Power supply glitch induced faults on FPGA: an in-depth analysis of the injection mechanism

Loïc Zussa, Jean-Max Dutertre, Jessy Clédière, Assia Tria

On-Line Testing Symposium (IOLTS), 2013 IEEE 19th International
http://dx.doi.org/10.1109/IOLTS.2013.6604060

Power supply glitch induced faults on FPGA: an in-depth analysis of the injection mechanism

Loïc Zussa* , Jean-Max Dutertre*, Jessy Clédière† and Assia Tria†
*École nationale supérieure des Mines de Saint-Étienne 13120 Gardanne, France
Email: {zussa, dutertre}@emse.fr
†CEA-LETI, 13120 Gardanne, France
Email: {assia.tria, jessy.clediere}@cea.fr

***Abstract*—Secure circuits are prone to a wide range of physical attacks. Among those are fault attacks based on modifying the circuit environment in order to change its behaviour or to induce faults into its computations. There are many common means used to inject such faults: laser shots, electromagnetic pulses, overclocking, chip underpowering, temperature increase, etc. In this paper we study the effect of negative power supply glitches on a FPGA. The obtained faults were compared to faults injected by clock glitches. As a result, both power and clock glitch induced faults were found to be identical. Because clock glitches are related to timing constraint violations, we shall consider that both power and clock glitches share this common fault injection mechanism. We also further studied the properties of this fault injection means.**

I. Introduction

Physical attacks (or *hardware* attacks) target the integrated circuits (ICs) which implement cryptographic algorithms for the purpose of providing security features. Amongst these physical attacks, this work focuses on *fault attacks* (FA), which consists in modifying the circuit environment in order to change its behaviour or to induce faults into its computations. There are many common means used to inject such faults: laser shots, electromagnetic pulses, overclocking, chip underpowering, temperature increase, etc.

There are three main subclasses of fault attacks: code re-routing, safe error (not addressed in this paper) and differential fault analysis. Code re-routing consists in replacing instructions executed by a micro-controller [2] to circumvent its security features (for example a PIN of a smartcard could be broken), or in weakening the strength of an iterative encryption algorithm by changing the number of its rounds [10]. Differential fault analysis (DFA) consists in retrieving the keys by comparing correct and faulted 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 [6]. From that time, many DFA schemes have been proposed to attack various encryption algorithms. Except for DES, most 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 transient voltage supply deprivation (the so-called power glitches). This fault injection mean is known and used since the beginning of FA [3]. However, there are few papers in the scientific bibliography ([17], [5], [14]), which report a deep investigation and understanding of the underlying fault injection mechanisms related to power glitches. Our contributions to that research field are:

- the experimental proof that power glitches create timing constraint violations (as clock glitches and underpowering do),
- a study of the properties and physical limitations of this injection means.

A programmable circuit (FPGA) was chosen as a test vehicle to conduct this study. It implements the advanced encryption standard (AES [15]), which is a secret key encryption algorithm.

This article is organized as follows: a remainder on timing constraints and an explanation of how faults may be injected by their violation are given in section II. A state-of-the-art on clock and power glitches induced faults is given in section III. The experimental set-up and the experiments outline are described in section IV. Then, the experimental results and their analysis are reported in section V. Finally a conclusion is drawn.

II. Timing constraints

In this section, the basics of timing constraints are firstly reminded. Secondly, the two means of inducing timing constraint violations for the purpose of fault injection are reviewed. Then, the experimental proof intended to demonstrate the equivalence of these two fault injection mechanisms is introduced.

A. Timing constraints

Almost all digital ICs 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).

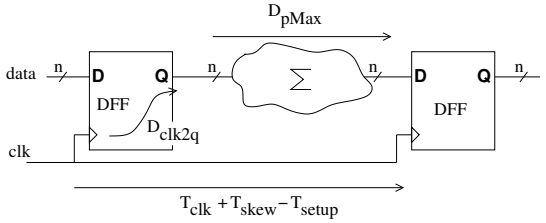

Fig. 1. Internal architecture of digital ICs

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 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 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:

$$T_{clk} > D_{clk2q} + D_{pMax} + T_{setup} - T_{skew} \qquad (1)$$

An illustration, at bit level, of the signal flow is given in figure 2-a for which the timing constraint is fulfilled. Note that the input of the downstream DFF ($D_{downstream}$) undergoes many logic glitches related to the calculations of the combinatorial logic before stabilizing. It exists a time margin (called the slack) between the last signal transition at the input of the downstream register and the setup time.

