

The lower the SNR is, the lower is also the correlation between the correct partial prediction of the power consumption and the power consumption of the device.

This SNR and the correlation coefficient of an attack using simulated data are simply related by the following equation:

$$r_{P,G} = \frac{1}{\sqrt{1 + \frac{1}{SNR}}}$$

It is important to remark that in an attack using simulated data, the noise is *only* algorithmic and thus produced by the unknown transitions in the device (*i.e.* $r_{G,M} = 1$). However, in practice, noise is also physical and induced by the measurements (*i.e.* $r_{G,M} < 1$). It can be written as the sum of the previously defined algorithmic noise and a physical noise: $N = N_a + N_p$. In this latter case, the SNR can simply be derived from Equation 4.

## VI. Countermeasures

Although numerous countermeasures have been proposed in the open literature, protecting implementations against power analysis is usually difficult and expensive. Moreover, most proposals only reduce the side-channel leakage and do not fundamentally prevent the attacks. In this context, the implementation cost of a countermeasure is of primary importance and must be evaluated with respect to the additional security obtained. This section provides a survey of side-channel countermeasures and discusses their applicability to FPGA implementations. In particular, we focus on (what is usually assumed to be) four of the most practical and efficient countermeasures, *i.e.* time randomization, noise addition, masking and dynamic and differential logic styles.

### A. Randomized countermeasures

Historically, the use of random process interrupts, clock skipping and dummy instructions was one of the first proposals to foil DPA techniques. A typical example was

presented at CHES 2001 (in [25]) and proposed to rename the registers randomly in order to hide the secret keys stored in a smart card. The extension of such ideas is a *non-deterministic* processor based on super-scalar architectures, for which the efficiency is a serious concern. Randomized exponentiation for modular exponentiation algorithms [46] and randomized addition chains for elliptic curve cryptography [17], [32] are additional examples of this kind of countermeasures.

However, if the resulting operational behavior of the circuit can be modeled by a probabilistic finite state machine, references [19], [33] demonstrated that the randomization can be analyzed. It notably allowed recovering the secret key of two randomized exponentiation algorithms proposed by Oswald and Aigner in [32]. More generally, re-synchronization techniques [12] usually allow bypassing the randomization. Moreover, the implementation cost in terms of resources and clock cycles is an additional bottleneck of these proposals. The practical effectiveness of such a countermeasure is discussed in [22].

### B. Noise addition

Noise addition is another traditional solution to counteract power analysis. It has the advantage of being relatively simple and it can be an effective way to resist attacks in practice. Although it does not provide any fundamental protections (the signal remains present and can still be recovered), its practical impact is easily evaluated by a simple statistical analysis. In general, noise addition may be expensive to implement and is obviously not an energy efficient solution. However, in the context of FPGA implementations, it is possible to add noise in a well chosen way so that we do not reduce the hardware efficiency, for example by using unrolled and pipelined implementations combined with additional designs running on the same circuit. Remark that combining noise addition and randomization can be an efficient (and relatively cheap) way to resist attacks in practice.

### C. Duplication and Boolean masking

*1) Description:* A general method to thwart DPA is to "mask" or "duplicate" all the intermediate data inside an implementation, so that the power consumption becomes unpredictable. These strategies are possible if all the fundamental operations used in a given algorithm can be rewritten in the masked or duplicated domain. This is easily seen to be the case in classical algorithms such as the DES or AES (see [2], [15]). Although these methods have been originally applied at the algorithmic level as well as at the gate level (*e.g.* in [44]), it has been shown recently that masking at the gate level involves critical security concerns. Reference [23] notably demonstrates that the glitching activity of masked logic gates offers a previously neglected leakage that seriously affects the security of the countermeasure. For this reason, this section will mainly discuss duplication and masking at the algorithmic level, using precomputed tables.

In a masked implementation, a random Boolean vector $r$ (denoted as the "mask") is XORed to the input data before applying the algorithm. Thereafter, during the algorithm execution, the data is always masked with random values. For example, a simple masked scheme is illustrated in Figure 8 for a key addition followed by an S-box. In this scheme, the $S'$ box allows the outputs of the encryption network to be masked with a known value $q$.

