

Side-Channel Attack Pitfalls

Kris Tiri

Platform Validation Architecture
Intel Corporation, USA

kris.tiri@intel.com

ABSTRACT

While cryptographic algorithms are usually strong against mathematical attacks, their practical implementation, both in software and in hardware, opens the door to side-channel attacks. Without expensive equipment or intrusive monitoring, these attacks bypass the mathematical complexity and find the cryptographic key by observing the power consumption or the execution time variations of the device in normal operation mode. The power traces of 8000 encryptions are for instance sufficient to extract the secret key of an unprotected ASIC AES implementation, which is orders of magnitude smaller than the 2^{128} tests required to brute force the algorithm. A careful implementation can address these vulnerabilities, yet the solutions conflict with the common design goals to optimize for area, performance and power consumption. This paper introduces the side-channel attack pitfalls, which help create or facilitate the observation of the information leakage, discusses mitigation strategies and identifies opportunities for future research.

Categories and Subject Descriptors

B.6 [Hardware]: Logic Design; B.7 [Hardware]: Integrated Circuits; E.3 [Data]: Data encryption.

General Terms

Design, Security, Verification.

Keywords

Side-Channel Attack, Differential Power Analysis, Encryption, Security IC.

1. INTRODUCTION

In traditional cryptanalysis, which views a cipher as a black box operation that transforms the plaintext into the ciphertext using a secret key, many ciphers have no practical known weaknesses, and the only way to unlock the secret key is to try all possible combinations. As a result, as long as the number of combinations is large enough such that a complete search becomes de facto

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

DAC 2007, June 4–8, 2007, San Diego, California, USA.

Copyright 2007 ACM 978-1-59593-627-1/07/0006 ...\$5.00.

impossible, the cipher is said to be secure. For instance, the RSA public key algorithm using 2048 bits can be used at least till the year 2030 before the expected computing power will be available to do the integer factorization of a 2048 bit number [1].

In reality, however, the cipher has to be implemented on a tangible device, which will leak additional information that can be used to determine the secret key. Indeed, similarly that feel or sound can help to find the combination of a padlock, the power consumption or time delay of the device can reveal the value of the secret key. Early smart card implementations, for instance, implemented the modular exponentiation of the RSA algorithm using the text book version of the square-and-multiply algorithm, in which the multiplication is only executed if the exponent bit equals 1. Since a multiplication does not have the same power signature as a squaring operation, it was possible to find out by observing the power consumption when a multiplication took place and thus to read off the private key from a single power trace.

The attacks, that use additional information leaking from the practical implementation, are also known as side-channel attacks (SCAs). First proposed in 1996 [2], they have since then been used to extract cryptographic key material of symmetric and public key encryption algorithms running on microprocessors, DSPs, FPGAs, ASICs and high performance CPUs [3],[4],[5],[6],[7] from variations in power consumption, time delay or electromagnetic radiation [3],[8],[9]. Note that SCAs are non-invasive attacks, which means that they observe the device in normal operation mode without any physical harm to the device, and making the device tamperproof does not protect against the attacks. For certain attacks, it is not even necessary to possess the device or be in close proximity as demonstrated with a remote attack that successfully found key material of an OpenSSL webserver from non-constant execution time due to conditional branches in the algorithm [10].

A careful design –we will discuss some mitigation strategies in section 3– can offer increased resistance against the vulnerabilities. As pointed out earlier [11], security adds a new dimension to a design in addition to area, performance and power consumption optimization. Side-channel attack resistance, which can be a showstopper to achieve security, is no exception and the thus far prevailing strategies rarely come cheap. It is also not very well understood how to analyze the strength of a design and the precise cost of a mitigations strategy is seldom fully and clearly communicated. This makes the resistance increase hard to quantify and the design trade-offs difficult to make.

It is interesting to notice that despite the overwhelming negative connotation of side-channel attacks, they might not be all bad. It is currently being proposed to use them to do device fingerprinting [12]. This would allow detecting cloned smart cards, which are programmed with the secret information and the logical functionality of the original, from genuine smart cards as both cards would not have the same characteristic leakage.

The remainder of this paper is organized as follows. The next section introduces the pitfalls that might open the door to side-channel attacks: resource sharing, optimization features and increased functionality. Section 3 presents some mitigation strategies and discusses the shortcomings. Finally, a conclusion will be formulated.

