Yet Another Fault Injection Technique : by Forward Body Biasing Injection

K. TOBICH[1,2], P. MAURINE[1], P.-Y. LIARDET[2], and T. ORDAS[2]

[1] LIRMM, CNRS, Université Montpellier II
161 rue Ada, 34392 Montpellier CEDEX 5, France
Philippe.Maurine@lirmm.fr

[2] STMicroelectronics, Smartcard ICs
Z.I. Rousset, 13106 Rousset CEDEX, France
{Karim.Tobich, Pierre-Yvan.Liardet, Thomas.Ordas}@st.com

Abstract Nowadays fault attacks are widely deployed against secure devices by hardware evaluation centers. While the least expensive fault injection techniques, like clock or voltage glitches, are well taken into account in secure devices by dedicated hardware, more advanced techniques, such as light based attacks, require huge investments. To protect devices against these types of attacks requires complex detection or correction policies. However more and more solutions against light attacks are now available. In this context, this paper presents a new way to induce faults in computation at a moderate cost (few thousand euros) that may defeat already in place hardware counter-measures. To demonstrate its effectiveness we applied this technique on an ASIC component equipped with hardware counter-measures. For this demonstration, fault exploitation is operated using the classic Bellcore attack [6] applied on a modular exponentiation supported by a modular arithmetic co-processor.

Keywords: Fault Attacks, Forward Body Biasing Injection, Electromagnetic Attacks, RSA, Chinese Remainder Theorem

1 Introduction

Fault injection techniques range from old-fashioned glitches [1] to more recent electromagnetic (EM) harmonic or pulse injections [15], and of course include light attacks [16]. As reported in [1], perturbations were first obtained by applying glitches on the supply or clock pads. Today, it is easy to try "glitch" attacks by purchasing an "unlooper" for few tens of euros from the web. Fortunately, counter-measures based on glitch detectors are commonly embedded into today's circuits which makes buying an "unlooper" a pure waste of money. As a matter of fact, because this attack is still very popular in Pay TV, chip manufacturers take particular care and have expensive dedicated materials to verify

that each new device addressing the secure microcontroller market is either not perturbed, or is able to detect all possible glitches applied on pads.

Nowadays, lasers remain the main tools used to inject faults into modern secure microcontrollers. However, the price of this equipment and the expertise required to mount an attack are high and represent a big investment. Additionally, when the microcontroller is a smartcard, it is often necessary to prepare the integrated circuit in order to make fault injection possible; this delicate preparation requires dedicated equipment and know how. Counter-measures such as light sensors, available in both academic and industrial communities [9] are already deployed in most secure devices, thus rendering the practice of laser-based attacks more and more difficult [5].

As an alternative to laser, EM waves have been recently proposed to inject faults through packages and front sides [15]. Indeed, it has been recently proven in [4] that EM harmonic waves can be used to manipulate the bias of a True Random Number Generator (TRNG). This result, obtained on ring oscillator based TRNGs embedded into an FPGA, shows the high potential of this technique. More recently in [8], the exploitation of faults obtained by EM pulses on a non-secure microcontroller was successful to obtain the secret key of a software AES implementation applying a Piret-Quisquater attack [14]. However, EM fault injection techniques are a new topic and their limitations and capabilities are not well known on secure microcontrollers.

Within this context, and as suggested in [1], our contribution is to propose a new and low cost technique that locally injects faults into an Integrated Circuit (IC). We describe the bench, the necessary setup to obtain exploitable faults, and provide our first experimental results on **a secure microcontroller whose countermeasures were disabled**. With this paper we would like to raise the interest of the community on possible effects of local glitches that could represent a major threat due to its moderate price.

In order to provide evidence that induced faults by this new injection technique leads to exploitable results, we have based our experiment on the well known Bellcore attack [6] operated on a modular exponentiation.

The paper is organized as follows: Section 2 presents the principle of the injection technique known as called the Forward Body Biasing Injection (FBBI) and the associated injection platform. Section 3 gives an overview of the Bellcore attack and presents the practical settings of the FBBI platform that lead to exploitable faults. In Section 4, experimental results are analyzed to demonstrate the effectiveness of this new fault injection technique to recover the key. Finally, a conclusion is drawn and perspectives are given.

