Investigation of timing constraints violation as a fault injection means

Loïc Zussa*,Jean-Max Dutertre*, Jessy Clédière‡, Bruno Robisson† and Assia Tria†
*†Département Systèmes et Architectures Sécurisées (SAS)
*École Nationale Supérieure des Mines de Saint-Étienne (ENSMSE), †CEA-LETI, Gardanne, France
{zussa, dutertre}@emse.fr {bruno.robisson, assia.tria}@cea.fr
‡CEA-LETI, Minatec Campus, Grenoble, France
jessy.clediere@cea.fr

***Abstract*—Secure circuits are prone to a wide range of physical attacks. Among them, fault attacks are based on modifying the circuit environment in order to change its behaviour or to induce faults into its computations. As a result, the security level of the circuit under attack may be weaken. Many means are of common use to inject such faults: laser shot, electromagnetic pulse, overclocking, chip underpowering, temperature increase, etc. However, the mechanisms involved in the fault injection process have not been yet deeply investigate. Especially, those that have a global effect linked to timing constraints violation. In this paper we provide an experimental proof of the uniqueness of the fault injection process by means of the target's clock, power supply, or temperature alteration. We also studied further the properties of these fault injection means. These insights are intended to give designers guidelines to strengthen fault countermeasures. It also enable to imagine broad-spectrum countermeasures against most of the fault injection means.**

***Index Terms*—Fault injection, timing constraint violation, overclocking, underpowering, overheating.**

I. Introduction

Many of our daily used electronic devices embed cryptographic features (e.g. smart cards, cell phones, pay TV, passports, etc.). The use of cryptographic algorithms is intended to provide to the end users confidentiality, authentication and data integrity services. As a consequence, the integrated circuits (ICs) implementing these security features are often targeted by malicious attackers. They seek to extract confidential information like encoding keys from their targets. To that end, a wide range of physical attacks against secure circuits have been introduced over the past decades.

Physical attacks (or *hardware* attacks) target the ICs which implement cryptographic algorithms. It exists three main kinds of physical attacks. The first one, called *invasive attacks*, refers to any technique based on the analysis or modification of the target's design by using invasive methods (e.g. probing the data buses of an IC [1] or using focused ion beams (FIB) to cut or change its interconnections). The second kind, called *side channel analysis* (SCA), is a passive attack. It exploits the information leakage related to some physical values of the chip, such as its power consumption, its electromagnetic emissions or the duration of its computations. Indeed, the strong correlation that exists between these quantities and the processed data has made it possible to develop statistical techniques to retrieve the secrets concealed in secure devices using SCA [2], [3]. The third kind of physical attack, called *fault attacks* (FA), consists in modifying the circuit environment in order to change its behaviour or to induce faults into its computations. Many means are of common use to inject such faults: laser shot, electromagnetic pulse, overclocking, chip underpowering, temperature increase, etc.

There are three main subclasses of fault attacks: algorithm modification, safe error and differential fault analysis. Algorithm modification consists in replacing instructions executed by a microcontroller [4] to circumvent its security features, or in weakening the strength of an iterative encryption algorithm by changing the number of its rounds [5]. Safe-error attacks are based on distinguishing between the normal and abnormal behaviours of the chip in presence of disturbances intended to induce faults. In order to retrieve the sensitive data processed by the target, they are correlated with the chip's behaviours [6]. Differential fault analysis (DFA) consists in retrieving the keys by comparing correct ciphertext and faulty ciphertexts (i.e. ciphertexts obtained from a faulted encryption). This technique was first introduced for public key encryption algorithms [7], and rapidly extended to secret key algorithms [8]. From that time, many DFA schemes have been proposed to attack various encryption algorithms. All of them are associated with strong timing, range and location requirements regarding the fault injection process. If the faults are not induced at the proper time in the algorithm, or affect the wrong bits, the entire DFA process fails.

