# Power Attacks Resistance of Cryptographic S-boxes with added Error Detection Circuits

Francesco Regazzoni [1], Thomas Eisenbarth [2], Johann Großschädl [3], Luca Breveglieri [4], Paolo Ienne [5], Israel Koren [6], and Christof Paar [2]

[1] ALaRI - University of Lugano, Lugano, Switzerland. Email: regazzoni@alari.ch

[2] Horst Görtz Institute for IT Security, RUB, Bochum, Germany. Email: {eisenbarth,cpaar}@crypto.rub.de

[3] Department of Computer Science, University of Bristol, Bristol, UK Email: johann.groszschaedl@cs.bris.ac.uk

[4] DEI - Politecnico di Milano, Milano, Italy. Email: Luca.Breveglieri@elet.polimi.it

[5] School of Computer and Communication Sciences - EPFL, Lausanne, Switzerland. Email: Paolo.Ienne@epfl.ch

[6] University of Massachusetts, Amherst, MA, USA. Email: koren@ecs.umass.edu

**Abstract**

*Many side-channel attacks on implementations of cryptographic algorithms have been developed in recent years demonstrating the ease of extracting the secret key. In response, various schemes to protect cryptographic devices against such attacks have been devised and some implemented in practice. Almost all of these protection schemes target an individual side-channel attack and consequently, it is not obvious whether a scheme for protecting the device against one type of side-channel attacks may make the device more vulnerable to another type of side-channel attacks. We examine in this paper the possibility of such a negative impact for the case where fault detection circuitry is added to a device (to protect it against fault injection attacks) and analyze the resistance of the modified device to power attacks. To simplify the analysis we focus on only one component in the cryptographic device (namely, the S-box in the AES and Kasumi ciphers), and perform power attacks on the original implementation and on a modified implementation with an added parity check circuit. Our results show that the presence of the parity check circuitry has a negative impact on the resistance of the device to power analysis attacks.*

## 1. Introduction

The increasing ubiquity of information technologies in all aspects of human life makes security issues one of the most critical aspects of system design. Due to the rapid growth of the internet and the parallel diffusion of embedded applications, such as mobile phones and personal devices, the need for securing communication is nowadays of utmost importance. In this scenario, cryptography plays a fundamental role, but the use of secure algorithms is not sufficient to guarantee reliable communications. Due to the nature of the algorithm, even a single error occurring while performing an encryption using the *Advanced Encryption Standard* (*AES*)[16] will usually result in a completely corrupted block in the encrypted data. This property, that is desirable from a cryptographic point of view, represents a problem when faults occur in a cryptographic device. Because of this, fault detection and possibly fault tolerance are undoubtedly desirable features in cryptographic systems. Furthermore, a robust scheme for providing fault detection is also a good countermeasure against

an attacker who may maliciously inject faults in order to extract sensitive information such as the secret key.

Attacks based on fault injection are not the only way to gain access to sensitive information. In the last few years, new techniques for attacking implementations of cryptographic algorithms have been studied and developed. Unlike direct attacks on the algorithm, these techniques, called *Side Channel Attacks*, attempt to gather knowledge of sensitive data that may leak from a particular implementation of the cryptographic algorithm. One of the most successful side-channel attacks exploits the correlation between the power consumption of a given device and the data being processed. These *Power Analysis Attacks* have particular relevance since for some of them, no knowledge regarding the implementation of the target device is needed in order to be effective.

Several solutions have already been proposed and implemented, at different levels, to counteract these unconventional forms of attacks [9, 13, 14, 15, 17, 18]. However, almost all the proposed countermeasures so far have targeted a single type of side-channel attack and only very few studies have addressed the effects that one specific countermeasure can have on the resistance to a different type of attack.

In this paper, we focus on the effect that a countermeasure against fault injection attacks may have on the resistance of a cryptographic device to power analysis attacks. To study such an effect we have experimented with hardware implementations of non-linear transformations that are present in the AES and the Kasumi algorithms. We have added parity bit circuitry to these implementations for the purpose of detecting injected faults and analyzed the effect that the additional parity bit has on the resistance against the most powerful variant of *Differential Power Attacks* [12] - *Correlation Power Attacks* [6]. To determine the power consumption of all logic gates we ran SPICE level simulations, measuring the current consumption of the circuit. The obtained traces of the two S-boxes and the corresponding ones with parity-based error detection were then attacked in order to compare the implementations.

