SoC security: a war against side-channels

Sylvain Guilley, Renaud Pacalet

SoC security: a war against side-channels

Sylvain Guilley* and Renaud Pacalet†

GET / Télécom Paris, CNRS LTCI, Département communication et électronique.

* 46 rue Barrault, 75634 Paris Cedex 13, France <sylvain.guilley@enst.fr>.
† Institut Eurecom BP 193, 2229 route des Crêtes, 06904 Sophia-Antipolis Cedex, France <renaud.pacalet@enst.fr>.

Abstract

This article presents the state-of-the-art of the physical security of smart devices.

Electronic devices are getting ubiquitous and autonomous: their security is thus becoming a predominant feature. Attacks targeting the physical layer are all the more serious as hardware is not naturally protected against them. The attacks typically consist in either tampering with the device so as to make it malfunction or in spying at some information it leaks. Those attacks, either active or passive, belong to the *side-channel attack* class.

Active attacks operate by writing on an *ad hoc* side-channel: a degree a freedom normally not available to the end-user is modified by force. Passive attacks consist in listening to a side-channel: the attacker is thus able to gain more information about the device operation than it is supposed to.

Counter-measures against both types of attacks have been proposed and we show that only some of them are relevant. Active attacks are forfeited by an appropriate detection mechanism and passive attacks by the removal of all sorts of information leakage. As a consequence, securing hardware consists in watching side-channels or removing them if possible.

The increase of security is mainly driven by two trends: integration of the system (on a SoC) for improved discretion and development of a dedicated symptom-free electronic CAD. SoC security is thus foreseen to become a discipline in itself.

Key words: Security, SoC, side-channels, integration, tamper resistance, symptom-free electronic, secured CAD.

1 Introduction

1.1 The emergence of hardware security

Pervasive computing summarizes the fact that information processing capability spreads into the environment, thus creating an *ambient intelligence*. Ubiquitous chips communicate with one another, perform automatic self-test or monitor other machines. Those chips make up open networks built out of many independent or collaborating devices. In such an environment, security becomes a major issue.

Moreover, e-business needs more confidence in consumer/productor authenticity. The so-called trusted processor architecture recommended by the TCG [2] aims at defining the hardware as a neutral land where neither party has power to importune the other.

The need for security, be it for network protection or trust in business transactions, is imposing as a major feature of nowadays electronic devices.

Software security is not a feature but rather a process: the protection against attacks on systems consists in following a set of good practices, like regular software updates. Those attacks are safely avoided provided an adequate control-panel is defined and enforced by a strict security policy.

However, a class of attacks do not target the software, but the hardware. As we will discuss below, any cryptographic shield is of little help when the attacker has physical access to the device and is able to alter or monitor it.

Hardware attacks are particularly dangerous for three reasons:

- Hardware cannot be patched, or even if it can be reprogrammed, it is by far less reconfigurable than software. Therefore, hardware security must be thoroughly thought at, since it shall hold for a long period of time. As a consequence, the confidence level in an electronic device upon release must be higher than the one of a piece of software.

- Hardware attacks were discovered only recently. Unlike software attacks or *cryptanalysis*, that can been made essentially impracticable on modern ciphers, hardware attacks have not revealed their full power yet.

- The attacks on the hardware can originate from a wide variety of media. They range from the mere eavesdropping of the chip activity to the chip three-dimensional layout reverse-engineering, assisted by chemistry, advanced optics and sophisticated shape-recognition programs. In the rest of this article, the medium by which the attack is conducted is referred to as a *side-channel*.

Current design flows seldom include a security evaluation, although a quality methodology has been introduced by the *common criteria* [1] worldwide standard. Those criteria attribute an evaluation assurance level (EAL) to a device (target of evaluation, or TOE) given a specified protection profile (PP).

To our knowledge, hardware is rarely evaluated. The main reasons are:

- From the marketing point of view, EAL is not seen as attractive: clients currently do no value the certification of a security product.