As a consequence, the ability to control precisely the fault injection process is a key element in carrying out any fault attack. A fine understanding of the various fault injection mechanisms is also mandatory to enable the design of fault resistant ICs. That's the reason why this paper focuses on an in-depth investigation of the most common fault injection means: those which are related to the violation of the ICs' timing constraints. These means are based either on modifying the clock signal (i.e. overclocking) or on increasing the data propagation time through the circuit's logic (i.e. voltage supply deprivation or temperature increase). They have a global effect on the targeted circuit because they affect similarly the whole logic. They are also of great concern because they are usually implemented with none expensive equipments.

These fault injection means are known and used since the beginning of FA [9]. However, there is few papers in the scientific bibliography ([10], [11], [12]), which report a deep investigation and understanding of the underlying fault injection mechanisms. Our contributions to that research field are:

- the experimental proof of the uniqueness of the fault injection mechanism related to the aforementioned means,
- a study of the reproducibility of the obtained results,
- a fine analysis of fault injection due to timing constraints violation (with a focus on the fault range and the ability to control faults' location).

To conduct this study we have chosen a programmable circuit (FPGA) as a test vehicle. It implements the advanced encryption standard (AES [13]), which is a secret key encryption algorithm.

This article is organized as follows. A remainder on timing constraints and an explanation of how fault may be injected by their violation are given in section II. The experimental setup and results are described in section III. These results are analysed in section IV. Finally, our findings are summarized in the concluding section V with some perspectives.

II. Timing constraints

In this section, the basics of timing constraints are firstly reminded. Secondly, the two means of inducing timing constraints violation for the purpose of fault injection are reviewed. Then, the experimental proof intended to demonstrate the uniqueness of the fault injection mechanism is introduced.

A. Timing constraints

Almost all digital ICs work according the principle of synchrony: they use a common clock signal to synchronize their internal operations. Figure 1 outlines a representation of their internal architecture: combinatorial logic (marked $\sum$) surrounded upstream and downstream by register banks made of D flip-flops (DFF) sharing the same clock signal (clk).

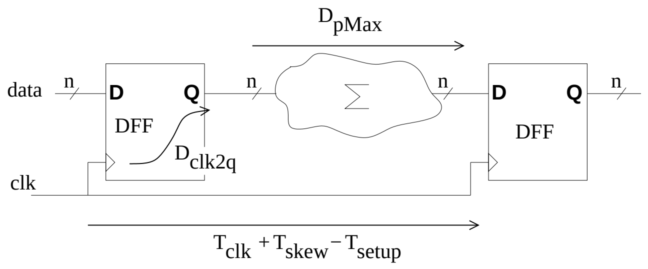

Fig. 1. Internal architecture of digital ICs

Data are released from the first register bank on a clock rising edge and then processed through the logic before being latched into the next register bank on the next clock rising edge. Thus, in first approximation the clock period (T_{clk}) has to be longer than the maximum data propagation time through the logic (D_{pMax}) to ensure correct operation. Besides, a precise writing of the timing constraint equation requires to take into account three other parameters: D_{clk2q} the delay elapsed between the clock rising edge and the actual update of a register's output; T_{skew} the skew or slight phase difference that may exist between the clock signals at the clock inputs of two different registers; T_{setup} the setup time which is the amount of time for which a D flip-flop input must be stable before the clock's rising edge to ensure reliable operation. It also exists an hold time (T_{hold}) which expresses the same constraint but after the clock edge. Hence, the timing constraint equation (eq. 1) is obtained:

$$T_{clk} > D_{clk2q} + D_{pMax} + T_{setup} - T_{skew} \tag{1}$$

An illustration, at bit level, of the signal flow is given in figure 2-a for which the timing constraint is fulfilled. It exists a time margin (called the slack) between the last signal transition at the input of the downstream register and the setup time.