2. SIDE-CHANNEL PITFALLS

Typical targets of side-channel attacks are security ICs used in embedded devices and smart cards. Not only are they an alluring target as they are dedicated to performing *secure* operations, they are also rather easy to analyze. In general, it is a simple device running a single process at a low to moderate clock frequency with direct access to the side-channel of interest and with (precise) control over synchronization and measurements.

Yet the interaction between software and a micro-architecture optimized for performance leads to similar vulnerabilities and in recent times, SCAs have been demonstrated that successfully attack software implementations of symmetric and public key encryption algorithms running on 32-bit microprocessors [7],[13]. In this section, we will use the cache attacks, which exploit side-channel information that is leaked by a microprocessor's cache, to illustrate the side-channel pitfalls: resource sharing, optimization features, and increased functionality. Without proper attention, the pitfalls are prone to contribute side-channel leakage and hence they can be used to identify potential problems. Given that a side-channel is in fact a covert channel without conspiracy or consent, it should not be a surprise that both share facilitators and consequently mitigation strategies [14],[15].

A cache attack works as follows. The cache is used to store recently used data in a fast memory block close to the microprocessor. Whenever this data is used subsequently, it can be delivered quickly. On the other hand, whenever the microprocessor requires data that is not in the cache, it has to be fetched from another memory with a larger latency. The time difference between both events is measurable and provides the attacker with sufficient data on the state and the execution of the algorithm to extract some or even all secret key bits [7],[13],[16],[17]. Note that numerous mitigations have been put forward to limit the information leakage (e.g. [18]), some of which have been incorporated into cryptographic tools and libraries (e.g. OpenSSL [19]).

2.1 Resource Sharing

Resource sharing, which reduces the amount of hardware needed to implement a certain functionality, can both create side-channel information and facilitate its observation.

This can be seen with our cache attack illustration as follows. The cache is shared between the different processes running on the same CPU and since the cache has a finite size, they compete for the shared resource. This means that one process can evict another

process's data from the cache if some of their data is mapped to the same cache line, where a cache line (aka. cache entry) is the smallest unit of memory that can be transferred between the main memory and the cache. This has two consequences. Firstly, a *spy* process is able to observe which cache lines are used by the *crypto* process and thus determine its cache footprint. Secondly, the spy process is able to clean the cache. It can remove all of *crypto*'s data from the cache, which will increase the number of cache misses of the *crypto* process as the cache does not contain any of its data. This also has the advantage that the attacker can readily hypothesize a known initial state of the cache, which is required to estimate the leakage.

2.2 Optimization Features

Optimization features, which improve a system's performance or cost, can create side-channel information. In general, only the typical case is being optimized, and hence the corner cases leak information.

This can be seen with our cache attack illustration as follows. The cache is implemented to overcome the latency penalty of an access to main memory. Without a cache, the microprocessor would have to fetch all data from main memory or a higher level cache. The main memory is much slower than the microprocessor, which would be stalled waiting for the data to become available. In general, it is expected that cache hits (when the required data is in the cache) are much more frequent than cache misses (when the required data is not in the cache) and thus that the overall latency is closer to the latency of the cache than to the latency of main memory. The cache is an optimization feature leaking information through the latency difference between a typical case (cache hit) and a corner case (cache miss).

2.3 Increased visibility/functionality

Increased visibility, or for that matter increased functionality, can facilitate the observation of side-channel information. In general new functionality increases the complexity and hence introduces new interactions that might ease the difficulty of mounting the measurements.

This can be seen with our cache attack illustration as follows. Special performance counters have been added to modern microprocessors to count a wide array of events that affect a program's performance. They are able to count the number of cache accesses, as the cache behavior is an important factor in a program's performance. Compared with the time stamp counter, which is currently used in the cache attacks, the performance counters increase the visibility. The counters can be programmed to solely count the events of interest. For instance, it is possible to specify to only measure the cache read misses and not the cache write misses. Time measurements, on the other hand, measure all events that influence the time delay. The performance counters paint a more accurate picture and—if for a moment we ignore the fact that they can only be accessed using privileged ring-0 instructions—, they could enable better and faster attacks than the timestamp counter.