2 Forward Body Biasing Injection (FBBI)

Reverse and Forward Body Biasing techniques are well known and commonly used in low power design solutions to mitigate process variations and/or control the static power dissipation of ICs. They consist in applying to the substrate, a forward or, reverse static bias of a few hundreds of milliVolts (mV) to modulate threshold voltages of MOS transistors. The next paragraphs explain how a transient Forward Body Bias of a few tens of Volts can be used by an adversary to inject faults into IC.

2.1 FBBI principle and Integrated IC Modeling

IC designers working with advanced technologies are aware of the potential causes of permanent or transient faults. Among them, one may identify dynamic voltage drops, ground bounces, temperature shifts, aging effects, electromigration process, conducted noises, EM perturbations, etc. In recent technologies, voltage drops are one of the main causes of temporary failure. Within this context, Figure 1 gives a first order model of the power / ground network of an IC to illustrate some of these causes and the corresponding techniques for intentional fault injections. As can be seen, the substrate appears as a forgotten backdoor enabling the injection of faults into any IC. Additionally, this backdoor should allow the injection of local faults, due to the spatial distribution of the resistances and capacitances of the supply network including the substrate. For the sake of simplicity, Figure 1 only provides a discrete model of this network. In this figure:

- *RGE* and *RGI* are respectively the external and internal resistors of the ground network,
- *RVddE* and *RVddI* stands respectively for the external and internal resistors of the supply network,
- *RSubInt* is the substrate resistor,
- *RGndSub* models the resistor of the contacts between the substrate and the ground network; contacts designed to force the substrate voltage to the ground, in normal operating conditions.
- and finally *CGndSub* (resp. *CVddSub*) is the capacitance between the substrate and the ground (resp. between the substrate and the supply).

The analysis of the connections between the supply network and the substrate indicates the possibility of inducing an intentional and local voltage drop by applying a positive voltage pulse. This pulse may also lead to a temporary cut-off of the supply. We confirmed this idea by electrical simulations.

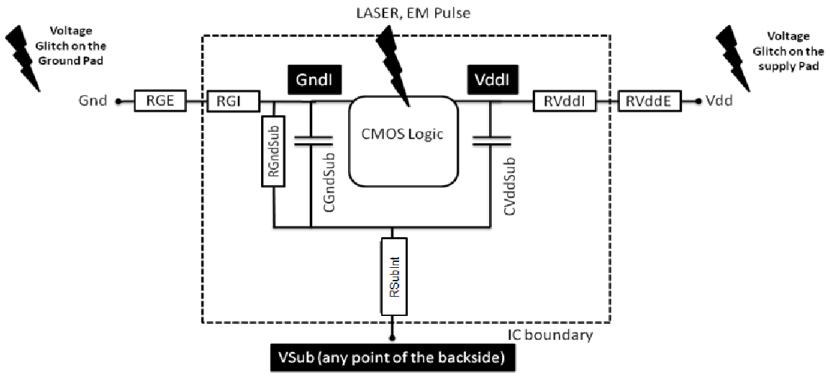

Figure1. First order IC of the Power/ Ground Network of an IC

Figure 2 gives the typical evolutions of the internal *VddI* supply and ground *GndI* voltages, under a forward body bias of several volts. On this Figure, $(VddI - GndI)$ is the swing seen by the logic gates that becomes null and even negative. In the following sections we analyse this surprising electrical behavior.

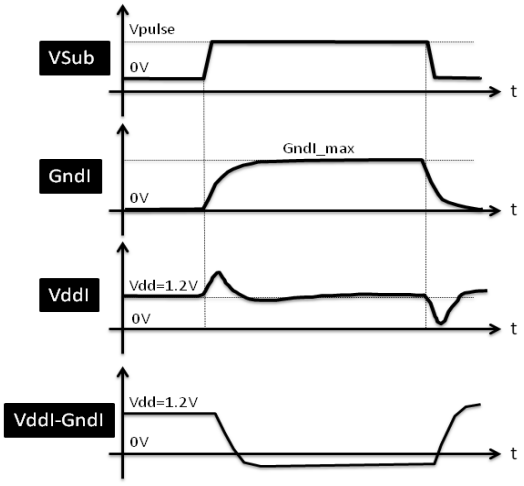

Figure2. Typical Simulated Waveforms of VSub, GndI, VddI and (VddI-GndI)