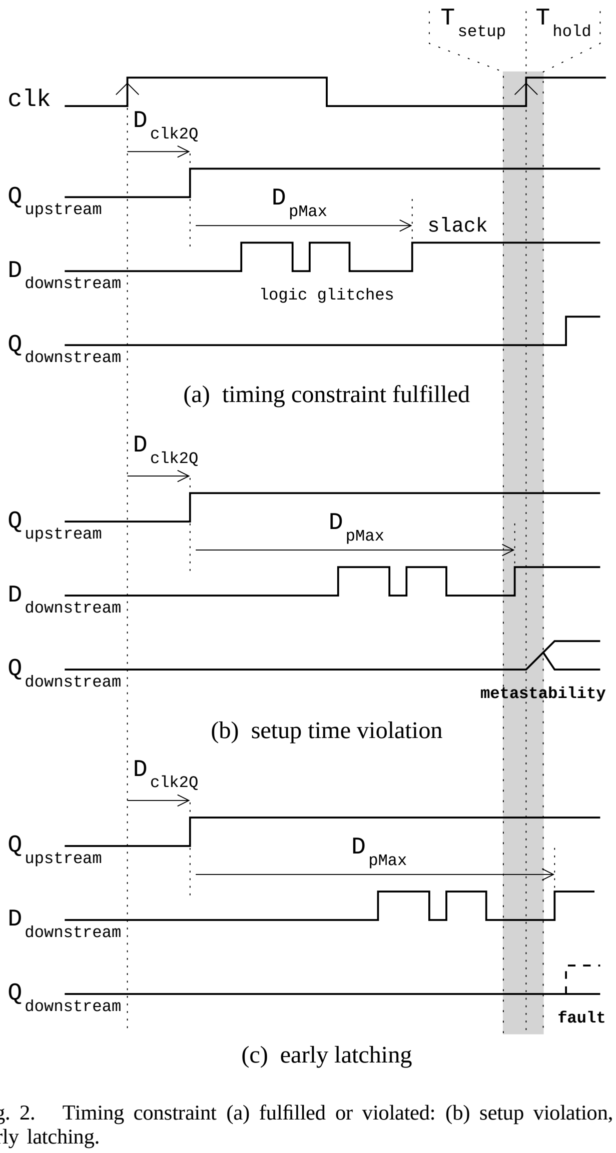

Fig. 2. Timing constraint (a) fulfilled or violated: (b) setup violation, (c) early latching.

The violation of this timing constraint is a straightforward means to inject faults into a circuit. Two stages of such violations are depicted on fig. 2-b-c. A shaded area around the clock rising edge delineates a time interval which corresponds to a non-deterministic behaviour of the DFF in case of any transition on its input. It extends before and after the clock edge from an amount of time equal to the setup and hold times respectively. A setup time violation arises if the last signal transition is too close to the clock rising edge (Fig.2-b). Then the DFF's output undergoes a metastable behaviour [14]: it may stabilize either on a high or low state. An error may occur or not. Fig.2-c introduces a second kind of faulty behaviour: an early latching. In this instance, an erroneous logic value is latched by the register: a fault is actually injected. The fault injection process is then purely assured and deterministic because there is no signal transition in the shaded area. Hereafter, we will refer to constraint timing violation for both cases.

The two next subsections reports the means to achieve such timing violations.

B. Overcloking

A straightforward approach to inject faults through timing constraints violation is overclocking. It consists in decreasing the clock's period until faults appear by setup time violation or early latching (Eq. 2).

$$T_{clk_{overclocking}} < D_{clk2q} + D_{pMax} + T_{setup} - T_{skew} \qquad (2)$$

An increase in the clock frequency does not provide any timing control: faults may be induced at any clock cycle of the circuit. An enhancement of that technique is the use of clock glitches, which is based on inducing a timing violation by modifying only one clock period [15].

C. Increasing propagation time

The second means of violating the timing constraint equation (cf. eq. 1) is by increasing its right handside part. It may be achieved by increasing the data propagation time through the logic (D_{pMax}). As shown by equation 3:

$$T_{clk} < D_{clk2q} + D_{pMax_{increased}} + T_{setup} - T_{skew} \qquad (3)$$

For the sake of simplicity, the data propagation time through a simple CMOS inverter as a function of the power supply and the temperature is recalled. The physical equations are obviously more elaborated for more complicated logic. However, the observable trends are alike. The inverter's architecture and waveforms are depicted in figure 3 where t_{pLH} and t_{pHL} are its propagation delays for an output's transition from low to high and high to low logic levels respectively.

