A methodology to localise EMFI areas on Microcontrollers

Maxime Madau

▶ **To cite this version:**

Maxime Madau. A methodology to localise EMFI areas on Microcontrollers. Micro and nanotechnologies/Microelectronics. Université Montpellier, 2019. English. NNT : 2019MONTS045 . tel-02478873

HAL Id: tel-02478873
https://theses.hal.science/tel-02478873

Submitted on 14 Feb 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE POUR OBTENIR LE GRADE DE DOCTEUR
DE L'UNIVERSITE DE MONTPELLIER

En SYAM - Systèmes Automatiques et Micro-Électroniques

École doctorale : Information, Structures, Systèmes

Unité de recherche LIRMM
Laboratoire d'Informatique, de Robotique et de Micro-électronique de Montpellier

A methodology to localise EMFI areas on Microcontrollers

Présentée par Maxime MADAU

Le 08 Novembre 2019

Sous la direction de Philippe MAURINE

Devant le jury composé de

M. Bruno ROUZEYRE, Professeur, Université de Montpellier	Président
M. Régis LEVEUGLE, Professeur, Grenoble INP	Rapporteur
M. Giorgio DI NATALE, Directeur de recherche CNRS, TIMA	Rapporteur
M. Michel AGOYAN, Ingénieur, STMicroelectronics	Examinateur
M. Yannick TEGLIA, Ingénieur, Thales	Examinateur

UNIVERSITÉ DE MONTPELLIER

Remerciement

J'aimerais tout d' abord exprimer ma gratitude aux personnes suivantes : M. Bruno ROUZEYRE, M. Régis LEVEUGLE, M. Giorgio DI NATALE, M. Michel AGOYAN, M. Yannick TEGLIA pour m'avoir fait l'honneur d'accepter d'être les membres de mon jury de thèse. Qui plus est, j'aimerais remercier M. Régis LEVEUGLE, M. Giorgio DI NATALE pour leur relecture et commentaires sur mon manuscrit de thèse.

J'aimerais ensuite remercier mon directeur de thèse M. Philippe MAURINE pour son encadrement. Ses nombreuses idées, et sa rapidité à rebondir en cas de mauvais résultat m'ont grandement aidé à progresser dans mes travaux. Mais aussi, son didactisme qui m'a permis de comprendre des concepts qui, au début de ma thèse, ne faisait pas partie de mon cœur de métier.

Je souhaite aussi remercier M. Michel AGOYAN pour son encadrement (coté industriel) et pour tout ce qu'il m'a appris. Mais aussi pour nos discussions tant sur les phénomènes physiques abordés durant ma thèse, que sur des sujets plus futiles (mais si le C++ c'est bien).

Je tiens à remercier M. Yannick TEGLIA pour la confiance qu'il m'a accordé en me proposant de travailler au sein de l'équipe AST; à la fois pour un stage de fin d'étude et dans le cadre de ce sujet de thèse. Mais aussi, pour tous les fous rire que nous avons eu et son intérêt pour mes travaux.

Cette thèse a été réalisé en collaboration avec le Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM) et STMicroelectronique. Je voudrais donc témoigner ma gratitude pour ces deux institutions. Tout d'abord, STMicroelectronics pour avoir financer cette thèse. Ensuite, le LIRMM pour l'encadrement tout au long de la thèse. Que ce soit, du point de vue administratif (facilitation des démarches) ou de l'accès aux formations.

Qui plus est, J'aimerais remercier les équipes en charge de la sécurité des systèmes embarqués au sein de STMicroelectronics pour leurs encouragements et leur aide durant ma thèse. En particulier, j'aimerais remercier : M. Thomas SARNOT, M. Albert MARTINEZ, (le rigoureux) M. Guillaume RAYMOND et M. Patrick HADDAD pour leur expertise et pour avoir supporté toutes mes répétitions de présentations. Je remercie aussi les tous les doctorants et post-doctorants du LIRMM avec qui j'ai pu travailler, notam-

ment Mathieu DUMONT et Rachid OMAOUAYACHE.

Une thèse est faite de haut de bas, mais j'ai toujours pu compter sur mes amis pour me remotiver : Anna (pusheen!!), Zakaria, Gaetan, Maryline, Christophe (クリストフー), Aymeric, Eric Tartière, JT, Brice, JC (Ahiiii), Guilhem, toutes les personnes de rêve de manga : Nicolas, Anastasia, Damien, Michael, Thomas ... merci à vous.

En particulier, j'aimerais remercier Mike MILLER (\m/) pour sa gentillesse, ses encouragements, et nos nombreuses blagues au travail.

Cette partie ne pourrait finir sans remercier les deux doctorants mais aussi amis avec qui j'ai partagé une bonne partie de ma thèse: Nicolas BRUNEAU et Lydie TERRAS. Merci à vous pour votre aide, votre bonne humeur et les discussions passionnantes dont tout l'open space a pu profiter.

Enfin et surtout, j'aimerais remercier ma famille, pour leur soutien, leur écoute et leur curiosité quant à mes travaux.

Abstract

Today, security of embedded devices is put in the limelight with the increasing market share of both IoT and automotive. To ensure a proper level of security to its customer such embedded components must undergo pentesting either to obtain some certifications to address security market but also to avoid tarnishing the name of the firm in case of vulnerability. Amongst the various attack paths, one of most threatening is the voluntary violation of operation condition to induce a fault on a circuit. These faults are then used for privilege escalation or combined with statistic tools to recover cryptographic keys. This thesis focuses on the use of electromagnetic field to generate such faults, this medium being the one that offers the best trade-off between cost and accuracy.

The efficiency of such family of attack has already been demonstrated in the literature. Yet, fault injection techniques shared a common problem which root cause is the amount of parameter an evaluator has to tweak to obtain a fault. Therefore, it is hard to state whether a target is protected against fault injection since evaluation is bounded in time, making exhaustive search not an option. Metrics or strategies should be defined to get the most out of up to date fault injection methods.

This thesis is a first step towards defining such metrics, and proposed to tackle the space complexity of EM fault injection. In other words, according to the attack scenario we developed metrics or strategy relying on both experimentation and the state of the art. The aims of those metrics/strategy being to reduce the space on the DUT that undergo electromagnetic emanation to the most likely to be faulted area.

In a first part, a criterion based on a basic model of the coupling between the injection probes and the circuit as well as today fault model will be developed. This criterion is then analysed and a refinement is proposed. Yet fault injection could also be used to nullify countermeasure that disable some attack vectors. Most of those countermeasures have in common the use of a true random generator. Thence in a second part we evaluate the robustness of an up to date true random number generator against electromagnetic perturbation. From this analysis we derived which parts of true random number generator are more relevant to be targeted using electromagnetic waves.

Résumé

De nos jours, la sécurité des systèmes embarqués prend une place de plus en plus importante. Notamment du fait de l'augmentation de la part du marché prise par l'IoT et le marché automobile. Afin de justifier un certain niveau de sécurité ces systèmes embarqués doivent subir des audits de sécurité afin soit d'obtenir une certification, qui peut s'avérer nécessaire pour adresser certains marché, ou alors plus simplement pour éviter de ternir le nom de l'entreprise en cas de faille. Le chemin d'attaque le plus efficace est probablement l'injection de faute obtenue par une violation volontaire des conditions d'utilisation d'un circuit. De cette faute différent scénarios sont possibles, soit celle-ci est couplée à des outils statistiques pour obtenir la clef secrète d'un algorithme cryptographique, soit elle permet une escalade de privilège.

Cette thèse se concentre sur les fautes induites par perturbation électromagnétique, qui est le support qui offre le meilleur compromis précision coût. Si les attaques par injections de fautes se sont montrées maintes fois efficaces dans la littérature, elles possèdent néanmoins un défaut conséquent dans le cadre de l'évaluation sécuritaire. Ce défaut vient du très grand nombre de paramètres offert par l'injection de faute à son utilisateur. Si on ajoute à cela les contraintes temporelles inhérentes à l'évaluation, on se rend compte de la difficulté de garantir la sécurité d'un produit contre de telles menaces. De ce constat il devient évident que des métriques ou stratégie sont nécessaire pour améliorer la qualité des évaluations.

Cette thèse est un premier pas dans cette direction et propose de résoudre la complexitée spatiales lors d'une campagne évaluation face à l'injection de faute électromagnétique. L'idée est de définir une métrique se basant sur des expérimentations ainsi que l'état de l'art pour réduire l'espace à tester à quelques positions qui vont presque certainement mener à une faute du fait de leur propriété physique.

Dans une première partie la création d'un tel critère est présentée. Celui-ci se base sur un modèle simplifié du couplage sonde d'injection circuit et sur le modèle de faute le plus récent. Ensuite les limites d'un tel critère sont analysées afin d'en trouver une amélioration. Cependant, l'injection de faute ne permet pas seulement d'attaquer directement une cible, elle peut aussi diminuer sa sécurité en visant ses contre-mesures. La plupart des contre-mesures ont en commun l'utilisation d'un générateur de nombre aléatoire, c'est

pourquoi la robustesse d'un générateur aléatoire récent sera évaluée dans une troisième partie. De cette analyse un chemin d'attaque sera dérivé dans le cadre de l'injection de faute via ondes électromagnétiques.

Contents

List of Figures

Chapter 1

Introduction

Context

Today security is a growing concern, regarding the typical topic of military communication protection or smartcard used for instance in banking operation. But there are not the only consumer of secure devices, other industrial fields also have security concerns as showed the not so recent *stuxnet* malware which was targeting Iran's nuclear plant. The two "newcomers" in the industrial world being the automotive sectors and the Internet of Thing (*IoT*).

Today, this two markets are put in the limelight as many research or magazine articles about hacking cars or botnet infecting IoT devices are written. In the case of botnet one can cite MIRAI, this worm makes a lot of noise in the press on December 2016 when huge duplicated denial of services were recorded. This botnet infected IoT camera and home routers by brute forcing dummy default password. Today it is still expanded to use zero-day exploit instead of just brute forcing. As of July 2018, thirteen versions of MIRAI have been reported infecting Linux IoT devices.

Usually, more security on a device comes along with the use of cryptographic algorithm. Different cryptographic algorithms and secure protocols have been design to address the following problematics:

- Confidentiality: is the property that guarantee nobody without the right to access a document, or a communication, can access it.

- Authentication: ensure the identity of somebody.
- Integrity: is the capacity to check whether a document has been altered or not.
- Non repudiation: is the ability to infer an action to someone.

Cryptographic functions are not proven to ensure its security but rely on mathematical problems that are hard to solve, where hard to solve means non computable in polynomial time. Therefore, in the case of cryptography we often talk of *one way function* and *trapdoor one way function*. A function f is a *one way function* if $f(x) = y$ is computable in a polynomial time but computing a x so that $f(x) = y$ is a hard to solve problem, i.e. non-computable in a polynomial time. When the computation of an image of x by f can be eased by the use of a secret value (often referred to as secret key) we speak of *trapdoor one way function*.

The security of a cryptographic primitive is ensured by this mathematical property, and is considered secure until someone finds a way to compute the inverse of the cryptographic function in polynomial time. Yet, this is true if and only if we stay at the mathematical level, once the algorithm is physically implemented on a device (microcontroller, smart card, …) this definition of secure does not hold anymore. Indeed, Kocher and al. in [47] showed that it is possible to bind secret key information with the target physical information (often referred to as *leakages*). For instance if we consider the following slice of code which may occur in the cryptographic algorithm RSA (Rivest Shamir Adleman):

```
1: for ( do i=L-1; i>=0; –i)
2:     s=s*s mod(p)
3:     if key bit = 1 then
4:         s=s*y mod(n)
5:     end if
6: end for
```

It is clear from the program flow that if the private key bit is 1 the computation takes longer than if the key bit is 0. Therefore, using physically measurable phenomenon such as the current consumption of the device one can retrieve the flow of the program. Indeed, by comparing the time taken by each loop iteration in the current consumption leakages, he can retrieve the key. The branch of cryptanalysis that study the different ways to retrieve the private key by observing the various channels of information is called *"side channel analysis"*.

Having access to a physical device also enable an attacker to alter the functioning condition of the device. This enables for instance to overclock the device so that it skips instructions or to inject glitch on the power network of the chip. Then using faulty ciphered text one is able to retrieve the secret key by using so called differential fault attack. The tool box of attackers have grown to become more precise and only alter a specific part of the circuit using LASER beam or electromagnetic fault injection. The later having the assets over the LASER of working without any preprocessing of the target and from both side of the chip. This kind of attacks does not only allow an attacker of bypassing pin verification and therefore threatening authenticate and confidentiality property. It also enables, as in side channels attacks, to retrieve the private key using a few faulty ciphered texts.

To mitigate cryptanalysis, cryptographic algorithms and their countermeasures mostly rely on true random number generator. Still generating truly random number is an intricate and open problem. To address this problem research project are funded such as the ”Hardware Enabled Crypto And Randomness” (*HECTOR*) project. This project regroups various academics and industrials on the topics of generating truly random number in a robust manner.

Therefore using cryptography on a device does not suffice to be secure. To address some security markets microcontroller producers should obtain a certification that requires the robustness of the cryptography against side channel and fault attack to be evaluated. Yet those tests are bounded in time and both LASER and electromagnetic fault injection offers a variety of parameters whose effects are not well-known. Therefore, both attacks offers a combinatorial complexity that is prohibitive and that impacts the quality of the evaluation. This thesis tries to address this problematic by presenting a criterion to reduce the area of the chip to be scanned during pulse electromagnetic fault injection evaluation.

In a second part, as this thesis was performed within STMicroelectronics and in the context of the European HECTOR research project an evaluation of the robustness of true number generator against pulse electromagnetic fault injection [1] is presented. Indeed, such

[1] a specific form of electromagnetic fault injection

kind of attack have not yet been tested against True Random Number Generator *TRNG*, at least in the literature, this thesis proposed to fill this gap by providing what is possible to achieve using pulse electromagnetic fault injection as well as how to mitigate these attacks against such a structure.

Thesis organisation

The first chapter is presenting a state of the art of the fault injection using the electromagnetic medium.

The second chapter introduces a new criterion to enable a faster security evaluation process of EM fault injection. To that purpose, the designed criterion enables to highlight parts of a circuit that are more likely to induce a fault on the targeted computation. In a second part a refinement of such criterion is discussed.

Finally the third chapter is about the impact of today electromagnetic platform against ring oscillator based true random numbers generators.

Chapter 2

Electromagnetic Fault Injection

The aim of the chapter is to have an in depth look to the different injection means that rely on electromagnetic field to inject faults. The topic range from the platform needed to lead such attacks to a study of the interaction with the device under test to finish with their countermeasure.

The most recent way to inject fault using EM emissions being Rowhammer; which exploits a flaw in DRAM to corrupt the content of memory. This flaw being triggered by software means.

The two other attacks rely on the use of an external EM field. In the case of harmonic EM fault injection, a continuous and amplified sinusoidal wave is emitted to target more specifically analogue devices. In terms of security the most relevant target of such platform being random number generation as it is the basic brick of numerous protections.

This thesis is focused on the last category, Pulsed EM fault injection which relies on a high voltage pulse outputs into an antenna to induce some sorts of reduced area glitch on the target. The most up to date description of the effect induced by pulse fault injections being the disruption of flip-flop elements of the circuits. This model being referred to as the Sample Fault model [70].

Yet to mitigate such attacks different countermeasures have been designed at various level. From the software level the main idea is to duplicate both data and instruction to ensure a correct operation. While on the hardware side several sensors working at the analogue or digital level have been created. Each of them having its own drawbacks and advantages.

Contents

2.1 Introduction

Since 1996 Electromagnetic Compatibility tests have to be passed for electronic devices to be sold. Those tests are performed to verify that devices EM emissions are not harmful but also to check whether or not the devices can compute under a hostile environment, i.e. under EM field emitted by nearby electronic devices. This highlights the fact that EM field can potentially be used to disrupt electronic devices such as System on Chip (*SoC*).

The first use of external EM field to purposely modify the state of a SoC was reported in 2002. Indeed, at that time, the authors of [78] demonstrated that using an EM coupling they were able to induce a sufficient perturbation in an Integrated Circuit (*IC*) to modify the content of memories (both RAM and EPROM). Even if there is little detail on the know-how, one can learn from this publication that authors used to couple with the *SoC* a simple coil wired around a scope probe alongside a flashgun capacitor to provide the voltage pulse.

One has to wait 2007 to see a second work [44] reporting EM injection as an efficient way to induce exploitable faults on *CRT-RSA* hardware implementation. Where exploitable means, it enables successful cryptanalysis attack. To that aim, authors used a spark gap to generate an EM pulse. Of course, with such an experimental setup, the produced EM injections were not characterized by high spatial and time resolutions. Similarly, duration and polarity of such EM injections were not controllable.

From these two pioneer experiments, Electromagnetic Fault Injection *EMFI* platforms have evolved and are now offering higher time and spatial resolutions (there is still room for improvements) as well as more reproducibility in the experiments. These gains are possible due to the control offer by newer platform. These controls mostly concern the characteristics of EM injections, among which the amplitude, the polarity, the duration. However, such capabilities come at the cost of a higher complexity while using EM injection and there effects are not yet well understand.

Within this context, this chapter aims at recalling what are the up to date EMFI plat-

forms. Indeed, there are now, according to the nature of the target, three different ways for injecting faults using EM field. These three techniques will be denoted afterward by: Harmonic EMFI, pulsed EMFI and Rowhammering. These recalls done, this chapter then gives a review about EMFI fault models more specially for pulsed EMFI since it is the main topic of this thesis. Finally, an overview of known countermeasures against EMFI is provided.

2.2 EMFI platforms

Today, there are three different ways to use the EM coupling for injecting faults into ICs. Two of them, namely the Harmonic Fault Injection and the pulsed EM Fault Injection, rely on different externally generated EM field sources. The third technique is more recent but has a reduced range of applicability. It is called Rowhammering and does concern only DRAM memories (and potentially some flash memory [46]). This technique exploits a flaw (the closeness of bitlines and wordlines) in the design of embedded memories which is becoming stronger and stronger with the scaling of technologies. This attack being very recent and restricted in its application it is not, to the best of our knowledge, required to address any security market.

The next sections will describe the different types of EMFI platforms. This description starts by the less important platform with respect to the subject of this thesis and ends by the description of the pulsed EM platform which is at the heart of this thesis. During the description of these platforms some basic principles on which they rely are recalled.

2.2.1 Rowhammer

It is widely known that EM waves can be the root cause of unintentional failures of ICs. Among the EM field related root causes of such faults, one can cite for instance EM compatibility problems between ICs or the crosstalk phenomenon [46], [72] between two internal wires of a same IC. As discussed above, EM waves can also be used to intentionally induce faults in such targets. The common assumption being that an external EM wave source is mandatory. Still, if this idea is most of the time verified, it is not fully true. Indeed, some EM phenomena occurring within ICs can be triggered to intentionally

induce exploitable faults. Crosstalk being one of such phenomena.