3. MITIGATION STRATEGIES

3.1 Side-Channel Attacks in a Nutshell

A side-channel attack works as follows (see figure 1): it compares observations of the side-channel leakage (*i.e.* measurement samples of the supply current, execution time, or electro magnetic radiation) with estimations of the side-channel leakage. The leakage estimation comes from a leakage model of the device requiring a guess on the secret key. The correct key is found by identifying the best match between the measurements and the leakage estimations of the different key guesses. Furthermore, by limiting the leakage model to only a small piece of the algorithm, only a small part of the key must be guessed and the complete key can be found using a divide-and-conquer approach. For instance, an attack on the Advanced Encryption Standard (AES) generally estimates the leakage caused by a single key byte and as a result the 128-bit key can be found with a mere $16 \cdot 2^8$ tests. Finally, as the observations might be noisy and the model might be approximate, statistical methods are often used to derive the secret from many measurements.

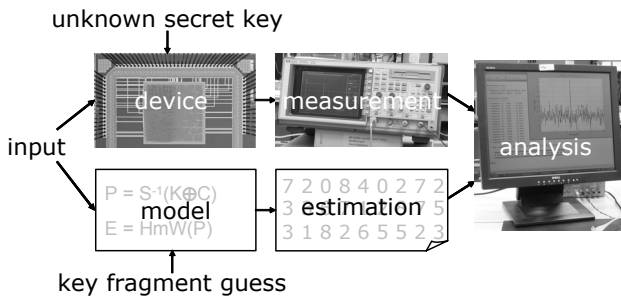


Figure 1. Side-channel attack methodology.

Attacks that use a single or only a few observations are referred to as *simple* side-channel attacks. The ‘simple’ refers to the number of measurements used and not to the simplicity of the attacks. In fact, they require a precise knowledge of the architecture and implementation of both the device and the algorithm and their effect on the observed measurement sample. As a result, they are relatively easy to protect from. For instance, the square-and-multiply algorithm can be implemented to perform the multiply independent of the exponent bit and only use the result if the exponent bit is actually one.

Attacks that use many observations are referred to as *differential* side-channel attacks. The timing attacks typically target variable instruction flow. Their focus is on public key ciphers as symmetric ciphers, which always perform the same operations, can easily –aside from the cache effects– be made constant time. The public key ciphers can be effectively protected using masking or blinding techniques that prevent collecting multiple measurements of the same operation on different data [2],[20].

The power attacks, and for that matter EMA attacks as electromagnetic fields are generated by the electric charge that flows, target data dependent supply current variations. These attacks are based on the fact that logic operations in standard static CMOS have power characteristics that depend on the input data. Power is only drawn from the power supply when a logic gate has a 0 to 1 output transition. As an example [21], figure 2

(left) shows the measurements of an unprotected ASIC AES with 128-bit encryption data path and on-the-fly key scheduling. Figure 2 (right) shows the result of a correlation attack, which correlates the power variations in the clock cycle of interest with an estimation using a guess on 8 key bits. The figure shows that the correct 8 bits are easily found despite the *algorithmic* noise (*i.e.* power consumption) of the other 120 bits in the data path and the complete key scheduling. Indeed, the differential power analysis is effective even if power variations are overshadowed with measurement errors and noise. After a sufficient number of measurements, a signal will emerge from the noise [22].

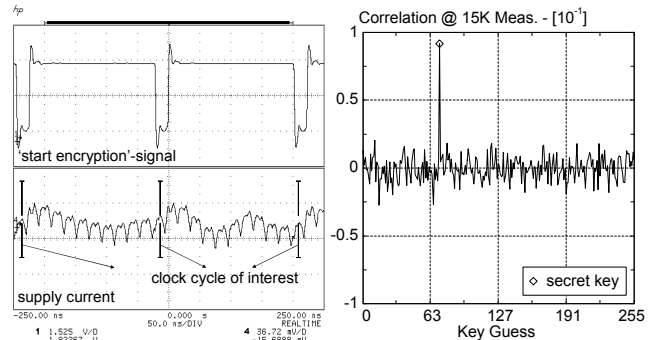


Figure 2. Cracking the secret key: supply current measurements (left); attack result after 15K measurements.

The next section discusses some selected power attack countermeasures and general challenges.

3.2 Countermeasures and Challenges

Many of the mitigations that are intuitively put forward, such as the randomization of the execution sequence or the addition of a random power consuming module or a current sink, hardly improve the resistance against the power attacks [23],[3],[24]. In the present state-of-the-art, the countermeasures try to make the power consumption of the cryptographic device independent of the signal values at the internal circuit nodes by either randomizing or flattening the power consumption. None of the techniques, however, provides perfect security instead they increase the required number of measurements.