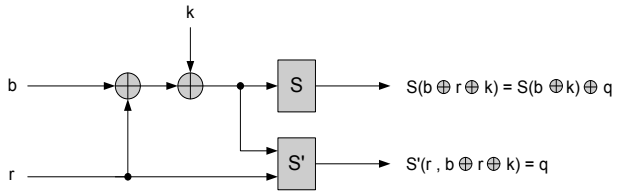


Fig. 8. Masked scheme.

In practice, for small S-box sizes a simple solution is to precompute $S'$ and store it into a large memory. As a typical example, a masked DES design can be efficiently implemented into FPGAs because its S-boxes are 6-bit wide. On the other hand, masking the AES Rijndael with a similar technique will require a prohibitive amount of memory. Solutions exist to improve the efficiency of the countermeasure in this latter context, *e.g.* in [34]. However, duplication and masking generally remain an expensive solution.

Remark from this description that the security of such a protection strongly relies on the fact that the mask is randomly updated for every new encryption. Then, the power analysis attack described in the previous sections is no more applicable, because the power consumption is not predictable in function of the key.

*2) Security against higher-order attacks:* Considering the security of a masked or duplicated implementation, it has been shown that they remain vulnerable to higher-order power analysis attacks. In general, higher-order attacks take advantage of some key-dependent statistical distributions of the power consumption in an actual design. While the original work of Messerges [26] was somewhat specific and only applied to smart cards, [45] demonstrated that higher-order power analysis is possible, without any additional hypothesis than usually assumed for first-order attacks. [39] proposed an extension of these techniques by considering a more general power consumption model and applied it to FPGAs. Higher-order attacks were finally improved in [35], using an approach based on a maximum likelihood recovery of the secret key. As a consequence of these results, one may conclude that duplication and masking do not sufficiently improve the security of a block cipher against side-channel attacks. Such implementations can be targeted in a practically tractable number of measurements. Again, however, the combination of duplication or masking with other countermeasures can lead to a certain level of practical security. The exact statistical evaluations of these attacks is a scope for further research.

### D. Dynamic and differential logic styles

As most DPA protections only reduce the side-channel leakage and do not fundamentally prevent a power analysis attack, an interesting alternative is to use a logic style for which the power consumption is independent of the data handled. Although it does not provide a theoretical countermeasure either (small power variations still appear in function of the input sequences), it has the advantage of making the attack significantly harder. Moreover, this solution can be combined with good performances if a good logic style is chosen.

A logic style is usually denoted as differential if the complementary data inputs and outputs are available in the circuit. The notion of dynamic logic gates refers to the fact that the gate operation is divided into two phases [36]. First, the output capacitance is charged. Then, during the evaluation, it is discharged according to the input values. When combining dynamic and differential logic styles, the charge and discharge of the output capacitance is therefore independent of the input data. As the output signal and its complement are available, there are always two capacitances loaded during the precharge and one of them is discharged during the evaluation, regardless of the input sequences.

Examples of such logic styles are discussed in [21], [41] with respect to side-channel concerns. In order to evaluate their actual security, it is important to remember that a power analysis efficiency depends on:

- The possibility to predict the power consumption of a device in function of its input data.
- The correlation between a theoretical prediction of the power consumption and its real measurement.

It is clear that the attack was applicable to CMOS devices because their power consumption significantly varies in function of their input data and can be easily predicted by simply evaluating the number of bit flips in the circuit. Regarding dynamic and differential logic styles, there are two effects that are susceptible to counteract power analysis.

First, the value of the power consumption normalized standard deviation (NSD) can be decreased and consequently increases the difficulty of having good measurements. From a purely theoretical point of view, this does not affect the attack efficiency because only the correlation values are relevant with this respect. That is, *under the assumption that an attacker can perfectly predict and measure the power consumption*, a circuit resistance is equal for any logic style. This is a simple consequence of the attack SNR defined in Equation 6. Indeed, if we only consider the algorithmic noise, decreasing the power consumption variances will affect all the design S-boxes (*i.e.* both the signal and the algorithmic noise) in exactly the same way. Nevertheless, *measurements are not perfect*. Regarding practical attacks, the measurement noise remains constant for any logic style and therefore causes a reduction of the SNR. Unfortunately, this observation highly depends on the attacker measurement setup and its theoretical evaluation is hard.