Indeed, in 2014, it has been showed in [46] that it is possible to corrupt DRAM content using specific memory accesses. This has led to the so called Rowhammering attack. The basic idea is to intensively read the same word in a DRAM and thus exploit crosstalk effect between neighbouring row. More precisely, this attack is based on the observation that at each reading access a small amount of charge is transferred to the neighbours wordlines. This phenomenon having as effect to modify the states of the other wordline, and the memory cell they manage, in the vicinity of the activated one.

Later in march 2015 Google's project zero team crafted a privilege escalation attack on Linux using this deficiency and brought this attack method in the limelight. In 2016 this attack became a fully remote attack by being triggered using javascripts and internet browser [38]. To understand how crosstalk, i.e. strong internal EM coupling, is exploited by Rowhammer attack, some reminders about DRAM organization and reading operation are given below.

DRAM is an extremely dense matrix of memory cells as depicted by Fig. 2.2. Each memory cells (Fig. 2.1) is composed of a single transistor and a capacitor used to store or not charges. This charging being used to encode a logic one or zero while the transistor act as a switch to access this value. In this matrix the lines are named Wordlines and the columns are named $bitlines$ because word values are read from $bitlines$ and a word is stored across one wordline. $Bitlines$ alternate with so called $\overline{bitline}$ which are complementary from a logic point of view. This means that a 0 on $bitline$ implies a 1 on its associated $\overline{bitline}$. Such an architectural solution facilitates the correct reading (and thus increases the operation speed of the whole DRAM) of the memory content. Such a gain is achieved by using differential sense amplifiers interpreting the difference of bias between $bitline$ or $\overline{bitline}$ and a reference voltage V_{ref}.

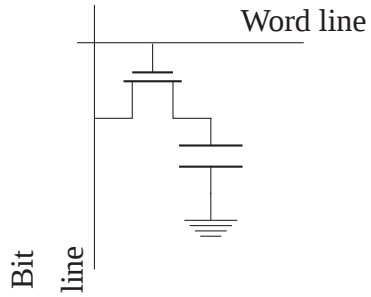


Figure 2.1: DRAM memory cell

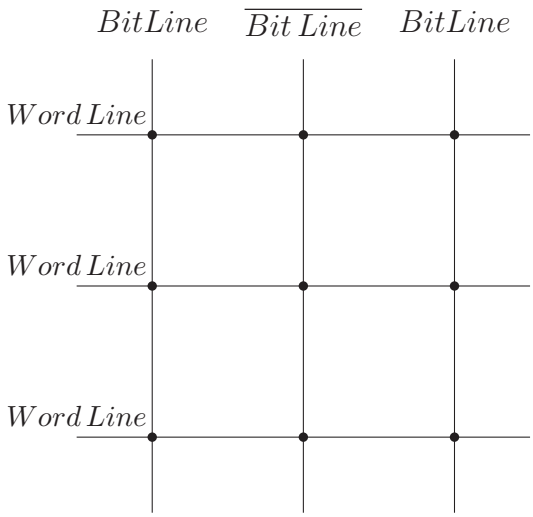


Figure 2.2: DRAM topology (a dot means a memory cell)

A read operation begins by pre-charging to half the supply voltage value all couples of ($bitline, \overline{bitline}$). Once, the pre-charge completed, the $wordline$ associated to the word to be read is activated. This turns ON all transistors of the bitcells storing the word to be read. As a result, according to the bit values stored by the bitcells, each couple ($bitline, \overline{bitline}$). Each induced voltage difference between a $bitline$ or a $\overline{bitline}$ is then converted into a logical value by a sense amplifier. This value is then stored in an output register.

Rowhammer attack simultaneously exploits the crosstalk phenomenon and a specificity of DRAM architecture. Indeed, from the reminder on the memory cell architecture one can notice that since memory cells are capacitors then once they are discharged their value is lost. To circumvent this, DRAM refreshes memory cell at regular time interval which is lesser than the retention time of a cell. Moreover, for the same reason the reading

operation also requires to restore the read word in the memory. That is why read operation in memory is followed by a write operation. Rowhammer will enable to reduce the retention time of some memory cells making the refreshing operation arriving too late. This is where the coupling effect is exploited. By activating repeatedly a specific Wordline the current variation in this line generate an EM field which will activate neighbouring Wordline. Then as Wordline is activated memory cells starts leaking current in their $bitline$ (or $\overline{bitline}$). By intentionally reducing the retention time of memory cell in a non-accessed Wordline an attacker is therefore able to modify memory state. Due to the alternation between ($bitline, \overline{bitline}$) the discharged cell will be seen as logic '1' or '0' this explaining the bitflip phenomena.

Algorithm 1 Rowhammer Principle

```
1: while true do
2:     Read at address 1
3:     Read at address 2
4:     Flush the cache (example clflush)
5: end while
```

If the principle of Rowhammer is quite simple, there are some practical difficulties. Among them one is the presence in modern DRAM of row buffers. Such a buffer stores, similarly to cache memory, the last read word. It is used to save energy when consecutively readings the same word; a read operation being energy hungry (especially the write back operation). Such a power optimisation is an inner protection of DRAM against the Rowhammering attack. Nonetheless, this countermeasure can easily be bypassed by Reading two different memory address in the same bank (this activates the same wordline). This strategy defines the so called double-sided Rowhammer attack [46], even if single sided row attacks has also be proven almost quite efficient [37].

The second problem to circumvent to lead a Rowhammer attack is bypassing the cache memories. But thinking the only presence of cache is enough to protect against Rowhammer is not true since there are many ways to bypass it. The first one consists in flushing the cache using for instance the $clflush$ instruction when working with x86-64 compliant processor architecture. The second one is to reverse engineered the cache eviction policy as presented in [38] and by project zero team. The third one is the use of "non temporal

instruction". These instructions are called like this since they are used on data that will not be used in the near future. Cache performance being governed by the fact that data are reused often "non temporal data" would harm performances if they were cached. To that end, specific instructions such as "memcpy" have been design to handle such data. However, handling those data means those instructions can bypass cache policy. That is why authors of [77] studied the possible used of this instruction family to lead Rowhammering attack. The result has been proven quite efficient.

Recently is has also been shown possible by relying on Direct Memory Access (*DMA*) for net transaction. Indeed, in [51] [84] authors show how using the Remote Direct Memory Access (*RDMA*) of some boards it is possible to stress the DRAM remotely. DMA and RDMA are hardware device that enables memory to memory operation in a very efficient way. This efficiency coming from the fact they bypass the CPU, which in terms of Rowhammer feasibilities means it bypasses the cache.

Despite its restriction to DRAM memory and its difficulty of use, Rowhammer attack has be proven effective as a security threat by both enabling privilege escalation or recovering secrets from the RSA algorithm (in [16]).

2.2.2 Harmonic EMFI platform presentation

Inducing faults using an EM field can also be achieved using the so-called Harmonic EMFI technique. It consists in exposing the ICs to a continuous EM waves and more precisely to an electrical field. The choice of continuous waves rather than EM pulses is explained by the characteristics of targets this technique aims at disturbing. Indeed, harmonic EMFI was developed to target analogue blocks of ICs whose operation is not clocked but continuous in time. Such a platform was used by F. Poucheret to induce faults in a clock generator [57] as well as in a True Random Number Generator [57]. A description of the induced faulty behaviours and of the effect of continuous EM field is given in section 2.3.

As depicted Fig. 2.3, a harmonic EMFI platform is composed of several functional

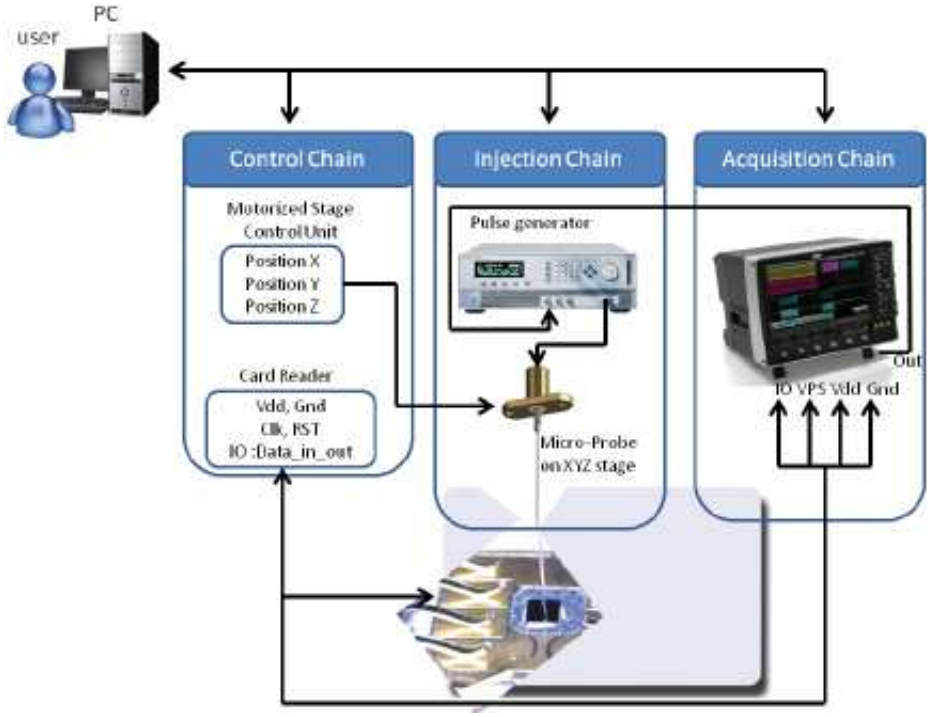


Figure 2.3: Harmonic Injection Setup [59]

blocks. Among them, the most specific and important are :

- The RF generator providing sine waves with a frequency ranging between 100MHz and several GHz.
- The HIFI power amplifier increasing the power of the sine waves from mW range to hundreds of watts.
- An antenna receiving the powerful sine wave and delivering a local and intense EM field. This antenna could be made up of a tungsten tip with a sharp end of diameter in the range of $10\mu m$ so that to create an intense and local electrical field.

2.2.3 Pulsed EMFI

This section introduces the pulsed EMFI platform that I built at STMicroelectronics to develop my Phd works. Because this platform is at the heart of the thesis, a greater attention is paid for it than for other EMFI platforms.

2.2.3.1 Structure of the pulsed EMFI platform

The pulsed EMFI platform I used is really close to the historical EMFI platforms introduced in the pioneer works [78] and [44]. It features different electronic equipments as depicted Fig.2.4 that shows our EMFI platform :

- A high speed generator providing to an EM probe a voltage pulse with an amplitude ranging from $50V$ to $400V$ and a duration ranging from $6ns$ to $100ns$. The produced pulses are also powerful since the generator delivers up to 8A.
- EM probes or probes built around a ferrite core. More details about the different types of EM probe are given in the next paragraphs.
- X, Y, Z motorized stages with an accuracy of $10\mu m$ to be able to scan with the EM probe the whole IC surface and thus draw EMI susceptibility maps.
- A digital sampling oscilloscope to synchronize the EM pulses with the IC operation by monitoring either the power or the EM radiations of the DUT.
- A low noise voltage amplifier ($40db$) to amplify the EM radiations of ICs collected to monitor their behavior and synchronize the EM shots.

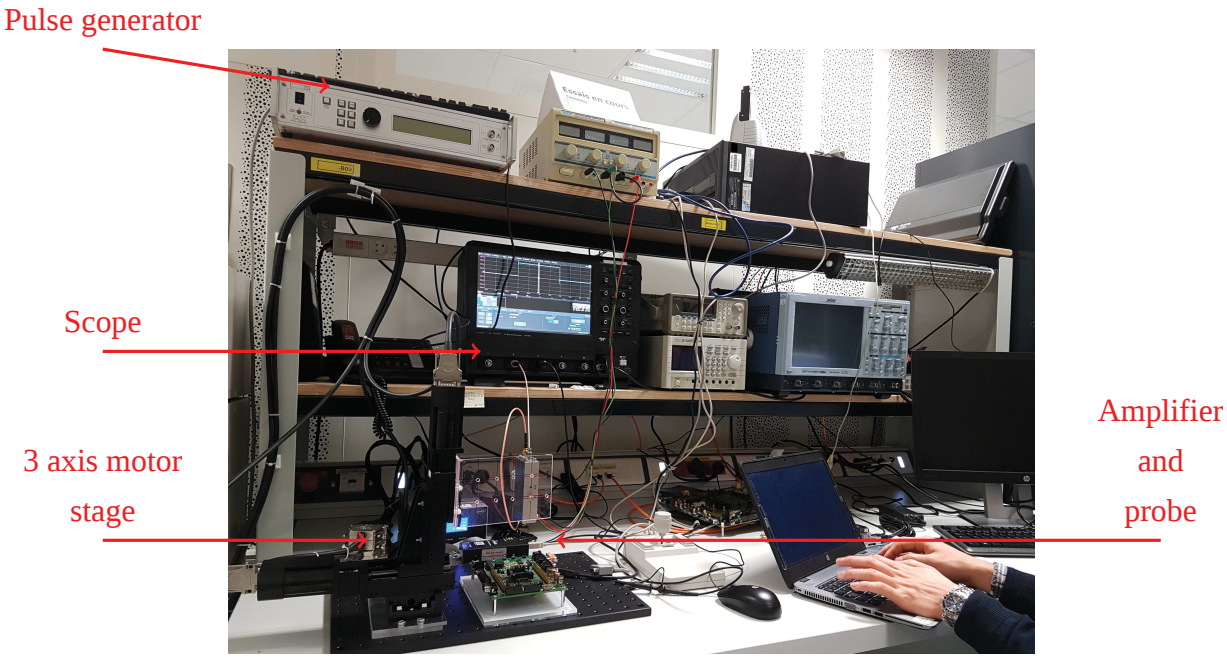


Figure 2.4: Our pulsed EMFI platform

2.2.3.2 Operation principle of our pulsed EMFI platform

The developed pulsed EMFI platform aims at generating a sudden and short variation of the magnetic field and to concentrate it on the smallest possible part of the IC surface (either frontside or backside). Because of the EM coupling between the IC and the EM probe, this results in inductive coupling creating many electromotive forces in a more or less wide area of the DUT. This area's width being tightly tied to the intensity of the EM shot as well as the EM probe characteristics.

The electromotive forces being necessarily generated in closed loop structures, one may expect to target mostly the power and ground networks of ICs because signal wires end, in CMOS circuits, on one or several MOS capacitors (gates of MOS transistors) characterized by a really high impedance. Thus, the electromotive forces induce a strong flow of parasitic currents in ICs and create strong and more or less local disturbances of the ground and power networks. As a final result, if these disturbances are sufficient, transient faults occur corrupting the course of the algorithm processed by the IC.

2.2.3.3 EM Pulse generation

In an EMFI platform, the chain of elements ensuring the generation of the EM pulse is of course of prime importance. There are different ways for generating an EM pulse even if all solutions share at least one common element. This element being the EMFI probe made up of a coil with or without a ferrite core in which a sudden variation of current is induced to produce the EM pulse. Despite this common element, these techniques differ in the way the current variation is produced. Two approaches are possible: one consisting in attacking the EMFI probe input with a voltage pulse, the other consisting in attacking the EMFI probe with a current pulse.

2.2.3.3.1 Comparison of the different pulse generation platform The use of a current pulse generator is the more natural solution. It is also the historical one. Indeed, [78] proposed the use of flash gun capacitor. To that end, the current of the flash gun is fed to the EMFI probe to produce the EMFI pulse. This is also the approach chosen in [44]. However, a gaz spark is directly used to generate the EM pulse.

Such an approach offers two significant advantages. The first one is the high power of the generated EM pulse. The amplitude of generated current is so high that EMFI probe without ferrite core can be used. The second advantage is the easiness with which one can match the impedances of the current generator and of the EMFI probe to avoid bouncy EM pulses. However, this approach has also two main drawbacks. The first one is the timing jittery of the EM pulse generation system; jittery due to that of the ionization time of the air gap or gaz tube. The second one is the lifetime. Indeed, the rapid ageing of such systems renders difficult the reproducibility of results over time. A last drawback of this approach is the impossibility of controlling the amplitude and duration of the generated EM pulse which is imposed by the characteristic of the gap.

Recently, the authors of [6] introduced another low cost injection bench to replace the use of gaz spark. This setup is the electronic current pulse generator depicted Fig.2.5. It delivers a 20A pulse. This value is to be compared to that delivered by sparks : $100-500A$.

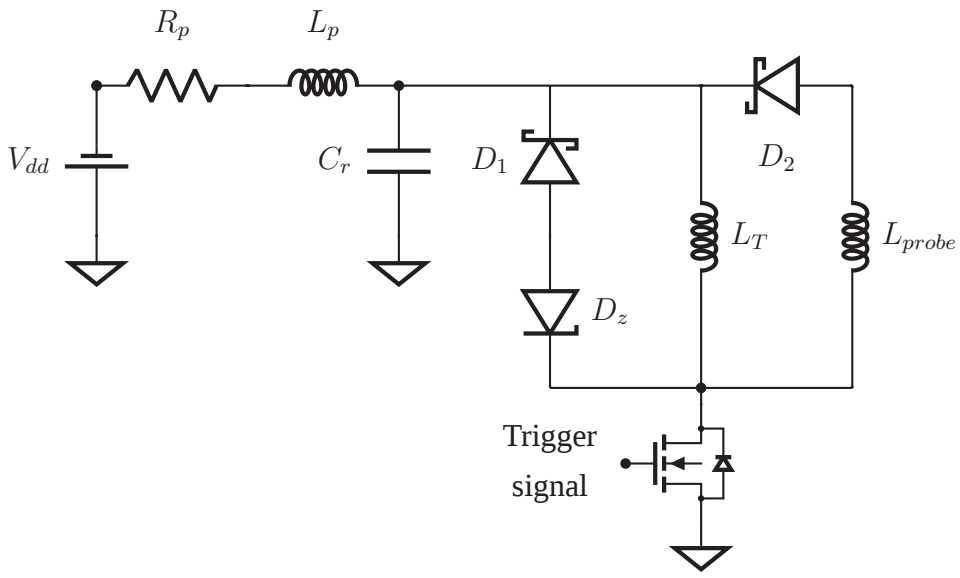


Figure 2.5: Low voltage injection circuit proposed in [6]

The alternative approach consisting in using a voltage pulse generator offers different advantages and drawbacks. Among the advantages one can identify the high timing resolution and the great stability and lifetime of nowadays voltage pulse generators.

The second main advantage is the controllability of the voltage pulse characteristics

that enables a fine-tuning of the EM pulse (amplitude, duration, polarity) with respect to the Device Under Test *(DUT)*. The main drawback of such an approach is the difficulty of generating high voltage pulse with a high current of sufficiently large bandwidth. As a result, the power of such EMFI systems is lower than current based pulse generators. This lower power might be partially compensated by the use of EMFI probe built around a ferrite core to concentrate the magnetic field lines and thus the EM power on a reduced area.