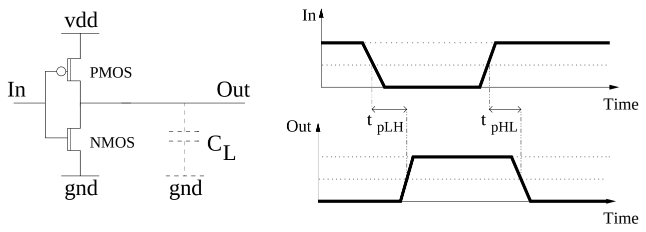

Fig. 3. Inverter: architecture and typical waveforms

t_{pLH} and t_{pHL} may not have the same value. Hence, the data propagation time through the inverter (and through any logic block) depends on the handled data: the propagation time is data dependent.

The propagation time, t_{pLH} (eq. 4), is obtained from a first order analysis [16] of the inverter's dynamic behaviour:

$$t_{pLH} = \frac{C_L \left[\frac{2|V_{th,p}|}{V_{DD} - |V_{th,p}|} + \ln\left(3 - 4\frac{|V_{th,p}|}{V_{DD}}\right)\right]}{\mu_p C_{ox} \frac{W_p}{L_p}(V_{DD} - |V_{th,p}|)} \qquad (4)$$

where V_{DD} is the power supply voltage, C_L the load capacitance, $V_{th,p}$ the PMOS threshold voltage, μ_p the holes mobility, C_{ox} the gate oxyde capacitance and (W_p/L_p) the aspect ratio of the PMOS. A similar equation for t_{pHL} may be derived from eq. 4 by substituting the parameters related to the inverter's NMOS (e.g. μ_n, (W_n/L_n), $V_{th,n}$) for those related to the PMOS.

1) Underpowering: as stated by eq. 4 any decrease of V_{DD} will induce an increase of the propagation delay of the inverter. By extension, the data propagation time through any logic block is increased when an IC is underpowered. Hence, underpowering is a common means to achieve fault injection by violation of the timing constraint.

2) Overheating: the two temperature-dependent parameters of eq. 4 are the charge carriers mobility and the threshold voltage. However, the temperature dependence of the propagation delay due to V_{th} is a few percent of that due to μ (the mobility for either holes or electrons). Therefore, only the temperature dependence of μ (see eq. 5) is considered at first order [17].

$$\mu(T) = \mu(T_0)\left(\frac{T}{T_0}\right)^{\alpha} \qquad (5)$$

where T is the temperature, T_0 and α fitting parameters. α varies approximately from -2.2 to -1.5 depending on the doping level [17]. Thus, by extension and from eq. 4 and 5, the data propagation time is increased when an IC is overheated.

D. Several fault injection means, a common mechanism

Hence, overclocking, underpowering and overheating are three suitable means to inject faults into a circuit by violation of its timing constraints [11], [10]. Intuitively, these three means are usually considered to rely to a same mechanism.

The novelty of our approach lies in the proposal of an experimental validation of this assumption. This proof is based on the analysis of the injected faults by means of these three techniques for a test chip handling the same data (the latter condition is due to the data-dependence of the propagation times, and consequently of the induced faults). The uniqueness of the injected faults for every of these three means is the core of that proof as reported in the next section.

III. Experimental validation

A. Experiments outline

The following experiments are devoted to the analysis of the faults injected into our test chip by violation of its timing constraints. Three means are investigated: overclocking, underpowering and overheating. They all have a global effect on the target (i.e. they impair similarly every of its part). Hence, injected faults will be located where the timing constraint is violated according the mechanism depicted in figure 2. Any fault may result from a setup time violation, which exhibit a metastable behaviour, or from an early latching, which is purely deterministic. Consequently, it may be a very difficult task to analyse the injected faults if they are too numerous because of the non deterministic behaviour of metastability. Thereby, we have chosen to focus on the first injected fault that may appear when the stress (i.e. overclocking, voltage deprivation or overheating) applied to the test chip is progressively increased. The obtained fault will also reveal the critical path of the design (i.e. the logic path with the longest propagation delay) associated with the current data set. Indeed, the propagation delay through the combinatorial logic is data dependent. As the test chip runs the AES encryption algorithm, the data that have an effect on the propagation delays are both the plaintext and the key. That's the reason why we used a dataset of 10,000 {Plaintext, Key} pairs choosen randomly.