In Figure 1, *RGE, RGI, RGndSub* and *RSubInt* constitute a voltage divider between the ground and the substrate. Therefore, a forward pulse (of *Vpulse*

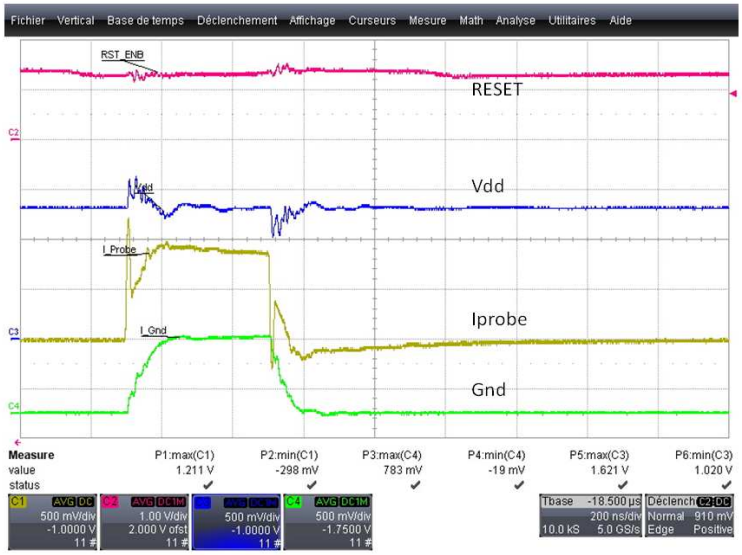

Figure3. Screenshot of VSub, Gnd, Vdd obtained on a 90 nm testchip

Volts) applied on any point of the substrate will load the capacitor, *CGndSub*, via *RGndSub* and *RSubInt*. At the end of this loading, the internal ground voltage, *GndI*, will reach the positive value:

$$GndImax = \frac{RGE + RGI}{RGE + RGI + RGndSub + RSubInt} \times Vpulse \quad (1)$$

At the same time, due to the coupling capacitance *CVddSub*, an overshoot and an undershoot of the internal supply voltage, *VddI*, occur during the rising and falling edges of the voltage pulse applied on *VSub*.
Thus, with an adequate choice of the parameters *Vpulse*, *RGE* and optionally *RSubint* (by reducing the thickness of the substrate), it is possible to shut down, locally and temporarily, the supply voltage. Moreover, it should be noted that transient faults may be induced in some logical gates.
To verify the correctness of this reasoning, an experimental platform was developed. This platform is described in the next section. Experimental results obtained on a 90nm testchip have confirmed the validity of simulated results. As an example, Figure 3 gives the experimental waveforms obtained with a 90 nm testchip. These waveforms are to be compared with the simulated waveforms of Figure 2.

2.2 Injection Platform

Figure 4 shows an architectural view of the FBBI platform. It is simply based on a medium voltage pulse generator (100V, 2A max) and includes a simple probe made of a thin tungsten rod (diameter of 20 μm). It also requires standard control elements such as a PC, an oscilloscope, and motorized stages to adequately establish, at the desired position, the contact between the probe and the substrate. Additionally, all necessary software has been developed in order