- Security constraints throughout the chips design and manufacturing are not economically implementable or simply not implementable at all. Formal proofs, design by successive refinements, and similar techniques are currently not mature.
- Counter-measures against some physical attacks are still unknown.

1.2 Side-channels

An hardware device that can be physically accessed is especially vulnerable at its physical layer (of its OSI representation). It can be investigated by an apparatus that interacts with a side-channel. On the one hand, we will say that the side-channel is *written* when the electric device is altered. For instance, when shorting a node to ground with a FIB (Focused Ion Beam), the side-channel "electrical interconnexion network" is overwritten. One has normally not access to this network, which is supposed to be buried below a thick passivation layer and encapsulated into a plastic box. On the other hand, we will say that a side-channel is *read* when a physical quantity is monitored. This is the case of the simple power analysis (SPA [21]), an attack that consists in correlating a block of output data with its instant power consumption, in order to retrieve information about the instruction sequence being executed.

Attacks consisting in writing to a side-channel are *invasive* or *active*. They willingly alter the device normal operation, so as to move it into a weak state or to gain information from a computation fault. Active attacks can be irreversible, like the chip destruction for reverse-engineering purposes. Or they can be reversible, like glitch attacks: in this case, they consist in punctually moving the device out of its specified operating conditions.

The other attacks, that read a side-channel, are also referred to as *passive*. Their purpose is to gain more information about the computation being executed than allowed, and to deduce secrets from the analysis of the correlation between the legal information output by the device and the side-channel "stolen" information.

Active attacks can be more powerful than a passive attack, since the attacker can accurately target a security fault. However they require an appropriate tool and a good knowledge of the circuit. Those tools actually already exist: foundries are equipped with test equipments that are suitable tools for the investigation of circuits. On the opposite, passive attacks are easily done: most often, they can be realized thanks to no more than an acquisition card and a personal computer.

In the rest of the article, we will neither describe the attacks into details nor mention tricks. However, we will insist on the general principles.

1.3 Counter-measures

As far as invasive attacks are concerned, two counter-measures can be thought at:

- Strengthening of the hardware "physical" shielding [8, 9]. Of course, this measure makes an active attack more difficult. But it does not prevent a future similar attack with more powerful tools. This scheme is clearly not sustainable: it is a race between the attackers and the defenders.

- Preventing the attack by detecting it. This counter-measure solves the problem out: should an active attack be led, the device reacts, either by resetting, or by erasing the key material, or, if necessary, by auto-destruction. However, this technique works only provided the active attacks can be properly detected. *Glitch attacks* on the clock or on the power consist in applying a transient change on the clock or on the voltage supply so as to induce a random error [19]. They often succeed because they are so rapid that the system fails to detect them. Or even if the system was designed to detect them, it would become so sensitive that any noise from the environment would induce a false alarm. Tamper detection is thus a trade-off between a security level and an usage convenience.

There are two types of counter-measures against passive attacks:

- Internal data obfuscation, in order to make the side-channels random. Two types of such counter-measures are usually used:

  1. when a module is idle, it is all the same fed with dummy data, so as to fool attacks based on the activity monitoring.
  2. or the data can be manipulated under an encrypted form. Data or key blinding are based on algebraic properties: computations are not led on the regular operands, but one the operands masked with a random number, without changing the computation result [25].

  Those techniques make the passive attack more difficult, but do not prevent them: the side-channel can still be read, and if by any chance the dummy data generator or the random mask are known to the attacker, the counter-measure becomes useless.

- Side-channels removal. The goal of this method is to remove the sources of information leakage. This approach is the most secured, but also the most expensive. As a matter of facts, electronic chips designed from regular design flows leak a lot of information. Information of interest for an attacker include the computation time (*timing attacks* [20]), the instant power consumption (SPA and DPA [21]) or the electromagnetic emissions (EMA [15].) Two counter-measures have been put forward:

  1. inventing a new *computer aided design* (CAD) methodology for security, characterized by the fact that the computations do not have any external syndrome depending on the data being processed, and/or
  2. decorrelating time and data by resorting to self-timed logic [26].