The violation of this timing constraint is a straightforward means to inject faults into a circuit. Two stages of such violations are depicted in 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 [13]: it may stabilize either on a high or low state regardless of its input's value. 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 timing constraint violations for both cases.

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

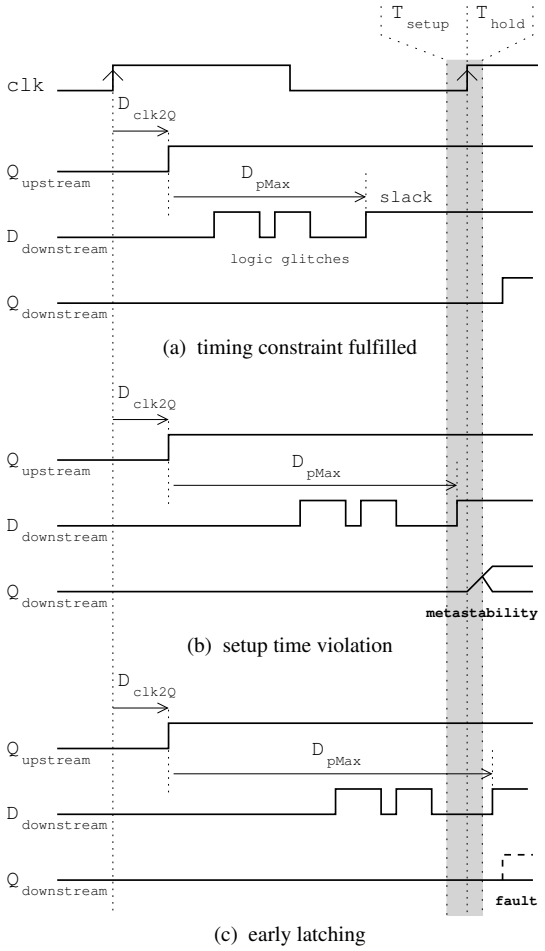

Fig. 2. Timing constraint (a) fulfilled or violated: (b) setup violation, (c) early latching.

B. Overcloking

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

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

Overclocking does not provide any timing control: faults may be induced at any clock cycle of the targeted IC. An enhancement of that technique consists in inducing a timing violation by modifying only one chosen clock period (see III).

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 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.

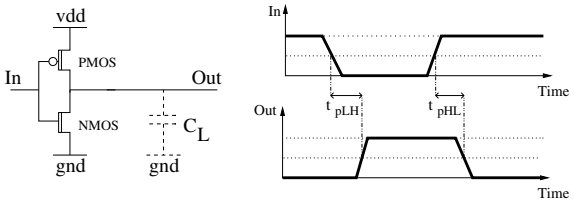

Fig. 3. Inverter: architecture and typical waveforms

Note that t_{pLH} and t_{pHL} may have different values. 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:

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

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.

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 as long as the IC is underpowered. Hence, underpowering is a common means to achieve fault injection by violation of the timing constraints.

D. Several fault injection means, a common mechanism

Therefore, overclocking and underpowering (and also overheating, however not studied in this paper for the sake of brevity) are two suitable means to inject faults into a circuit by violation of its timing constraints [5], [17]. Intuitively, these two means are usually considered to originate in a same mechanism. Underpowering is characterized by a static behaviour (i.e. the chip's supply voltage is set to a constant value outside its nominal range), which make it relatively easy to investigate and understand the corresponding injection mechanism.

Whereas power supply glitches consist in a transient perturbation of the power supply voltage. The assumption that power glitches induce faults by violation of the target's timing constraints is often made [18], [3]. However, no evidence of that assumption is given in the scientific bibliography. Indeed, the fact that power glitch induced faults are due to (and only to) timing constraint violations is questionable because it may involve some dynamic behaviour related to the fast modification of the power supply. Hence, 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 both clock and power supply glitches on 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 equivalence of the injected faults for these two means is the core of that proof as reported in the next sections.

III. State-of-the-art of clock and power glitches

A. Clock glitch induced faults

The use of clock glitches to induce faults into the computations of an IC is well known. It consists in decreasing one clock period until a fault is injected due to timing constraint violations. Recent papers have introduced dedicated platforms on FPGAs [12], [1], [11]. This technique provides the ability to choose precisely the stress applied to the target (i.e. the time decrement of the targeted clock period), and thus the number of the injected faults. It also, by nature, makes it possible to select a given calculation cycle, which is a useful property to meet the requirements of DFA's schemes.

