# Power Analysis Attacks against FPGA Implementations of the DES

François-Xavier Standaert[1], Sıddıka Berna Örs[2⋆],
Jean-Jacques Quisquater[1], Bart Preneel[2]

[1]UCL Crypto Group
Laboratoire de Microélectronique
Université Catholique de Louvain
Place du Levant, 3, B-1348 Louvain-La-Neuve, Belgium
standaert,quisquater@dice.ucl.ac.be

[2]Katholieke Universiteit Leuven, Dept.ESAT/SCD-COSIC,
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium.
siddika.bernaors,bart.preneel@esat.kuleuven.ac.be

**Abstract.** Cryptosystem designers frequently assume that secret parameters will be manipulated in tamper resistant environments. However, physical implementations can be extremely difficult to control and may result in the unintended leakage of side-channel information. In power analysis attacks, it is assumed that the power consumption is correlated to the data that is being processed. An attacker may therefore recover secret information by simply monitoring the power consumption of a device. Several articles have investigated power attacks in the context of smart card implementations. While FPGAs are becoming increasingly popular for cryptographic applications, there are only a few articles that assess their vulnerability to physical attacks. In this article, we demonstrate the specific properties of FPGAs *w.r.t.* Differential Power Analysis (DPA). First we emphasize that the original attack by Kocher *et al.* and the improvements by Brier *et al.* do not apply directly to FPGAs because their physical behavior differs substantially from that of smart cards. Then we generalize the DPA attack to FPGAs and provide strong evidence that FPGA implementations of the Data Encryption Standard (DES) are vulnerable to such attacks.

## 1 Introduction

Since their publication in 1998 [1], power analysis attacks have attracted significant attention within the cryptographic community. So far, they have been successfully applied to different kinds of (unprotected) implementations of symmetric and public-key encryption schemes. Most published attacks apply to smart cards and only a few articles assess the vulnerability of FPGA implementations to power analysis attacks [2, 3]. In this paper, we demonstrate the specificity of this kind of platform in the context of Differential Power Analysis. First, we

⋆ Sıddıka Berna Örs is funded by research grants of the Katholieke Universiteit Leuven, Belgium. This work was supported by Concerted Research Action GOA-MEFISTO-666 of the Flemish Government and by the FWO "Identification and Cryptography" project (G.0141.03).

show that the original attack described in [1] and its most recent improvements [4] do not work properly for FPGAs because their physical behavior is different than smart cards. Then we generalize the power consumption model and apply it to FPGAs. Finally, we describe a correlation attack [4, 5] in which we correlate theoretical predictions of the power consumption with real measurements in order to make an efficient use of all the collected data. The resulting attack is more efficient than the popular "multiple-bit" DPA and allows interesting theoretical predictions of the attacks with simulated data. All these techniques are successfully applied to an FPGA implementation of the DES. This is the first result on the vulnerability of an FPGA implementation of a block cipher to power attacks.

This paper is organized as follows. Section 2 presents the hypothesis used to carry out the DPA and Sect. 3 gives a short description of the DES algorithm. Section 4 describes the original DPA attack and underlines why it is not applicable to FPGAs. Sections 5 and 6 investigate a generalized power attack. Section 7 presents some theoretical predictions of the generalized attack and Sect. 8 applies it to real measurements. Finally, conclusions are presented in Sect. 9.

## 2 Hypothesis

In Differential Power Analysis, 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 (*e.g.* 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.

While little information is available on the design and implementation of FPGAs (much of the information is proprietary), we can make assumptions about how commercial FPGAs behave at the transistor level. The most popular technology used to build programmable logic is static RAM[1], where the storage cells, the logic blocks and the connection blocks are made of CMOS gates. For these circuits, it is reasonable to assume that the main component of the power consumption is the dynamic power consumption. For a single CMOS gate, we can express it as follows [7]:

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

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. However, for the attacker, the relevant question is to know if this data-dependent behavior is observable. This was confirmed by the following test.