The remainder of this paper is organized as follows. Section 2 presents a summary of previous research efforts involving fault injection and power analysis attacks. Section 3 briefly describes the Rijndael (AES) and Kasumi algorithms. Our proposed implementation of the two S-boxes and the used parity check scheme are introduced in Section 4. The simulation environment and the experimental results of the power attacks are described in Section 5. Section 6 concludes the paper.

## 2. Related Work

The evolution of side-channel cryptanalysis attacks introduced new problems for designers of embedded systems performing cryptographic algorithms, in particular, smart cards. Attacks that target the device where the algorithm is performed rather than the mathematical structure of the algorithm are very powerful and often reasonably cheap, thus a large number of previous works addressed this problem. Among the so called *side-channels* attacks, time, power, electromagnetic emanation and the deliberate injection of faults are of particular relevance [2, 11, 12].

Despite the large amount of previous research, a perfect protection against such attacks is still not available. A common approach for defeating power analysis attacks is to remove as much as possible the correlation between the power consumption and the data being processed, by using a combination of countermeasures. The proposed countermeasures act at different levels of the design process, ranging from algorithmic techniques [9, 18], through architectural approaches [13, 14] down to hardware-related methods [15, 17].

Countermeasures against fault injection attacks [4, 5] have also been proposed. One approach is

to rely on error detection codes which have been traditionally used in data transmission, especially for dealing with noisy channels.

Several classical codes were adapted to the needs of cryptographic applications, e.g., parity checking. Additionally some new solutions tailored to the specific needs have been proposed, mainly based on *Concurrent Error Detection (CED)* techniques. Usually CED suppresses the normal execution of the algorithm every time an error is detected, thus an attacker is not able to view and analyze the faulty output.

One possible way to check the correctness of the output involves the duplication of the hardware. The results produced by the two identical circuits are compared, and if not equal, no output is produced. The duplication roughly doubles the area needed by the circuit, hence is rather expensive. A different approach is to use the same circuit and recompute the result a second time before comparing. In this case, the area requirements are kept low, but the execution time is doubled.

In addition to these general approaches, some works focus on a particular cryptographic algorithm or on a particular class of algorithms. Wolter et. al. [19] presented an implementation of the IDEA algorithm in which the data are first encrypted and then, as a check, decrypted with the result compared to the original plaintext. Public key algorithms are analyzed in [8] where the authors propose an approach for providing error detection and correction by means of redundant arithmetic based on finite rings. Although comprehensive, the proposed implementation is complex and results in a higher area overhead compared to other methods.

In [10], Karri et al. propose a CED that is tailored to substitution-permutation network ciphers and compares the modified parity of the input with the parity of the output.

The CED scheme proposed for AES in [3] uses one parity bit for every internal state byte of the AES. This scheme, which requires a limited amount of area for its implementation, detects all odd errors and in many cases even errors as well. Due to its simplicity and low overhead, we selected this approach for implementing the error detection in the S-boxes analyzed in our paper.

## 3. Overview of the algorithms

In this section we provide an overview of the two algorithms we use: Rijndael (AES) [7] and Kasumi[1]. We focus in this paper on the S-boxes of the ciphers, which for AES is the only non-linear transformation.

### 3.1. Advanced Encryption Standard

Rijndael algorithm, that became officially AES [16] in 2001, implements a block cipher for symmetric key cryptography and supports key sizes of 128, 192 and 256 bits, while the block size is restricted to 128 bits. The algorithm works on a two dimensional representation of the input block called state, that is initialized to the input data block and holds the intermediate result during the encryption and decryption processes, and ultimately holds the final result when the process is completed. All the transformations within the algorithm are grouped in a set called round, that is iterated a specific number of times depending on the key size.

In the encryption process, the round function is composed of the following four transformations: *ShiftRows* cyclically shifts to left the bytes in the last three rows of the state with different offsets, *SubBytes (or S-box)* is a non-linear byte substitution and operates independently on each byte of the state, *MixColumns* multiplies modulo $x^4+1$ the columns of the state by the polynomial $\{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$, and finally *AddRoundKey* adds a round key to the state. Each round key

consists of four words produced by a key schedule process, that generates all of them by expanding the given secret key.