1.4 Actors

Many security systems, if not all, rely on one secret key. Attacks aim at discovering this secret. If this key is known, the security system is defeated. Security is thus a struggle between attackers and defenders.

As far as side-channel attacks are concerned, the attackers can be sorted into three classes [6]:

- **Class I (clever outsiders):** They are often very intelligent but may have insufficient knowledge of the system. They may have access to only moderately sophisticated equipment. They often try to taken some advantage of an existing weakness in the system, rather than trying to create one.
- **Class II (knowledgeable insiders):** They have substantial specialized technical education and experience. They have varying degrees of understanding of parts of the system but potential access to most of it. They often have highly sophisticated tools and instruments for analysis.
- **Class III (funded organizations):** They are able to assemble teams of specialists with related and complementary skills backed by great funding resources. They are capable of in-depth analysis of the system, designing sophisticated attacks and using the most advanced analysis tools. They may use Class II adversaries as part of the attack team.

Defenders are security system designers and operators. The research community is getting involved in hardware security: for instance, resistance against side-channel attacks has been proposed as an evaluation criterion for the AES candidates [11, 14]. Moreover, the conference *Cryptographic Hardware and Embedded Systems* (CHES [3]) is encountering increasing success year after year.

The rest of the article is a review of the state-of-the-art in physical security. It is organized as follows: The security of the systems made up of several communicating chips is discussed in Sec. 2. The single-chip systems (of SoCs) face specific security issues, analyzed in Sec. 3. The increase of the integration density challenges physical security in many aspects. This question is treated in the Sec. 4 devoted to the prospective of physical security.

2 Multi-chip security issues

Many systems are comprised of several chips on a circuit board. This architecture is mandatory if the system is so complex that it cannot be integrated on one single chip or if it requires large amounts of memory. Those systems are vulnerable to attacks on the bus interconnexions. If the buses are not protected, they constitute a side-channel, onto which an attacker can couple. Without special care in the bus design, an attacker is able to substitute or read data in cleartext from the bus. For this reason, secured applications encipher data before sending it on the bus, and symmetrically, only accept enciphered data from the outside.

The typical architecture of the interface between a processor and an enciphered bus is depicted in Fig. 1. A secret key K, stored into the processor, is used to encipher the data (ED) and the addresses (EA).

- Every data d is mixed with its address a before encryption. This precaution aims at making it impossible to search for repetitions among the enciphered data d'. Moreover, the data words are full lines of cache, so as to avoid attacks

based on a tabulation of the instructions. Actually, if the data or the instructions were enciphered individually, a *cipher instruction search* attack [22] would be possible.

- The addresses a are enciphered in order to randomize the memory accesses. This measure prevents the reverse-engineering of the code.

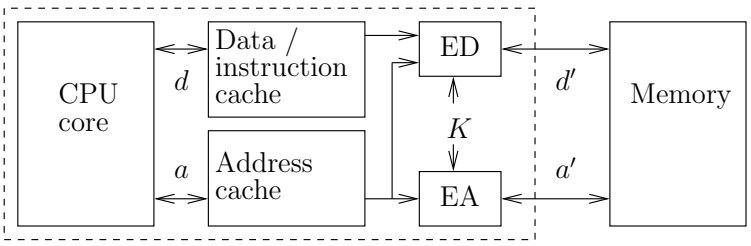

Figure 1: *Bus-encryption processor (dashed box) plus external memory.*

This architecture prevents cipher instruction search attacks, but does not protect against memory overwriting. The solution to this fault consists in checking the memory integrity using an hierarchical signature scheme [16, 23]. However, hash algorithms require many iterations (for instance 80 for SHA-1). Those iterations can be repeated $\mathcal{O}(\log(n))$ times, where n is the memory size, for every memory access. There does not exist processors able to encipher and sign so quickly.

Therefore, systems are better protected if they lie on a single chip. SoC are thus the candidates of choice for secured applications. Moore's law [5] is in this respect fostering security.