AES is a standard established by the NIST [13] for symmetric key cryptography. It is a substitution and permutation network based on four transformations (i.e. SubBytes, ShiftRows, MixColumns, AddRoundKey) used iteratively in rounds as depicted on fig.4. The test chip (Xilinx Spartan 3A fpga) embeds an hardware 128-bit version of this algorithm (AES-128). It processes data blocks of 128 bits (usually represented as 4x4 bytes matrix, called the AES' state) in ten rounds (after round 0). The round keys (K1 to K10) used during every round are calculated on-the-fly by a key expansion module. Hence, a full encryption is completed in eleven clock periods. The test chip nominal clock period is 100 MHz, and its core nominal voltage is 1.2V. In this work, AES is mainly used as

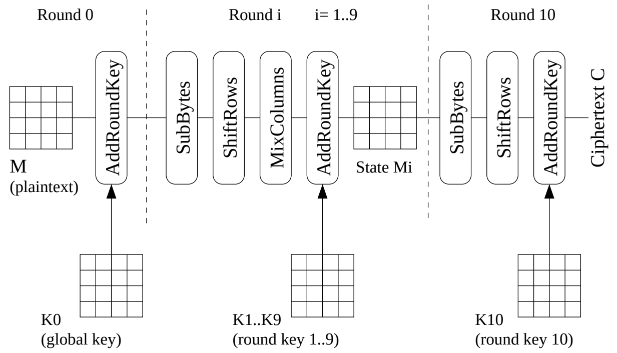

Fig. 4. AES-128 encryption algorithm.

a test element. Thus, we will not go deeper into its properties. However, because this algorithm is likely to be subject to DFA, the obtained results are yet of interest. As the delays in the encryption module are greater than the delays in the key expander module we can assume that the encryption module will be faulted before every others modules.

B. Overclocking

A first experiment was conducted by using overclocking as a fault injection means. For each {Plaintext, Key} pair of the dataset the following process was followed: send the plaintext and key to the test chip, launch a first encryption at nominal settings to obtain a correct ciphertext used as a reference, increase successively by an elementary step the stress applied to the target (i.e. the clock frequency) until a first faulty ciphertext is obtained. The elementary frequency step was set to 200 kHz. Then, the faulty ciphertext was processed by reversing the encryption (the key is known) and a comparison was made between the intermediate states of the computations to retrieve the injected fault. The gathered data were added to the dataset: {Plaintext, Key, Ciphertext, Fault}.

The obtained faults were single-bit with a rate slightly greater than 90%. Figure 5 gives the spreading of the single-bit faults over the AES' rounds. Besides, the value of the

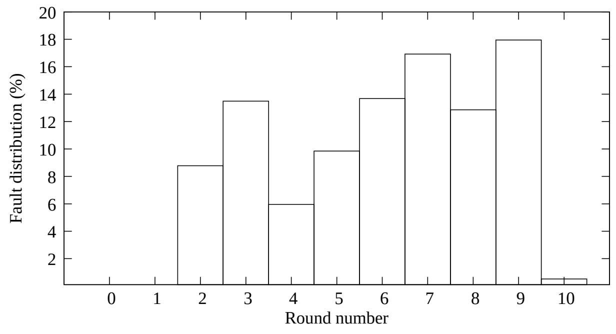

Fig. 5. Faults spreading over the AES' rounds (9,000 single-bit faults).

frequency associated with every injected fault gives the corresponding critical time (the accuracy of these measures is given apart from T_{setup}, T_{skew} and D_{clk2q}). These critical times were found between 7,418 ps and 8,741 ps with an average value of 7,968 ps.

C. Underpowering

A second set of experiments was conducted by using underpowering as a fault injection means according to the process described in III-B: successive decreases of the target's power supply (its nominal voltage is 1.2V) by elementary steps of 2mV until a first fault is injected. The whole {Plaintext, Key} pairs related to the injection of single-bit faults by overclocking were tested. For every pair, the injected fault was the same: a matching rate of 100% between the faults induced by overclocking and voltage deprivation was obtained. Faults were induced for a power supply ranging from 1.061V to 0.979V, with an average value of 1.02V.