Decryption is similar to the encryption process and uses the same basic transformations, but inverted. The key schedule is identical to the one described for the encryption process but starts with the last round key.

### 3.2. Kasumi

Kasumi[1] is the base for the standardized algorithm of the *3GPP* (3rd Generation Partnership Project). It is a Feistel block cipher with eight rounds that produce a 64-bit output from a 64-bit input using a secret key of length 128 bits. During the encryption, the input $I$ is divided into two 32-bit strings, called $L_0$ and $R_0$. Then, for each round $i$, $L_i$ and $R_i$ are computed as $R_i = L_{i-1}$ and $L_i = R_{i-1} \oplus f_i(L_{i-1}, RK_i)$, where $f_i$ denotes the round function within $L_i$ and $RK_i$ is the round key. The round function $f_i$ is constructed from sub-functions and has two different forms depending on whether it is an even or an odd round. It uses two *S-boxes*: *S7* which maps a 7-bit input to a 7-bit output, and *S9*, which maps a 9-bit input to a 9-bit output. Both of the S-boxes have been designed in such a way that they can be easily implemented using either a look-up table or combinatorial logic.

## 4. Implementation of the Substitution Boxes

Figure 1 depicts the basic configuration for the power analysis attacks used in this work, that reflects the configuration commonly used in these scenarios. The plaintext is added to the secret key and the result of this xor operation is used as input for the S-box function. The output of the substitution function is stored into a bank of registers. In order to always have the same initial condition, a reset signal is applied to the registers at the end of each write operation. Although a real implementation of the full algorithm can be slightly different from this simplification, the purpose of this work is to estimate the impact of a parity bit concatenated to the S-box on the resistance to a power analysis attack. Thus, the considered approximation that is well known and accepted as an accurate model for attacking the part of a cryptographic algorithm that is more exposed to side channel attacks, is adequate for our needs.

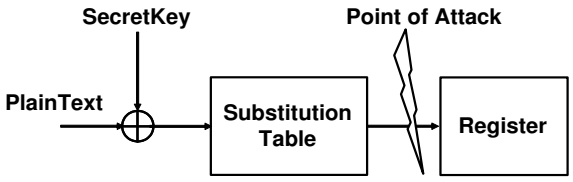


**Figure 1. Overview of the considered part of the Kasumi and AES Algorithm.**

For both of the considered S-boxes (*Kasumi S7* and *AES*), we implemented two versions of the non-linear function. In the first we used a straightforward implementation of the standard, while in the second we added a bit for parity checking, following the description in [3]. As depicted in Figure 2 that shows as an example the S-box of AES, this additional bit is used to verify the even parity twice: once at the input and then at the output of the S-box. When new data enter the S-box, the check bit is separated from the data bits and the parity is checked. If the calculated parity is

even, the 8 data bits enter the S-box circuit, that produces 9 bits: the byte with the result of the non-linear transformation plus its parity bit. At this point the second check is performed, again as described before. If the results of both verifications are correct, then the 9-bit output of the S-box is composed of the correct result of the substitution with a concatenated parity bit. If, instead, an error is detected, the S-box generates a faulty output consisting of the string *000000001*, where the only bit set to logic level one is the parity bit. The block diagram of the parity control added to the S-box is shown in Figure 2, where the number of bits for each set of wires is represented by a digit at the top of the line.

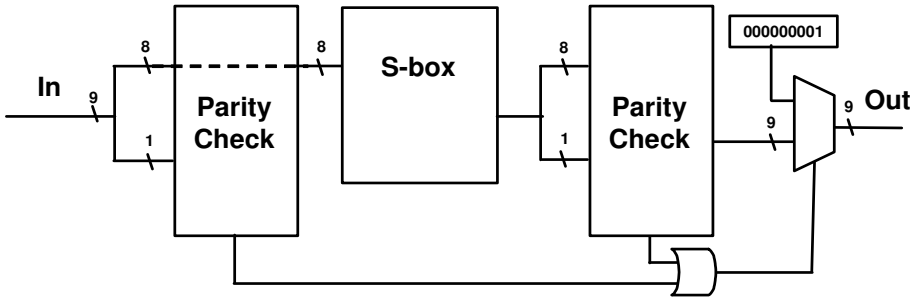


**Figure 2. Block diagram of the error detection scheme applied to the 8 bit S-box of the AES algorithm.**

For all our implementations, the S-box module has been described using VHDL at the behavioral level. Because of this, it has been synthesized by the tool as a combinatorial network rather then a memory look up table. Thus, it is not necessary to protect the address decoder against injected faults since this component is not present in the synthesized implementation of the substitution table. This approach is not a limitation since it reflects a typical situation of designing a cryptographic coprocessor, where the full coprocessor is specified using a *Hardware Description Language* and then synthesized by the tool without imposing specific implementation constraints. Thus, the S-box module is frequently realized as combinatorial logic.