We have designed our own clock glitch fault injection platform based on these previous researches. This design allows us to set the modified clock period with an accuracy of $35ps$.

As reported in [1], [11], [14], clock glitch induced faults are characterized by two main properties:

- faults are data-dependent (i.e. if the processed data change so does the injected faults and the associated critical times),
- the fault injection process may undergo a metastable behaviour (as exemplified in fig. 2-b) when the stress applied to the target is still low. As the stress increases the injection process may become deterministic (see fig. 2-c as an illustration).

These two properties are related to any fault injection means based on timing constraint violation.

B. Power glitch induced faults

Power supply glitches are often used to induce faults into secure ICs. Its use (mainly on micro-controllers) has been extensively reported [3], [8], [18], [4] for glitches consisting in a sudden negative change of the power supply voltage.

However, very few papers have been dedicated to a deep investigation of the underlying fault injection mechanism. The most significant paper [9] studied accurately the effects of a voltage glitch on the DFFs of a CMOS circuit. It showed, on a simulation basis, that power supply glitches cannot induce faults into DFFs. It also confirmed, according electrical simulations, that the faults injected by negative voltage glitches are due to timing constraint violations. The latter are caused by an increase of the combinatorial logic propagation delays.

Other explanations of the fault injection mechanism related to power glitches may be found. The authors of [19] stated that due to voltage glitches *"different sub-circuits might be*

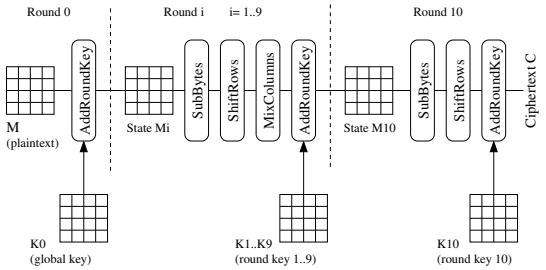

Fig. 4. AES-128 general outline.

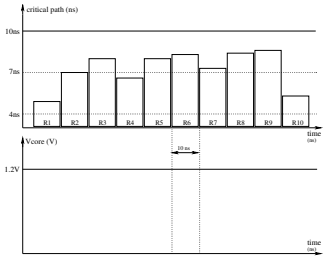

Fig. 5. Most critical paths in every AES rounds in nominal conditions

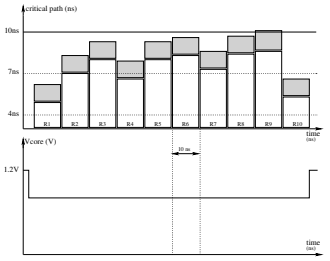

Fig. 6. Most critical paths in every AES rounds when subject to underpowering

powered at different voltages, hence, enabling fault injections". However, they did not provide any evidence of this phenomenon.

At the time being, no experimental proof of the assumption that power glitch induced faults are injected by timing constraint violations has been provided. The main intent of this paper is to provide such an experimental proof.

IV. Experimental set-up

A. AES *test chip, board and voltage pulse generators*

AES is a standard established by the NIST [15] 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 in fig. 4. The test chip (Xilinx Spartan 3A FPGA) embeds a hardware 128-bit version of this algorithm (AES-128). It processes data blocks of 128 bits (usually represented as a 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. In fig. 4, the 4x4 bytes matrices also point out where the registers storing the AES' state are located in our design: just before the SubBytes transformation. The design is shaped in a loop encompassing the four AES transformations and the registers bank used to store the AES' state. Hence, a full encryption is completed in eleven clock periods. The test chip nominal clock period is 100 MHz. In this work, AES is mainly used as 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 still of interest.

The component is mounted on a board with voltage regulators that provide the voltage supplies of its I/O ports and of its internal core logic : 3.3V and 1.2V respectively. Communication interfaces with this device are also provided. Many capacitances are connected between the board's ground and the supply rails of the chip. We have opened the supply rail of the FPGA's voltage core to make it possible to inject a voltage glitch into its logic. We have also de-soldered the capacitance of the voltage core supply rail to improve the efficiency of the fault injection process. An SMA connector has been soldered in place of a capacitance which was close to the core voltage input of the FPGA, in order to diminish the reflection phenomena that will affect the injected voltage pulse.

For the purpose of injecting power supply glitches we used two pulse generators (Agilent 814A and Picosecond 10,300B).