Jointly, we used our frequency generator to measure the critical time linked to every setting of the power supply. Figure 6 depicts the critical time as a function of the power supply for three {Plaintext;Key} pairs of the dataset.

D. Overheating

The third set of experiments was performed by using overheating (a manually controlled thermal air-blower was used) as a fault injection means. Since, heating an IC is done with long time constants only ten {Plaintex, Key} pairs of the dataset were tested according to the process described in

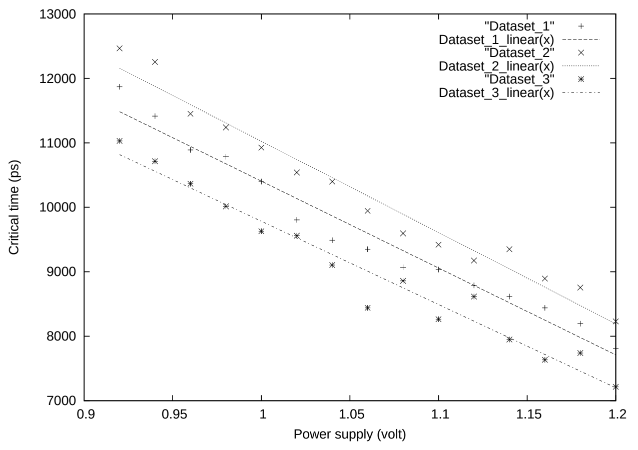

Fig. 6. Critical time versus power supply.

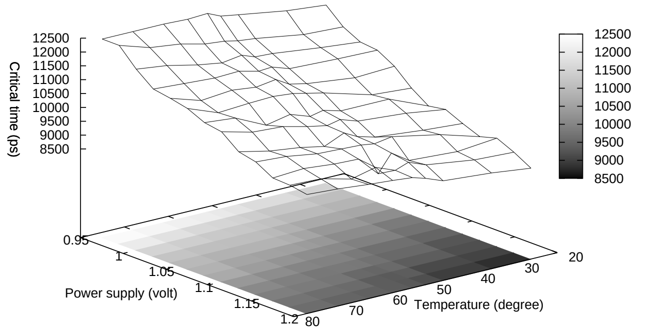

Fig. 8. Critical time versus power supply and Temperature

III-B and III-C. As expected (see eq. 5) an increase in the chip's temperature (measured outside the package) led to an increase in its critical time until faults are injected when the critical time goes beyond the nominal clock period. For every pair, identical faults with those induced by overcloking and underpowering were obtained. Figure 7 presents the critical time as a function of the temperature for three experiments.

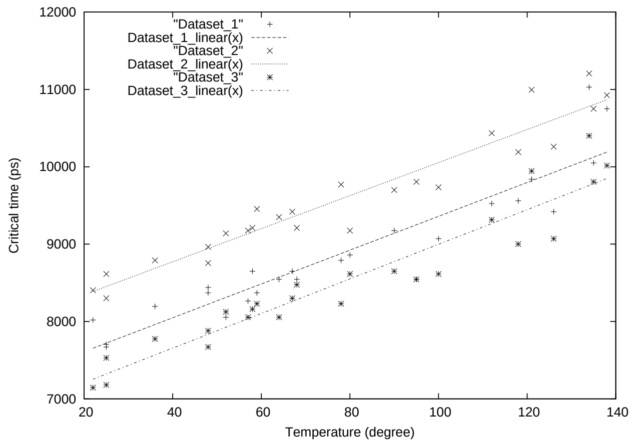

Fig. 7. Critical time versus Temperature

E. Underpowering and overheating

Since, underpowering and overheating both lead to an increase of the target's critical time we used these two injection means in combination to build a 3D curve of the critical time for a given dataset (presented on fig.8).

IV. Results analysis

A. Experimental proof

We drawn in section II the hypothesis that faults injected by overclocking, underpowering or overheating were induced according to a common mechanism based on timing constraints violation. For the purpose of validating this assumption we have conducted three sets of experiments reported in section III in order to obtain an experimental proof. The same single-bit faults were always obtained by using these three fault injection means. As a consequence, we believe that this is a valid experimental proof of the uniqueness of the fault injection mechanism related to overclocking, underpowering and overheating.