## 5. Results

In this section we describe the attacks we mounted on the conventional implementations of the *Kasumi S7* and *AES* S-boxes and we compare the results with those for attacks on the implementations of the same substitution tables but with an error detection scheme as described in [3].

In order to evaluate the robustness of our hardware implementations, we attacked the current absorption traces obtained with transistor level simulation. The simulation was preformed using Synopsys Nanosim at 10ps time resolution. All the circuits have been described using VHDL, synthesized with Synopsys Design Compiler, placed and routed using Cadence Silicon Ensemble and simulated at SPICE level using the design flow for the technology library UMC 0.18$\mu$m. We first used a noise-free simulated environment: neither white (thermal) noise nor algorithmic one (produced by other components) appear in the power trace. Furthermore, the simulation was performed with a very high resolution both for current (1 $\mu$A) and time (10 ps), resulting in the best possible condition for an attacker. In this noise-free environment an attack against a CMOS technology is always successful. Thus, in order to obtain a more realistic situation (closer to the one experienced in a real attack), we added white Gaussian noise to the traces.

| S-Box | CPA without noise | CPA with noise |
|---|---|---|
| AES | 256/256 | 35/256 |
| AES w/parity | 256/256 | 51/256 |
| Kasumi | 128/128 | 63/128 |
| Kasumi w/parity | 128/128 | 87/128 |

**Table 1. Average number of key bits found when mounting CPA attacks.**

We performed a Correlation Power Analysis (*CPA*) that is a more powerful variant of the Differential Power Analysis. A CPA typically consists of the following steps: After the selection of a target point for the attack, that is usually an intermediate key dependent result, the attacker encrypts (decrypts) a certain number of known plaintexts (ciphertexts) and measures the corresponding power consumption. Based on a key guess, hypothesized intermediate values are then calculated and used to determine the Hamming weight of the targeted S-box output. All the key guesses are then evaluated using the statistical correlation between the power consumed and the hypothesized Hamming weight. Finally, the key which shows the strongest correlation is chosen.

The first series of attacks was performed on noise-free traces generated by Nanosim. Under these conditions, the number of traces needed to succeed with the attack is equal to the number of all possible plain text, i.e., 128 for Kasumi and 256 for AES. As can be seen from the second column of Table 1, this is the best condition for the attacker: all the key bits were found in both implementations of the algorithms. Furthermore, for all the correct guesses the value of the correlation was approximately 1, clearly indicating a successful attack. This result shows that in similar situations, where the attacker has a sufficient number of traces to completely filter out the noise, the attack is always successful. Thus, the presence of a parity bit does not change the situation.

The next step was to attack the traces with an addition of thermal noise modeled as white Gaussian noise. We have repeated the CPA attack several times using in this case too, all the possible keys for both of the algorithms. Each attack was performed on different sets of traces, since the value of the noise added to the traces generated at SPICE level was different every time, but the level of the noise was kept fixed. Table 1 reports the average number of secret key bits found while attacking the two different implementations and shows that the presence of a parity bit has a negative impact on the resistance to power analysis attacks. In fact, in the implementation with parity bit, the number of key bits found increases by up to 38% for the Kasumi algorithm and by up to 45% for AES. We want to stress that the attacks were mounted in a simulation environment, thus in ideal conditions for an attacker, both in terms of a very accurate sampling rate and the absence of algorithmic noise.

We also attacked traces obtained by adding to the output of the SPICE level simulation white Gaussian noise of different amplitudes. As before, each attack was performed on a different set of traces, not only because the value of the noise was randomly generated but also because its level was changing in every set. The experiments show that the number of key bit found depends, in both cases, on the level of white noise. In fact, below a specific threshold (that depends on the measurement setup), all the keys were found in all the implementations. A dual situation happens above another threshold: none of the key bits can be found. Thus, when the noise is too high or almost completely absent, the presence of a parity bit does not have any impact on the attack. When the noise level is between these two tresholds, the presence of a parity bit helps the attacker and thus has a negative impact on the resistance against power attacks. An example of this situation can be

seen in Figure 3, that depicts, for the Kasumi algorithm, the number of key bits found as a function of the level of white noise. The dotted line that corresponds to the number of key bits found for the S-box implemented with parity check, shows the advantage that an attacker has in this situation.