3 SoC security issues

Integrating a whole system onto a single substrate makes side-channel attacks more difficult. On a SoC, interconnexions between blocks are internal (thus hidden) and physical access to the chip is tedious. In addition, the smaller the chip, the smaller the information it leaks. Nonetheless, SoC are not intrinsically immune to active or passive side-channel attacks.

3.1 Invasive attacks on SoC

Destructive attacks have been shown to be powerful. The TAMPER laboratory, in Cambridge, has carried out several sorts of destructive attacks [8] to retrieve secrets. The principle of some such attacks can be found in [9] (Chip Rewriting Attacks) and [10] (Non-Differential Fault Attacks).

But secrets can also be discovered without damaging the SoC. Non-destructive invasive attacks on SoC include:

- RAM overwritting [27]; current induction using an alternative current in a coil can be used to force the state of a static RAM point.

- optically induced faults [30]; illumination of a target transistor causes it to conduct, thereby inducing a transient fault.
- clock or power glitch attacks [19]; a glitch on the clock or the power supply induces internal errors.

Of course, it is difficult to resist a destructive attack. Nevertheless, non-destructive fault attacks can also be efficient, as underlined in [13]: one single error in an RSA signature using *Chinese Remainder Theorem* (CRT, page 610 of [25]) allows the modulus factoring. Smartcards are equipped with a wide variety of sensors [8]. However, an attacker can bypass them if she operates cautiously. Consequently, the best defense against invasive attacks is their detection.

Self-timed circuits are by design immune to fault attacks [4]. A single error in an asynchronous circuit, like an event insertion, makes it crash. In [26], a more controllable counter-measure based on a *self-checking logic* is proposed. Every bit of data is encoded on two bits, interpreted as shown in Tab. 1. Moreover, the gates are designed to propagate the *alarm* signal, so that every single error is reported to the system. It is then up to the system to decide which action to undertake.

Table 1: *Data signalization with error-detection used in [26].*

Bits value	Encoding
00	clear
01 or 10	logical 0 or 1
11	alarm

3.2 Passive attacks on SoC

An attacker can adopt two strategies to passively spy a SoC. She can either read its state when it is idle, taking advantage of physical phenomenon like memory remanence [9, 27], or spy it during its dynamic operation. We assume that memories can be used securely (*e.g.* cleared after usage) and thus tackle in this section to dynamic passive attacks [28].

Those attacks can be classified according to the degree of accuracy required to perform them. We distinguish coarse-grained attacks that operate at the algorithmic level from accurate attacks that operate at the clock period level.

3.2.1 Passive attacks at the algorithmic level

The typical example is the *timing attack* [20]. An algorithm overall execution time is generally a function of the secret if the algorithm is not carefully designed (either obfuscated or balanced.)

Some attacks, like the SPA [21], consist in the recognition of execution patterns. The algorithm shows different power consumption patterns depending on the input data. Conditional branching can be detected by this method. The same attack can be realized based on electromagnetic measurements [15].

```
Inputs : M , K
R = 1 ;
for i = |K| − 1; i ≥ 0 ; i − − do
  R = R^2 ;
  /* Unbalanced branching */
  if K_i == 1 then
    R = R × M ;
  end if
end for
Return R = M^K ;
```

Figure 2: *Regular* square-and-multiply *exponentiation algorithm.*

```
Inputs : M , K
R = 1 ; S = M ;
for i = |K| − 1; i ≥ 0 ; i − − do
  /* Balanced branching */
  if K_i == 1 then
    R = R × S ; S = S^2 ;
  else
    S = S × R ; R = R^2 ;
  end if
end for
Return R = M^K ;
```

Figure 3: Montgomery ladder *exponentiation algorithm [18].*

The passive attacks at the algorithmic level are typically defeated with balanced versions of the algorithms. Properly balancing an algorithm is not trivial: merely replacing conditional branches by dummy operations if nothing is to be done is a counter-measure that introduces an fault. The dummy operations can be tested by faults attacks: the attacker can test whether an operation is dummy or not, which reveals the conditional branches that were supposed to be hidden. The execution time can also be obfuscated: implementations of RSA by the library OpenSSL were vulnerable to the timing attack until the exponent was blinded [12]. However, the blinding technique works only for some algorithms. Moreover, the blinding counter-measure works only provided the blinding algorithm is not faulty.