Let three 4096-bit vectors be defined as follows. Initially, $a_0 = 00000...001$ and $b_0, c_0 = 00000...000$. Then:

$$a_{i+1} = SL(a_i)$$

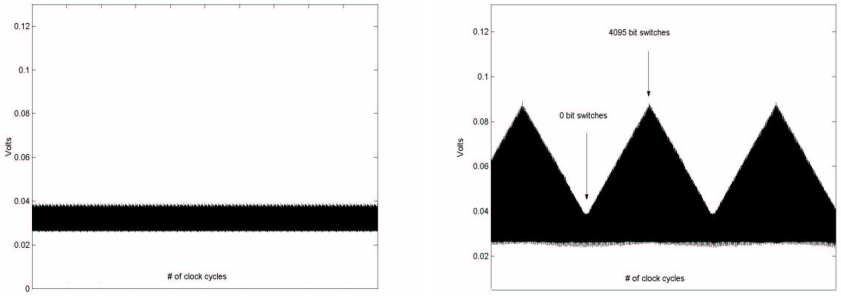
[1] For all the experiments, we used a Xilinx Virtex XCV800 FPGA [6].

$$b_{i+1} = b_i \oplus a_i$$
$$c_{i+1} = c_i \oplus b_i \ ,$$

where $SL$ is the shift left operator and consecutive values $(x_i, x_{i+1})$ are separated by a register. It is easy to see that:

- $a$ is a bit-vector with a constant Hamming weight ($H(a) = 1$). The position of the 1-bit inside the vector is periodically incremented from 0 to 4095.
- $b$ is a bit-vector for which the Hamming weight is incremented/decremented from 0 to 4095.
- $c$ is a bit-vector for which the number of bit switches between two consecutive states is incremented/decremented from 0 to 4095.

A design that generates these three vectors was implemented in the FPGA. Figure 1(a) illustrates[2] the power consumption of the vectors $a$ and $b$. Figure 1(b)




(a) Hamming weight (b) Transitions

**Fig. 1.** Preliminary test.

illustrates the power consumption of vectors $a$, $b$ and $c$. From this experiment, we conclude that the power consumption clearly depends on the number of transitions in registers but not on the Hamming weight of the data in the registers.

Based on these considerations, we used the following **hypothesis** to mount power attacks against FPGAs: "an estimation of a device power consumption at time $t$ is given by the number of bit transitions inside the device registers at this time." Predicting the transitions in registers is reasonable since registers usually consume a large part of the power in a design.

## 3 The Data Encryption Standard

In 1977, the Data Encryption Standard (DES) algorithm [8] was adopted as a Federal Information Processing Standard for unclassified government communication. Although a new Advanced Encryption Standard (AES, [9]) was selected

[2] Measurement setups for DPA have already been intensively described in the literature. In Fig. 1, we observe the voltage variations over a small resistor inserted in the supply circuit of the FPGA. Every trace was averaged 10 times in order to remove the noise from our measurements.

in October 2000, DES is still widely used, particularly in the financial sector. DES encrypts 64-bit blocks with a 56-bit key; its main operations are permutations, substitutions and XOR operations. DES is an iterated block cipher that applies 16 key-dependent transformations called rounds to the plaintext. This structure allows for very efficient hardware implementations.

The plaintext is first permuted by a fixed permutation *IP*. Next the result is split into two 32-bit halves, denoted with $L$ (left) and $R$ (right) to which a round function is applied 16 times. The ciphertext is calculated by applying the inverse of the initial permutation *IP* to the result of the 16th round.

The secret key is expanded by the key schedule algorithm from 56 bits to sixteen 48-bit subkeys $K_i$; each round uses a different subkey $K_i$. The key schedule consists of bit permutations and rotations. As a consequence, if one can find any subkey, one can derive the complete key immediately (the missing 8 bits can be found by exhaustive search over 256 values).

Finally, the round function is represented in the grey part of Fig. 2(a); it can be described as follows:

$$\begin{aligned} L_{i+1} &= R_i \\ R_{i+1} &= L_i \oplus f(R_i, K_i), \quad i = 0, \ldots 15 \ . \end{aligned}$$

Here $L_{16} \| R_{16}$ is the ciphertext. The details of the nonlinear function $f$ are provided in Fig. 2(b): the right part $R_i$ is first expanded to 48 bits with the $E$ box, which duplicates some bits. Next, the 48-bit subkey $K_i$ is added bitwise modulo 2 (XORed) to $E(R_i)$ and the result of the *XOR* function is sent to eight non-linear S-boxes ($S$). Each of them has six input bits and four output bits. The resulting 32 bits are permuted by the bit permutation $P$.