A second drawback of voltage pulse generators is their output impedance which is usually greater ($\sim$ 50 Ohm) than that of EMFI probes ($\sim$ 1 Ohm). This drawback induces bouncing of the voltage pulse between the generator and the EMFI probe. This is typically a problem currently tackled by the FUI project CSAFE+ in which are involved both STMicroelectronics and the LIRMM.

Finally the last limiting points of such platforms is the price of the pulse generator. This is what motivated the BADFET project [26] where authors proposed a low cost high voltage pulse generator where they use a high capacity capacitor. Despite being proven efficient to bypass U-boot, this platform falls in the same platform as the one introduced in [6]. Indeed, all the tweaking offered by the voltage pulse generator are hardwired and impose by the chosen components. Therefore, a trade-off has to be chosen between price and accuracy.

2.2.3.3.2 Analysis of the impact of voltage pulse characteristics on fault injection

Despite these drawbacks, voltage pulse based platform has been demonstrated really efficient to inject faults into IC thanks to its flexibility offered by the high controllability of the EM pulse characteristics. This explains why such type of platform was used during my thesis.

This flexibility and high controllability takes the form, using our platform, of four degrees of freedom, which are:

1. the pulse width or duration between $5ns$ and $100ns$ with an accuracy of $10ps$.
2. the pulse amplitude between $50V$ and 400 by step of $0.1V$.
3. the pulse polarity.

4. and of the moment of the EM injection which can be set between $360ns$ and $1s$ with an accuracy equal to 1% of this value.

Each parameter has its own effect on the EMFI. However, even if these effects are not well understood, as of now, experience lead by the community seems to converge to the following informal conclusion.

Regarding the pulse width there is no known study on the repercussion of this parameter on the pulse injection outcome. Still, there is a more and more widely accepted empirical hypothesis on the behaviour of the EMFI. This hypothesis being the following: faults are more easily obtained if the voltage pulse (EM pulse) is applied just before a rising edge of the clock and more precisely if the second edge of the voltage pulse occurs around (during) a clock rising edge. Therefore, to fault a specific operation it is a common practice to synchronize the pulse injection with the EM traces of the computation under evaluation at the given position.

This empirical observation has been confirmed and explained by simulation by a PhD student who is currently working at STMicroelectronics. Experimental validation of the enhanced sampling fault model [70] is on going and will be published really soon.

The second main parameter of EMFI is the pulse amplitude. This parameter impacts the IC area affected by the EM pulse as well as the probability of getting a fault that results in DUT destruction. Indeed, it has been experimentally observed by S. Ordas [70], that, for a given EMFI probe position above a given DUT:

- There is a minimal amplitude to induce a fault in an IC.
- Increasing the amplitude above this threshold enlarges the timing window during which the fault can be induced, but also the area above which the EMFI probe can be placed to obtain this fault.
- Increasing the amplitude above a value leads to stop the operation of the DUT and to obtain no response.

Thus there is trade-off to find that makes this parameter quite tedious to set. Indeed, on one hand, to induce a fault into a specific IC, a minimal amplitude should be set (otherwise no fault occurs and the evaluator miss some interesting attack entry points). On the other

hand, rising too much the amplitude can disrupt too strongly some parts of the DUT (such as the clock generator) or induce some latch-up, leads the DUT to mute state or getting irrelevant faults in terms of cryptanalysis.

Finally, the last main degree of freedom is the polarity of the EM injection, i.e. the sign of the voltage pulse applied at EMFI probe input. This important parameter can condition, for a given EMFI probe position, the occurrence of a fault or not.

2.2.4 EMFI probe design

An important part of an EMFI chain are the EMFI probes and this especially when working with a voltage pulse generator rather than a current pulse generator as discussed in the preceding sections.

Directly connected to the pulse generator output, they are in charge of transforming the voltage pulses into an EM pulse and in concentrating them into the smaller volume possible. Their shape and characteristics are therefore extremely important, and should be chosen properly.

At the origin EMFI probes were designed to have a technology close to the ones used for EM side channel analysis, i.e. for listening purposes thence no current is injected in it. Because this was a first and rough approach, authors of [69] analysed by simulation different probe designs for pulsed EMFI. From this analysis, it appears that one good way to design EMFI probes was to use a ferrite core around which is wrapped 4 to 5 spires. In addition, it was shown in [68], that sharpening the tip end following special rules is an interesting solution to concentrate the magnetic field on a reduced area.

In accordance with the simulations reported in [69], in [71] and [22] it seems that in practice 4-8 coils are indeed a good trade off.

Two kinds of probes were used during my thesis. The first one, may be the most common one, is the cylindrical EMFI probe shown by the topmost part of Fig. 2.6. The magnetic field lines flow (Fig. 2.7) from one end to the other and the amplitude of the

Figure 2.6: Examples of EMFI probes I used during my work

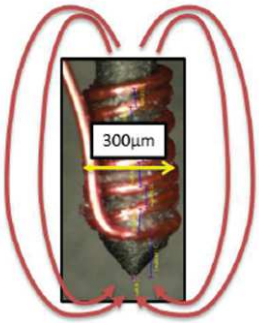


Figure 2.7: Magnetic field lines generated using a cylindrical and EMFI probe with a sharp tip end

magnetic field is maximum below the EMFI probe and decreases quickly with the distance after the edge of the probe.

The second type of probes I used are the U-shape probes. One of them is shown on the lower part of Fig.2.6. The gain in accuracy coming from the gap in the ferrite core. This gap having as effect to concentrate the magnetic field line instead of being along the whole probe. It has been designed to go a step further by enhancing the locality and the power of the EMFI. However, this accuracy comes at the price of another parameter to set during characterization campaign. Indeed, the magnetic field lines now flow according to the probe axis. Thus, the direction of the probe has to be taken into account. By direction, we mean either horizontal or vertical, when considering the Z axis as the rotation axis. This parameter is important since, according to the orientation the probe, EM field will affects mainly the horizontally or vertically routed metal layers.

In the quest of having more localized effect, the idea of recycling old tape record

reading head might be relevant. The physical phenomenon exploited by EMFI being very similar to one used by the old tape record system. In addition, these heads have been tailored to have a very local effect thanks to the use of a much smaller air gap in the ferromagnetic components. This plus the fact that those probes are manufactured with a better precision than the handmade probes make them as good potential EMFI probes. However, the only drawback of these probes is that they are not designed to work with voltage value as high as $400V$ or high amperage value i.e. non of the up to date bench can be used as is.

Still having the more localized effects is not a mandatory property of fault injection. Having a large area under cover enable to perform multi bits fault which can be used by some attacker scenarios like byte zeroing used in Ineffective Fault Analysis (IFA) [23] or the lesser constraint attack described in [32]. The classical Piret [75] and Giraud [35] attacks can also be achieved using byte faults.

2.2.5 X,Y and Z Positioning

The last degree of freedom an evaluator has to set during the characterization of a DUT EMFI robustness is the (X, Y, Z) position of the EMFI probe tip end above the IC surface. This positioning strongly influences the probability to induce a fault and its nature as it can be observed in [70].

To control this parameter the bench I developed features 3 motorized axes in order to set the (X, Y, Z) coordinate of the probe tip end with an accuracy of $10\mu m$. This accuracy was decided as sufficient regarding the dimensions ($\sim 1mm$) of our EMFI probes but also with respect to EMFI experiments. Those experiments showing that the effects of EMFI using our probes do not significantly change for positioning difference lower than $25\mu m$.

An additional argument to limit the accuracy of our motorized stages is related to the DUT's area which in the case of my PhD is in the order of magnitude of $10mm^2$. For such DUT, performing EM susceptibility scans of their whole surface with X and Y displacement steps of the probe lower than $10\mu m$ is extremely time-consuming. More than 5 days of experiments can be required to achieve a scan because the DUT has to be re-

initialised after the occurrence of each faulty behaviour to get reliable results. Therefore, to perform as much test as possible a trade-off between accuracy and the time taken by scanning the whole die of the DUT has been found. As a result, during my work, EM susceptibility scans were performed with steps greater than $100\mu m$, a value well above the accuracy of the chosen motorized stages.

As aforementioned, the EMFI probe positioning has a great influence on the effect produced in the IC. This can be explained by the characteristics of EM induction which acts only on closed loop structures whose inner areas has to be crossed by magnetic field lines to be affected. Among the constituting elements of IC, the ones forming closed loops are the power and ground networks which are usually routed on the top-level metal layers. Thus, according to the EMFI probe positioning and to the distance between its position and the targeted operating functional block, EMFI can have no effect or can induce exploitable (or not) faults[1]. The probe positioning is therefore a key point during characterisation of DUT against EMFI as it is the case for LASER Fault Injection.

2.2.6 Summary

In the preceding sections three ways of using EM coupling to threaten security have been presented. The threat ranging from privilege escalation to ”breaking” cryptographic algorithm implementations.

The first one, namely RowHammer, use software to exploits a design flaw in DRAM memory and thus does not require any expensive equipment.The two others, namely the Harmonic EMFI and pulsed EMFI exploit EM induction to disrupt the operation of ICs. Both are based on the use of an external EM field generator.

Literature shows that Harmonic EMFI seems to be more tailored to inject faults into analogue blocks whereas pulsed EMFI is more general purpose even if it is more indicated to disrupt the behaviour of ICs during a few (1, 2, ...) clock cycles.

Pulsed EMFI, which is at the heart of my work, allows injecting faults into ICs by gen-

[1] where exploitable is from a cryptanalysis point of view

erating in a reduced volume close to IC surface sudden variations of the magnetic field. Moreover, those variations are bounded in time. This variations induce by EM induction parasitic currents in the closed loops of the power ground networks of the DUT. The exploitation of EM induction offers a key advantage to EMFI over other injection means. This advantage being the ability of injecting faults from ICs frontside or the backside and this without necessarily removing the package. This advantage has been highlighted by several papers about EMFI.

Yet, despite its efficiency both pulsed and harmonic EMFI offer a wide range of parameters to tweak in order to achieve a fault. This problematic of setting the EMFI parameters, in the scope of characterisations conducted in a reduced time, remains however unaddressed in the literature. The next chapter of this document will address this problematic, the next sections are first related to EM fault models; these models will be exploited to limit the complexity of EMFI characterisations.

2.3 EM Fault Models

All the platforms presented in the preceding sections have been demonstrated efficient to mount successful attacks on various types of ICs. Yet either to enhance the accuracy of the security characterizations, or to design efficient countermeasures with the right cost, the underlying principle as well as the logical and electrical effects of the EMFI must be understood.

The understanding of how faults occur usually results in defining what is usually called a Fault Model. In fact, several fault models have to be defined for a same injection mean. Indeed, it is typical to define one fault model for each abstraction level considered during the design flow of ICs. Thus, a fault model can be built at the electrical level, at the logical level, at the RTL level and even at the software level. However, most of the time, two models are set up: one at the software level and one at the hardware or electrical level.

Developing a fault model at software level requires the understanding of effect from a software point of view. This means how data manipulated by a program can be modified by a fault injection. Such a fault model also considered the instruction side of a program

and aims at determining how the program flow can be perturbed. Thence, a software model should state how instructions and data can be disrupted by a fault injection.

Because software level models are quite abstracted, they usually offer the advantage of being agnostic to the considered type of injection mean (LASER, EM, …). However, this is just a rule of thumb, and there are differences, even small, on what can be done with the various fault injection means. Therefore, this agnosticism comes at the cost of a rough modeling of the fault effects.

On the other hand, hardware/electrical level fault models are based on physic and usually allow simulating fault injection with electrical simulation tools like spice ([64]) or spectre[2]. They thus allow representing with a really high accuracy what is really going on in a chip submitted to a fault injection. However, this accuracy has a cost, simulations are usually long and thus have to be limited to more or less small part of ICs.

The rest of this section is organised as follows. First, software EM fault models are discussed before introducing state-of-the-art information about the EM hardware fault models.

2.3.1 Software / Logic level fault models

At software level, a fault model aims at summarise all possible incorrect behaviours that could occur at runtime. One of the direct interest of such a model, is that by knowing all possible kinds of incorrect behaviours it becomes possible to design optimise countermeasures. Where optimise means with a minimal overhead on both memory and runtime while allowing the detection of fault injection attempts or mitigating their effects. Of course, here incorrect behaviour means either a corruption of the data or the program expected flow.

Such types of models might seem less accurate than hardware or electrical models

[2] https://www.cadence.com/content/cadence-www/global/en_US/home/tools/custom-ic-analog-rf-design/circuit-simulation/spectre-circuit-simulator.html

because they do not try to explain all the effects of a fault injection but rather its impacts on data manipulated by software. However, this lesser accuracy turns into an advantage as this model is very generic. Indeed, because detailed physical aspects being partially disregarded, such models are often common to several fault injection means (at least some aspects are shared) and remain valid for different embedded devices.

Faults injected during RowHammer Attacks have already been characterized. They are bit flips in memory cells [46]. The bits which are flipped depend on the inner structure and organization of the memory under test. More precisely, Rowhammer, according to the structure of the memory, sticks at 0 or at 1 some bits. Such a model can be perceived as a logical model. However, from this simple information and the knowledge of both the embedded software and architecture of the DUT one can derived a software model.

Even if there is no specific papers related to software EM fault models, EMFI is now recognized to induce instruction skips and branch corruptions. More precisely, it has been shown in [61] that as a Rowhammer Attack, EMFI can flip bits but not only in memory. In addition, it has also been shown that EMFI can induce bitsets and bitresets (as well as *stuck at one* and *stuck at zero*). Those two scenarios apply on one to several bits.

In addition to data corruptions, the bitflips also have an impact on the algorithm execution flow. Such flow can be modelled using graph theory, this graph is named a Control Flow Graph (*CFG*). It can be defined as the graph of all possible paths that can be taken by a basic block; a block being a sequential piece of code, which entry point is the target of a jump operation, and such a block contains one jump operation at his end.

For the purpose of illustration, let us consider the following piece of code Fig. 2.8 that performs a pin code testing and its CFG Alg. 2 Fig. 2.8. Those two figures highlight the correspondences between the different Basic Blocks (BBi) and the code lines using colors.

Algorithm 2 Dummy code for Control Flow example

```
1:  int a = 0;
2:  bool transactionApproved = false;
3:  repeat
4:      ReadUserPinCode(a);
5:      if TestPinCode(a) then
6:          puts("Transaction approved");
7:          bool transactionApproved = true;
8:      else
9:          puts("Transaction not Approved");
10:     end if
11: until transactionApproved
12: return;
```

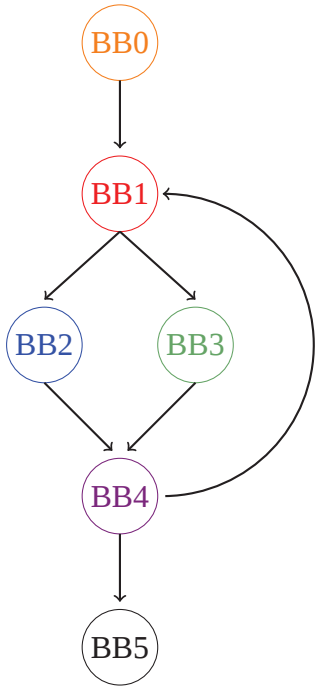


Figure 2.8: Control flow graph

Considering this basic dummy example 2, one may wonder what could happen if an adversary modifies the opcode of the TestPinCode instruction by flipping one or several bits. There are two possibilities.

First, the faulted opcode takes an invalid opcode value (there no instruction with such opcode in the instruction set). Depending on the target, an invalid opcode might be considered as a nop (no operation) and in this case the adversary has skipped the pin code verification. Otherwise, the invalid opcode generates a 'hard fault'. In the second case, the faulted opcode takes the value of another opcode and in this case another operation is performed or a branch operation is created. Thus, generating a valid opcode could also allow the adversary to skip the pin code verification step.

In case a fault injection has modified data rather than opcode values, the adversary might change the address used in some branch instruction. This could result, considering again our basic example, in executing the branch BB2 rather than the branch BB1 even if a wrong pin code has been dialed. Such an effect being similar to that of skipping the pin verification step.

Therefore, an adversary able to modify the execution of program so that to not follow the CFG constitutes a serious threat on secure products. Another possible effect of data corruption could be the reduction of the iteration number of a loop. This also which can have some dramatic results in terms of security. For instance, one can lower the cryptographic resistance of an AES by reducing the number of executed rounds [30].

To conclude, pulsed EMFI and Rowhammer injection techniques share the same software fault model except that Rowhammer is only efficient to disrupt certain DRAMs. Indeed, both fault injection means induce bitflips and thus can be used to perturb a program execution by inducing for example instruction skips. In addition, Rowhammer and EMFI have similar impacts at different abstraction levels. First, at high level of abstraction, they can be considered as corrupting the control flow graph, while at logical level their effects can either be model by bitsets, bitresets, bitflips. However, because pulsed EMFI targets are not limited to DRAM the area of an IC being vulnerable to EMFI is much wider and ranges, from a software point of view, from the pipeline to memory content. As a consequence, it seems easier to induce exploitable faults in specific instructions using EMFI rather than with Rowhammer.

2.3.2 Hardware / Electrical Fault Models

Hardware/Electrical fault models have the same goal as software fault model: explain how faults occurs within DUT. However, this time they do not address to software developers but to IC designers.

This section focuses on hardware / Electrical models of faults induced by external sources of EM waves. It describes physical and logical level aspects of the fault injection mechanisms.

First the case of harmonic fault injection fault models is considered before talking about pulsed EM fault model.

2.3.2.1 Harmonic EMFI

Harmonic EMFI has been designed to disrupt analog blocks and more particularly clock generators, True Random Numbers Generators (TRNG), Power Regulators, etc. After the development of associated platforms, some works demonstrated its efficiency in altering the operation of embedded clock generators [76] or the entropy sources of TRNG designed with Ring Oscillators [11].

In the latter work, authors exploited the ability of harmonic EMFI in imposing the phase between outputs of freely (asynchronously) running ring oscillators to bias the random number flow of a TRNG. The TRNG considered in this publication being build around the phase jitter of ring oscillators, which is an unpredictable phenomenon commonly used to generate randomness. Yet such TRNG are often build using different oscillators (of same frequency) and comparing the different state of the oscillators at a time close to the oscillators period. Because of the phase jitter that is different for each oscillator the rising edge will not occur exactly at the same time for all oscillators, and this property is then used to generate random number. To perturb such TRNG authors of [58] have used a phenomenon quite similar to that observed in 1665 by Christiaan Huygens who had observed the progressive synchronization of clocks sharing a same physical support. In the case of TRNG all the oscillators share almost the same phase signal and therefore the source of randomness does not exist anymore or at least is biased toward the same value.