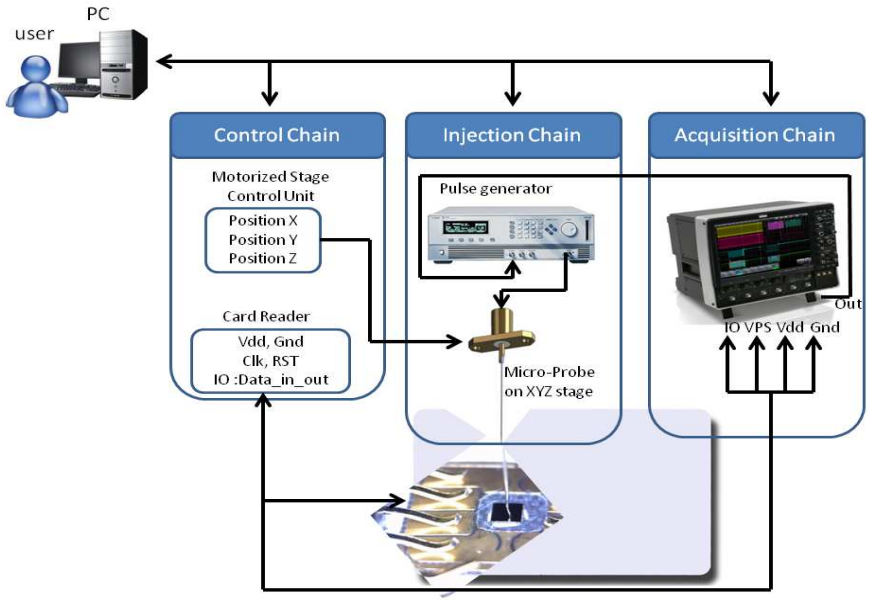

Figure4. Forward Body Bias Injection Platform

to apply different parameters and steps on the integrated circuit surface to map (chip scan) fault effects.

3 Exploitation of Faults

Since the seminal paper [6], many papers are dedicated to fault attacks. A majority of them presents more or less complex cryptanalysis exploiting faults in several fault models. The beauty of the fault model described in [6] (recalled in § 3.1) resides in its simplicity and in the fact that the fault model is very comfortable for the attacker. Indeed, the attacker's task is to induce an incorrect computation within a time frame that represents almost all the activity of an

RSA-CRT. More precisely he or she has to induce a fault during one of the two exponentiations and we will see later in § 4.1 that there are several vectors to obtain a fault.

3.1 Attack Description

Let n be the product of two large prime integers, p and q, each $n/2$ bits long. To sign a message $M \in \mathbb{Z}_n$ using RSA, one needs to compute $S \equiv M^d \mod n$, with d the private exponent satisfying $e.d \equiv 1 \mod ((p-1).(q-1))$ where e is the associated public exponent.

Obviously, the factorization of n is necessary to derive d from public quantities e and n, by inducing a fault while computing a given implementation, we will see that the factorization of n is straight forward.

In order to reduce computation time, the Chinese Reminder Theorem (CRT) [13] is usually used and an RSA signature S is computed as [7]

$$S = q.\alpha_q.S_p + p.\alpha_p.S_q \tag{2}$$

where

$$\begin{cases} S_p \stackrel{\text{def}}{\equiv} S \mod p \text{ and } \mathrm{S_p} \equiv M^{d_p} \mod p \\ S_q \stackrel{\text{def}}{\equiv} S \mod q \text{ and } \mathrm{S_q} \equiv M^{d_q} \mod q \end{cases}$$

with $d_p \stackrel{\text{def}}{\equiv} d \mod (p-1)$ and $d_q \stackrel{\text{def}}{\equiv} d \mod (q-1)$ and where α_p and α_q are defined such that

$$\begin{cases} q.\alpha_q \stackrel{\text{def}}{\equiv} 1 \mod p \\ p.\alpha_p \stackrel{\text{def}}{\equiv} 1 \mod q \end{cases}$$

Classically, equation 2 is optimized for implementation by

$$S \equiv [([(S_q - S_p) \mod q].\alpha_p).p + S_p] \mod n \tag{3}$$

but for the sake of simplicity, let us consider that an attacker can induce a fault during the computation of S_p to obtain $\tilde{S_p}$. As a result, if the component is able to continue computation with this error, and even if computation is done with 3, the outcome will be

$$\tilde{S} = q.\alpha_q.\tilde{S_p} + p.\alpha_p.S_q \tag{4}$$

and by subtracting the result of equation 2 from equation 4 is comes

$$S - \widetilde{S} \equiv q.\alpha_q.(S_p - \tilde{S_p}) \mod n.$$

Under the weak assumption that p does not divide $S_p - \tilde{S}_p$, computing the greatest common divisor of n and $S_p - \tilde{S}_p$ the attacker gets