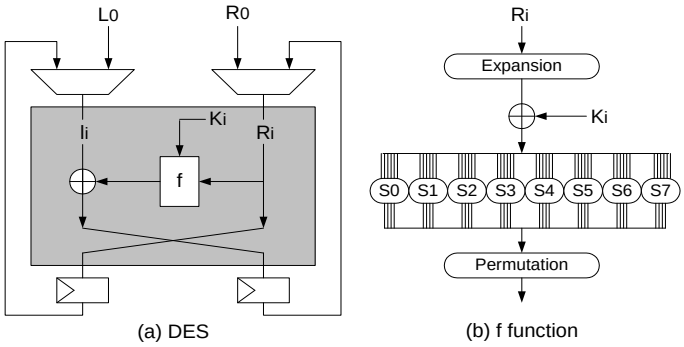



**Fig. 2.** Data Encryption Standard.

We have performed our experiments on the sequential DES implementation of [10] that takes one clock cycle to perform one round. It is represented in Fig. 2(a).

## 4 Original attack

In its original form [1], Differential Power Analysis of DES requires a selection function $D(C, b, K_{Sb,15})$ that we define as computing the value of a bit $b$ which is part of the intermediate vector $L_{15}$ (Fig. 3(a)). One can write $b$ as follows:

$$b = \text{one output bit of } Sb\,((\text{six bits of } L_{16}) \oplus K_{Sb,15}) \oplus \text{one bit of } R_{16}\ .$$

To implement the DPA attack, an attacker first observes $m$ encryptions and captures the power traces $T_i$ $(1 \leq i \leq m)$ and their associated ciphertexts $C_i$. No knowledge of the plaintext is required. By guessing six key bits $K_{Sb,15}$, the function $D$ can be computed for each value of $i$ and we can divide the traces into two sets: one set corresponding to $D_i = 0$ and the other one with $D_i = 1$. The traces in each set are then averaged to obtain two average traces $A_0$ and $A_1$ and we can compute the difference $\Delta = A_0 - A_1$.

If $K_{Sb,15}$ is correct, the computed value for $D$ will be equal to the actual value of target bit $b$ with probability 1. As the power consumption is correlated to the data, the plot of $\Delta$ will be flat, with spikes in regions where $D$ is correlated to the values being processed. If $K_{Sb,15}$ is incorrect, $\Delta$ will be flat everywhere. Finally, in a multiple bit attack, the selection function outputs $d$ bits with $d > 1$. It allows to improve the SNR of the attack: if a single-bit DPA attack using $N$ traces has a signal to noise ratio $SNR_1$, then an *all-zeros-or-all-ones* $d$-bit DPA attack using $N$ traces will have a ratio $SNR_d = d \cdot SNR_1$.

According to [1], the selection function was chosen because, at some point during a software DES implementation, the software needs to compute its value. When this occurs or any time data containing the selection bits is manipulated, there will be a slight difference in the amount of power dissipated, depending

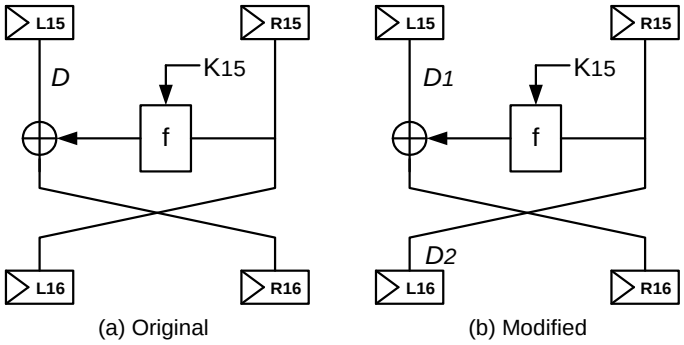



**Fig. 3.** Selection functions $D$,$D'$.

on the values of these bits. However, in the case of RAM-based FPGA implementations, this function does not correctly match the physical behavior of the devices. In a multiple-bit attack, one tries to distinguish bit vectors of different Hamming weights, although it is clear from Fig. 1 that the most significant

power differences are related to the switching activity between two states. In the next section, we propose to modify the selection function in order to take into account the physical behavior of the FPGAs.