Unconditionally safe algorithms implementations execute in constant time. For example, the *Montgomery powering ladder*, illustrated in Fig. 3, is a constant time solution [18] for the modular exponentiation problem.

3.2.2 Passive attacks at the clock period level

Differential attacks (not to be confused with *differential cryptanalysis*) target a single logical gate. Out of a collection of power traces, the value of a given bit is guessed by a maximum likelihood statistical analysis. The DPA [21] and DEMA attack [7, 15] are based on this principle.

The attacks at the clock level are powerful, since whatever the noise, they asymptotically reveal a symptom (or side-channel) from a chosen gate. The removal of all symptom implies (at least) four requirements, that are not met by standard gates [17]:

1. Equilibrating gates in computation time.

2. Synchronizing the inputs, so that the starting date of the computation does not depend of the operands arrival order. This synchronization requires the use of C-Elements [29].

3. Making sure the power consumption is independent of the input data. It is important to notice that not only the average value but also the power profile must be identical.

4. Leaving no evidence of a previous computation. This condition is not straightforward in CMOS logic, where parasitic capacitances memorize the state of the internal nodes of a gate. A reset or clearing step is mandatory between two computations.

Such gates can be realized provided their truth table contains as many 0s as 1s [26]. Otherwise, the gate must comprise dummy material, so as to artificially build a balanced implementation. An example of a symptom-free `NAND` gate is illustrated in Fig. 4. Every bit of data is encoded on a dual-rail with the signalization of Tab. 1. In addition, every valid data (either 01 or 10) is followed by a null (00) spacer, so as to start over the next computation in the same conditions as the previous one. The C-Elements and the OR gates making up the `NAND` gate of Fig. 4 must be designed in such a way any charges otherwise left in the parasitic capacitances are removed.

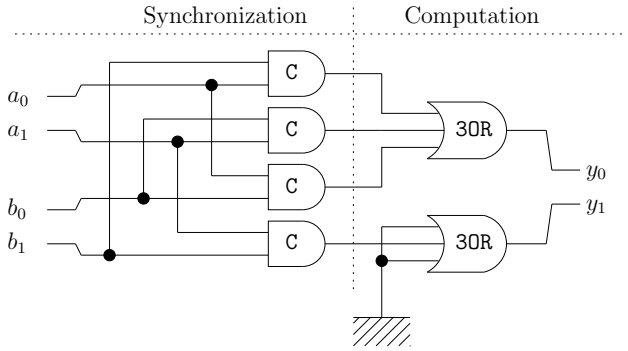

Figure 4: *An example of a `NAND` gate without side-channels.*

Those new gates require a new methodology for CAD: synthesizers, place and route tools. Those tools must have a *security check* feature. Thus a *trusted electronic CAD*, along with evaluation tools, are to be invented and developed.

Security against side-channel attacks is thus foreseen to become part and parcel of the SoC design methodologies.

4 Prospective

4.1 The cost of "design for security"

The previous sections have shown that there exists counter-measures against side-channel attacks. The implementation of the counter-measures implies an additional design cost and decreases the system performances.

Much often, design for performance and design for security seem antinomic. Today's circuits are secured or not, but few circuits are designed to meet a specified level of security.

In order to design a circuit with both performance and security specifications, new methodologies and tools are needed. The cost and the impact on the system performances of the specified level of security is a knowledge database that currently lacks. Such a database could feed to an architectural exploration tool that takes into account both security and performances constraints. The developpement of such tools is the key to "design for performance" and "design for security" co-design.

4.2 Side-channel attacks connexions with SoC complexity increase

The increase of density of integration in SoCs is still considered a sustainable trend [5]. This raises the question of the impact of chips complexification on their security. On the one hand, the increase of the system size multiplies the side-channels, and on the other hand, the increase of complexity makes the side-channels more noisy.