$$\gcd(S - \tilde{S}, n) = \gcd(q.\alpha_q.(S_p - \tilde{S}_p) \mod n, n) = q.$$

In a similar way, faults induced while computing $\tilde{S}_q$ will lead to the discovery of p. At this point once a fault is injected during either S_p or S_q, it is easy to factorize n and then compute the private exponent d from the public exponent e. In practice, a signing device never computes twice the same signature, so that S is often unknown by the attacker but, in many cases, the message M is available and as shown in [12,10], with the same assumption on fault one can easily obtain q for instance if fault is induced while computing S_p with:

$$\gcd(M - \tilde{S}^e \mod n, n) = q$$

where e is the public exponent. To summarize, with a known message and a faulty signature obtained by perturbation of one of the two RSA-CRT modular exponentiations, we can derive the private key of the device computing the signature.

4 Experimental Results

This section describes the experiment's procedures used to apply the Bellcore Attack on a RSA-CRT running on a microcontroller. It also provides a detailed analysis of the obtained results.

4.1 Fault Injection Procedure

To apply the attack, it was necessary to synchronize our FBBI platform with the device operations in order to inject a fault during S_p or S_q computation. Figure 5 has been obtained with a simple oscilloscope and shows the communications (yellow trace) with the IC and its power consumption (green trace). One can clearly observe, between two I/O activities corresponding to the end of the command transmission and the start of the chip answer, that the power consumption shows two big patterns corresponding to the two modular exponentiations. With this observation it was easy to setup a trigger to inject the fault during a modular exponentiation. The triggering process defined, the main difficulty was to provide the correct parameters to the pulse generator to induce the required perturbation. More precisely, we have to determine the pulse width, its amplitude, and the delay separating the triggering of the generator using the scope from the effective pulse release. To do this, we performed the following steps:

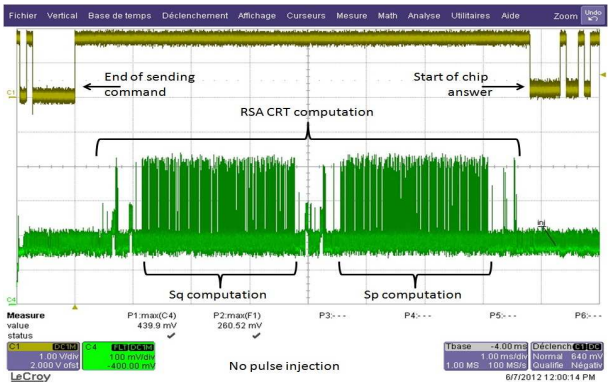

Figure5. Monitored signals to inject the pulse during S_p (in yellow : the communications with the IC, in green the power consumption)

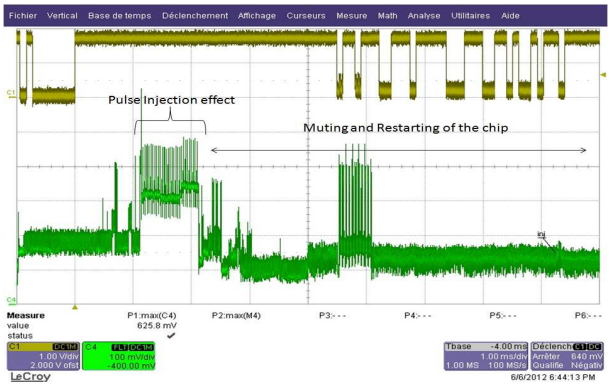

Figure6. Injection causes a reset of the device.

1. Set the pulse width so that to observe, by direct measure of the Gnd signal with the scope, a stabilization of the ground voltage to its final value (*GndImax* in Figure 2).
2. Set the pulse amplitude by observing the effect of several injections. More precisely, we gradually increased the amplitude until faulty answers (including mute ones) were obtained. During these two last steps the probe was placed at the center of substrate.
3. Use motorized stages in x, y to fine-tune the location of injections. In fact, a cartography was launched and the results of the injection at different coordinates were stored in the PC.
4. fine-tune the delay, δ, between the scope trigger and the effective pulse release by the generator, so that we can target different moments of S_p com-

putation. For each delay value, some FBBI were performed (typically 10) and their results recorded for further analysis.