## 5 Modified selection function

In its original form, the selection function is defined as computing the value of a bit $b$ which is part of the intermediate vector $L_{15}$. For multiple bit attacks, $d$ bits are computed and we denote them by $D = L_{15}[p_0, p_1, \ldots, p_{d-1}]$, where $p_i$ is the position of the $ith$ bit guessed. Distinguishing $D =$"$00\ldots0$" from $D =$"$11\ldots1$" therefore implies distinguishing vectors of different Hamming weights. A modified selection function can be defined as follows. Let $D_1$ be the original selection function that involves bits $L_{15}[p_0, p_1, \ldots, p_{d-1}]$. As $L_{16}$ is part of the ciphertext, we can access it. With the notation $D_2 = L_{16}[p_0, p_1, \ldots, p_{d-1}]$, we define a new selection function correlated with the switching activity of the device: $D' = H(D_1 \oplus D_2)$. Based on this selection function, we have mounted successful 4-bit attacks against FPGA implementations of the DES. However, as a multiple bit attack only considers the texts that give rise to 0 or $d$ switches, it is far from optimal and a lot of texts are actually not used. Next, we propose an improved attack based on the correlation between the theoretical power consumption files and the practical measurements.

Note that the same model is used implicitly for software implementations in smart cards: Brier *et al.* clearly state in [4] that the DPA model is based on the Hamming distance between the data handled and an unknown but constant reference state. This constant reference state simply corresponds to the address of an instruction. As a software implementation will load the instruction before loading the data, a DPA attack actually models the switching activity between two states, but one of these states (*i.e.* the instruction address) is constant. Our selection function (with two variable states) is therefore a generalization of the original DPA model.

## 6 Improved attack

A correlation attack [4, 5] against an FPGA implementation of the DES is divided into three steps. Let $N$ be the number of plaintext/ciphertext pairs for which the power consumption measurements are available. Let $K$ be the secret key used to encrypt. When simulating the attacks, we assume that $K$ is known to the attacker (when the attack is implemented, $K$ is of course unknown).

**Prediction phase:** For each of the $N$ encrypted plaintexts, the attacker first selects a target S-box for the selection function $D'$ (cf. supra). Then, he predicts the value of $D'$ (*i.e.* the number of bit flips inside a target register between rounds 15 and 16) for the $2^6$ key guesses. The result of the prediction phase is an $N \times 2^6$ **selected prediction matrix** containing integers between 0 and 4. For simulation purposes, it is also interesting to produce the **global prediction**

**matrix** that contains the number of bit flips inside all the registers[3] of the design, for all the cycles. That is, if the encryption is performed in 16 clock cycles, we obtain an $N \times 16$ matrix, containing integers between 0 and 64. This is only feasible if the key is known. According to the hypothesis of Sect. 2, these matrices give estimations for the power consumption of the device.

**Measurement phase:** During the measurement phase, we let the FPGA encrypt the same $N$ plaintexts with the same key, as we did in the prediction phase. While the chip is operating, we measure the power consumption for the 16 consecutive clock cycles. Then, the power consumption trace of each encryption is averaged 10 times in order to remove the noise from our measurements and we store the maximum value of each encryption cycle so that we produce a $N \times 16$ matrix with the power consumption values for all the texts, cycles. We denote it as the **global consumption matrix**.

**Correlation phase:** In the correlation phase, we compute the correlation coefficient between the 16th column of the global consumption matrix (corresponding to 16th round targeted by the prediction phase) and all the columns of the selected prediction matrix (corresponding to all the $2^6$ key guesses). If the attack is successful, we expect that only one value, corresponding to the correct key guess, leads to a high correlation coefficient.

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

$$C(T, P) = \frac{E(T.P) - E(T).E(P)}{\sqrt{Var(T).Var(P)}} \ . \tag{2}$$

Here $E(T)$ denotes the mean of the set of traces $T$ and $Var(T)$ its variance. If this correlation is high, it is usually assumed that the prediction of the model, and thus the key hypothesis, is correct.

Finally, theoretical predictions of the attack can be performed by using the global prediction matrix in place 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, *i.e.* an attack where the correct key guess leads to the highest correlation coefficient. This is investigated in the next section.