Randomizing the power consumption is done with masking techniques that randomize the signal values at the internal circuit nodes while still producing the correct ciphertext. This can be done at the algorithmic level where a random *mask* is added to the data prior to the encryption and removed afterwards without changing the encryption result (e.g. [25]) or at the circuit level where a random mask-bit equalizes the output transition probabilities of each logic gate [26],[27]. Flattening the power consumption is done at the circuit level such that each individual gate has a quasi data-independent power dissipation. This is done with dynamic differential logic, sometimes also referred to as dual rail with precharge logic to assure that every logic gate has a single charging event per cycle [28]. In self-timed asynchronous logic [29], the terminology refers to dual rail encoded data interleaved with spacers. As an example [21], figure 3 shows the measurements and the attack result of an AES ASIC functionally identical to one of figure 2, which has been protected using the

secure digital design flow of [30] to flatten out the power consumption. 1.5M measurements are not sufficient to find the key byte under attack.

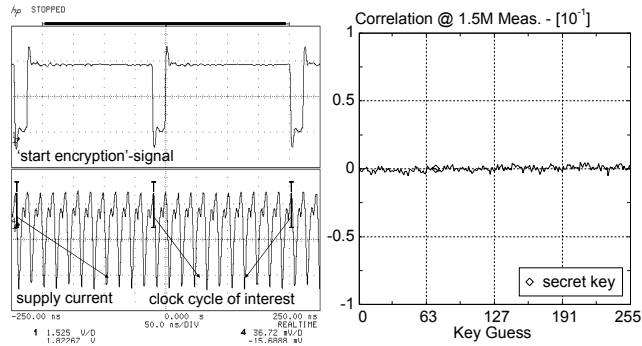


Figure 3. Secure Digital Design Flow: supply current measurements (left); attack result after 1.5M measurements.

We will now enumerate some opportunities, but also possible pitfalls, for future research.

The increased power attack resistance does not come for free. The algorithmic level masking has a factor 1.5 overhead when compared with a regular (unprotected) design [25]. The masked logic styles have a factor 2 and 5 area overhead [26],[27]. The dual rail logic styles have a factor 3 area overhead [21]. Yet, the figures for the algorithmic and logic masking do not include the random number generator. It is thus important that the full implementation cost of a countermeasure is clearly communicated and taken into account for evaluation. Several techniques have been proposed to reduce the area overhead. For instance, custom logic cells can be made more compact than compound standard logic cells and security partitioning reduces the part of the design that has to be protected [21]. *Can these techniques be optimized further or can a breakthrough mitigation technology be developed with a lower overhead?*

Yet, one has to be careful to declare a (new) mitigation as secure. A visual inspection, or even the standard deviation of the power consumption, does not provide any indication [31]. Thus far, the best figure of merit is probably the required number of measurements for a successful attack on a realistic circuit. The success of an attack, however, depends both on the information in the power consumption and on the strength of the attacker, which encompasses the measurement setup but also the leakage estimation and the statistical technique used. Indeed, if the power estimation is more accurate the attack will be more successful. The statistical analysis technique to compare the measurements with the estimations is also important: the difference of means test requires more measurements than the correlation test, which requires more measurements than the Bayesian classification. Some work has been done to distinguish the quality of an implementation from the strength of a side-channel adversary [32], but it is not clear how in a practical way a design can be evaluated without an attack and with abstraction of the statistical tool or distinguisher. *Can an expression be found that based on design parameters such as the activity factor or a power consumption profile indicates the strength of a design?*

Aside from algorithmic techniques for software running on microprocessors, resistance can not be added as an afterthought. Some design flows have been developed to automatically create more secure designs [30],[33],[34],[35]. Yet, design time security assessment remains a crucial design phase [36]. The quality of the assessment, however, is only as good as the power consumption simulation model used. It is important to note that side-channel resistance can not be isolated at one abstraction level: a technique that can be proven secure at a high abstraction level is not necessarily secure when gate delays or load capacitances are taken into account. This has been proven true both for the masking techniques and for the current flattening techniques. Glitches make attacks on the former possible [37]. Early propagation of data enables attacks on the latter [38]. These issues could be addressed with strict control over all the signal arrival times. *How can this be done in a 20K+ design? Building a correct simulation model of the side-channel leaks for use in the design time security assessment is not easy. Minor differences, even second order effects, in the power model can have a big influence in the resistance assessment [31]. What is the proper power simulation model that can be used to correctly yet quickly evaluate a design at design time? Given that minor differences have a big influence, how can process related variations of deep submicron technology be taken into account to assure a high yield despite intra- and inter-die variations?*