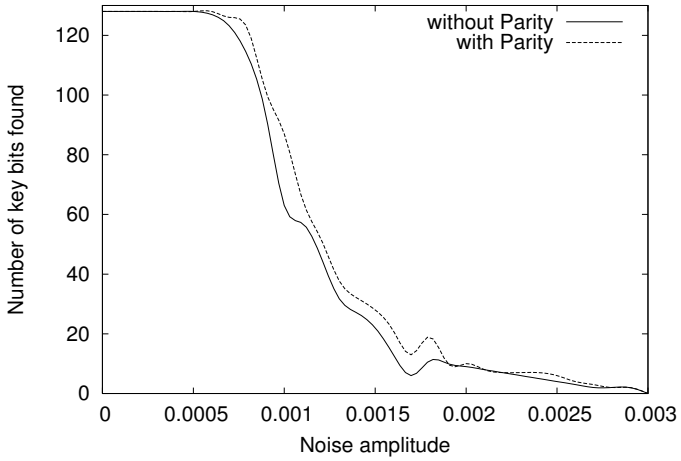


**Figure 3. The number of key bits found as a function of the level of white noise for the Kasumi's S-Box.**

Finally, we performed a classical DPA attack against the implementations of the S7 S-boxes of Kasumi and AES, with and without fault detection. In Kocher's DPA [12] only one bit of the S-box output is used in the differential function. We thus performed attacks leveraging all the output bits of each of the S-boxes. In the attacks on the fault detection implementation, we also hypothesized the parity bit. Using the noise-free traces, the value of the peak for the attacks on the parity bit, depicted in Figure 4, has approximately the same height as the ones for the attacks performed on all other single bits, showing that an attack on the parity bit is just as strong as an attack on any other bit. Also, when considering the traces with the Gaussian noise, there is no difference between an attack on the parity bit and an attack on another bit. In fact, the average number of key bits found is the same. This result is important when considering protecting circuits from side channel attacks: the parity bit must be secured as well as the other bits.

## 6. Conclusion

In this paper, we evaluated the effect that an error detection circuit may have on the resistance to power analysis attacks on hardware implementations of cryptographic substitution boxes. We discussed the role played by noise and provided a quantitative analysis of the impact for the selected implementation. Our results show that the added parity bit helps the attacker, since the number of key bits found increased by more then 35% with respect to an implementation without a fault detection circuit. Furthermore, we showed that the parity bit of a critical value carries as much information as the other bits of that value. Consequently, when adding countermeasures against Power Analysis in the presence of error detection circuitry, the parity bit needs to be protected as well.

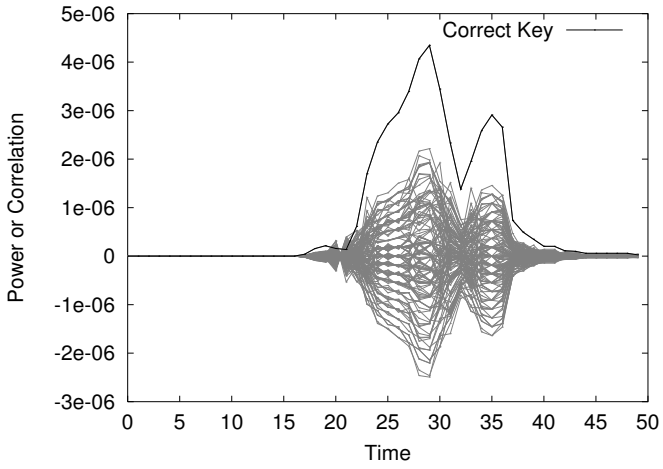


**Figure 4. DPA on the parity bit of the Kasumi Algorithm using noise-free traces.**

## References


[1] 3GPP 35.202 Technical Specification version 3.1.1. Kasumi s-box function specifications. *www.3gpp.org/ftp/Specs/archive/35_series/35.202/*, 2002.