## 7 An attack using simulated data

Let our target for the selection function $D'$ be the 4 bits of the register $L$ that are affected by the 6 Most Significant Bits (MSBs) of the round key 16. It cor-

[3] Note that since the same key is used for all the measurements, the power consumption of the key schedule is fixed and may be considered as a DC component that we can neglect as a first approximation.

responds to the output bits of S-box $S0$. Let the number of measurements be $N = 4096$. A theoretical prediction of our attack can be performed by running it with simulated data.
In the first step of the simulated attack, we produce the **selected prediction matrix** and **global prediction matrix** as defined in the previous section. Thereafter, we perform the correlation phase between these two matrices. If the attack is successful, we expect that only one value, corresponding to the correct key guess, leads to a high correlation coefficient. Figure 4 shows that this expectation is fulfilled and the correct 6 MSBs of the last round key guess are $1\mathrm{E}_{hex} = 30_{dec}$.

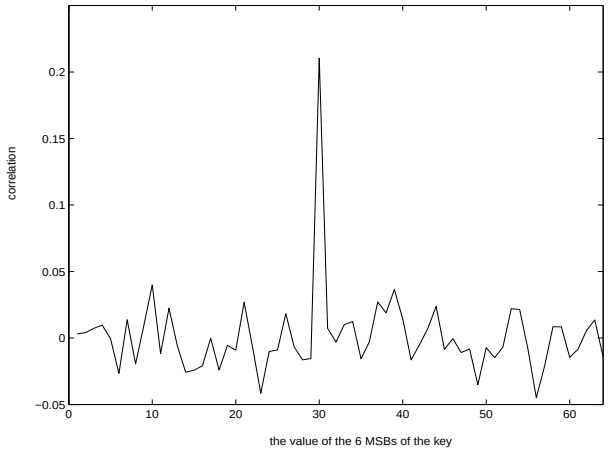



**Fig. 4.** A correlation attack using simulated data

As an attacker would like to learn the minimum number of plaintexts that are necessary to find the key, we have also calculated this correlation coefficient for different values of $N : 0 \leq N \leq 2\,000$. As shown in Fig. 5, after approximately 400 plaintexts the right 6 key-bits can be distinguished from a wrong guess. We may therefore say that the attack is **theoretically successful** after about 400 texts.

## 8 An attack using practical measurements

When attacking a device in practice, the selected prediction matrix stays unchanged while we replace the global prediction matrix by the measured **global consumption matrix**. Therefore, we let the FPGA encrypt the same $N = 4096$ plaintexts with the same key as we did in the previous section and produced the matrix as described in Sect. 6.

In order to identify the correct 6 MSBs of the final round key, we used the correlation coefficient again. As it is shown in Fig. 6, the highest correlation occurs when the key guess is $1\mathrm{E}_{hex} = 30_{dec}$. This value corresponds to the correct 6 MSBs of the round key 16. As a consequence, the attack is **practically**

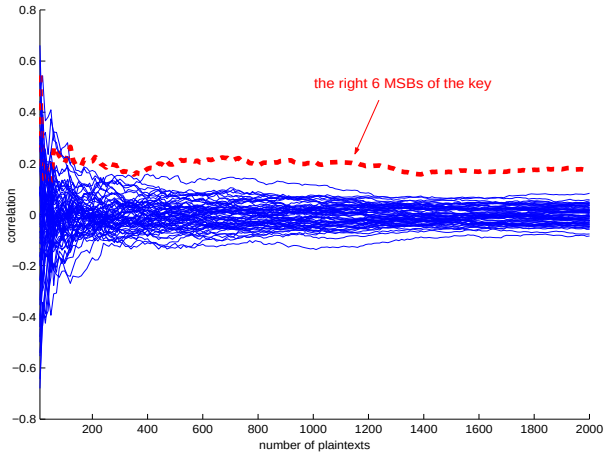



**Fig. 5.** A correlation attack using simulated data for different $N$ values.

**successful**, *i.e.* the selected prediction matrix is sufficiently correlated with the real measurements and we can extract the key information. Remark that comparing Fig. 4 and Fig. 6 clearly allows to evaluate the effect of the noise in our measurements.

It is important to note that more bits of the final subkey may be found using exactly the same set of measurements. The attacker only has to modify the selected prediction matrix in order to target different key bits. As every subkey consists of 48 bits and the master key of 56 bits, we can easily find the last 8 key bits by exhaustive search.

Finally, a more accurate prediction of the FPGA power consumption could allow to improve the efficiency of the attack. A notable feature of FPGAs is that they contain different components (*e.g.* logic blocks, connections) with a different power consumption because of a different effective load capacitance. As a consequence, the power consumption of FPGA designs does not only depend on their switching activity but also on the internal components used. Recent works [11] tried to identify these important resources in the FPGA architecture and to characterize their power consumption. This could be used to improve the power consumption predictions.