The first point emphasizes the fact that side-channels are definitely going to be a growing threat on SoCs. The second point is not relevant. The denser integration, the weaker the side-channels (emitted by the activity of a single gate) and the more noisy the overall (resulting for the activity of all the gates) signal an attacker can spy. But in parallel with the secured SoCs shrink, the investigation tools become more accurate and possibly can operate more locally, thus ignoring the noisy activity of the rest of the circuit.

In addition, from a theoretical point of view, it has been shown that whatever the technology used, a computing machine must dissipate energy during its operation [24]. The side-channels are thus a necessity and not a computing artefact. The advance of device integration will lower the side-channels (without making them disappear) and in the meantime more accurate measurement appartus will allow attacker to measure them. The side-channels threat is not lowered by Moore's law.

5 Conclusion

Since the publication in 1996 of the timing attack and in 1999 of the power attacks by Paul Kocher, hardware security has become an active field of research. It appeared that hardware, whatever its implementation (microcontrollers, processors executing software, ASIC or FGPA), was naturally not shielded against those attacks. In the scope of physical attacks, vulnerabilities always result from the exploitation of a side-channel. Active or invasive attacks tamper with a side-channel, whereas passive attacks only spy at a side-channel. The SoC are the most secured devices since their integration makes access to their side-channels especially difficult.

Active attacks can be made more difficult by strengthening the device tamper-resistance. But the hardware is better protected by an appropriate system of intrusion detection.

Two classes of counter-measures against passive attacks have been put forward in the scientific literature. On the one hand, data obfuscation, like data or key blinding, consists in randomizing the side-channel. However, this counter-measure does not prevent an attacker from spying the side-channel. One the other hand, removing the side-channel ensures that no information leaks.

Specific methods and tools are to be developed to enforce those counter-measures. In parallel, a systematic evaluation of the design security will be required for any step in the design flow.

References

[1] Common Criteria website. `http://www.commoncriteria.org/`.

[2] `TCG` (formerly `TCPA`) website. `https://www.trustedcomputinggroup.org/`.

[3] `CHES` conference website. `http://www.chesworkshop.org/`.

[4] `G3Card` website. `http://www.g3card.org/`.

[5] `IRTS` (International Technology Roadmap for Semiconductors) website. `http://public.itrs.net/`.

[6] Abraham (D.), Dolan (G.), Double (G.), and Stevens (J.). Transaction Security System. *IBM Systems Journal*, 30(2):206–229, 1991.

[7] Agrawal (D.), Archambeault (B.), Rao (J. R.), and Rohatgi (P.). The EM Side-Channel(s): Attacks and Assessment Methodologies. *IBM report*. `http://www.research.ibm.com/intsec/emf-paper.ps`.

[8] Anderson (R.) and Kuhn (M.). Tamper Resistance – a Cautionary Note. *Proc. of the Second Usenix Workshop on Electronic Commerce*, pages 1–11, November 1996.

[9] Anderson (R.) and Kuhn (M.). Low Cost Attacks on Tamper Resistant Devices. *Proc. of IWSP: 5th International Workshop of Security Protocols*, 1361:125–136, April 7–9 1997. Paris (France).

[10] Biham (E.) and Shamir (A.). Differential Fault analysis on secret key cryptosystems. *Proc. of CRYPTO'97*, 1294:513–525, 1997.

[11] Biham (E.) and Shamir (A.). Power Analysis of the Key Scheduling of the AES Candidates. *Proc. of the Second Advanced Encryption Standard (AES) Candidate Conference*, 1999. `http://csrc.nist.gov/CryptoToolkit/aes/round1/conf2/aes2conf.htm`.

[12] Boneh (D.) and Brumley (D.). Remote timing attacks are practical. *Proc. of the 12th Usenix Security Symposium*, 2003. `http://crypto.stanford.edu/~dabo/papers/ssl-timing.pdf`.

[13] Boneh (D.), Demillo (R. A.), and Lipton (R. J.). On the Importance of Checking Cryptographic Protocols for Faults. *Proc. of Eurocrypt'97*, pages 37–51, 1997. `http://theory.stanford.edu/~dabo/abstracts/faults.html`.