Later, authors of [76] demonstrate the ability of harmonic EMFI in directly and locally injecting power in an embedded clock generator and thus the possibility of overclocking DUTs.

Despite this result, the most threatening use case of harmonic EMFI remains the disruption of True Random Generator (TRNG). Indeed, random numbers generators being widely used to generate secret keys or to provide the entropy required by some efficient side channel countermeasures [19], [36]. The robustness of such countermeasures relies on the randomness of the entropy source. Thus inducing a bias in embedded TRNG results in significantly lowering the security of embedded platform.

2.3.2.2 Pulsed EMFI

In this section the two fault models associated to pulsed EMFI are introduced following the historical order of appearance; namely the timing fault model [88] and the sampling fault model [70]. Before the description of these models, some basics about the operation of clocked integrated circuits are recalled.

To explain the operation of synchronous Integrated Circuits, let us first consider the highly simplify schematics of a circuit given on Fig. 2.9. In this figure, D-type Flip-Flop (DFF) are the memory elements commonly used to sample the output of the logic block. This operation is performed at the clock frequency, i.e. on each rising edge on the CLK input signal of the flip-flop. More precisely, their role is to sample the results of calculation done at the n^{th} clock cycle and to transmit them to the next logic blocks that will process them at the $(n^{th} + 1)$ clock cycle. They thus ensure the synchronization and sequence of operations in circuits.

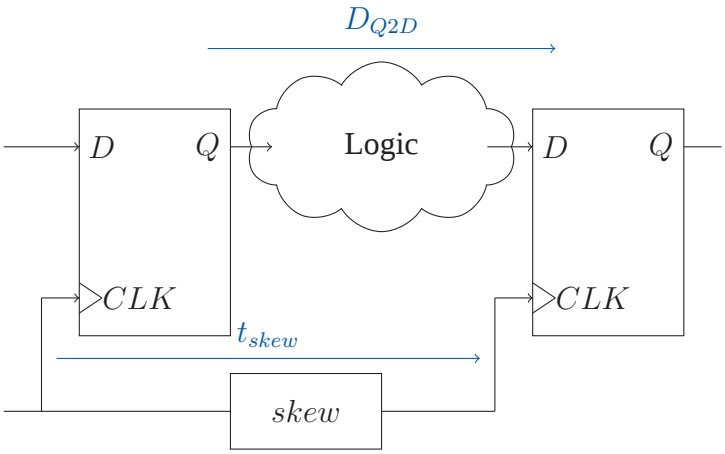


Figure 2.9: Cartoonish of synchronous circuits

To ensure such a role, DFFs sample the value of the data available on its input D at the rising clock edge (arriving at its input CLK). The stored value is then output on Q output signal. To operate correctly, all signals, including the supply voltage and the ground, must be stable since a time of t_{setup} seconds before the rising clock edge and must remain stable during t_{hold} seconds after.

These timing constraints have to be met for each DFF in a circuit, imposing timing constraints during the design flow of ICs. Indeed, looking at Fig.2.9, it comes that the following inequality must be met:

$$T_{CK} \geq CK2Q + Q2D + t_{setup} + t_{skew} \tag{2.1}$$

Where $CK2Q$ is the propagation delay of the DFF on the left, $Q2D$ is the propagation delay of the logic between the two DFF, and t_{skew} is the delay between the arrival times of the clock signal at the inputs of the two DFFs. Indeed, when the n^{th} rising clock edge arrives at the input of the left DFF, a new data is pushed forward from D to Q in $CK2Q$ s. This new data then propagates from Q to the input D of the right DFF in $Q2D$ s. To be correctly sampled by the DFF on the right at the next rising clock edge, this new value must be stable since t_{setup} s before the next rising clock edge i.e. since $T_{CK} - t_{setup} - t_{skew}$ s after the preceding rising clock edge. Hence, the Eq. 2.1 which is illustrated on figure 2.10.

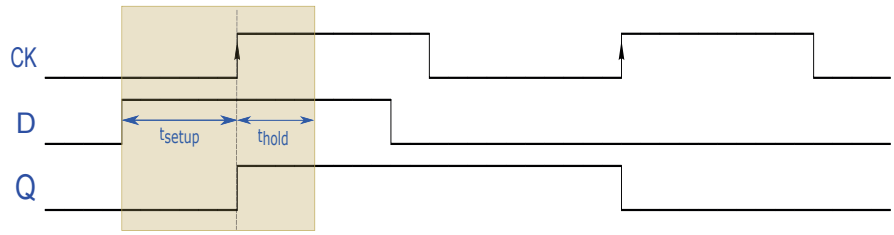


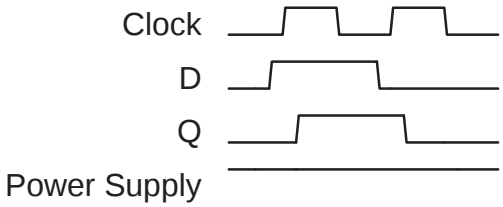
Figure 2.10: Illustration of the so-called setup and hold time timing contraints

Timing fault models are based on the inequation 2.1. Indeed, it has been sustained in [27] that EMFI produces timing faults because EMFI induces fluctuations in the supply voltage rails. This fluctuation then induces increase of the $Q2D$ timing and therefore violations of the Eq.2.1.

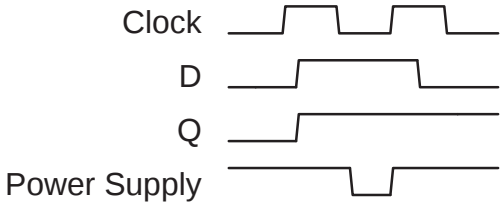
Let's have a look to the resultant timing diagram and the effect of EMFI on DFF sampling operation (when copying the input signal). In Fig.2.1 DFF's normal operating is represented while Fig.2.2 gives a possible outcome of EMFI on the timing diagram.

On the normal operating timing diagram D respects the t_{setup} timing constraint; this implies that after the second rising clock edge the output signal Q is a logical '0'.

Now let's analyse Fig. 2.2. In this case, an EMFI occurs during the first clock period. As a result, the power supply experiences a voltage drop. This drop shifts in time the second edge of D after the second rising clock edge and thus D violates the t_{setup} constraint. As a result, the DFF samples a '1' instead of the expected '0'.



Timing diagram 2.1: Normal operation



Timing diagram 2.2: Faulted operation

Still this first model was not able to explain all possible faults obtained by pulsed EMFI. Indeed, in [70] authors shown they were able to inject faults in DFFs that were not triggered by the clock signal; in their experiments the clock was forced to zero. Therefore, during their experiment, timing constraints cannot be violated since there is no clock and thus no timing constraints. However, set and reset signals are clearly at stake in this faulty behaviour. This evidence the existence of a broader fault model.

To go further, authors of [70] performed additional tests. They ran an AES on both FPGA and SoC. Then they monitored if the ciphertext was the expected one or a faulty one while shifting in time (all along the course of the AES) the occurrence of EM injection (using the same injection's parameters). This experiment highlighted that the probability to induce a fault follows a periodical pattern with a period equal to the clock period. This also highlights that the probability to induce a fault follows a periodic pattern with a period equal to the clock period, but the width of the timing window during which faults occur are

independent of the clock period. Therefore, it demonstrated that the timing fault model is might not the right model for EMFI. The occurrence of timing fault windows independent of the clock frequency combined with the fact that the content of DFF can be disrupted even when the clock signal is idle has led to consider that DFF should be the faulted component instead of the logic glue. This means that EMFI induces faults by disrupting the DFF input signals while DFF are switching. Thus, DFF are the most EMFI susceptible digital elements in an IC.

Those considerations have led to the so-called sampling fault model which states that the EM susceptibility of ICs is:

- Is not constant over time.
- Is periodic with a periodicity equal to the DUT clock frequency, F_{CK}.
- Is maximal during short time windows centered around the rising clock edges; these maxima being fixed by the susceptibility of DFF. These time windows correspond to time slots within which it is possible to disrupt the content of DFF by faulting their input signals.
- Is minimal the rest of the time.

Such a behaviour of the EM susceptibility defines the sampling fault model which is summed up by Fig.2.11.

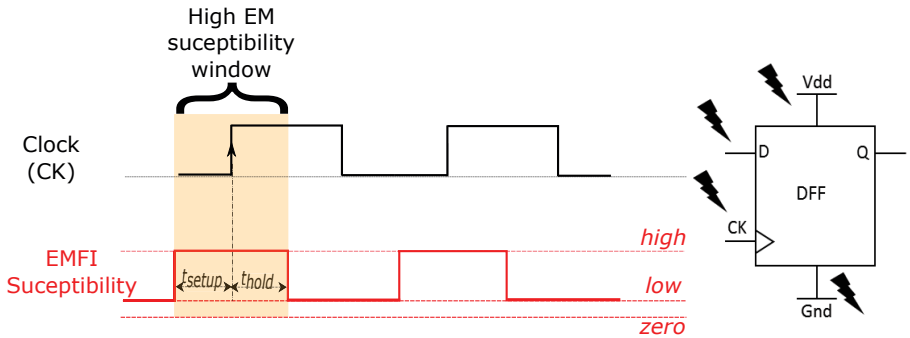


Figure 2.11: Sampling fault model

As a summary, two fault models have been introduced to explain the effects of pulsed EMFI. If they explain completely differently the occurrence of faults (one through an

increase of propagation delays, the other through the direct disruption of the switching process of DFF), they are not fully incompatible from a software point of view. Indeed, both fault models tell that EMFI induces bitsets and bitresets.

2.3.3 Fault Model Summary

In the preceding sections different fault models have been introduced for each EM coupling based fault injection mean.

Rowhammer flaw exploits many fast accesses to DRAM architecture to discharge some memory cells; the logical state of a cell being defined by both the charge of the cell and that of the bitline it is connected with. Therefore, discharging the cell induces sticking some bits of DRAM to either ’1’ or ’0’.

Considering Harmonic EMFI, its seems that the use of sine wave allows controlling the phase and modifying the frequency of freely oscillating structures by injecting directly a periodic signal in the common supply network of analogue blocks such as clock generator and jitter based TRNGs. It can also the bias TRNG which is a key element of many security structure.

Finally, considering pulsed EMFI, the latest fault model states that DFFs are the most susceptible elements. They can be set or reset by applying a sufficiently powerful EM pulse just before a rising clock edge.

2.4 Countermeasures

By mainly presenting the different flaws exploited by EM fault injection in the preceding sections one can get quite pessimistic about the design of resilient ICs against EM fault injection. However by knowing how EMFI interact with ICs and their embedded software, it becomes possible to design efficient countermeasures.

Countermeasures can be divided in two categories depending on the policy used to detect the faults. On one hand, one can think of either instrumenting or exploiting some

properties of the algorithm under test. The fault model gives clue on what to expect from the fault injection which is either data corruption or instruction corruption. Therefore, this first category will present technique that aims at enforcing CFG and monitoring the data flow.

On the other hand, one can also think of monitoring the DUT electrical conditions. This category is therefore relying on the hardware fault model and the use of sensors tailored to monitor the various side effects of EMFI. There are two main families of sensors the one that acts in the analogue world and the one working in the digital world. The first category is mainly focused on locking phenomenon detection and monitoring the timing constraint of flip-flops while the second category exploit DFF susceptibility to EMFI.

Finally, for the sake of completeness, two other techniques are also presented in this chapter which are namely infective programming and code polymorphism. However, such countermeasures are quite debatable, since in the case of the first one all implementation has been proven inefficient. In the case of the second countermeasure, it aims at making the targets more difficult to attack by adding noise in the EM traces. Therefore, an attacker could not use this channel anymore to extract information to ease the attack. Yet, it is quite hard to state whether or not the noise added by code polymorphism is efficient.

This chapter aims at describing some countermeasures related to the three categories but also at discussing of their own advantages and drawbacks. Rowhammer being a specific fault injection based on a design flaw of memories, this chapter ends by a section introducing some countermeasures against this attack.

2.4.1 Detection countermeasure

This section aims at presenting the different ways to detect that an IC undergo fault injection attack by looking at EMFI side effects on the computation. As presented mostly in the software fault model those side effects either affect the datapath or the control flow of the program. Therefore, the countermeasure presented in this chapter aim at enforcing those two metrics.

This section is organised as follows, first redundancy based technique will be introduced as it is the most widely used technique but also the most reliable. Then the use of parity code to detect data modification to finish with Control Flow Graph enforcement methodology.

2.4.1.1 Redundancy

Maybe the most widely used countermeasure against fault attack is redundancy. The name coming from the fact that this countermeasure relies on either doing the same computation twice or duplicating the data. The former one can be design both at hardware level or at instruction level while the second one is purely at software level. In a second time redundancy compares the two redundant data or computation result and states whether a fault occurs or not. Notice that in this chapter the policy taken after fault detection is not addressed.

2.4.1.1.1 Time and spatial redundancy To ensure that an algorithm follows the proper operation one can compute twice the algorithm under test. These computations can be done at two different instants, in this case we talk of time redundancy, which can be applied for both hardware and software case. On the other hand, in the specific case of hardware implementation one can use two hardware instances of the algorithm to compute the expected output and compare the result.

Such a methodology has the advantage of being simple to implement. However, in the case of time redundancy the time between when the fault occurs and when it is detected can be quite long. On the other hand, Spatial redundancy is quite heavy both in terms of space on the silicon and of power consumption of the target.

To palliate to the time redundancy drawback a specific solution has been design for the hardware case. This implementation is called the Double Data Rate architecture and has been introduced in [55], [56]. The key idea being to compute both computation almost at the same time, to that end the IP to secure is design to compute on both falling edge and rising edge. This enables for instance in the case of a cipher algorithm two ciphers on both edges and to compare the result at the end of the computation. Using such an architecture the fault detection happens at the end of the cipher operation without waiting

for a second cipher to be computed. Yet it does not solve the consumption cost induces by the two computations.

This idea of computing two computations at the same time can also be implemented at the software level. In this case one can take advantages of SIMD instruction as presented in [20]. SIMD stands for Single Instruction Multiple Data, it provides instruction that compute on a vector of data instead of one data. To enable such operation a specific set of register is often provided with a larger size than the one of the target therefore the extra space can be used to store a copy of the data and the same computation can be performed on both value within the same clock cycles. In [48] authors have widened the scope of such countermeasure by mimic SIMD register with regular register. Indeed, they work on a 32 bits architecture and implements an 8 bits AES so that the 32 register act as SIMD register with the following content: DATA / COPY /DATA2 / COPY2.

The gain of such design is twofold, first one can compute both ciphers at the same time therefore there is no latency between the two ciphers. Moreover, it enables to compare the two data at any time during the algorithm thus one can reduce the detection time. This technique of duplicating the data is closely related to the information redundancy introduced just after. Still, as in the hardware case the consumption is still high because of the two computations.

2.4.1.1.2 Information redundancy Information redundancy contrarily to the instruction redundancy introduce after aims at monitoring the dataflow of a program. As introduced before, one can use SIMD to fully duplicate the data used by the algorithm. But such approach is very memory hungry and does not fit for all targets. Therefore, a trade-off must be found between accuracy and memory footprint.

To reduce the overhead technique based on Error Correction Code *ECC* has been proposed. One can also notice that this approach was introduced before the one using SIMD. The theory behind Error Correction Code has been pioneer by Richard hamming in 1940. It was first developed to ensure that a transmitted message through a noisy channel wasn't altered. To that end the whole message is transmitted as well as an error code computed from the message. Then the receiver compute the error code using the message and com-

pare both code; if there are different the message (or the checksum) has been altered.

One can then model cryptographic algorithm as particular communication channel where the whole message is drastically modified. This implies that care must be taken when choosing an error code. Therefore, in this scenario the checksum as it contains data from the original message can be seen as redundant information and should be able to undergo the manipulation perform by the cryptographic algorithm. The advantages of using ECC is that the information is duplicated in a more compact format than the whole duplication of data. This compact format also has a cost which is only a part of the information is encoded which implies a loss of accuracy compared to the whole message duplication.

In the case of RSA, modular code [85] offers a pretty good trade-off between accuracy and overhead. For AES, polynomial codes seem to be better [14],[87],[45]

The use of Error Correcting Code enables to overcome the memory footprint problematic of full data redundancy, but at the cost of a lesser accuracy in detecting faults.

2.4.1.1.3 Instruction redundancy When talking about software level another form of redundancy, focused on instruction, has been introduced in both [8], [62]. This form takes place at instruction level which means that instead of computing once the same instruction, it is done multiple times (twice or thrice) and aims at protecting against pulsed EMFI. This is introduced in the first paper and refers to as instruction duplication or triplication.

The idea behind instruction duplication being that at least one of the two operations won't be faulted because of a too short timing window between the two instructions. Indeed, as it was presented in the platform presentation section, up to date platform are using pulse generator that require a certain latency between two shots. For instance the injection platform used in this thesis requires around $100ns$ after the trigger event to shoot. Moreover, the pulse generator can work at a frequency of up to some Hertz which is very slow compares to up to date microcontroller clock frequency (e.g. the targets considered in this thesis run at 64 or 80 Mhz). Then the result of the two instructions is compared and an alarm is risen if an error occurred. The use of triplication enables to correct the

injected fault since at least two results are required to determine which of the three results have been faulted.

In the second paper authors formalize the concept of duplication with a use case on the ARM Thumb-2 instruction set. To that end they introduce the notion of idempotent instruction, which means an instruction that can be directly duplicated as is since they do not modify the content of the register they use. For instance the instruction $movr1, r2$ if duplicated it won't alter the result of the instruction which is to put $r2$ content in $r1$. The second class of instructions are the instructions that can be re-written as a sequence of idempotent instruction. For example a secure version of $add \quad r1, r1, r3$ would be $mov \quad rx, r1;$ $movrx, r1;$ $add \quad r1, rx, r3;$ $add \quad r1, rx, r3$ where rx is an available register.

Finally there are instructions that cannot be secured since they cannot be expressed as a sequence of idempotent instructions. Then to secure a program the programmer only has to duplicate idempotent instructions. If they are non idempotent they can be break down to idempotent instructions and duplicate or if they can't be break down they stay as is. In [9] authors implement this strategy as a LLVM's pass to automatize the method. LLVM is a compiler framework formerly known as Low Level Virtual Machine, and a pass means an optimisation module that works on the intermediate representation. The main problem of such countermeasure is both the cost in terms of execution time doing the same operation twice and in terms of memory usage.

2.4.1.1.4 Redundancy summary Redundancy is probably the most used countermeasure against fault injection that targets a cryptographic algorithm. It is simple to implement, with the highest coverage rate and it can sometimes be automated at compiler level. It is quite versatile since it takes place at different levels of abstraction which enables to implement some variants depending on the target resources. Indeed, its main flaw being it is quite resource heavy from memory to hardware. Some trade-off have been found such as the use of Error Correction Code but it is not the only one and those trade-offs often sacrifice accuracy to reduce overhead.