In practice, more accurate estimations about the most power hungry components of an FPGA design can be derived from the delay information that is generated by most implementation tools [12]. As an input delay represents the delay seen by a signal driving that input due to the capacitance along the wire, large (*resp.* small) delay values indicate that the wire has a large (*resp.* small) capacitance. Based on the reports automatically generated by implementation tools, one may expect to recover a very accurate information about the signals that are driving high capacitances. The knowledge of the implementation netlists with delay information is therefore relevant. It will allow an attacker to improve the attack.

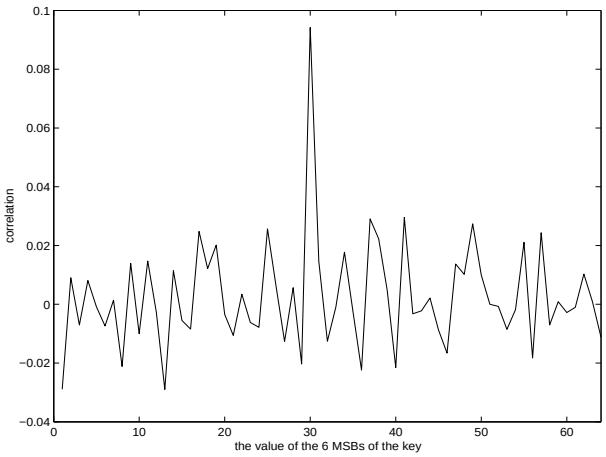



**Fig. 6.** A correlation attack with real measurements

## 9 Conclusion

This paper demonstrated the specific properties of SRAM-based FPGAs in the context of Differential Power Analysis. As the original attack of [1] does not apply 'as it is' to these reconfigurable devices, we generalized the model of power attacks in order to take into account the physical behavior of FPGAs. The resulting attack is effective with a reasonable number of measurements. It is more efficient than the popular "multiple-bit" DPA and allows interesting theoretical predictions of the attacks with simulated data. Moreover, the power consumption model and therefore the efficiency of the attack could be improved in different ways, for example by taking advantage of implementation netlists and delay information as we suggest in Sect. 8. Other block ciphers (*e.g.* AES Rijndael) are also vulnerable to our methods. These results confirm that power analysis presents a realistic threat for FPGA implementations of block ciphers.

## References

1. P.Kocher, J.Jaffe, B.Jun, *Differential Power Analysis*, in the proceedings of CRYPTO 99, Lecture Notes in Computer Science 1666, pp 398-412, Springer-Verlag.
2. 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, Springer-Verlag.
3. F.X. Standaert, L.van Oldeneel, D.Samyde, J.J. Quisquater, *Power Analysis of FPGAs, How Practical is the Attack?*, in the proceedings of FPL 2003, Lecture Notes in Computer Science, vol 2278, pp 701-711, Springer-Verlag.
4. E.Brier, C.Clavier, F.Olivier, *Optimal Statistical Power Analysis*, IACR e-print archive 2003/152.
5. 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.
6. Xilinx: *Virtex 2.5V Field Programmable Gate Arrays Data Sheet*, http://www.xilinx.com.
7. J.M.Rabaey, *Digital Integrated Circuits*, Prentice Hall International, 1996.
8. National Bureau of Standards. *FIPS PUB 46*, The Data Encryption Standard, Jan 1977.
9. NIST Home page, http://csrc.nist.gov/CryptoToolkit/aes/.
10. G.Rouvroy, F.X.Standaert, J.J.Quisquater, J.D.Legat, *Design Strategies and Modified Descriptions to Optimize Cipher FPGA Implementations: Fast and Compact Results for DES and Triple-DES*, in the proceedings of FPL 2003, LNCS 2778, pp 181-193, Springer-Verlag, 2003.
11. L.Shang, A.Kaviani, K.Bathala, *Dynamic Power Consumption in Virtex2 FPGA Family*, FPGA 2002, Monterey, California, 2002.
12. L.T. Mc Daniel, *An Investigation of Differential Power Analysis Attacks on FPGA-based Encryption Systems*, Master Thesis, Virginia Polytechnic Insitute, May 29, 2003.