From a practical point of view, a second critical concern is *the predictability of the power consumption*, which *may not be perfect either*. To understand this last statement, one should remember the origin of the power consumption differences in the different logic families. In CMOS gates, the useful component of the power consumption is dynamic and depends on the probability of a $0 \rightarrow 1$ output transition. The consumption differences directly depend on the load (or lack thereof) of the output capacitance and therefore, are predictable in function of the input transitions without any knowledge about the circuit design.

In case of dynamic and differential circuits, the situation strongly differs because the output capacitance is loaded independently of the input transitions. The consumption differences are due to the presence of parasitic capacitances in the design and therefore, they cannot be predicted without a precise "transistor-level" knowledge of the circuit. As a consequence, an attacker can only target one specific implementation and preliminarily needs to build a table containing the power consumption differences in function of the circuit input data ($i.e.$ an information that is usually not made available to the users). Therefore *the correlation values will be reduced according to the precision of the power consumption model used for the predictions*. At this point also, the NSD probably have a practical impact and this would require further research, *e.g.* on the exact relation between the power consumption model and the logic style. Note again that, in theory, a precise power consumption model could always be obtained in the context of template attacks [11], *e.g.* using artificial neural network techniques. Therefore, the countermeasure is only expected to increase the complexity of a power attack and does not prevent it theoretically.

For circuit complexity reasons, these logic styles are not susceptible to be used in reconfigurable hardware devices. However, a similar behavior (*i.e.* dynamic and differential) can be obtained at the gate-level in FPGAs, as suggested in [42]. Compared to an original FPGA design, the modified synthesis procedure involves delay and area increases that make the proposal somewhat comparable to duplication in terms of efficiency. The secure and efficient combination of these gate-level masking methods would be worth further research.

### E. Other solutions

The previous subsections underlined that obtaining resistance (even in practice) against power analysis attacks is challenging. Actual security can be improved by the combination of different countermeasures, but such an approach does not provide theoretical security. In general, a unified evaluation of side-channel countermeasures is an interesting scope for further research. Remark that in addition to the protections described in this article, a number of other possibilities exist and have not yet been formally investigated. We mention the following examples:

1. The use of re-keying techniques. As any cryptanalytic technique, power analysis attacks require the access to a number of power consumption measurements and all these measured encryptions must have been performed with the same key. As a consequence, a straightforward countermeasure is to use encryption modes where the key is changed sufficiently often. With respect to efficiency, this involves specific requirements for the implementations in terms of key agility. Regarding security, although other attack contexts (*e.g.* template attacks) theoretically allow targeting such implementations, this solution could probably defeat certain attackers.

2. The modification of present block cipher structures. For example, the addition of a preliminary transform to the cipher such that its output values would be *unpredictable* (because of high diffusion) and *empty* would be of particular interest. Secret permutations (*i.e.* wire crossings) could be a convenient tool with this respect.

## VII. Conclusions

Power analysis attacks (and more generally side-channel attacks) present a very practical threat for the security of cryptographic algorithm implementations. However, these attacks are also less general than classical cryptanalysis and usually target one specific circuit. For this reason, it is extremely important to determine what an attacker is able to do and what knowledge of the design can be used. These assumptions allow a developer to have a framework which helps him to choose efficient countermeasures. It is also important to consider the security of an embedded platform as a whole, no level being excluded from the analysis.

In general, comparing FPGA designs and software implementations, it must be observed that it is basically more difficult to attack hardware than software because parallel computing causes a dilution of the attacks SNR. High work frequencies can also make the sampling process critical. This is specially true if the FPGA does not only act as an encryption machine, but combines different digital signal processing applications, *e.g.* compression, watermarking, filtering, ... The discussions of this article allow one to evaluate the effect of complex designs onto the attack feasibility and underline that it may become a bottleneck for certain attackers.