B. Reproducibility and metastability

An analysis of the reproducibility of the fault injection process was also performed. As exemplified in fig. 2 it exists two kind of faulty behaviour related either to a setup time violation or to an early latching. The latter case is characterized by a 100% reproducibility rate for any experiment carried out with the same experimental settings (dataset, power supply, clock period and temperature): each encryption leads to the same fault. The behaviour is slightly different when a setup time violation occur, because it creates a metastable behaviour of the impacted flip-flop. For a given experimental settings the fault may be induced or not. Figure 9 reports this metastable behaviour induced by overclocking for three different datasets. It gives the fault occurrence rate as a function of the clock period. Consider *bit1*: for a clock period (T_{clk}) beyond 8,800ps

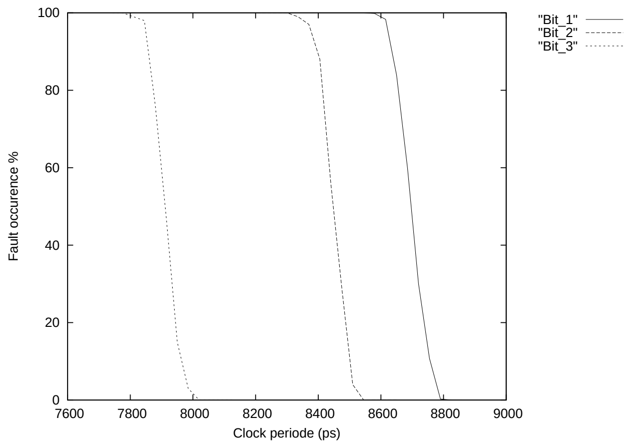

Fig. 9. Metastability: fault occurrence rate versus clock period

no fault is injected, for T_{clk} = 8,700ps the fault occurrence rate is 28% (i.e. 28 encryptions out of 100 will lead to a fault), for T_{clk} below 8,500ps a fault is consistently injected (early latching). The injected faults were always the same (100% reproducibility), however there may be no fault.

The 100% reproducibility rate mentioned above was obtained for single-bits faults. In case of multi-bits faults this rate is lower because of a cumulative effect between bits affected by a metastable behaviour. To be more specific, the reproducibility rate is decreasing as the number of faulted bits is increasing. Because of this phenomenon, it is often considered that the reproducibility of faults injected by timing constraints violation is low. This statement should be mitigate because we have proved that a 100% reproducibility rate may be achieved with a careful choice of the experimental settings.

C. Fault analysis

1) Fault range: : a careful and progressive increase in the stress applied to the test chip has permit to obtain single-bit faults with a success rate slightly beyond 90%. As regards DFA, this fault model is the most difficult to achieve and the most alarming. Part of the 10% remaining faults were multi-bits faults related to the simultaneous violation of two (or more) critical paths. Yet, most of this 10% faults originates from faults induced in different rounds. This phenomenon is illustrated in figure 10. In fact, the fault injected by violation

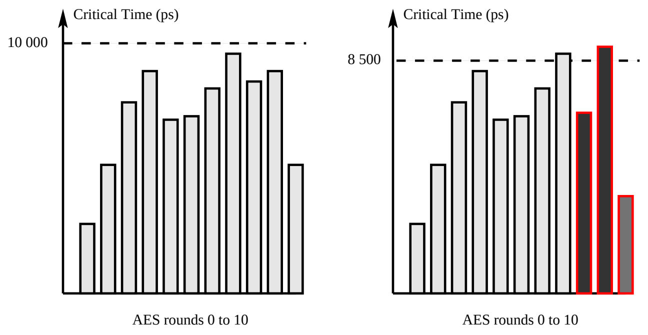

Fig. 10. Path modification after a fault

of the critical path of the seventh round (left diagram) induced a modification of the data handled during the subsequent rounds. Consequently, these rounds' critical times are changed (redrawn on the right diagram). Then, if one of the modified critical time is greater than the clock period (as illustrated) a second fault is injected. This phenomenon also explain the non-equality of the number of faults injected over the AES' rounds as depicted in fig. 5.