B. Outline of experiments

1) Methods: For a given set of data (plaintext and key) processed by the AES test chip, we used both clock and power supply glitches to gather and study the injected faults. For the sake of simplicity, we focused on the first injected fault obtained for every round of the AES, while increasing the stress applied to the FPGA (i.e. a step by step decrease of the faulted period duration or increase of the amplitude of the negative voltage glitch).

The clock glitch generator allowed to target independently the ten rounds of the AES. We did not succeed in inducing fault during the AES initial round because it has a very short data propagation time. Any clock glitch short enough to violate the corresponding critical time also faults the controlling FSM of the device driving it into a non-recovering fail state. While the data propagation times of the subsequent ten rounds are the longest of the design, we were always able to induce faults in these rounds by means of clock glitches. Besides, the first injected fault reveals the corresponding critical time (as detailed in [14]). Then, for each data-set, we gathered the injected faults while targeting each AES round (except the initial one) and their critical times. Figure 5 reports the critical times obtained for a given data-set at nominal supply voltage (1.2V). They are obviously shorter than the clock period (10ns). As expected, the paths measured for each round were found different (because the data processed during each round were also different).

The utilized pulse generators are able to provide a DC voltage in addition to pulses. We used this feature to power the test chip internal core. Moreover, it allowed us to carry out underpowering attacks as illustrated in fig. 6. As the core voltage is decreased, the critical times of each round are increased (the grey shaded parts in fig. 6). A measure of these critical times was made by using simultaneously clock glitches. In the instance of fig. 6, the critical time of the 9^{th} round goes beyond the clock period. Then, a fault is injected.

Our main intent was to compare faults induced by power supply glitches with those induced by clock glitches. To do so, we used negative power supply glitches as depicted in figures 7 and 8.

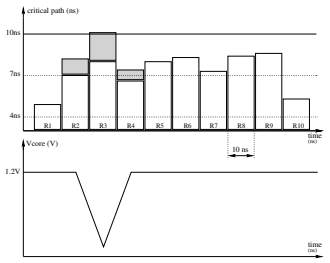

Fig. 7. Most critical paths in every AES round when subject to a power glitch centered on round 3

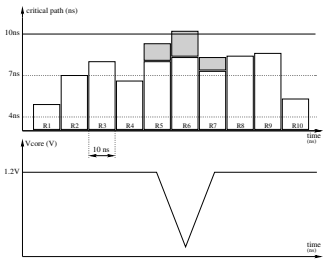

Fig. 8. Most critical paths in every AES round when subject to a power glitch centered on round 6

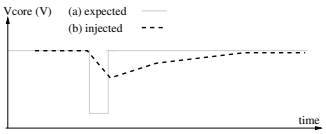

Fig. 9. Expected and altered glitches with only one generator (hypothetical view)

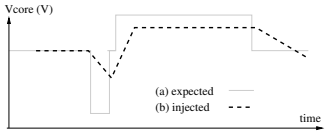

Fig. 10. Expected and altered glitches with two generators (hypothetical view)

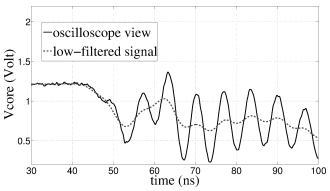

Fig. 11. Expected and actually injected glitch with only one generator

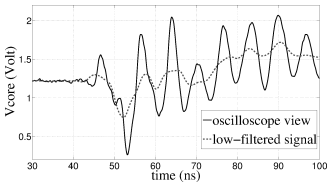

Fig. 12. Expected and actually injected glitch with two generators

2) Library of clock glitch induced faults: A data-set library of 1,000 random {Plaintext, Key} couples was built. For each couple a first fault-free encryption was run, and the obtained correct ciphetext was added to the library. Then, for each AES round of each encryption, a fault injection experiment by means of clock glitches was carried out. The corresponding clock faulted period was reduced progressively by steps of 35ps until a first fault was induced. Next, the faulted ciphertext was processed by reversing the AES encryption (the key and plaintext being known), in order to find out the injected fault and to check its instant of appearance. It allowed us to study the fault nature (i.e. the number of faulted bits and their location). Finally, the injected faults were added to the library.

3) Library of power supply glitch induced faults: The same data-set library of 1,000 random {Plaintext, Key} couples was used to conduct fault injection experiments by means of power supply glitches. These experiments were carried out in a similar way: a progressive increase of the applied stress until a first fault appears. The injection time was varied according to the principle illustrated in fig. 7 and 8, in order to target all the AES rounds. Finally, the injected faults were added to the library.