2.4.1.2 Control Flow Integrity

The fault model chapter introduces the following software fault model where all possible instructions can be skipped or data corrupted to enable jump to unwanted target in a software. Therefore, EM fault is very likely to perform transformation on the Control Flow Graph. Method that ensure that a program is following the control flow graph at runtime are called Control Flow Integrity.

The typical scheme is the following [1]. For each indirect branch target in a program a unique id is computed. Then at each indirect jump a test is performed to know whether or not the address is a valid indirect jump target. In the same vein signature modeling technique proposed to compute a signature of a block of instruction, most commonly basic block, at compile time (or after). Then at runtime the signature is calculated either by the OS/virtual machine (for instance jvm) or a hardware watchdog. This dynamically computed signature is then compared to the static one. When this scheme is apply at instruction level, it is referred to Continuous Signature Monitoring [86].

2.4.1.2.1 Counter based CFI Still some pure software control flow graph monitoring has been developed such as [50] which targets smart card and does not rely on virtual machine or OS. The proposed counter-measure uses a counter to verify function's execution path at runtime. The counter behaviour being the following: before entering a function, it is initialised and then inside the function its value is compared with the expected value. If they are the same then the counter is incremented otherwise an error has been detected. The counter is also checked by the caller of the function when the callee returns. This schema has also been tailored to support if construction as well as loop. In this implementation the counter is playing the role of the signature used in the previous implementation. To avoid a huge overhead on the code memory and execution time the authors proposed the following simulation on C code to detect vulnerable part of the code. The idea is to test from all the points inside a function the effect of an arbitrary jump within the bound of the function. From this simulation C code line are classed according to their impact from the attackers point of view. If a line within a function leads to benefit earn by the attacker the function is marked as to be secured.

2.4.1.2.2 Shadow Stack and call graph integrity A subset of CFI is also possible if neither the memory footprint nor the addition of watchdog (either in software or hardware) is possible but at the cost of a lesser accuracy. This countermeasure focuses on a sub-graph of the CFG called the ”call-graph” instead of verifying all basic block. The ”call-graph” is a directed graph where all nodes are a function and each edge represents a call from one node to the other node according to the edge direction.

An efficient method to monitor this graph is using a so-called shadow stack[3]. The shadow stack works as follows: when in the prologue[4] of a function the caller address is stored in the shadow stack and when in the epilogue of a function this value is compared to the actual return address. If they differ either we have jump in another function and passed over its prologue or a fault have compromised the return address.

This technique does not enable a fine grain control flow integrity as the token technique presented before and as quite a long time before detecting the fault, but those lacks are compensated by its impact in both execution and memory footprint.

2.4.1.2.3 CFI based countermeasure Summary CFI countermeasures have the assets of being quite generic by enabling to protect cryptographic algorithm as well as privilege escalation attack. Its main drawback being its memory footprint which can be leverage at the cost of a hardware watchdog. Yet it is by nature focused on block of instructions (or function graph), making this family of countermeasure far less accurate than redundancy which also check dataflow.

2.4.2 Sensors based methods

2.4.2.1 Analogue sensors

Yet CFI is not the only side effect that can inform of a potential EMFI attack. Indeed, as introduced in the fault model section EMFI also have physical effect that can monitor at hardware level. Notably both harmonic and pulse EMFI relies on EM coupling which

[3] `http://www.angelfire.com/sk/stackshield/`

[4] the prologue of a program is the few assembly lines that settle the stack and register of the callee

is a physical effect that can be measured on a circuit. Hardware only detectors have the assets of relying on hardware fault model and thus on measurable physical effect. That is to say in the specific case of harmonic EMFI, the fault model states that phase signal can be altered, but they can also be monitored which the base principal of harmonic EMFI detectors. On the other hand, pulsed EMFI analogue sensors will aim at monitoring timing constraint on DFF.

This section is sorted in this order first sensors based on unwanted coupling is presented. Then sensors based on phase signal monitoring are presented. And this section ends on the pulsed EMFI specific sensors.

2.4.2.1.1 EM coupling sensor One of the analogue sensors that is tightly bind to EMFI effect is the one proposed in [42]. They propose to embedded in the circuits some sensors made of a LC circuit. This circuit having the assets of oscillating at a certain frequency, frequency which can be modified by coupling with another electromagnetic components, i.e. in our case the attacker probes. The LC frequency being:

$$f_{LC} = \frac{1}{2\pi\sqrt{(L-M)C}}$$

notice the coupling parameter M which is modified by the probes and the EM field generated during injection. Therefore, by monitoring the frequency variation of the LC circuits, it is possible to deduce that the coupling parameter has been modified, i.e. the circuits undergo an attack.

2.4.2.1.2 Harmonic EM fault model based sensor Sensor can also be used to monitor physical effect described by fault model. First the harmonic EMFI fault model which states that EM coupling is likely to lock oscillators to the phase and frequency of the injected EM wave. Two sensors are based on monitoring the phase lock between a ring oscillator and different architectures namely Phase Locked Loop and Hogge Phase detector.

Phase locked loop is a control loop which outputs a signal with the same phase as the input signal. It is composed of phase comparator which enables to measure the error

between the phase of the output signal and the input signal. This signal feeds a Voltage Control Oscillator (*VCO*) that generates the output signal. To enhance the accuracy of the PLL a first order low pass filter is inserted between the VCO and the phase comparator. One of the common use case of such structure is to generate multiple of a sinusoidal signal which will serve to generate various clock signal in a IC from the same external source. Therefore, there is a first coefficient before the phase comparator and another one in the feedback of the control loop. When the phase of the input signal is equal to the phase of the output signal, the PLL is in the locked states. Hogge Phase detector is a structure that is often used in Phase Locked Loop's phase comparator.

The principle of the sensors is the following: in the normal case after a certain amount of time the PLL (or Hogge phase detector) will be locked on the phase and frequency of the input Ring Oscillators and it stays locked unless an event modify the property of the RO. Therefore, if an attacker, uses harmonic EM waves to perturb the IC, this will change the phase signal of the monitored oscillators. This implies that either phase locked loop (see [60]) or Hodge Phase Detector ([18]) will not be in locked state during a certain amount of time. Therefore, if those components unlock it means that the circuits is under attack. However, this kind of sensor is quite heavy in terms of circuit footprints and it requires multiple detectors to cover the whole chip.

2.4.2.1.3 Pulsed EM fault model based sensor In the case of pulsed EM fault model, analogue sensors mostly rely on the timing violation. As a reminder, this fault model states that pulsed EM faults are caused by DFF's timing constraint violation. In [88] authors introduced the following scheme to detect those timing violations Fig.2.12. They propose to use a clock glitch detector to detect EM fault injection. In this scheme, a delay block is inserted between the clock signal and the clock input of a DFF that samples its clock signal. The delay block is set to be greater than critical path time and smaller than t_{ck}. By taking as example the timing diagram on 2.3 where everything work as expected. The flip-flop will sample a 0 and output a 0. Then on the figure 2.4 where the IC undergo a pulsed EMFI attack, the DFF will output a 1. Indeed, the extra delay induce by EMFI will make the DFF to sample during the rising edge of the Clock signals, this will triggers the alarm.

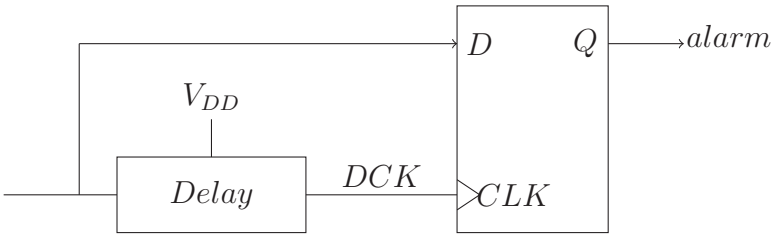
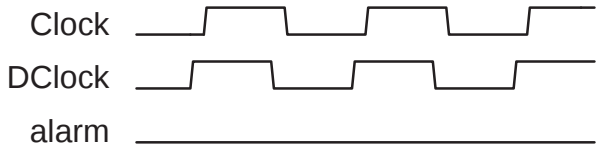
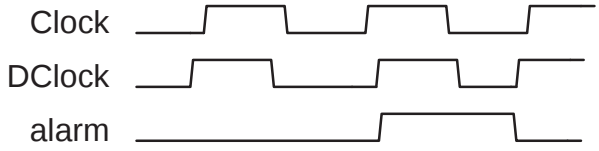


Figure 2.12: Glitch detector implemented in [88]



Timing diagram 2.3: Normal operation of the sensor



Timing diagram 2.4: Faulted operation of the sensor

The main drawback of analogue detectors is that they are quite difficult to set up as they rely on physical effect. For instance one can cite the delay of the delay bock in the [88]. Moreover, phase detectors have a high surface footprint which can be problematic on some devices. To circumvent this problematic, digital versions of such kind of sensors have been design.

2.4.2.2 Digital sensors

Digital sensors mainly rely on the pulsed EMFI hardware fault model and only aim at detecting pulsed EMFI fault injection. Recently[5] a fully digital version of a sensor based on measuring event relying on DFF timing constraint has been developed. The architecture of the sensors is based on a delay chain, which is commonly used to propagate a carry value see Fig. 2.13. This delay chain is composed of buffers and each buffers output is sampled

[5] i.e. year 2018

by flip-flop. The input signal of this sensor being the clock signal the circuit. Therefore, if the delay chain is tailored to have a size of half a clock period the content of the flip-flops will be either $000...1111$ or $111...0000$ depending on whether we are sampling a rising or falling edge. Hence, assuming faults are following the timing sampling model, then the clock edges will be shifted in time modifying the content of flip-flops to this $110...111$ or $001..111$ (with an offset of 0 or 1).

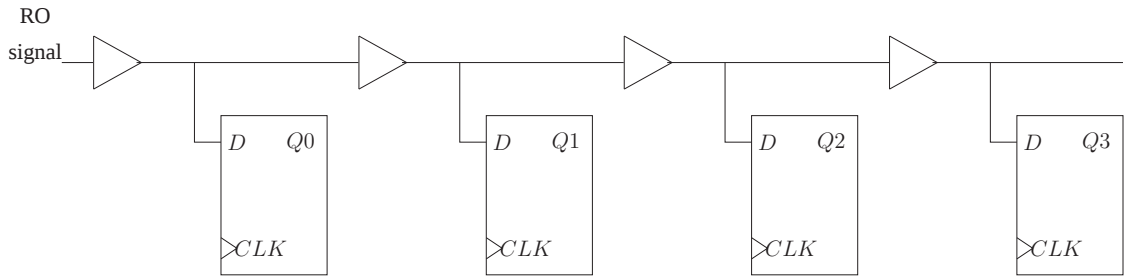


Figure 2.13: Sensor structure

The detector then acts like this if the pattern stored in the flip-flop is the same as the one without fault there is no fault otherwise an alarm signal is rise.

Nonetheless there is another fault model in the specific case of pulsed EMFI, which name is sampling fault model. This fault model states that DFF are more likely to be perturbed by EMFI than the other components. Therefore, the use of DFF as a sensor has been studied in [33]. Notice that it is also the first fully digital sensors to detect EM fault attack. The sensor is composed of two cells that are sensible on both clock's edges see Fig. 2.14. As the considered fault model states that the most susceptible part of an IC are the DFF and more specially on clock rising edges using a clock and its complementary enable to have a longer susceptibility window for the sensor. Indeed, due to the "not" logic gate the DFF1 and DFF3 are sensible during falling edges while DFF0 and DFF3 are sensible on rising edge. If we consider only one cell Fig. 2.14 the two flip-flop are initialised with different values so that "xoring" the two outputs should be equal to one. Otherwise, the signal will be zero. To test all the different combinations the second cell differs in initialization value. Therefore, all possible transitions $1 \rightarrow 0$ and $0 \rightarrow 1$on both clock edges are monitored. The end of the sensor is then the and operation on the xor of cells output Fig.2.15.

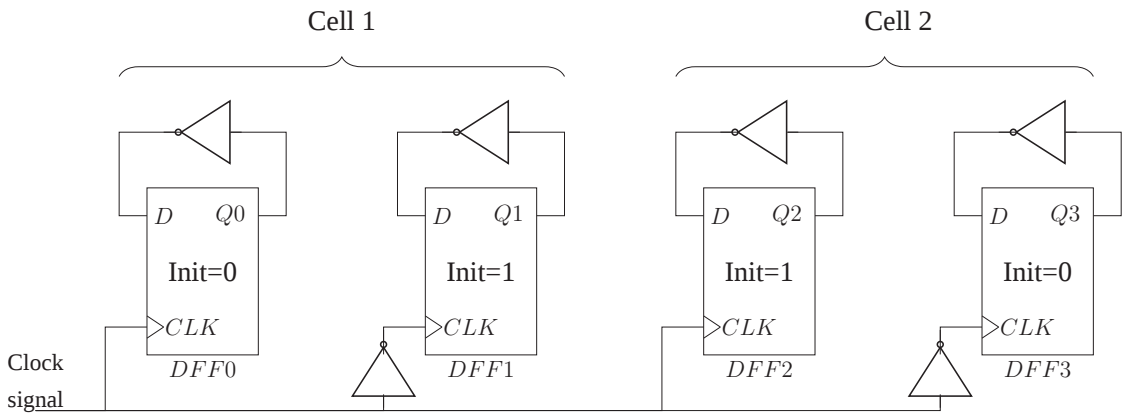


Figure 2.14: High susceptibility cell[33]

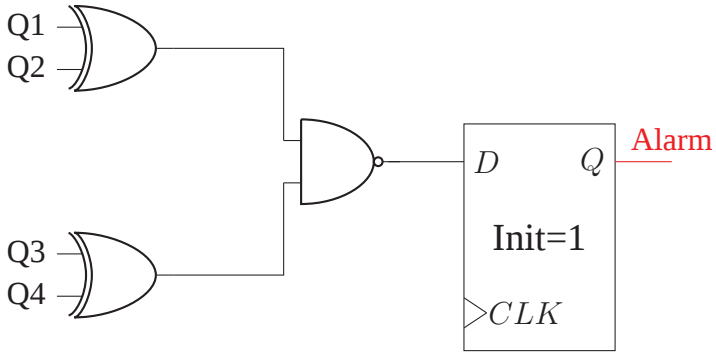


Figure 2.15: Sensor [33]

This detector has the assets of being quite light in terms of hardware footprints. However, it has many false positive, i.e. the sensors detect a fault at a position while the probe is at the extreme opposite.

2.4.3 Rising difficulty countermeasure

The previous section presented how to detect fault injection. However, detecting is not the only possibility to prevent an attacker from bypassing the security.

Another policy could be to make the life of the attacker harder. Different approaches can be taken, one can try to make the cipher text unusable from a differential fault attack

point of view if a fault occurs. It is the idea behind infective programming. The other methods rely on the following observation: to generate a usable fault an attack should know two things when and where to shoot. The time information being contained in the power consumption of the algorithm (and therefore in the EM analysis traces). The aims of this second countermeasure will be to mask this consumption by adding noise. The main flaw of this strategy being that measuring the noise addition is tedious problems that depends on ICs design.

This section is organised as follows first infective programming idea is presented and then a common implementation of noise addition is analysed.

2.4.3.1 Infective Programming

The purpose of infective programming is to turn faulted data into unusable information for an attacker. The key idea is to remove the link between the faulty cipher text and the cipher text acquired by the attacker, so that Differential Fault Analysis cannot be applied. To that end if a fault is detected some random values are added to the faulty cipher text. Different implementations were provided in [73], [34] and are composed in two parts : detecting a fault tentative (mainly by comparing result see the section Redundancy) and adding randomness to the faulty cipher text. Yet most of infective programming methods have been shown to be vulnerable to attacks [52], [10]. As of today, there is no viable infective programming scheme.

2.4.3.2 Code Polymorphism

Instead of reacting in response to a fault, one can also rise the difficulty of making a fault on the algorithm. The main drawback of this method is that its efficiency can barely be measurable. The common strategy to add noise at software level is the use of code polymorphism. It relies on using different versions of the algorithm to perform the same computation. In other words, each time the function is invoked the code is organised in a different way but performs the same algorithm. The idea is to disable an attacker to pinpoint in the EM emissions the part related to the protected algorithm. Thus, an attacker cannot use EM emissions to have some insight about when and potentially where

to perform the injection.

This strategy can take place at compile time or at runtime. Mainly the different versions proposed to shuffle either at basic block or at instruction level. The other idea is to add a random number of ”useless” instruction in the code to protect (to hide) the consumption of the algorithm in the one caused by the useless instruction. Polymorphism also implies to be able to change the register used by an instruction which is called register renaming between each call to the function. Finally, one can also replace instruction by one that are semantically equivalent or by a sequence of code that compute the same result using different operation sequences.

To implement these kinds of countermeasure different methods have been explored. Code polymorphism as a security mechanism against fault attack has been studied in [4] [2] and [25] or [13]. The authors of [4] only shuffle basic block using runtime code generator. In [13] the authors proposed the same sort of polymorphism, but they use a dedicated IP which takes place between the processor and the instruction cache to shuffle basic block. This method has the advantages of being faster and lighter in terms of memory. In [2] authors proposed to shuffled at runtime both instruction and Basic Block of a program as well as register renaming and semantically equivalent code. To that end, they embedded in the program a runtime code generator which relies on static analysis (i.e. compile time) information. In [25] authors embedded some small compiler element so-called ”compilette” that works on a pseudo assembly like language to generate the different version of the program at runtime. Those different versions include register renaming and semantically equivalent code fragment as well as the addition of dummy instruction to induce unsynchronised traces between each run.

2.4.4 Rowhammer countermeasure

Rowhammer is not relying on an external EM field and its fault model being very specific explains why this section dedicated to it. Rowhammer is a flaw that targets data (and instruction) contained in DRAM memory. It can stick memory cells to either at ’0’ or ’1’ regarding the bitline logic by discharging cells in DRAM. To do so it requires accessing

memory (DRAM) the fastest as possible to bypass refreshing property.

As before this subsection is organised with the same top to bottom approach. Rowhammer researcher and designer of countermeasure are playing a game of cat and mouse where a new countermeasure is designed then a new attack to bypass it is done and so on. This making countermeasure presented in some hierarchical order.

2.4.4.1 software countermeasure

From Rowhammer fault model one can notice that direct access to the DRAM can be problematic and that cache is by nature a way to block Rowhamer as it prevents to fetch the data from DRAM if they have already been readen. Moreover, the first Rowhammer attack publish by Google project zero team presneted at Blackhat 15[6] was relying on the x86 instruction ”clflush” which enables to flush the cache from user space. Therefore, most software countermeasure aims at disabling the user from cleaning or bypassing the cache.