However, in a highly secure context, no single countermeasure presently provides theoretical security. Rather, the combination of different techniques (including noise addition, randomization, duplication, masking, ...) allows reaching a certain level of actual security. A rigorous statistical evaluation of side-channel countermeasures is an interesting scope for further research. In a long term perspective, the need of provably secure implementations against side-channel attacks is also a serious concern.

## Appendix

### Computation of the correlation coefficient $r_{P,M}$

Let us assume that the triple $(P, M, G)$ has a multidimensional normal distribution that we express as follows:

$$\begin{pmatrix} P \\ M \\ G \end{pmatrix} = \mathbf{N}\left(\begin{pmatrix} \mu_P \\ \mu_M \\ \mu_G \end{pmatrix}, \begin{pmatrix} \sigma_P^2 & \rho_1 & \rho_2 \\ \rho_1 & \sigma_M^2 & \rho_3 \\ \rho_2 & \rho_3 & \sigma_G^2 \end{pmatrix}\right)$$

This can be rewritten as:

$$\left(\begin{array}{c} P \\ M \\ \hline G \end{array}\right) = \mathbf{N}\left(\begin{pmatrix} \mu_P \\ \mu_M \\ \mu_G \end{pmatrix}, \left(\begin{array}{cc|c} \sigma_P^2 & \rho_1 & \rho_2 \\ \rho_1 & \sigma_M^2 & \rho_3 \\ \hline \rho_2 & \rho_3 & \sigma_G^2 \end{array}\right)\right)$$

And thus

$$\left(\begin{array}{c} P \\ M \\ \hline G \end{array}\right) = \mathbf{N}\left(\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \sum_{12} \\ \sum_{21} & \sum_{22} \end{pmatrix}\right)$$

From these expressions, we can compute the conditional distribution $(P, M)|G$. According to [16], this conditional distribution is normal with mean $\mu_1 + \sum_{12} \sum_{22}^{-1} (G - \mu_2)$ and covariance matrix $\sum_{11} - \sum_{12} \sum_{22}^{-1} \sum_{21}$.

Therefore, the covariance matrix equals:

$$\begin{pmatrix} \sigma_P^2 & \rho_1 \\ \rho_1 & \sigma_M^2 \end{pmatrix} - \begin{pmatrix} \rho_2 \\ \rho_3 \end{pmatrix} \frac{1}{\sigma_G^2} \begin{pmatrix} \rho_2 & \rho_3 \end{pmatrix}$$

Which is equivalent to:

$$\begin{pmatrix} \sigma_P^2 - \frac{\rho_2^2}{\sigma_G^2} & \rho_1 - \frac{\rho_2 \rho_3}{\sigma_G^2} \\ \rho_1 - \frac{\rho_2 \rho_3}{\sigma_G^2} & \sigma_M^2 - \frac{\rho_3^2}{\sigma_G^2} \end{pmatrix}$$

Finally, we simply observe that if an attacker knew the global transitions $G$, there would be nothing to gain in knowing the global consumption $M$. This means that we have the conditional independence between $P$ and $M$ that we can express as follows:

$$P \perp\!\!\!\perp M|G \Rightarrow \rho(P, M|G) = 0$$

The condition on the covariances is therefore:

$$\rho_1 - \frac{\rho_2 \rho_3}{\sigma_G^2} = 0$$

And for the correlation coefficients, we find:

$$r_{P,M} = r_{P,G} \times r_{G,M}$$

## References


[1] M.L. Akkar, R. Bevan, P. Dischamp, D. Moyart, *Power Analysis, What Is Now Possible*, in the proceedings of Asiacrypt 2000, Lecture Notes in Computer Science, vol 1976, pp 489-502, Kyoto, Japan, December 2000, Springer-Verlag.

[2] M.L. Akkar, C. Giraud, *An Implementation of DES and AES Secure againts Some Attacks*, in the proceedings of CHES 2001, Lecture Notes in Computer Sciences, vol 2162, pp 309-318, Paris, France, May 2001, Springer-Verlag.

[3] P. Barreto, V. Rijmen, *The KHAZAD Legacy-Level Block Cipher*, Submission to NESSIE project, available from http://www.cosic.esat.kuleuven.ac.be/nessie/.