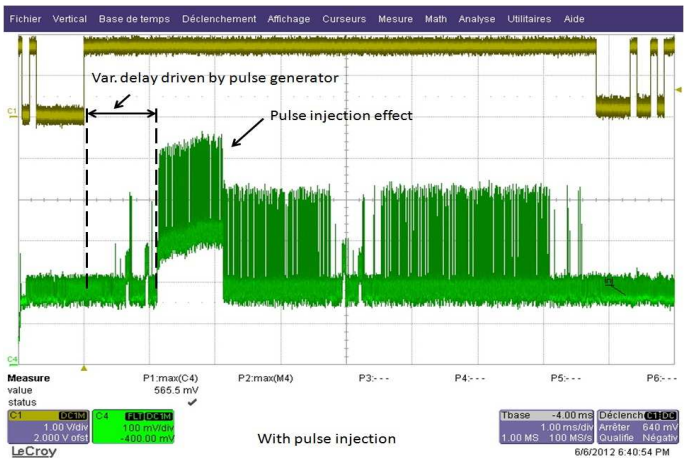

Figure7. Injection Effect

Before launching the above procedure and obtaining a fault cartography, preliminary tests were done to decide which data had to be recorded.

More precisely we looked at the power consumption. We observed mainly three effects. The first one is similar to the one shown in Figure 5 and corresponds to a case were injection has no effect on the secure circuit. On the second one, shown on Figure 6, the FBBI has caused a reset of the device and no exploitable results were obtained. Finally in the third one, Figure 7, the device has been strongly affected but the computation has been completed to the end. In this last case the injection does not systematically lead to an exploitable result. After these preliminary tests, the injection procedure was automated such as to provide a log file to facilitate the analysis of the results.
The vectors that can be perturbed by our injection are:

- CPU execution (that chain co-processor operations),
- RAM access (by CPU or directly by the co-processor),
- Co-processor execution.

In fact, we collected various data. Among them, we of course collected the (x, y) coordinates so that to draw a fault cartography in order to analyze the correlation between fault locations and the device floorplan. We also collected the delay, δ, to analyze the eventual effect of a synchrony between the injection and the clock.

Finally, we of course collected the correct and faulty results of the RSA-CRT computation as well as flags of security registers.

4.2 Analysis

Our first task was to extract all the results provided by the secure microcontroller under injection in order to identify the types of faults that were injected. The table lists the different types of answers obtained from the IC under FBBI during a cartography.

Table1. Fault types

Log information	%
Numbers of Injections: 19830	100
Correct answers	66.04
No answers (Mutes)	6.10
Faulty but unexploited answers	27.64
Faulty and exploited answers	0.22

As can be seen in Table 1, most of the injections did not produce any fault and 6.10% produced a complete stop of the circuit. The remaining 27.64% produced erroneous answers after a complete execution of the instructions by the IC. However, only 0.22% of the obtained faults were exploitable. From this first analysis, we concluded that the proposed injection technique was effective and probably local.

To definitively determine if the FBBI technique produces local faults or global ones, we finally drew four cartographies. The scanned area was $1900\mu m$ $\times$ $1900\mu m$ and the probe displacement step was $100\mu m$. Figure 8 shows the results. The first, second, third, fourth cartographies show where exploitable faults were obtained for a release of the pulse at Time 1 = $t_{trigger} + 0ns$, Time 2 = $t_{trigger} + 50ns$, Time 3 = $t_{trigger} + 100ns$, and Time 4 = $t_{trigger} + 200ns$, respectively. Note that the pulse width was fixed to have a duration of 4 clock cycles.

On these cartographies we may first observe that faults appear at different coordinates depending on the time at which the fault is injected. As a second observation, one may notice that for a given time of injection, faults appears at several positions.