For the dual rail circuit styles to be effective it is crucial that the load capacitances at the differential output are matched. Figure 4 shows the load capacitance decomposition. Matching the interconnect capacitances of the signal wires is essential. This can be done with differential routing [30] or backend duplication [35]. *Yet with shrinking deep submicron technology, how can we make sure that the nets are matched sufficiently and what does sufficiently mean?*

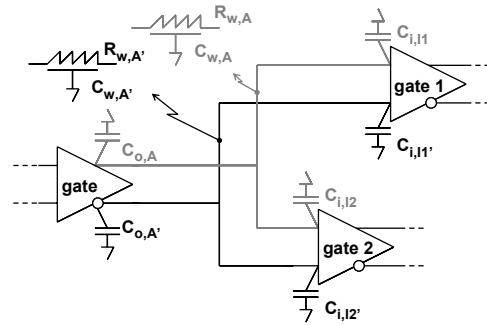


Figure 4. Interconnect capacitance decomposition.

Finally, to give an example of the complexity of designing and evaluating a countermeasure, consider recent developments regarding algorithmic masking, which was conceived as a mathematically proven countermeasure. Earlier work has already shown that it was vulnerable against higher order attacks. These attacks can combine multiple samples to remove an unknown mask because of the Hamming weight or distance leakage estimation model used [39]. But now using template attacks, in which the leakage model is built from the measurements, the authors of [40] conclude that masking has zero improvement on the security of an implementation.

4. CONCLUSIONS

Using side-channel information, it can be very easy to gain secret information from a device. Protecting against the attacks exploiting the information, however, can be a challenge, is costly and must be done with care. This paper discussed the side-channel attack pitfalls, which give rise to the information leakage, reviewed prevailing mitigation strategies, and identified opportunities but also various pitfalls that must be avoided for future research.