[4] R. Bevan, E. Knudsen, *Ways to Enhance Differential Power Analysis*, in the proceedings of ICISC 2002, Lecture Notes in Computer Science, vol 2587, pp 327-342, Seoul, Korea, November 2002, Springer-Verlag.

[5] E. Biham, A. Shamir, *Differential Cryptanalysis of the Data Encryption Standard*, Springer Verlag, 1993.

[6] E. Brier, C. Clavier, F. Olivier, *Correlation Power Analysis with a Leakage Model*, in the proceedings of CHES 2004, Lecture Notes in Computer Science, vol 3156, pp 16-29, Boston, USA, August 2004, Springer-Verlag.

[7] W. Bryc, A. Dembo, A. Kagan, *On the Maximum Correlation Coefficient*, Technical Report of the Departement of Statistics, Stanford University, 2002-25, August 2002.

[8] P. Buysschaert, E. De Mulder, S. B. Örs, P. Delmotte, B. Preneel, G. Vandenbosch, I. Verbauwhede, *Electromagnetic Analysis Attack on an FPGA Implementation of an Elliptic Curve Cryptosystem*, in the proceedings of EUROCON 2005 - The International Conference on Computer as a Tool, IEEE, 4 pages, 2005.

[9] V. Carlier, H. Chabanne, E. Dottax, H. Pelletier, *Electromagnetic Side Channels of an FPGA Implementation of AES*, IACR e-print archive 2004/145, http://eprint.iacr.org, 2004, 2004.

[10] S. Chari C. Jutla, J. Rao, P. Rohatgi, *Towards Sound Approaches to Counteract Power-Analysis Attacks*, in the proceedings of Crypto 1999, Lecture Notes in Computer Science, vol 1666, pp 398-412, Santa Barbara, California, USA, August 1999, Springer-Verlag.

[11] S. Chari, J. Rao, P. Rohatgi, *Template Attacks*, in the proceedings of CHES 2002, Lecture Notes in Computer Science, vol 2523, pp 13-28, Redwood Shores, California, USA, August 2002, Springer-Verlag.

[12] C. Clavier, J.S. Coron, N. Dabbous, *Differential Power Analysis in the Presence of Hardware Countermeasures*, in the proceedings of CHES 2000, Lecture Notes in Computer Sciences, vol 1965, pp 252-263, Worcester, Massachusetts, USA, August 2000, Springer-Verlag.

[13] J.S. Coron, P. Kocher, D. Naccache, *Statistics and Secret Leakage*, in the proceedings of Financial Crypto 2000, Lecture Notes in Computer Science, vol 1972, pp 157-173, Anguilla, British West Indies, February 2000, Springer-Verlag.

[14] J. Daemen, V. Rijmen, "*The Design of Rijndael. AES - The Advanced Encryption Standard*," Springer-Verlag, 2001.

[15] L. Goubin, J. Patarin, *DES and Differential Power Analysis*, in the proceedings of CHES 1999, Lecture Notes in Computer Science, vol 1717, pp 158-172, Worcester, Massachussets, USA, August 1999, Springer-Verlag.

[16] F.A. Graybill, *Theory and Application of the Linear Model*, DuxBury Press, 1976.

[17] J.-C. Ha, S.-J. Moon, *Randomized Signed-Scalar Multiplication of ECC to Resist Power Attacks*, in the proceedings of CHES 2002, Lecture Notes in Computer Science, vol 2523, pp 551-563, Redwood Shores, CA, USA, August 2002, Springer-Verlag.

[18] A. Hald, *Statistical Theory with Engineering Applications*, Wiley, 1952.

[19] C. Karlof, D. Wagner, *Hidden Markov Model Cryptanalysis*, in the proceedings of CHES 2003, Lecture Notes in Computer Sciences, vol 2779, pp 17-30, Cologne, Germany, September 2003, Springer-Verlag.

[20] P. Kocher, J. Jaffe, B. Jun, *Differential Power Analysis*, in the proceedings of CRYPTO 99, Lecture Notes in Computer Science, vol 1666, pp 398-412, Santa Barbara, USA, August 1999, Springer-Verlag.