4) Experimental proof: Then, clock and power glitches induced faults were compared in order to check whether they were identical or not. If identical, it would be an experimental proof that faults induced by clock and power glitches are due to timing constraint violations, i.e. that they originate in the same injection mechanism. The experimental results we have obtained are reported in the next section.

V. Experimental results

A. Clock glitch results

We observed, as expected, that this injection process is data-dependent and also that the metastability phenomenon exemplified in fig. 2-b is observable. Faults were injected successfully at each round of the AES. The obtained faults were single-bit with a rate slightly greater than 90%.

B. Power supply glitch results

1) Experimental issues: The settings and methods reported in this work were found experimentally after many trials and errors. We finally found out that we need to inject a pulse in the 20ns range similar to those depicted in figs. 7 and 8 at a negative amplitude beyond 30V to go through the internal low filtering effect originated in the core voltage supply port: the pulse amplitude inside the component was highly attenuated. However, we were still unable to match the timing accuracy we had with clock glitches. After reversing the encryption of the faulty ciphertexts, we found that faults were often injected in two or three subsequent rounds. We drew the hypothesis that the injected pulse was distorted and its effect was extended over a larger period of time as illustrated on figure 9.

The remedy to this lack of accuracy was to use the second pulse generator (Picosecond 10,300B) to boost the rising part of the actually injected glitch as shown in figure 10 (As a result, the core power supply is also increased during the following rounds. However, it had no adverse effect on the computations of the IC). The correcting pulse was positive, with a 20V amplitude and a 100ns duration. Fig. 11 and 12 display the core supply rail voltage captured by an oscilloscope during pulsed injection performed with both techniques. Many oscillations appear. We also drew a low-filtered view of the voltage to figure how it may be inside the component (after passing through the supply port). This confirms the assumptions of fig. 9 and 10.

However, because critical times are data-dependent, a different setting of the faulting pulse's amplitude was to be found for each round of each data-set. Consequently, a different setting of the correcting pulse was needed. This led to very long tuning steps. Thus, we simultaneously used a constant modification of the power supply and a voltage glitch to ease the process. The glitches settings were kept constant (for both faulting and correcting pulses). The core power supply value and the timing parameters were the only varying parameters of our experiments. This technique is illustrated in fig. 13 where the low grey shaded squares represent the critical time increase due to underpowering and the deep grey shaded squares represent the critical time increase due to the voltage pulse. In this instance, round 6 was faulted.

2) Results: Due to the long time needed to carry out power glitch experiments, we report here the injection results from 140 different rounds. The single-bit fault injection success rate was also slightly greater than 90%. As expected, the data-

dependent nature of the injection process and also the metastability phenomenon, exemplified in fig. 2-b, were observed. The experimental results are :

- 70% of the injected faults were identical to those obtained by using clock glitch injection,
- 10% of the injected faults were induced by violating the timing constraints of the second most critical path of the round (as further clock glitches experiments revealed),
- 20% of the injected faults were injected in neighbouring rounds of the targeted one.

The latter behaviour is exemplified in figure 14: a round with a critical path shorter than those of its neighbouring rounds may be *unreachable*.

C. Results analysis

For both studied fault injection means (clock and power glitches), we have observed the two main distinguishing features of timing constraint violations: data-dependency and the occurrence of a metastability phenomenon. Moreover, 70% of the injected faults were found to be identical. The remaining 10% and 20% were explained respectively by the violation of the 2^{nd} or 3^{rd} most critical path and by fault injection in neighbouring rounds. We believe that this is an effective experimental proof of the uniqueness of the injection mechanism. Even if the power glitch injection setup is far less accurate than the clock glitches one in some specific cases it could be easier to inject fault on the power supply line than on the clock line.

We did not report in this paper our experiments of fault injection with positive power supply glitches. The reason is that we were unable to induce faults on our FPGA board set-up by using our pulse generators, despite their wide range of available settings in amplitude and time. Further experiments have to be carried out before being able to draw a conclusion.