2.4.4.1.1 Cache based countermeasure The first countermeasure to rely on such mechanism was to remove from user space instruction that can flush the cache. However, using cache side channel it is possible to find a cache eviction pattern that can be exploited to lead attacks as demonstrated in [38]. Furthermore, it is sometimes required from performance point of view to be able to bypass the cache mechanism. One can think of the use of DMA in server ([51] [84]) or some instructions such as ”memcpy” used to initialise data that won’t be used in a near future. Removing those functions ends up with data that pollutes the cache and thence lower the performances.

Yet knowing that code that trigger Rowhammer relies on specific instruction might enable to detect malicious code. This was proposed as MASCAT countermeasure ([43]), where authors use static analysis of binary to detect code pattern likely to trigger Rowhammer. Where static analysis means, it does not run the targeted code to run.

[6] https://www.blackhat.com/docs/us-15/materials/us-15-Seaborn-Exploiting-The-DRAM-Rowhammer-Bug-To-Gain-Kernel-Privileges.pdf

Looking at cache eviction policy can also be countered. In ANVIL [5] author proposed to use CPU performance counter to monitor the amount of cache miss done by a program. If the number of cache miss is superior to a threshold then the accessed addresses that have cause cache miss are logged and nearby rows are refreshed. Still authors of [37] have designed a version of Rowhammer that is running in an enclave (e.g. using SGX instruction set for x86 architecture). Note that an enclave is an environment which runs with user privileged but is isolated from the other applications by having its memory encrypted and is excluded from CPU performance counter. Therefore, using enclave and cache policy knowledge it is still possible to run Rowhammer without being detected.

2.4.4.1.2 Memory access based countermeasure Since Rowhammer is based on specific memory access another approach than monitoring the cache is to directly monitor DRAM. First, to enable Rowhammer attacker requires accessing two different addresses to bypass rowbuffer (cf rowhammer subsection). The most efficient pattern being called the double-sided Rowhammer where the targeted wordline is just between the wordlines that contains the two address the attacker is reading. Another point is that the addresses should be in the same DRAM bank for the double-sided Rowhammer to happen. Therefore, monitoring the DRAM access pattern seems to be a relevant countermeasure, yet in [37] author explained how to perform Rowhammer using one single address.

Therefore the memory access should be monitored from another point of view. The CATT countermeasure [17] relies on the Memory Management Unit to isolate physically the kernel space memory from user land. This makes the assumption that Rowhammer is only usable on memory and that the only attack vector is to flip a bit in page table entry to make it point to a kernel page. However, it can be used to perform instruction skip see [37], and CATT isolation does not apply to buffer that are both owned by the kernel and the user space (for instance video buffer). Those kinds of buffer can then lead to privilege escalation as demonstrated in [21]. As we can see, all pure software countermeasures have shown vulnerable to a variant of Rowhammer.

2.4.4.2 Rowhammer Hardware countermeasure

From a hardware point of view the main objective of Rohammering DRAM is to lower memory cells' refresh rate to induce bitflips. Based on this observation the first countermeasures that were provided were to rise refresh rate and to use Error Correcting Code (ECC) [46]. However, the DDR4 example (which use ECC) shows that it was not sufficient. Furthermore, some recent research show how to generate bit flips despite ECC [24]. Moreover, some proofs of concept against DDR3 with a double refresh rate have also been crafted [7].

Therefore policy to refresh potential victim row has been developed. For instance in PARA ([46]), the approaches is the following: it counts the activation of a wordline and compute a probability to refresh neighbours faster according to this counter. The device in charge of counting the activation taking place at the memory controller level. By doing this it triggers row refresh before bitflips and thus should mitigate Rowhammer deficiency. In the same vein Target Row Refresh increase a counter for each memory row adjacent to an activated row. When a threshold is reached, the row is refreshed. Still, DDR4 with TRR enables has been shown vulnerable in [8].

As of today it seems that all Rowhammer countermeasure can be bypassed using one of the variants of Rowhammer. Still, the requirement to lead a successful attacks keeps increasing (running in an enclave, knowing the cache policy and using single sided Rowhammering).

2.4.5 Countermeasures Summary

The preceding section presented the different strategy used by software engineer or designer to protect an IC against EM based fault injection.

The first conclusion in the case of Rowhammer is that all countermeasures tries to rise

[7] https://www.blackhat.com/docs/us-15/materials/us-15-Seaborn-Exploiting-The-DRAM-Rowhammer-Bug-To-Gain-Kernel-Privileges.pdf

[8] https://gruss.cc/files/us-18-Gruss-Another-Flip-In-The-Row.pdf

the level of difficulty to trigger the DRAM effect but none is completely able to disable the attack.

In the case of the external EM field based attack two strategy are mostly used and sometimes works in pair. The strategy being either detection or rising the difficulty of leading an attack. Detection strategy are all based on the fault model and therefore are specific to EM fault attacks. The sensors developed can either be fully analogue or digital. The main flaw of analogue sensors being that they require specific setting which can be troublesome. Looking at software level the proposed countermeasure often implies high memory cost as well as performance overhead. Still, by focusing on protecting against a software behaviour instead of physical property they have the assets of functioning for other fault family.

Chapter 3

EM Fault Injection Susceptibility Criterion

This chapter presents a method to reduce the time taken by pulse EM fault injection evaluation. The biggest flaw of such attacks being its number of parameters. While having numerous parameter enable to fine tune the EM injection (and thence the fault obtained) can be seen as an asset, when it comes to security characterisation it becomes a drawback. This paradox is due to the lack of criterion to reduce the combinatory between each parameter. The criterion designed in this chapter focuses on one of the most time-consuming operation of EMFI which is the whole scanning of the DUT for each set of pulse parameter.

EMFISC criterion relies on ranking positions on the circuits by their susceptibility against EMFI. Where the susceptibility is defined according to the sampling fault model as well as antenna reciprocity property. These two metrics being computed from EM analysis traces which are considerably faster to acquire compared to injection. ***This first criterion has been published at the Smart Card Research and Advanced Application Conference taking place in 2017 [54].***

To enhance the accuracy of the criterion a refinement is proposed. This refinement is based on clustering position according to their EMFI susceptibility. This having the assets of taking into account the size of the probe compared to the size of the circuit as well as potential bias between EM analysis map and injection.

Contents

3.1 Introduction

Despite being proven efficient, pulsed EMFI still suffers nowadays from a major flaw. Indeed, when it comes to security characterisation, performed with timing constraint, pulsed EMFI suffers from a too wide set of control parameters. As a remainder, the list of parameters the evaluator has to settle is :

- V_{pulse}, the pulse amplitude, that can be set between $50V$ and $400V$ by step of $0.1V$ using our EMFI platform,
- PW, the pulse width that ranges between $6ns$ and $30ns$,
- the pulse polarity that can be either positive or negative,
- the position of the EMFI probe above the IC surface that can be set with an accuracy of $10\mu m$,
- the choice of the probe, among a set of 3 different geometries (cylindrical probe, conic probe, U-shape probe; all declined with different sizes) and electrical characteristics (number of loops, loop spacing, wire diameter ...),
- the moment at which is delivered the EM pulse with respect to the DUT operation. Given a time reference (a triggering signal), this moment can be delayed up to 1s with our platform.

Considering this list, it is obvious that in a reduced evaluation time, the characterisations of DUTs robustness against EMFI could only be very partial, since exhaustive search cannot be performed. As a result, the validity of such characterisations is only based on the experience of the evaluators.

However one can observe that this flaw is not specific to pulsed EMFI. It is in fact quite general to all fault injection means. The complexity of ICs robustness evaluations against laser fault shots suffer from the same flaw. Indeed, many control parameters are involved in the setting of laser injection. Only few fault injection means are characterized by a lower complexity. Among them, one can cite the use of voltage pulses on the power or ground pads. The complexity of such injection mean is naturally lower because the absence of any spatial or positioning parameter of the injection tool. Furthermore, the complexity of such technique has been further reduced using genetic algorithm in [74].

Despite this paper, there is to the best of my knowledge, no other research focusing on reducing the combinatorial complexity of fault injection and more particularly in the case of EMFI

Within this context, this chapter aims at introducing a criterion developed during this thesis to reduce the complexity of EMFI tests. And thus to enhance the quality of robustness evaluations against EM fault injection. The problematic that has been addressed by this criterion is the reduction of spatial complexity using the duality [1] of the EM antenna on a circuit.

The problematic was not chosen randomly, it comes from the analysis of which operation is the most time-consuming during EMFI security characterisation. From this analysis two operations appeared to be far in front of the others. The first one is the operation of reflashing the IC in case of no response from the DUT or latchup. As the root cause of IC no response is not known it is more robust to reflash the DUT with the code under test since these behaviours might have for side effect to corrupt memory content, i.e. the code being tested. If it is the case it might induce the evaluator in error because of false positive induced by prior shoots. However, this operation is quite costly and often has to be done for each repetition on the same X, Y, Z position and for various positions depending on the different parameters combinations. The second one is scanning the whole die for **all the different maps**. That is to say for each parameter change we have to test all the positions even the ones that will not lead to a fault for any pulse parameter, therefore the time penalty is quite obvious. Moreover, reducing the area to test could also impact the number of no response position and thus reducing reflashing. Therefore, reducing the scan area is one of the most interesting problematic for an evaluator to be able to characterize efficiently a DUT.

This chapter is organized as follows. First, the physical properties as well as the raw materials on which the criterion is designed are presented. Then the two metrics used by the criterion are introduced. Because the second property is very related to EM reverse engineering, a brief state of the art about such methods is provided. Here, reverse engineering means either finding position above the IC surface at which the EM leakage is

[1] Duality refers to both emission and reception of EM field

the strongest or IP block location. After these recalls, the second part of this chapter introduces the criterion and validates it. Then the criterion results are analysed and debated which leads to a final part introducing a potential enhancement.

3.2 Physical considerations behind the EM Fault Injection Criterion

Why first focusing on EM techniques allows locating functional blocks and / or positions above which the EM signal can easily be exploited to extract secret information using Side Channel Attacks ? Two main arguments explain this choice. The first one being that evaluators aim at inducing exploitable faults for instance the Differential Fault Analysis (DFA) and thus aim at corrupting either data path and/or the IPs computing intermediate values of some cryptographic algorithms. The second one is related to the richness of the EM side-channel that contains all the information required to visualize data paths or locate functional blocks.

As an illustration of the information richness carried by EM waves let's consider the following case. We are in a white box context where the evaluator knows the code (Alg.3) executed by an IC and is observing the EM trace of Fig.3.1 collected above this IC. Let's suppose the evaluator wants to characterize the memory access part of Alg.3 which are LDR/STR/LDR instruction. He can first observe the 9 peaks related to the first ADD operation. Combine with instruction timing, given by the instruction set, it enables to identify the timing window of the part of the code that is under study. Then by ensuring that the 11 peaks caused by the 11 ADD instructions are executed he can estimate at which sample on the EM traces the memory pattern is executed. As one can observe there are more than 11 peaks after the memory access which are related to the epilogue of the function under evaluation. This simple example gives a direct demonstration that the EM side channel enclosed all necessary timing information.

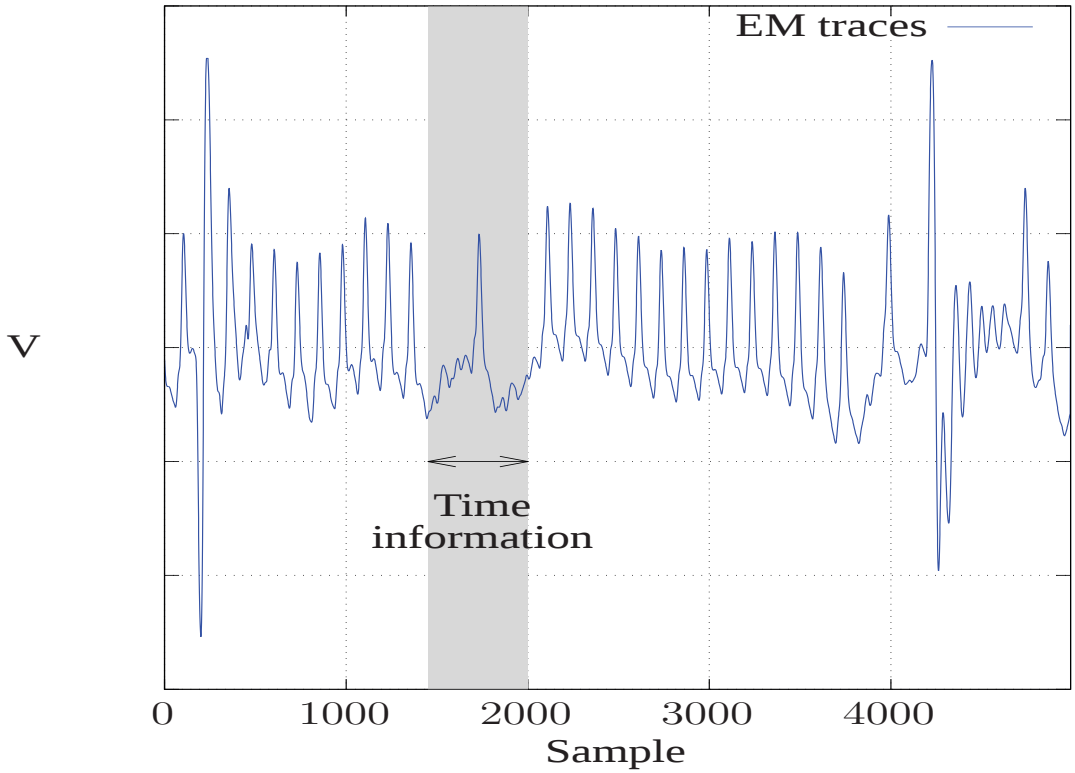


Figure 3.1: EM traces example

Algorithm 3 Pattern (AddrSRAM1 (R0), AddrSRAM2 (R1))

1: ADD R0,R0,#0; 9 times
2: LDR R2,[R0]; read AddrSRAM1
3: STR R2,[R1]; write at AddrSRAM2
4: LDR R3,[R1]; read back AddrSRAM2
5: ADD R0,R0,#0; 11 times

In addition to timing information, spatial ones can also be recovered from the EM side channel by comparing EM traces acquired at different positions. Among the hundred of techniques to compare EM traces, the most intuitive being undoubtedly: the closest of an active area the EM probe is the stronger the collected EM signal is. Therefore, analysing a set of EM traces constituting a complete EM scans of a DUT, using either temporal or frequency analysis tools, must allow localizing data paths and /or functional blocks involved in the course of any algorithm.

Anyway, among the most wanted information while developing a technique to guide EMFI tests, one can identify the EM susceptibility of each coordinate of the IC surface; the later being defined as its ability to collect EM waves generated above it. Thus, a position with a high EM susceptibility is a good entry point to inject parasitic currents in the DUT and thus a greater chance to induce a transient fault. This quantity is directly related to the quality of the coupling between the EM antenna and the IC and therefore, for a given antenna, to the electrical and structural properties of the IC in the close vicinity of the considered position. However, estimating the local EM susceptibility of a position is a hard problem.

A direct approach to estimate the EM susceptibility of a position, and more precisely its magnetic susceptibility, is to measure its emissivity considering that the duality/reciprocity of passive antennas (they can be either used for emission and reception). This notion is the topic of section 3.2.1. However, EM probes for near field scan are designed to increase both the signal-to-noise ratio and the spatial resolution while EM injectors are designed to deliver an intense magnetic field variation in the lower volume as possible.

EM probes and EM injectors have thus different form factors and different electrical characteristics and these differences have to be considered to transpose measured EM emissivity values into EM susceptibility values. This transposition requires the complete characterisation of EM probes and EM injectors which is a difficult problem and a tedious task. This problem is not addressed in this thesis. Indeed, it has been decided to workaround it by collecting emissivity values using the same EM injectors knowing this approach gives lesser accurate value from an EM reverse engineering point of view but more trustable results in terms of EM susceptibility.

Within the specific context of EMFI tests, considering only EM susceptibility could not be sufficient. Indeed, evaluators aims at inducing faults into data paths or functional blocks involved in the execution of the target algorithm. Therefore, the EMFI susceptibility criterion cannot be only the EM susceptibility. It must allow identifying hotspots above ICs with a high EM susceptibility and EM emissions related to the course of the target algorithm.

At that point, one may wonder if performing an EM near field scan before performing EMFI tests is a cost-efficient approach. Well in practice, using our bench, 8 hours are spent to perform a complete EM near field scan of an IC surface of $20mm^2$. These 8 hours are to be compared with the 3 days spent to perform a single EMFI scan of the same surface in EMFI mode with a single set of parameters. So spending 8 hours one time to reduce the area to be scanned during EMFI tests could be very cost-efficient if several sets of EMFI parameters have to be considered, which represent the most common case. It remains at developing the desired EMFI susceptibility and this is the topic of the next sections.

3.2.1 EM Coupling, Emissivity and Susceptibility

The main line of our approach being defined, this section focuses on the link between EM emissivity and EM susceptibility. To begin the discussion, let us consider Fig.3.2 and Fig.3.3.

Fig.3.2 gives the basic modelling of what happened when EM radiations of a DUT are collected with an EM probe. The left part of this figure models a part of the DUT as an active antenna (an emitting antenna), i.e. the injection case. On the other hand, the right part models the EM probe as a passive antenna delivering u_{probe} to a low noise amplifier or digital sampling oscilloscope, i.e. a side channel scenario. Between these two antennas one can observe the mutual inductance $M_{x,y}$ modelling the EM coupling between the considered part of the DUT and the EM probe.

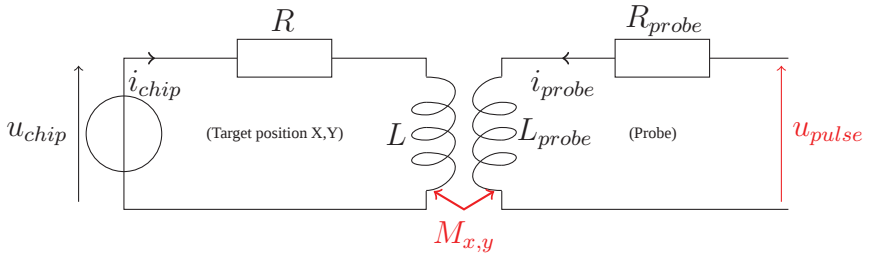


Figure 3.2: Lumped element model of EMFI applied to a DUT

Fig.3.2 depict the basic modelling of what happened when EM injection is performed above a part of a DUT. Similarly, to Fig.3.2, the left part models the DUT as an antenna and

the right part as an active antenna receiving a voltage pulse u_{pulse} on its input. Between them, the same mutual inductance is used to model the EM coupling if and only if the same antenna is used.