5. REFERENCES

- [1] Cryptographic Key Length Recommendations, <http://www.keylength.com/> <online>.
- [2] P. Kocher "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, Other Systems" CRYPTO, pp. 104-113, 1996.
- [3] P. Kocher, J. Jaffe, B. Jun "Differential Power Analysis" CRYPTO, pp. 388-397, 1999.
- [4] C. H. Gebotys, R. J. Gebotys "Secure Elliptic Curve Implementations: An Analysis of Resistance to Power-Attacks in a DSP Processor" CHES, pp. 114-128, 2002.
- [5] F.-X. Standaert, L. van Oldeneel tot Oldenzeel, D. Samyde, J.-J. Quisquater "Differential Power Analysis of FPGAs : How Practical is the Attack?" FPL, pp. 701-709, 2003.
- [6] S. B. Ors, F. Gurkaynak, E. Oswald, B. Preneel "Power-Analysis Attack on an ASIC AES implementation" ITCC, 2004.
- [7] Y. Tsunoo, T. Saito, T. Suzaki, M. Shigeri, H. Miyauchi "Cryptanalysis of DES Implemented on Computers with Cache" CHES, pp. 62-76, 2003.
- [8] J.F. Dhem, F. Koeune, P.-A. Leroux, P. Mestré, J.-J. Quisquater, J.-L. Willems "A practical implementation of the timing attack" CARDIS, pp. 167-182, 1998.
- [9] K. Gandolfi, C. Mourtel, F. Olivier "Electromagnetic Analysis: Concrete Results" CHES, pp. 251-261, 2001.
- [10] D. Brumley, D. Boneh "Remote timing attacks are practical" USENIX Security Symposium, pp. 1-14, 2003.
- [11] P. Kocher, R. Lee, G. McGraw, A. Raghunathan, S. Ravi "Security as a New Dimension in Embedded System Design" DAC, pp.735-760, 2004.
- [12] K. Mayes "Smart Card Platform Fingerprinting" IPAM Workshop IV: Special purpose hardware for cryptography: Attacks, Applications, 2006.
- [13] C. Percival "Cache missing for fun, profit" BSDCan, <http://www.daemonology.net/papers/htt.pdf> <online>, 2005.
- [14] B.W. Lampson "A Note on the Confinement Problem" Communications of the ACM, vol. 16, 1973.
- [15] O. Sibert, P. A. Porras, R. Lindell "The Intel 80x86 Processor Architecture: Pitfalls for Secure Systems" IEEE Symposium on Security, Privacy, pp. 211, 1995.
- [16] D. A. Osvik, A. Shamir, E. Tromer "Cache Attacks, Countermeasures: The Case of AES" CT-RSA, 2006.
- [17] M. Neve, J.-P. Seifert "Advances on Access-driven Cache Attacks on AES" SAC, 2006.
- [18] E. Brickell, G. Graunke, M. Neve, J.-P. Seifert "Software mitigations to hedge AES against cache-based software side channel vulnerabilities" IACR ePrint, rep. 2006/052, 2006.
- [19] OpenSSL Project, <http://www.openssl.org/> <online>.
- [20] J.-S. Coron "Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems" CHES, pp. 292-302, 1999.
- [21] K. Tiri, D. Hwang, A. Hodjat, B.-C. Lai, S. Yang, P. Schaumont, I. Verbauwhede "AES-Based Cryptographic, Biometric Security Coprocessor IC in 0.18- μ m CMOS Resistant to Side-Channel Power Analysis Attacks" VLSI SYMPOSIUM, pp. 216-219, 2005.
- [22] T. Messerges, E. Dabbish, R. Sloan "Examining smart-card security under the threat of power analysis attacks" IEEE TC, Vol. 51, Issue: 5, pp. 541-552, 2002.
- [23] C. Clavier, J. Coron, N. Dabbous "Differential Power Analysis in the Presence of Hardware Countermeasures" CHES, pp. 252-263, 2000.
- [24] A. Shamir "Protecting Smart Cards from Passive Power Analysis with Detached Power Supplies" CHES, pp. 71-77, 2000.
- [25] N. Pramstaller, F. Gürkaynak, S. Häne, H. Kaeslin, N. Felber, W. Fichtner "Towards an AES Crypto-chip Resistant to Differential Power Analysis" ESSCIRC, pp. 307-310, 2004.
- [26] D. Suzuki, M. Saeki, T. Ichikawa "Random Switching Logic: A Countermeasure against DPA based on Transition Probability," IACR ePrint, rep. 2004/346, 2004.
- [27] T. Popp, S. Mangard "Masked Dual-Rail Pre-charge Logic: DPA Resistance without the Routing Constraints," CHES, pp. 172-186, 2005.
- [28] K. Tiri, M. Akmal, I. Verbauwhede "A Dynamic, Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards" ESSCIRC, pp. 403-406, 2002.
- [29] S. Moore, R. Anderson, R. Mullins, G. Taylor "Balanced selfchecking asynchronous logic for smart card applications," J. Microprocess. Microsyst., vol. 27.9, pp. 421-430, 2003.
- [30] K. Tiri, I. Verbauwhede "A digital design flow for secure integrated circuits" IEEE TCAD, vol. 25.7, pp. 1197-1208, 2006.
- [31] K. Tiri, I. Verbauwhede, "Simulation Models for Side-Channel Information Leaks" DAC, pp. 228-233, 2005.
- [32] F.-X. Standaert, E. Peeters, C. Archambeau, J.-J. Quisquater "Towards Security Limits in Side-Channel Attacks" CHES, pp. 30-45, 2006.
- [33] K. Kulikowski, A. Smirnov, A. Taubin "Automated Design of Cryptographic Devices Resistant to Multiple Side-Channel Attacks" CHES, pp. 399-413, 2006.
- [34] Aigner M., Popp T., Mangard S., Trifiletti A., Renato M., Olivieri M., Scotti G., "Side Channel Analysis Resistant Design Flow", ISCAS, 2006.
- [35] S. Guilley, Ph. Hoogvorst, Y. Mathieu and R. Pacalet, "The "backend duplication" method", CHES, pp. 383-397, 2005.

- [36] H. Li, A. Marketos, S. Moore “Security Evaluation Against Electromagnetic Analysis at Design Time” CHES, pp. 280-292, 2005.
- [37] S. Mangard, K. Schramm “Pinpointing the Side-Channel Leakage of Masked AES Hardware Implementations” CHES, pp. 76-90, 2006.
- [38] K. J. Kulikowski, M. G. Karpovsky, A. Taubin “Power Attacks on Secure Hardware Based on Early Propagation of Data” IOLTS, pp. 131-138, 2006.
- [39] E. Oswald, S. Mangard, C. Herbst, S. Tillich “Practical Second-Order DPA Attacks for Masked Smart Card Implementations of Block Ciphers” CT-RSA, pp. 192-207, 2006.
- [40] E. Oswald, S. Mangard “Template Attacks on Masking - Resistance Is Futile” CT-RSA, pp. 243-256, 2007.