[14] Chari (S.), Jutla (C.), Rao (J. R.), and Rohatgi (P.). A Cautionary Note Regarding Evaluation of AES Candidates on Smart-Cards. *Proc. of the Second Advanced Encryption Standard (AES) Candidate Conference*, 1999. `http://csrc.nist.gov/encryption/aes/round1/conf2/aes2conf.htm`.

[15] Gandolfi (K.), Mourtel (C.), and Olivier (F.). Electromagnetic Analysis: Concrete Results. *Proc. of CHES'01*, 2162:251–261, 2001.

[16] Gassend (B.), Clarke (D.), Suh (G. E.), van Dijk (M.), and Devadas (S.). Caches and hash trees for efficient memory integrity verification. *Proc. of the Ninth International Symposium on High Performance Computer Architecture (HPCA-9)*, February 2003.

[17] Guilley (S.), Hoogvorst (P.), Mathieu (Y.), Pacalet (R.), and Provost (J.). CMOS Structures Suitable for Secured Hardware. *Proc. of DATE'04*, pages 1414–1415, February 2004.

[18] Joye (M.) and Yen (S.-M.). The Montgomery Powering Ladder. *Proc. of CHES'02*, pages 291–302, 2002.

[19] Kömmerling (O.) and Kuhn (M.). Design Principles for Tamper-Resistant Smartcard Processors. *Proc. of the Usenix Workshop on Smartcard Technology (Smartcard'99)*, pages 9–20, May 1999.

[20] Kocher (P.), Jaffe (J.), and Jun (B.). Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. *Proc. of CRYPTO'96*, 1109:104–113, 1996.

[21] Kocher (P.), Jaffe (J.), and Jun (B.). Differential Power Analysis: Leaking Secrets. *Proc. of CRYPTO'99*, 1666:388–397, 1999.

[22] Kuhn (M. G.). Cipher Instruction Search Attack on the Bus-Encryption Security Microcontroller DS5002FP. *IEEE Transactions on Computers*, 47(10):1153–1157, oct 1998.

[23] Lauradoux (C.) and Keryell (R.). CryptoPage-2 : un processeur sécurisé contre le rejeu. *Proc. of RENPAR'15 / CFSE'3 / SympAAA'2003*, October 2003.

[24] Matherat (P.) and Jaekel (M.-T.). Dissipation logique des implémentations d'automates – dissipation du calcul. *Technique et Science Informatiques*, 15(8):1079–1104, 1996. `http://www.comelec.enst.fr/~matherat/mes_publis/tsi96/`.

[25] Menezes (A. J.), van Oorschot (P. C.), and Vanstone (S. A.). Handbook of Applied Cryptography. 1997. CRC Press. ISBN: 0-8493-8523-7.

[26] Moore (S.), Mullins (R.), Cunningham (P.), Anderson (R.), and Taylor (G.). Improving smart card security using self-timed circuits. *Proc. of Async'02*, pages 211–218, April 2002.

[27] Nève (M.), Peeters (E.), Samyde (D.), and Quisquater (J.-J.). Memories: a Survey of their Secure Uses in Smart Cards. *Proc. of IEEE SISW 2003*, October 2003. Washington DC, USA.

[28] Oswald (E.). *On Side-Channel Attacks and the Application of Algorithmic Countermeasures*. PhD thesis, May 2003.
`http://www.iaik.tu-graz.ac.at/aboutus/people/oswald/papers/PhD.pdf`.

[29] Shams (M.), Ebergen (J.), and Elmasry (M.). Modeling and comparing CMOS implementations of the C-element. *IEEE Transactions on VLSI Systems*, 6(4):563–567, 1998.

[30] Skorobogatov (S. P.) and Anderson (R. J.). Optical Fault Induction Attacks. *Proc. of CHES'02*, 2002.
`http://www.cl.cam.ac.uk/~sps32/ches02-optofault.pdf`.