[21] F. Mace, F.-X. Standaert, I. Hassoune, J.-D. Legat, J.-J. Quisquater, *A Dynamic Current Mode Logic to Counteract Power Analysis Attacks*, in the proceedings of DCIS 2004.

[22] S. Mangard, *Hardware Countermeasures against DPA - A Statistical Analysis of Their Effectiveness*, in the proceedings of CT-RSA 2004, Lecture Notes in Computer Science, vol 3376, pp 222-235, San Francisco, USA, February 2004, Springer-Verlag.

[23] S.Mangard, *Side-Channel Leakage of Masked CMOS Gates*, in the proceedings of CT-RSA 05, Lecture Notes in Computer Science, vol 2964, pp 351-365, San Francisco, USA, February 2005, Springer-Verlag.

[24] M. Matsui, *Linear Cryptanalysis Method for DES Cipher*, in the proceedings of Eurocrypt 1993, Lecture Notes in Computer Science, vol 765, pp 386-397, Lofthus, Norway, May 1993, Springer-Verlag.

[25] D. May, H. Muller, N. Smart, *Randomized Register Renaming to Foil DPA*, in the proceedings of CHES 2001, Lecture Notes in Computer Sciences, vol 2162, pp 28-38, Paris, France, May 2001, Springer-Verlag.

[26] T.S. Messerges, *Using Second-Order Power Analysis to Attack DPA Resistant Software*, in the proceedings of CHES 2000, Lecture Notes in Computer Sciences, vol 1965, pp 71-77, Worcester, Massachusetts, USA, August 2000, Springer-Verlag.

[27] T.S. Messerges, E.A. Dabbish, R.H. Sloan, *Examining Smart-Card Security under the Threat of Power Analysis Attacks*, IEEE Transactions on Computers, vol 51, num 5, pp 541-552, May 2002.

[28] National Bureau of Standards, *FIPS PUB 46, The Data Encryption Standard*, Federal Information Processing Standard, NIST, U.S. Dept. of Commerce, Jan 1977.

[29] National Bureau of Standards, *FIPS 197, Advanced Encryption Standard*, Federal Information Processing Standard, NIST, U.S. Dept. of Commerce, November 2001.

[30] S.B. Ors, E. Oswald, B. Preneel, *Power-Analysis Attacks on an FPGA – First Experimental Results*, in the proceedings of CHES 2003, Lecture Notes in Computer Science, vol 2279, pp 35-50, Cologn, Germany, September 2003, Springer-Verlag.

[31] S.B. Ors, F. Gurkaynak, E. Oswald, B. Preneel *Power-Analysis Attack on an ASIC AES implementation*, in the proceedings of ITCC 2004, Las Vegas, April 5-7 2004.

[32] E. Oswald, M. Aigner, *Randomized Addition-Substraction Chains as a Countermeasure against Power Attacks*, in the proceedings of CHES 2001, vol 2162, pp 39-50, Paris, France, May 2001, Springer-Verlag.

[33] E. Oswald, *Enhancing Simple Power-Analysis Attacks on Elliptic Curve Cryptosystems*, in the proceedings of CHES 2002, Lecture Notes in Computer Science, vol 2523, pp 82-97, Redwood Shores, CA, USA, August 2002, Springer-Verlag.

[34] E. Oswald, S. Mangard, N. Pramstaller, V. Rijmen, *A Side-Channel Analysis Description of the AES S-box*, in the proceedings of FSE 2005, Paris, France, February 2005.

[35] E. Peeters, F.-X. Standaert, N. Donckers, J.-J. Quisquater, *Improved Higher-Order Side-Channel Attacks With FPGA Experiments*, in the proceedings of CHES 2005, Lecture Notes in Computer Science, vol 3659, pp 309-323, Edinburgh, Scotland, August 2005.

[36] J.M. Rabaey, *Digital Integrated Circuits*, Prentice Hall International, 1996.

[37] F.-X. Standaert, S.B. Ors, B. Preneel, *Power Analysis of an FPGA Implementation of Rijndael: is Pipelining a DPA Countermeasure?*, in the proceedings of CHES 2004, Lecture Notes in Computer Science, vol 3156, pp 30-44, Boston, USA, August 2004.