2) Fault location: : the experimental results also confirm the data dependence of the fault injection process. Any modification of the data has led to a modification of the injected fault. By doing so, faults were injected in every bytes and in half of the 128 bits of the AES state. Changing the data provides a loose control on the faults' location.

V. Conclusion

In this paper we have provided an experimental proof of the uniqueness of the fault injection mechanism by means of the target's clock period, power supply or temperature alteration. The proof lie in the nature of the injected faults: they were exactly the same for a given dataset irrespectively of the injection means used (overclocking, underpowering or overheating). Besides, we have conducted an in-depth study of these faults properties. It has revealed the ability to induce single-bit faults with a success rate beyond 90% and a reproducibility rate of 100%. The data dependence of the injected faults also allowed to control loosely the faults' location and timing (i.e. the affected round).

We hope these results will contribute to a better understanding of the threats related to fault injection by means of timing constraints violation. We are already testing a first version of a countermeasure based on these findings.

References

[1] O. Kömmerling and M. G. Kuhn, "Design principles for tamper-resistant smartcard processors," in *Proceedings of the USENIX Workshop on Smartcard Technology*, 1999, pp. 9–20.
[2] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *CRYPTO*, 1999, pp. 388–397.
[3] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *CHES*, 2004, pp. 16–29.
[4] J. Balasch, B. Gierlichs, and I. Verbauwhede, "An in-depth and black-box characterization of the effects of clock glitches on 8-bit mcus," in *Fault Diagnosis and Tolerance in Cryptography (FDTC), 2011 Workshop on*, 2011.
[5] J.-M. Dutertre, A.-P. Mirbaha, D. Naccache, A.-L. Ribotta, A. Tria, and T. Vaschalde, "Fault round modification analysis of the advanced encryption standard," in *IEEE Int. Symposium on Hardware-Oriented Security and Trust*, 2012.
[6] S.-M. Yen and M. Joye, "Checking before output may not be enough against fault-based cryptanalysis," *IEEE Transactions on Computers*, vol. 49, no. 9, pp. 967–970, 2000.
[7] D. Boneh, R. DeMillo, and R. Lipton, "On the importance of checking cryptographic protocols for faults," in *EUROCRYPT '97*, ser. Lecture Notes in Computer Science, vol. 1233, 1997, pp. 37–51.
[8] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Advances in Cryptology - CRYPTO '97*, ser. Lecture Notes in Computer Science, vol. 1294, 1997, pp. 513–525.
[9] H. BarEl, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan, "The sorcerer's apprentice guide to fault attacks," in *Special Issue on Cryptography and Security*, 2006, pp. 370–382.
[10] N. Selmane, S. Bhasin, S. Guilley, and J. Danger, "Security evaluation of asics and field programmable gate arrays against setup time violation attacks," *Information Security, IET*, vol. 5, no. 4, pp. 181–190, 2011.
[11] A. Barenghi, G. Bertoni, L. Breveglieri, M. Pellicioli, and G. Pelosi, "Low voltage fault attacks to aes," in *HOST*, 2010, pp. 7–12.
[12] Y. Li, K. Ohta, and K. Sakiyama, "New fault-based side-channel attack using fault sensitivity," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 1, pp. 88–97, 2012.
[13] NIST, "Announcing the advanced encryption standard (aes)," *Federal Information Processing Standards Publication 197*, 2001.
[14] J. Horstmann, H. Eichel, and R. Coates, "Metastability behavior of cmos asic flip-flops in theory and test," *Solid-State Circuits, IEEE Journal of*, vol. 24, no. 1, pp. 146 –157, feb 1989.
[15] M. Agoyan, J. Dutertre, D. Naccache, B. Robisson, and A. Tria, "When clocks fail: On critical paths and clock faults," *Smart Card Research and Advanced Application*, pp. 182–193, 2010.
[16] B. Razavi, *Fundamentals of Microelectronics*. Wiley, 2008.
[17] D. Ha, K. Woo, S. Meninger, T. Xanthopoulos, E. Crain, and D. Ham, "Time-domain cmos temperature sensors with dual delay-locked loops for microprocessor thermal monitoring," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 99, pp. 1–12, 2011.