These two observations suggest that the pulse propagates inside the chip. This propagation necessarily depends on the physical structure of the IC and more precisely on the positions of various contacts between the substrate and the ground signals, on the resistivity and the thickness of the substrate, and on

the way the ground signal is routed. Thus, we must conclude that FBBI may be local by some aspects but can produce effects at different positions in the circuit according its physical structure and the time at which the pulse is released. In others words, the FBBI characteristic strongly depends on the characteristics and the way the circuit is designed. Thus, even if further work is necessary to understand how the pulse propagates through the substrate and across the ground network, we may conclude that FBBI is an enhanced (more local) way to perform a fault injection glitch attack on the GND pad.

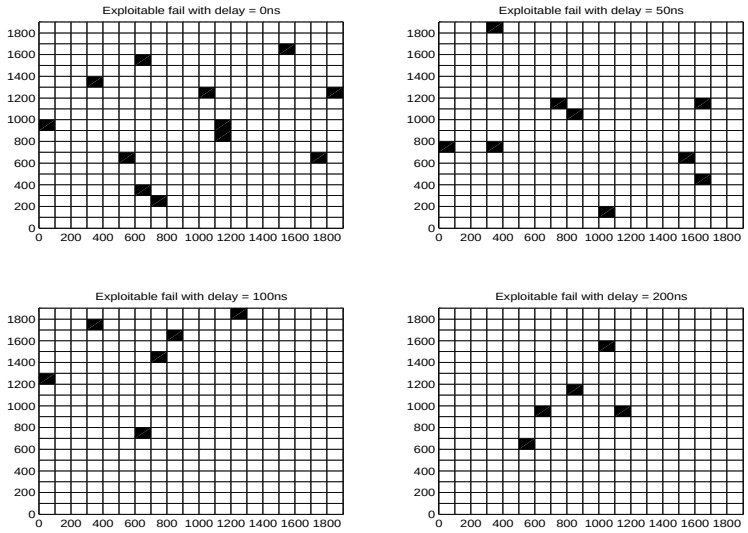

Figure8. Timing Sweep Effect

We may notice that security mechanisms were disabled in our experimental analysis. The principal effect of these mechanisms is to detect intrusions and reset the IC if an intrusion is confirmed. This will induce the erasing of registers and the loss of data.

In order to evaluate FBBI, the reset security has been deactivated.

During our evaluation, security flags were stored in PC. The analysis of these flags shows that detectors have detected FBBI attacks.

One of these flags corresponds to Glitch detector on VPS. For each Exploitable faults in our experimental results (see table 1) , Glitch detector has always detect the attack.

4.3 Assessment of the FBBI complexity to set up

When a new means attack is introduced, it is important to evaluate its threat potential. This is usually done according to the common criteria.

Let us thus start by analyzing the time we spent setting up our FBBI bench and develop all the related software. Because the equipment we used is commonly available in many electronic laboratories, and also because we benefited from the experience obtained during the development of a laser benches, the FBBI platform was mounted in only a few days. Additionally, we spent no more than three weeks identifying the ranges of all parameters to be applied to the developed FBBI platform to produce the expected effects.

Moreover, the attack was performed using common equipment; and it isn't excluded to continue investigation with more specialized equipment, in particular a more powerful and lower jitter pulse generator that can be fully driven by software. In our opinion, such equipment is the key to refine the fault injection parameters and obtain perturbations that are not detected by general purpose glitch detectors. Because the FBBI perturbations were detected by hardware countermeasures embedded in the targeted microcontroller, we must conclude that no exploitable fault will be obtained in a real application. However, because we didn't explore the whole range of possible parameter settings due to the lack of an advanced pulse generator, we must be cautious. Indeed a single and undetected fault targeting an RSA-CRT computation is easy to turn into a successful attack with the computation of a simple greatest common divisor as shown in Section 3.

Another point that increases the level of threat of the FBBI technique is that the required knowledge of the component is very reduced. We have only identified the RSA commands and its SPA signature to set-up the attack. We do not need any information about the software. Additionally, the physical characteristics of the IC were not considered during the set-up of the attack, because their effect is compensated by the appropriate setting of the FBBI parameters as described in the previous paragraphs.

To evaluate the FBBI technique, we have to know if this technique implies an access to the back side of the component and if several samples were destroyed during the set-up of the attack. In our case, two ICs, out of a set of ten, were opened without destroying their functionality. These two circuits were used to evaluate the FBBI technique.