[2] F. Bao, R. H. Deng, Y. Han, A. B. Jeng, A. D. Narasimhalu, and T.-H. Ngair. Breaking public key cryptosystems on tamper resistant devices in the presence of transient faults. In B. Christianson, B. Crispo, T. M. Lomas, and M. R. Roe, editors, *Security Protocols, 5th International Workshop*, volume 1361 of *Lecture Notes in Computer Science*, pages 115–124. Springer Verlag, 1998.

[3] G. Bertoni, L. Breveglieri, I. Koren, P. Maistri, and V. Piuri. Error analysis and detection procedures for a hardware implementation of the advanced encryption standard. In *IEEE Transactions on Computers*, volume 52, pages 492–505, 2003.

[4] E. Biham and A. Shamir. Differential Fault Analysis of Secret Key Cryptosystems. *Advances in Cryptology - CRYPTO*, page 513, 1997.

[5] D. Boneh. On the Importance of Eliminating Errors in Cryptographic Computations. In *Journal of Cryptology*, volume 14, pages 101–119. Springer, 2001.

[6] E. Brier, C. Clavier, and F. Olivier. Correlation power analysis with a leakage model. In M. Joye and J.-J. Quisquater, editors, *Cryptographic Hardware and Embedded Systems — CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 16–29. Springer, 2004.

[7] J. Daemen and V. Rijmen. AES Proposal: Rijndael. *http://csrc.nist.gov/CryptoToolkit/aes/rijndael/Rijndael.pdf*, 1999.

[8] G. Gaubatz and B. Sunar. Robust Finite Field Arithmetic for Fault-Tolerant Public-Key Cryptography. In *2nd Workshop on Fault Diagnosis and Tolerance in Cryptography - FDTC'05*, 2005.

[9] T. Izu, B. Möller, and T. Takagi. Improved elliptic curve multiplication methods resistant against side channel attacks. In A. J. Menezes and P. Sarkar, editors, *Progress in Cryptology — INDOCRYPT 2002*, volume 2551 of *Lecture Notes in Computer Science*, pages 296–313.

Springer Verlag, 2002.

[10] R. Karri, G. Kuznetsov, and M. Gössel. Parity-based concurrent error detection of substitution-permutation network block ciphers. In C. D. Walter, Çetin Kaya Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems — CHES 2003*, volume 2779 of *Lecture Notes in Computer Science*, pages 113–124. Springer, 2003.

[11] P. C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In N. I. Koblitz, editor, *Advances in Cryptology — CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer Verlag, 1996.

[12] P. C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In M. J. Wiener, editor, *Advances in Cryptology — CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer Verlag, 1999.

[13] D. May, H. L. Muller, and N. P. Smart. Non-deterministic processors. In V. Varadharajan and Y. Mu, editors, *Information Security and Privacy — ACISP 2001*, volume 2119 of *Lecture Notes in Computer Science*, pages 115–129. Springer Verlag, 2001.

[14] D. May, H. L. Muller, and N. P. Smart. Random register renaming to foil DPA. In Ç. K. Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems — CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 28–38. Springer Verlag, 2001.

[15] S. W. Moore, R. J. Anderson, P. Cunningham, R. Mullins, and G. Taylor. Improving smart card security using self-timed circuits. In *Proceedings of the 8th International Symposium on Asynchronous Circuits and Systems (ASYNC 2002)*, pages 193–200. IEEE Computer Society Press, Apr. 2002.

[16] National Institute of Standards and Technology (NIST). Announcing the Advanced Encryption Standard (AES). *http://www.nist.gov/*, November 2001.

[17] K. Tiri, M. Akmal, and I. M. Verbauwhede. A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards. In *Proceedings of the 28th European Solid-State Circuits Conference (ESSCIRC 2002)*, pages 403–406, Sept. 2002.

[18] C. D. Walter. MIST: An efficient, randomized exponentiation algorithm for resisting power analysis. In B. Preneel, editor, *Topics in Cryptology — CT-RSA 2002*, volume 2271 of *Lecture Notes in Computer Science*, pages 53–66. Springer Verlag, 2002.

[19] S. Wolter, H. Matz, A. Schubert, and R. Laur. On the VLSI implementation of the international data encryption algorithm IDEA. In *IEEE International Symposium on Circuits and Systems, ISCAS'95*, volume 1, pages 397–400, 1995.