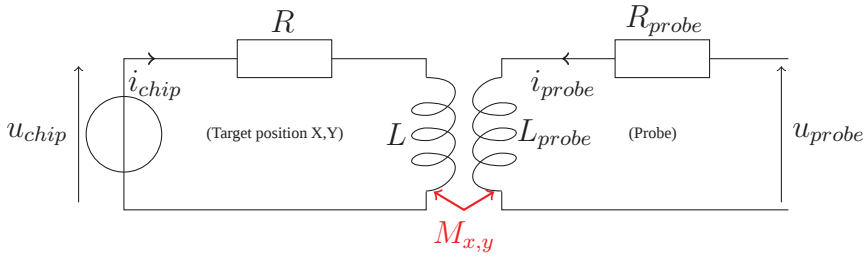


Figure 3.3: Lumped element model of a DUT EM analysis.

These two figures are really similar. The only difference is that in case of EMFI a strong voltage pulse u_{pulse} is applied to generate a strong variation u_{chip} of the internal supply voltage, while in case of an EM analysis, an image, u_{probe}, of the DUT activity, u_{chip}, is collected. This difference leads to eq. 3.1 and 3.2 showing that in case of EMFI, u_{chip} is mainly imposed by the derivative of the current flowing in the EM injector while in case of an EM analysis, u_{chip} fixes the collected u_{probe} variations.

$$u_{chip} = Ri_{chip} + L\frac{\mathrm{d}i_{chip}}{\mathrm{d}t} + M_{x,y}\frac{\mathrm{d}i_{probe}}{\mathrm{d}t} \tag{3.1}$$

$$u_{probe} = Ri_{probe} + L_{probe}\frac{\mathrm{d}i_{probe}}{\mathrm{d}t} + M_{x,y}\frac{\mathrm{d}i_{chip}}{\mathrm{d}t} \tag{3.2}$$

From these two equations it is obvious that an EMFI and EM analysis performed with a same EM probe (or EM injector) placed at the same position above the DUT shares the same EM coupling conditions which are modelled by the mutual inductance $M_{x,y}$. These equations also confirm the soundness of using EM analysis to estimate the quality of the EM coupling but also sustain its use to compare the quality of the EM coupling at different positions above the DUT. However, in the later case, it is assumed that i_{chip}, the current flowing in an area of the DUT is roughly the same anywhere. In practice this is not exactly the case as some functional blocks remain inactive during the execution of an algorithm. Yet this is not an issue in our case since the criterion aims at comparing position at which

a current is flowing during the target algorithm execution. The assumption thus reduces to consider that active functional blocks are characterized by roughly identical current consumptions.

From all the preceding, we can thus consider that EM analysis traces collected at coordinate (x, y) above the IC contain all required information to estimate the EM coupling between the EM probe / EM injector and the DUT during an EMFI, if and only if the EM probe used for collecting the traces is the same as that used to perform the EMFI. In the present context, this also means that an area of the DUT characterized by powerful EM emissions is very likely to receive well the energy of an EMFI pulse. Where ”receive well” means the yield in terms of power transmitted by the injector to the DUT is high.

Using the above conclusion our problematic, consisting in finding the positions with the highest EMFI susceptibility, can be redefined as finding the positions with the highest EM radiations. However, some simple experimental tests have shown this is not sufficient. Two reasons explain this.

First, a position with a high EM susceptibility is not necessarily linked to the course of the algorithm the evaluator aims at faulting. Indeed, there are functional blocks that are always-on. Among them, one can cite for example the internal clock generator or voltage regulator. Perturbing these blocks is not of great interest since this often leads to quite global and unworkable faults. It is thus necessary to discriminate positions with a high EMFI linked to the target algorithm to those with no relation with it.

A second reason explaining why finding positions with the highest emissions is not sufficient to disclose EMFI hotspots is the frequency content of the EM emissions. Indeed, according to the EM sampling fault model, the logic gates with the greatest EMFI susceptibility [70] are the D-type-Flip-Flops which toggle at the clock frequency. Thus, one must focus on positions related to potential radiations of these DFFs which are clocked elements. That is to say positions with high EM emissions at the clock frequency (or its harmonics) should be privileged with regard to positions with lower emissions at this frequency.

3.2.2 data path extraction

3.2.2.1 EM reverse engineering

After having explained why positions with high EM emissions at the clock frequency should be privileged, this section focuses on finding positions with EM emissions related to the execution of target algorithm. High EM emissions at clock frequency are not necessarily due to an electrical activity induced by the execution of the target algorithm. Many reasons explain this. First there are always-on parts in an IC and second the clock is broadcast on a large DUT area.

As the ElectroMagnetic Fault Injection Susceptibility Criterion (*EMFISC*) relies on specific part of the circuit localisation, in other words finding the DFF exploited by the targeted algorithm, a state of the art about IP localisation is recalled. Several techniques t-test TVLA [82], pearson correlation [80], Anova NICV [15], BCDC [31], Spectral Coherence [29], linear regression, have been proposed to localize positions at which are performed a computation in an IC. All share one of the two following principles:

- principle 1 : compare two sets of EM traces acquired when the algorithm (or functional block) is active and when it is not.

- principle 2 : compare sets of EM traces acquired when the algorithm (or functional block) is active but processes different inputs.

Despite this common aspect, the techniques available in the literature differ in how sets of EM traces are compared. These differences are usually explained by the author's goal.

In t-test TVLA [82], pearson correlation [80], Anova NICV [15], the goal is to find the (X, Y) coordinates at which the leakage of a cryptographic algorithm is the strongest (the easiest to exploit). Because side-channel leakages occur on really short time intervals, most of the proposed techniques performs vertical comparisons of EM traces (an EM trace being a vector $O = [o_1, ...o_n]$ of n successive EM observations) using a side-channel distinguisher among the t-test [82], Pearson's correlation [80], the analysis of variance [15]. Such an approach results in a trace (a vector) of the considered statistic ($T = [t_1, ...t_n]$ when using the t-test, $\rho = [\rho_1, ...\rho_n]$ when using the correlation, etc.) highlighting when

there are leakages if any.

The other solutions which are correlation [80], coherence [29], BCDC [31] follow an horizontal analysis of EM traces. This means that rather performing n comparisons of observation samples, several vectors O of observations are directly compared. This can be done using a correlation [81], a statistical distance the BCDC [31], a statistical similitude the coherence [29] or linear regression [49]. At the end of the comparison a real value is obtained scoring in average the difference between the vectors.

If the vertical approach is well suited for side-channel leakage detection in view of applying a vertical side-channel attack, it seems less indicated for reverse engineering or for our purpose. Indeed, in this approach all time aspects are disregarded and it seems difficult to find positions with EM emanations at the clock frequency induced by the execution of the target algorithm. A horizontal approach thus seems more indicated. Let us analyse in more detail the formerly proposed horizontal techniques.

Among the techniques adopting a horizontal approach, the one proposed in [81] uses correlation to disclose positions involved in a computation. To that end authors do propose to first perform a full scan of the chip to gather EM emissions. Then the traces collected at the different positions are compared using correlation. This implies that a correlation map is computed for each (X, Y) position. This map shows the correlation between traces measured above this position and all other positions. Positions that have a high correlation coefficient are then assumed to be observations of a same logical activity source. According to this assumption, authors finally propose to gather maps obtained at different (X, Y) coordinates in clusters according to the source they highlight. This is done by applying a 'two-dimensional cross-correlation' between each pair of maps; the obtained correlation value being compared to a threshold and if greater, the two maps are gathered in a same cluster.

[79] introduces also a horizontal reverse engineering technique to locate where computations take place in a DUT. It is based on an estimate of the Signal to Noise Ratio. In this paper, the SNR is estimated by computing an indicator called SNI (Signal to Noise

Indicator):

$$SNI = \frac{\sum\{|O_{active} - O_{inactive1}|\}}{\sum\{|O_{active1} - O_{inactive2}|\}} \tag{3.3}$$

Where O_{active} is an EM trace collected when the functional block to be discovered is active and $O_{active1}$, $O_{active2}$ are EM traces acquired when this block is inactive.

With such an approach, authors were able to locate the positions of an AES Sboxes implemented in a FPGA with a great accuracy. Unfortunately, this horizontal approach fully works in the time domain and does not allow considering frequencies characteristics of the activity contrarily to the one described in the next paragraphs.

Another horizontal approach has been proposed in [28]. The latter is based on Magnitude Squared Coherence. Magnitude Squared Coherence (MSC) is a tool allowing estimating the similitude between two time domain signals, even if it works in the frequency domain. As a remainder, MSC between two time domains signals $s_1(t)$ and $s_2(t)$ is computed using:

$$MSC_{s_1,s_2}(f) = \frac{|psd_{s_1,s_2}(f)|^2}{psd_{s_1,s_1}(f) \cdot psd_{s_2,s_2}(f)} = \frac{|C_{s_1s_2}(f)|^2}{C_{s_1s_1}(f)C_{s_2s_2}(f)} \tag{3.4}$$

MSC at frequency f takes value in the $[0, 1]$ where 0 fmeans the two signals do not share any characteristic at the frequency f, while 1 means that the behaviours of the signals at this frequency are exactly the same. In practice, the MSC values between two frequency values are averaged to get a score indicating how much the signal waveform are similar.

In [28], authors propose the use of the Magnitude Squared Coherence to rank different runs of the algorithm by the DUT; each run being performed with different data. The idea is thus to localize parts of the DUT manipulating the data by identifying positions with data depending on harmonics, and thus a low level of coherence between runs. However, to give more importance to powerful harmonics, the coherence at each frequency is weighted by its relative amplitude with regard to the most powerful harmonic in a bandwidth (BW) of interest. This is done using the Weighted Global Magnitude Squared Incoherence criterion.

$$WGMSI = \sum_{f \in BW} \frac{1 - Coherence_{s_1,s_2}(f)}{nf} \frac{A_{s_2}(f)}{\max_{f \in BW}(A_{s_2}(f))} \tag{3.5}$$

where $A_{s_2}(f)$ denotes the s_2's harmonics amplitude at the frequency f.

The main advantage of this method is the low number of EM traces required to compute an accurate (stabilized) WGMSI value. In the paper authors only use 5 different runs of the same algorithm to highlight the zones of interest. This advantage with its ability of finely analysing the behaviour of EM radiations at specific frequencies, are really interesting to meet your objective : detect EMFI hotspots. Its use to that end is described in next section.

3.2.2.2 Finding computation positions

From the previous section one can conclude that finding EM probe positions above the IC surface characterized by the strongest SNR is not sufficient as they potentially have no link with the algorithm executed by the DUT. Similarly, finding positions with EM emissions the most strongly related to the data or instructions processed by the DUT could also be ineffective. Indeed, among such position of interest a part of them could have a very low EM coupling and therefore relying only on a coupling estimation might make the criterion avoiding some are of interest. Thus, at that stage, the question is still : what is a good EM injector position for an efficient EM fault injection ?

Because we are interested only by pulsed EMFI, let us rely on the related fault model, namely the sampling fault model [70]. The latter, which has been formerly described in the state of the Art, states that most susceptible elements of ICs are DFF. It is even more precise than that since it claims that the susceptibility is maximal when DFFs are switching i.e. at each rising clock edge. This means that analysis can be restricted to the emanations at the clock frequency or at its harmonics.

Considering these indications but also the state of the Art about near field scan techniques for reverse engineering, one can decide to use the spectral coherence to localize positions at which DFF are sampling data processed by the algorithm. This resumes in computing for each considered position of the EM probe the following incoherence score at frequency equal to f_{CK}:

$$incoherence_{s_1,s_2}(f_{CK}) = 1 - \frac{psd_{s_1,s_2}(f_{CK})^2}{psd_{s_1,s_1}(f_{CK}) \cdot psd_{s_2,s_2}(f_{CK})} \tag{3.6}$$

$$incoherence_{s_1,s_2}(f_{CK}) = 1 - \frac{Coherence_{s_1,s_2}(f_{CK})^2}{Coherence_{s_1,s_1}(f_{CK}) Coherence_{s_2,s_2}(f_{CK})} \tag{3.7}$$

Spectral incoherence is the contrary of spectral coherence (formula is given by Eq.3.7): a score of 1 means both signals have different spectra. It is used for the sake of readability since we are looking for spectral differences. Still, that means that at least two maps with different inputs given to the algorithm under test are required. However, this is not a major drawback since getting an EM analysis map is considerably faster than getting an injection map. It is possible to compare those two maps and thus to highlight positions related to DFFs involved in the algorithm execution. However, these positions does not necessarily correspond to positions with a strong SNR.

3.2.2.3 EMFI Susceptibility Criterion (EMFISC)

At that point, formerly proposed techniques to localize electrical activity points in ICs using EM near field scan have described and discussed with regard to our objective. We can now introduce guidelines to identify EMFI hotspots and derive from these guidelines a EMFI susceptibility criterion.

Considering the sampling fault model and its lessons, it comes that EMFI hotspots should be points :

- *guideline 1:* emitting the strongest signal (in terms of power) associated to the clock signal or clock tree. This implies that EMFI hotspots should be characterized by a great Power Spectral Density at the clock frequency. Here $PSD(f_{CK})$ is used as a measure of the EM coupling strength between the EM injector and the IC at the considered position.

- *guideline 2:* emitting signal tightly bind to both targeted algorithm and clock frequency (f_{CK}). Considering the discussion in the preceding paragraph, the strength of this link could be measure using spectral incoherence at $f = f_{CK}$ which would be written as $incoherence_{s_1,s_2}(f_{CK})$.

If $PSD(f_{CK})$ and $incoherence_{s_1,s_2}(f_{CK})$ are sound regarding lessons of the sampling fault model, considering them jointly is problematic. Indeed, incoherence values range between 0 and 1 whereas $PSD(f_{CK})$ values are real values. There is thus a scale factor problem. One solution to circumvent this problem and thus equalize the scales of the two metrics, one can center and reduce the values obtained during measurement campaigns. The resulting variables are denoted $psdn$ and $incn$ in the remaining of this thesis.

This rescaling of power spectral density values done, one can aggregate these two figures of merit in a criterion. The EMFI susceptibility criterion, denoted EMFISC we proposed is :

$$EMFISC_{x,y} = \sqrt{\begin{array}{l}(1-a)\cdot(psdn_{x,y} - \min_{x,y}(psdn_{x,y}))^2 + \\ a\cdot(incn_{x,y} - \min_{x,y}(incn_{x,y}))^2\end{array}} \tag{3.8}$$

where a is a constant allowing to weight the two guidelines. A value of a equal to 1 reduces the criterion to the incoherence while a value of 0 reduces it to the spectral density.

3.3 EMFISC protocol

The application of the EMFI susceptibility criterion requires meeting some experimental constraints. In order to highlight them, let us, for sake of simplicity, consider in the following scenario where an evaluator has to characterize the EMFI susceptibility of a DUT running a hardware AES.

Firstly, it is clear that to apply Eq.3.8, the knowledge of the clock frequency at which run the DUT or a hardware cryptographic implementation (for instance a Hardware AES) is required, f_{CK}. This knowledge could be obtained from documentations related to the DUT in case the evaluation is done in a white or grey box approach. In case of a black box approach, this value can be directly extracted from EM measurements analysed either in the frequency domain or time domain. Thus, this first constraint can be easily met.

Secondly, the computation of Eq.3.8 and especially that of $inc_{x,y}$ requires the acquisition, at different coordinates of the IC surface, of EM traces associated to signal s_1 and

s_2. Where s_1 and s_2 refers to EM traces related to the same AES algorithm execution but with different sets of inputs. Those signals should satisfy some experimental rules in order to disclose positions at which data, instructions and addresses involved in the AES computation are manipulated. These experimental rules being the following:

1. rule 1 : for each (x, y) coordinate, a set of n EM traces corresponding to the execution of the AES with exactly the same inputs (same plaintext and same key) is required to accurately compute $psd_{x,y}$. This set is thus a collection of n measurements of the same signal s_1.

2. rule 2 : for each (x, y) coordinate, at least two sets of n EM traces corresponding to the execution of the AES with at least two different inputs (either two different plaintexts (de)ciphered by the AES or two different keys) are required to compute $inc_{x,y}$. These sets are thus at least two collections of n measurements of two EM signals s_{11} and s_{12} with different plaintexts or key.

With such a collection of experimental data it is then possible using the following protocol 4 to compute maps of EMFI susceptibility of an IC running a given application. In this protocol 4, $psd_{x,y}$ and $inc_{x,y}$ are standardized at line 5. After standardization, the obtained values are re-mapped to get only positive values at line 6 (the minimal values are shifted to zero). Finally, at line 7, only the (x, y) coordinates with $emfisc_{x,y}$ values greater than q_α are conserved as EMFI hotspots where q_α is the α quantile. Thus, if α is fixed at 0.75 in the protocol, only 25% of positions with the highest $emfisc_{x,y}$ are kept as hotspots.

It should be observed that for the sake of simplicity, the protocol is reduced to two signals s_{11}, s_{12} but can be extended to more signals according to the degree of freedom (that is to say the inputs) let to the evaluator for the evaluated algorithm run by the DUT. Moreover, the Euclidean norm has been tailored to let the evaluator favour either PSD or spectral incoherence by changing the value of ”a” parameter. This tweaking has been added since experimental results will show that favouring either PSD or spectral incoherence can significantly impact the results.

Algorithm 4 EMFISC

Input: f_{CK}, matrix of s_{11} and s_{12},
α (% chip to keep),
a (weight *psd* compared to *incoherence*)
Output: $emfisc_{x,y}$

1: **for** X,Y positions **do**
2: compute $psd_{s_1}(f)$
3: compute $inc_{s_{11},s_{12}}(f)$
4: **end for**
5: $psdn_{x,y}$ and $incn_{x,y}$ = center reduce $psd_{x,y}$ and $inc_{x,y}$ population
6: remap $psdn_{x,y}$ and $incn_{x,y}$ populations to get only positive values
7: compute $emfisc_{x,y} = \sqrt{(1-a) * psdn_{x,y}^2 + a * incn_{x,y}^2}$
8: quantile($emfisc_{x,y}$,α)

3.4 Validation protocol

To demonstrate the efficiency of our methodology and the correctness of our EMFI susceptibility criterion, we applied it to two different devices. This section gives information about these testchips and the algorithm they executed during our tests. It also presents the different types of faults that were obtained during EMFI campaigns on the two considered targets. Finally, after describing the experimental validation protocol we applied, metrics are introduced and used to quantify the efficiency of our method.

3.4.1 Devices Under Test

To be as independent as possible of the nature of the testchips during our experimental validation campaign, two modern micro-controllers were chosen. These micro-controllers were designed in two different CMOS technologies by two different founders. Both testchips have their clock signal generated by an internal clock signal generator based on Phase Locked Loop and an internal RC oscillator.

The first one (testchip1) features an ARM Cortex M4 core operating at $80\,MHz$, a Memory Protection Unit (MPU), a Floating Point Unit (FPU) and a Digital Signal Pro-

cessor instructions. It also embeds $96\,kb$ of SRAM, $32\,kb$ of bootable RAM and $1\,Mb$ of flash memory. The cache memory between the flash and the cortex was deactivated on both targets during experiments. Designed in a $90\,nm$ process technology, it has an area equal to $12\,mm^2$.