Thus an adversary will need few samples in order to successfully prepare some of them. Additionally, if we assume that for the targeted component, the location at which the probe must be placed to produce undetected faults is known, the adversary will have to spend only a few hours to perform the attack.

5 Conclusion

We have introduced a new means for injecting transient faults into an IC. This technique is known as Forward Body Biaising Injection. We have also provided a first experimental evaluation of the threat through the application of the "Bellcore Attack" on an RSA-CRT signature computation implemented on a microcontroller.

The evaluation of the FBBI potential is far from complete and various work directions have been identified. Among them, the understanding of the way the pulse injected in the substrate propagates in the circuit is an important task. It will allow a fair and complete evaluation of the threat raised by this new fault injection technique. Nevertheless and even if further work is necessary, we may now conclude that the FBBI is at least an enhanced way, i.e. in a more local manner, to perform a fault injection or glitch attack on the ground pad.

References

1. R.J. Anderson and M.G. Kuhn. Tamper Resistance - A Cautionary Note. In The Second USENIX Workshop on Electronic Commerce Proceedings. November, 1996, pp. 1-11.
2. R.J. Anderson and M.G. Kuhn. Low Cost Attack on Tamper Resistant Devices. In M. Lomas, et al. (eds.) Security Protocols, 5yh International Workshop. Springer-Verlag, volume 1361 of LNCS, 1997, pp. 125-136.
3. R.J Anderson and S. Skorobogatov. Optical fault induction attack. In Cryptographic Hardware and Embedded Systems volume 2523 of LNCS, 2002, pp. 2-12.
4. P. Bayon, L. Bossuet, A. Aubert, V. Fischer, F. Poucheret, B. Robisson, P. Maurine. Contactless Electromagnetic Active Attack on Ring Oscillator Based True Random Number Generator COSADE 2012, pp. 151-166.
5. R. Bekkeers and H. König. Fault Injection, a Fast Moving Target in Evaluations FDTC 2011.
6. D. Boneh, R. A. DeMillo, and R. J. Lipton. On the importance of checking cryptographic protocols for faults. In W. Fumy, editor, Advances in Cryptology Eurocrypt'97 : International Conference on the Theory and Application of Cryptographic Techniques, volume 1233
7. H. Cohen. A Course in Computational Algebraic Number Theory Springer-Verlag, volume 138 of GTM 1993, pp.19-21.
8. A. Dehbaoui, J.M. Dutertre, B. Robisson, P. Orsatelli, P. Maurine and A. Tria. Injection of transient faults using electromagnetic pulses-Practical results on a cryptographic system. IACR Cryptology ePrint Archive, 2012, 123.
9. O. Derouet. Secure Smartcard Design against Laser Fault Injection in Proc. FDTC, 2007.
10. M. Joye, Arjen K. Lenstra, and J.J. Quisquater. Chinese remaindering based cryptosystems in the presence of faults. Journal of Cryptology: the journal of the International Association for Cryptologic Research, 12(4) 1999, pp. 241-245.
11. Olivier Kömmerling, Marcus G. Kuhn. Design Principles for Tamper-Resistant samartcard processors In procedings of the USENIX Wokshop on Smartcard Technology WOST'99, 1999
12. A.K. Lenstra. Memo on RSA Signature Generation in the Presence of Faults. Manuscript, 1996. Available from the author at arjen.lenstra@citicorp.com

13. A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone. Handbook of applied cryptography. CRC Press, 2000 N.W. Corporate Blvd., Boca Raton, FL 33431-9868, USA, 1997.
14. G. Piret and J.J. Quisquater. A Differential Fault Attack Technique against SPN Structures, with application to AES and Khazard. In Cryptographic Hardware and Embedded Systems CHES 03, volume 2779 of LNCS, 2003, pp. 77-88.
15. F. Poucheret, K. Tobich, M. Lisart, L. Chusseau, B. Robisson, P. Maurine. Local and Direct EM Injection of Power Into CMOS Integrated Circuits FDTC 2011, pp. 100-104.
16. E. Trichina and R. Korkikyan. Multi Fault Laser Attacks on Protected CRT-RSA FDTC, 2010, pp.75-86.