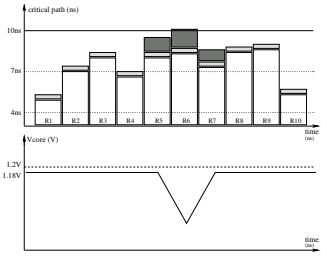

Fig. 13. Critical paths of every AES rounds affected by a transient pulse and underpowering

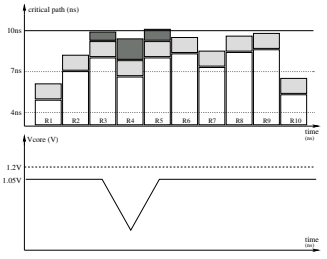

Fig. 14. Illustration of the *unreachable* round phenomenon (the 5^{th} round is faulted in place of the 4^{th})

VI. Conclusion

In this paper we have provided an experimental proof of the equivalence of the fault injection mechanism by means of clock and power supply glitches. The proof lies in the nature of the injected faults: they were the same or very similar for a given data-set irrespectively of the injection means used. Besides, we have conducted an in-depth study of the properties of these faults. It has revealed the ability to induce single-bit faults with a success rate beyond 90%. Power supply glitches also make it possible to fault almost every encryption round of our test chip (near 80% success rate). As, 10% of the injected faults were induced by violation of the 2^{nd} or 3^{rd} most critical path of the targeted ound, it suggests that there may also be a spatial effect associated with power glitches. This assumption would be studying in further researsh work.

References

[1] M. Agoyan, J.M. Dutertre, D. Naccache, B. Robisson, and A. Tria. When clocks fail: On critical paths and clock faults. Smart Card Research and Advanced Application, pages 182–193, 2010.

[2] 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, pages 105 –114, sept. 2011.

[3] 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, pages 370–382, 2006.

[4] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache. Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures. Proceedings of the IEEE, 100(11):3056–3076, 2012.

[5] Alessandro Barenghi, Guido Bertoni, Luca Breveglieri, Mauro Pellicioli, and Gerardo Pelosi. Low voltage fault attacks to aes. In HOST, pages 7–12, 2010.

[6] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In Advances in Cryptology - CRYPTO '97, volume 1294 of Lecture Notes in Computer Science, pages 513–525, 1997.

[7] D. Boneh, R.A. DeMillo, and R.J. Lipton. On the importance of checking cryptographic protocols for faults. In Advances in Cryptology - EUROCRYPT '97, volume 1233 of Lecture Notes in Computer Science, pages 37–51, 1997.

[8] Hamid Choukri and Michael Tunstall. Round reduction using faults. Fault Diagnosis an Tolerance in Cryptography – FDTC 2005, pages 13–24, 2005.

[9] A. Djellid-Ouar, G. Cathebras, and F. Bancel. Supply voltage glitches effects on cmos circuits. In Design and Test of Integrated Systems in Nanoscale Technology, 2006. DTIS 2006. International Conference on, pages 257 –261, sept. 2006.

[10] 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.

[11] S. Endo, T. Sugawara, N. Homma, T. Aoki, and A. Satoh. An on-chip glitchy-clock generator for testing fault injection attacks. J. Cryptographic Engineering, 1(4):265–270, 2011.

[12] T. Fukunaga and J. Takahashi. Practical fault attack on a cryptographic lsi with iso/iec 18033-3 block ciphers. In Proceedings of the 2009 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC '09, pages 84–92, 2009.

[13] J.U. Horstmann, H.W. Eichel, and R.L. Coates. Metastability behavior of cmos asic flip-flops in theory and test. Solid-State Circuits, IEEE Journal of, 24(1):146 –157, feb 1989.

[14] Y. Li, K. Ohta, and K. Sakiyama. New fault-based side-channel attack using fault sensitivity. IEEE Transactions on Information Forensics and Security, 7(1):88–97, 2012.

[15] NIST. Announcing the advanced encryption standard (aes). Federal Information Processing Standards Publication 197, 2001.

[16] Behzad Razavi. Fundamentals of Microelectronics. Wiley, 2008.

[17] N. Selmane, S. Bhasin, S. Guilley, and J.L. Danger. Security evaluation of application-specific integrated circuits and field programmable gate arrays against setup time violation attacks. Information Security, IET, 5(4):181–190, 2011.

[18] Peter Tummeltshammer and Andreas Steininger. On the role of the power supply as an entry for common cause faults. 13th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems, pages 152–157, 2009.

[19] Asier Goikoetxea Yanci, Stephen Pickles, and Tughrul Arslan. Characterization of a voltage glitch attack detector for secure devices. 2009 Symposium on Bio-inspired Learning and Intelligent Systems for Security, pages 91–96, 2009.