[38] F.-X. Standaert, S.B. Ors, J.-J. Quisquater, B. Preneel, *Power Analysis Attacks against FPGA Implementations of the DES*, in the proceedings of FPL 2004, Lecture Notes in Computer Science, vol 3203, pp 84-94, Antwerp, Belgium, September 2004.

[39] F.-X. Standaert, E. Peeters, J.-J. Quisquater, *On the Masking Countermeasure and Higher-Order Power Analysis Attacks*, in the proceedings of ITCC 2005, Embedded Crypto Track, Las Vegas, USA, April 2005.

[40] F.-X. Standaert, F. Macé, E. Peeters, J.-J. Quisquater, *Updates on the Security of FPGAs against Power Analysis Attacks*, UCL Crypto Group Technical report, CG-2006/1.

[41] K. Tiri, M. Akmal, I. Verbauwhede, *A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards*, in the proceedings of ESSCIRC 2002.

[42] K. Tiri, I. Verbauwhede, *Synthesis of Secure FPGA Implementations* , in the proceedings of the International Workshop on Logic and Synthesis (IWLS 2004), pp 224-231, June 2004.

[43] K. Tiri, I. Verbauwhede, *Place and Route for Secure Standard Cell Design*, in the proceedings of CARDIS 2004, pp 143-158, Toulouse, France, August 2004..

[44] E. Trichina, *Combinatorial Logic Design for AES SubByte Transformation on Masked Data*, IACR e-print archive 2003/236, http://eprint.iacr.org, 2003.

[45] J. Waddle, D. Wagner, *Towards Efficient Second-Order Power Analysis*, in the proceedings of CHES 2004, Lecture Notes in Computer Science, vol 3156, pp 1-15, Boston, USA, August 2004.

[46] C.D. Walter, *MIST: An Efficient, Randomized Exponentiation Algorithm for Resisting Power Analysis*, in the proceedings of CT-RSA 2002, Lecture Notes in Computer Science, vol 2271, pp 53-66, San Jose, CA, USA, February 2002, Springer-Verlag.

[47] Xilinx: *Virtex 2.5V Field Programmable Gate Arrays Data Sheet*, http://www.xilinx.com.

[48] Xilinx: *Spartan 2.5V Field Programmable Gate Arrays Data Sheet*, http://www.xilinx.com.

**François-Xavier Standaert** was born in Brussels, Belgium in 1978. He received the Electrical Engineering degree and PhD degree from the Université catholique de Louvain, respectively in June 2001 and June 2004. In 2005, he was a Fulbright visiting researcher at Columbia University (Networks Security Laboratory) and MIT Medialab. He is now a post doctoral researcher funded by the FNRS (Funds for National Scientific Research, Belgium), at the UCL Crypto Group. His research interests include digital design and FPGA's, cryptographic hardware, design of cryptographic primitives and side-channel analysis.

**Eric Peeters** was born in Brussels, Belgium, in 1979. He received the Electromechanical Engineering degree from the Université catholique de Louvain in June 2002. He is currently a PhD student of the UCL Crypto Group, under supervision of Pr. Jean-Jacques Quisquater. His research interests include digital design and FPGAs, design and hardware implementation of asymmetric ciphers and side-channel analysis.

**Gael Rouvroy** was born in Brussels, Belgium, in 1978. He obtained the Electromechanical Engineering degree and the PhD degree from the Université catholique de Louvain in 2001 and 2004. He is currently a post-doctoral researcher at the electrical engineering department of the same university. His research interests include digital design and FPGAs, cryptographic hardware, block ciphers, cryptanalysis, image processing (JPEG 2000) and watermarking.

**Jean-Jacques Quisquater** is professor of cryptography and multimedia security at the Department of Electrical Engineering, University of Louvain, Louvain-la-Neuve, Belgium. He is responsible, at least at the scientific level, of many projects related to smart cards (protocols, implementations, side-channels), secure protocols for communications, digital signatures, payTV, protection of copyrights and security tools for electronic commerce. He was the main designer of several coprocessors for powerful smart cards: CORSAIR (Philips) and FAME (Philips). He holds 17 patents in the field of smart cards. He is co-inventor of the so-called GQ cryptographic identification scheme.