The second testchip (testchip2) is organized around a ARM cortex M3 core operating at $64MHz$. It also features a MPU but no FPU. However, it only embeds $64\,kb$ of SRAM, $512\,kb$ of Flash. Designed in a different $90\,nm$ process, it has an area around $16\,mm^2$.

3.4.2 Algorithm Under Test

During all our experimental validations, EMFI have targeted Alg. 5 that was executed by the two Devices Under Test. Alg. 5 mainly consists in reading a chosen word at a chosen address in a first SRAM (AddrSRAM32 in Alg. 5), then writing the result of the reading in another SRAM (AddrSRAM96 in Alg. 5) and finally re-reading it to check if all operations have been performed correctly.

Algorithm 5 Pattern (AddrSRAM32, AddrSRAM96)

```
1: PUSH { lr, RO,R1 }
2: ADD R0,R0,#0; 11 times
3: LDR R0,[R0]; read SRAM32
4: STR R0,[R1]; write SRAM96
5: LDR R1,[R1]; read back
6: ADD R0,R0,#0; 11 times
7: POP { pc,R0,R1 }
```

One can wonder about the choice of this algorithm instead of a cryptographic algorithm. Such a pattern has been chosen for two reasons. First memory operation are a very common operation used by all algorithm therefore our test algorithm is not uncorrelated to any other algorithm. Secondly it is design to focus on measuring the relevancy of the criterion. We wanted the algorithm to be as fast as possible to enable exhaustive search on as most parameter as possible. Indeed, we are testing a method to reduce the number of position on the circuits to be tested, so we need to perform an exhaustive search and shoot

on every position of the circuit with a small stepping. Because we do not address the shoot timing in our criterion we need to test as most injection time instant as possible to make the comparison between the criterion and experiments as fair as possible. Otherwise, if we do test for a few time instant we might lose some error position which would bias the comparison between the criterion analysis and the real result. Moreover, this algorithm offer various degree of freedom. One can change the inputs, i.e. both value and addresses, or the set of registers ($R0$ and $R1$ in 5). All of this implies many days of experiment thence this quickness requirement on the algorithm.

The Alg. 5 has also been made to ease the EM analysis part, we can observe ADD instructions that are repeated 11 times before and after performing the two reads and the writing operation in SRAM. This repetition was inserted to isolate our target from the rest of the code and be able to interpret what happens during an EMFI. It also acts as a hard-coded delay after the trigger signal delivered to the EM pulse generator. Finally, it can also be seen as syntactic sugar to ease the setting to locate in time the target operations and thus guide the setting of EMFI and EM analysis equipments. To avoid cache misses and flash latency, that could make analysis curves a bit more difficult to read, we decided to execute the code from SRAM. By doing so it enables instruction to execute in the time indicates by ARM documentation. Therefore, we can bind type of faults to a specific operation (either the load or store), this enables to better classify the various kind of faults we get.

Thence this algorithm has been crafted to have a fine grain analysis (test various EM fault injection parameter) and to provide the best figure of merits to test our criterion. Furthermore, $LOAD$ and $STORE$ are quite widely use operation in embedded software it is not completely disconnected from a real scenario.

3.4.3 Preliminary tests with our EMFI platform

Before analysing how well our EMFI susceptibility criterion performs for our test cases, we led experiences with our EMFI platform (the one described in the state of the Art) and more particularly we analysed the kind of faults it can generate.

3.4.3.1 Analysis and taxonomy of faults

To be able to sort the faults according to the effect different detection mechanisms were implemented and different fault categories were defined. Remembering that we are targeting alg. 5 which manipulates with a fixed data and address sets, faults can be sorted using the following categories:

1. category n^o 1: corrupted value. This category groups all faults corresponding to a change of one or several bits in the expected results.
2. category n^o 2: corrupted address. This category gathers all erroneous reads at an address not enclosed in our pre-defined address set.
3. category n^o 3: corrupted memory. This category is formed by all faults corresponding to a memory content modification.
4. category n^o 4: infinite loop. This category is made up of all faults corresponding to a disruption of the reset signal or program counter.[2].
5. category n^o 5: garbage. Sometimes UART instead of sending coherent values send garbage value (value = "????????"). This could be explained by a modification of the baud rate value. This category groups all faults of this type.
6. category n^o 6: no response. This category gathers all faults forcing the IC to no more respond.

It should be noted that arrangements have been made in order to properly classify faults obtained during experiments. For instance, in order to detect faults falling in the corrupted address category, the SRAM memory banks were preliminary filled with the value $0xDDDD$
$DDDD$. With such an arrangement, if the chip respond with a $0xDDDDDDDD$ response or some $0xDD$ in its answer are likely to be due to a corruption of the address to be read or store. That is to say the LDR instruction or the STR instruction of 5 has been use on a modified address caused by EMFI. It should be noticed that our pattern does not allow the differentiation of faults disrupting the read and store instructions. Indeed, a fault impacting the store instruction necessarily implies a wrong load operation but a wrong load operation does not necessarily imply the corruption of the store instruction, it could be due to the disruption of the load instruction itself.

[2]The program counter is a register containing the address of the next instruction or the current instruction or some architecture

In order to detect corruptions of the memory content, an embedded Cyclic Redundancy Check (CRC) was used. The CRC is an error detection code mainly used to detect changes in values. During our experiment, we used it to compute a CRC signature of the memory region containing our data and code before and after each EMFI attempt. These CRC signatures were then compared in order to detect memory content changes.

During our experiments, testchip A was exposed to EM perturbations generated with the amplitude of our pulse generator set to of $50V$ and of $130V$ whereas for testchip B a finer analyse was done. Indeed, for this testchip, the amplitude of the voltage pulse was varied, by step pf $5V$ between $160V$ and $225V$. In both cases, the pulse width was set to $9.5ns$. All faults observed during these two experimental campaigns were analysed.

For testchip A, it was observed that most faults of type 'no response' have been obtained by setting the EMFI injector above either the ARM cortex or the power supply block. The faults falling in the infinite loop category have been observed while the EMFI injector was placed above the reset block or when the power consumption of the DUT was too much increased. For such faults, in order to carry on the experiment, the power supply has to be shut down before re-booting the chip.

The analysis of faults falling in the corrupted value bin, showed that most faults are bitsets and bitresets of one or several bits and this with both pulse's polarity. For instance, EMFI could transform an expected $0xFFFFFFFF$ into a $0xBFEFFFFF$ value or an expected $0xAAAAAAAA$ into $0xAAAABAAA$. This means that EMFI can alter a complete word, a single byte or even a single bit. In addition, regarding the effect of pulse polarity, it was observed that changing the pulse polarity does not necessarily change bitsets into bitresets or conversely.

3.4.3.2 Amplitude parameter effect

In the previous sections we claimed that the larger the pulse amplitude the larger the fault areas are. This claim is also supported by an experiment we led. This experiment has consisted in performing fault injection maps with specifics sets of pulse amplitudes and widths. However, to compare comparable things, the moment at which the EM pulse is generated was swept along the time window of the algorithm under test. The idea being to find the moment at which the more faulty responses are induced during the processing of

the target algorithm. This moment identified, the pulse width value was chosen by using the same policy.

Then EMFI campaigns were launched with different pulse amplitudes ranging between $160V$ and $220V$. Fig.3.6 and Fig.3.4 show the results for the two extremal values. These maps correspond to a full scan of the testchip2 with a displacement step of the EMFI injector equal to $100\mu m$.

Orange pixels correspond to EMFI injector positions at which the $LOAD$ as well as the $STORE$ instructions involved in Alg.5 can be faulted. Green ones highlight coordinates at which only the first $LOAD$ is faulted while the light blue ones are associated to $infinite\_loop$ fault category (i.e. fault on reset signal or program counter).

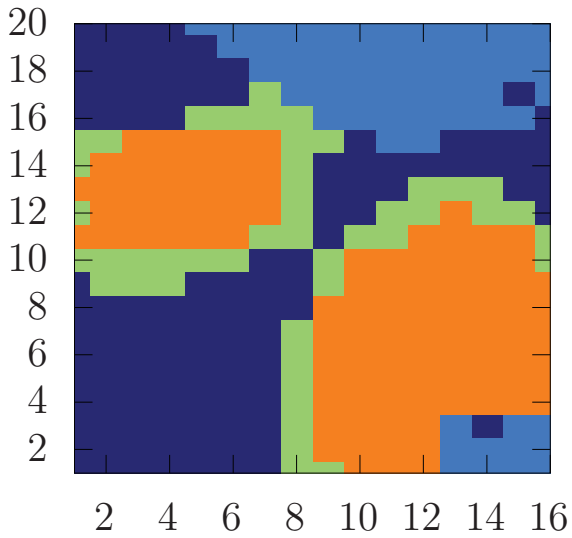


Figure 3.4: Fault map for pulse amplitude of 160V

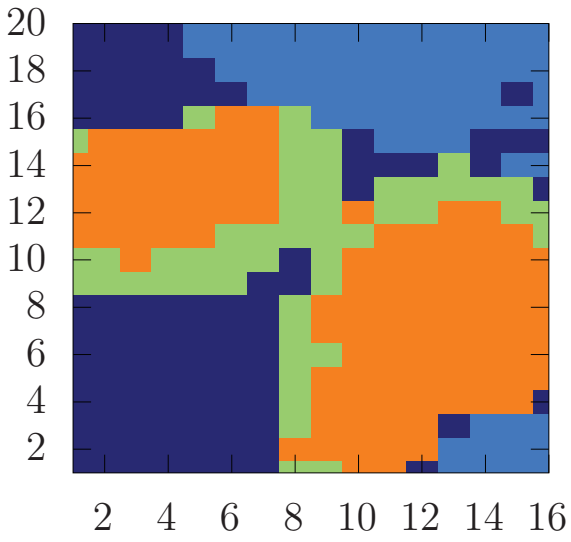


Figure 3.5: Fault map for pulse amplitude of 190V

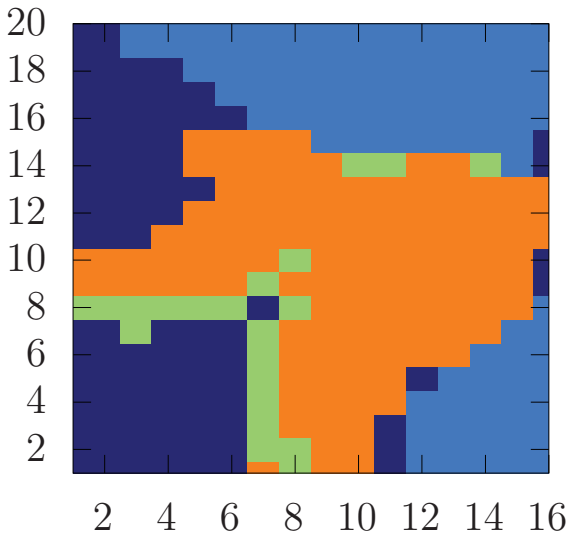


Figure 3.6: Fault map obtained with pulse amplitude of $220V$

The comparison of the two maps clearly illustrate that increasing the pulse amplitude enlarges the fault areas. For instance, at $160V$, 63% of positions over the chip correspond to positions at which a fault is induced while at $220V$ this score is equal 71%. Moreover, the effect the EMFI also evolves with the amplitude increase and become more stressful from the circuit point of view. Indeed, part of the green pixels corresponding a single instruction fault (*green area*) has been changed into orange (disruption of two instructions) or light blue pixels.

3.4.4 Experimental validation protocol

The experimental validation protocol has consisted in performing EM near field scan of testchips with an EMFI probe. The X and Y map steps were of $100\mu m$ for both testchips. At each position, 1000 EM traces corresponding to the execution of target algorithm with specific input were acquired with a sampling rate equal to $10GS/s$ for testchip1 and $1GS/s$ testchip2. This relatively high number of measurements at each position was imposed by the presence of noise generated by the analogue part of these circuits. Collected traces were then gathered in small set of traces to generate a reduced set of median traces. In the absence of such noise a significantly lower number of traces would have be sufficient by position. So without noise due to analogue parts those acquirements have been reduced to 900 curves divided in 300 acquirement for different sets of value and different addresses to read/write from (for a given position). After the EM near field scans, EMFI maps were performed. They were drawn with different pulse amplitudes using the same EMFI probe than the one used to perform the near field scans. The X and Y displacement steps for EMFI maps were fixed to the same values as EM near field scans. For testchip1 (respectively testchip2) pulse amplitudes of $\pm 50V$ and $\pm 130V$ (respectively $\pm 198V$) were considered.

3.4.5 Experimental results

To quantify the efficiency of the proposed hotspot localisation method, we defined two figures of merit. This solution was preferred to a visual approach consisting in comparing EMFISC maps with fault maps.

The first figure of merit we defined is the Coverage Rate (CR). It is the percentage of all considered coordinates (EM probe positions) above the IC surface leading to a fault that are discovered by our methodology, i.e. that falls in the quantile q_α. Of course, because CR depends on α, the evolution of CR with regard to this parameter has to be analysed rather than particular values. If the hotspot localisation method is efficient CR must remain high for high values of α.

$$CR = \frac{cardinal(\,(Position\,in\,criterion) \cap (Faulted\,Position)\,)}{cardinal(Faulted\,Position)} \tag{3.9}$$

The second figure of merit is the False Positive Rate (FPR) defined as the percentage of positions ranked in the category highly EMFI susceptible, i.e. falling in the quantile q_α that do not lead to a fault. Once again, the evolution of this figure of merit with regard to α is considered in the rest of the paper rather than a single value. If our localisation methodology is correct FPR must be low for high values of α.

$$FPR = \frac{cardinal(\,(Position\,in\,criterion) \cap \overline{(Faulted\,Position)}\,)}{cardinal(Area\,of\,the\,chip\,considerd)} \tag{3.10}$$

If these figures of merit give scores to measure the efficiency of the hotspot localisation method, they can also give a score to any other localisation method. In order to set a reference and be able to evaluate the performance of the EMFISC approach, we have chosen to compare it to a random approach consisting in selecting randomly the set of points as highly EM susceptible (in red on fig.3.7, 3.8). We could have considered a smarter approach. However, to the best of our knowledge there is none in the literature aiming at finding EMFI hotspots. Fig. 3.7 and 3.8 give the evolution of the defined figures of merit with regard to α in % for our two testchips submitted to EMFI. The first panel of these figures reports an averaged EM analysis traces showing the time windows on which the $EMFISC_{x,y}$ values are computed. The second panel gives the evolution of CR with regard to α for $a = 0.25, 0.5, 0.75$.

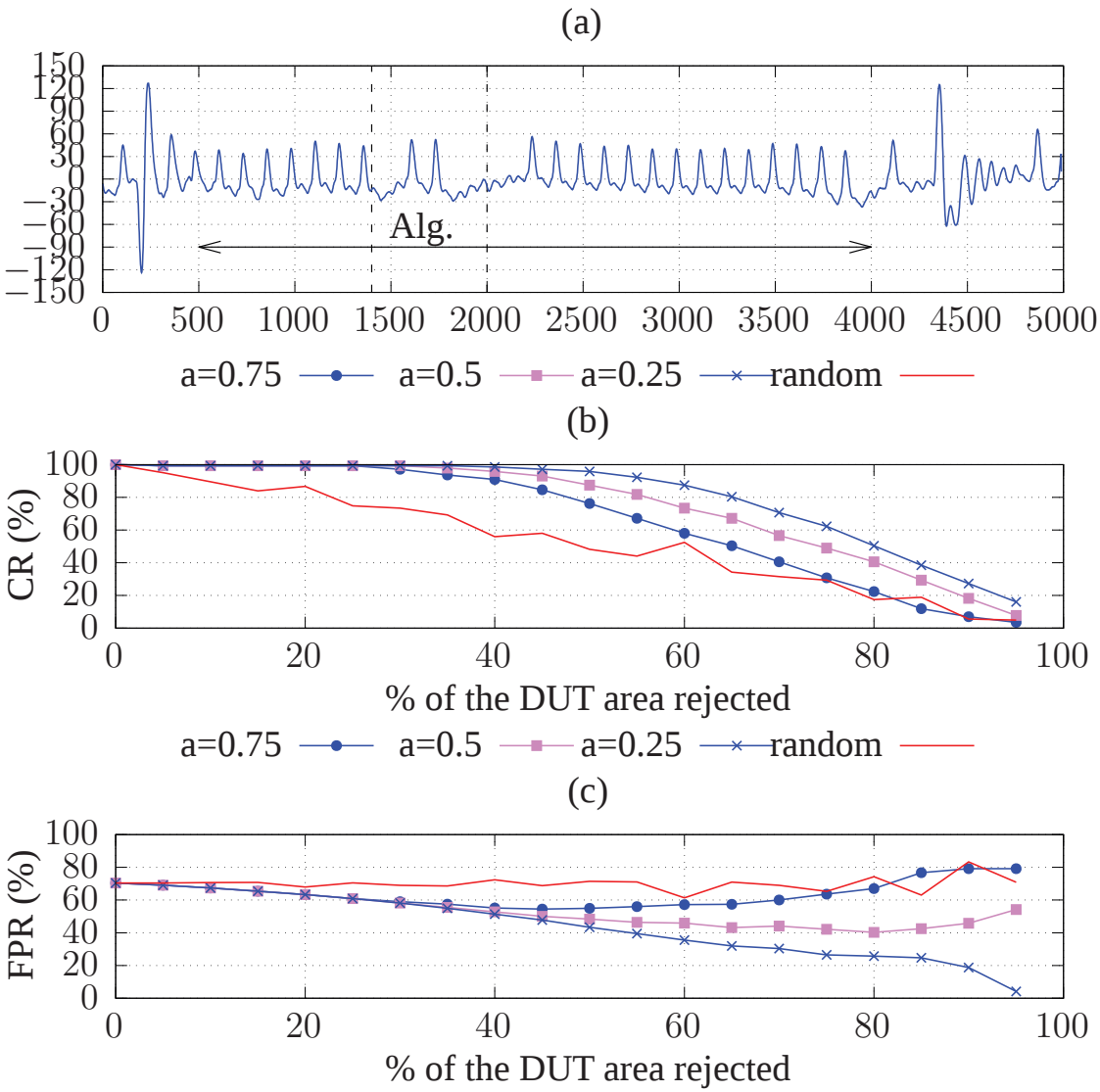


Figure 3.7: Testchip 1 (pulse amplitude $\pm 130V$) :(a) averaged EM traces at a given position, (b) evolution of CR with regard to α for $a = 0.25, 0.5, 0.75$ (c) evolution of FPR regarding α for $a = 0.25, 0.5, 0.75$, red curve = randomly selected point over the IC

.

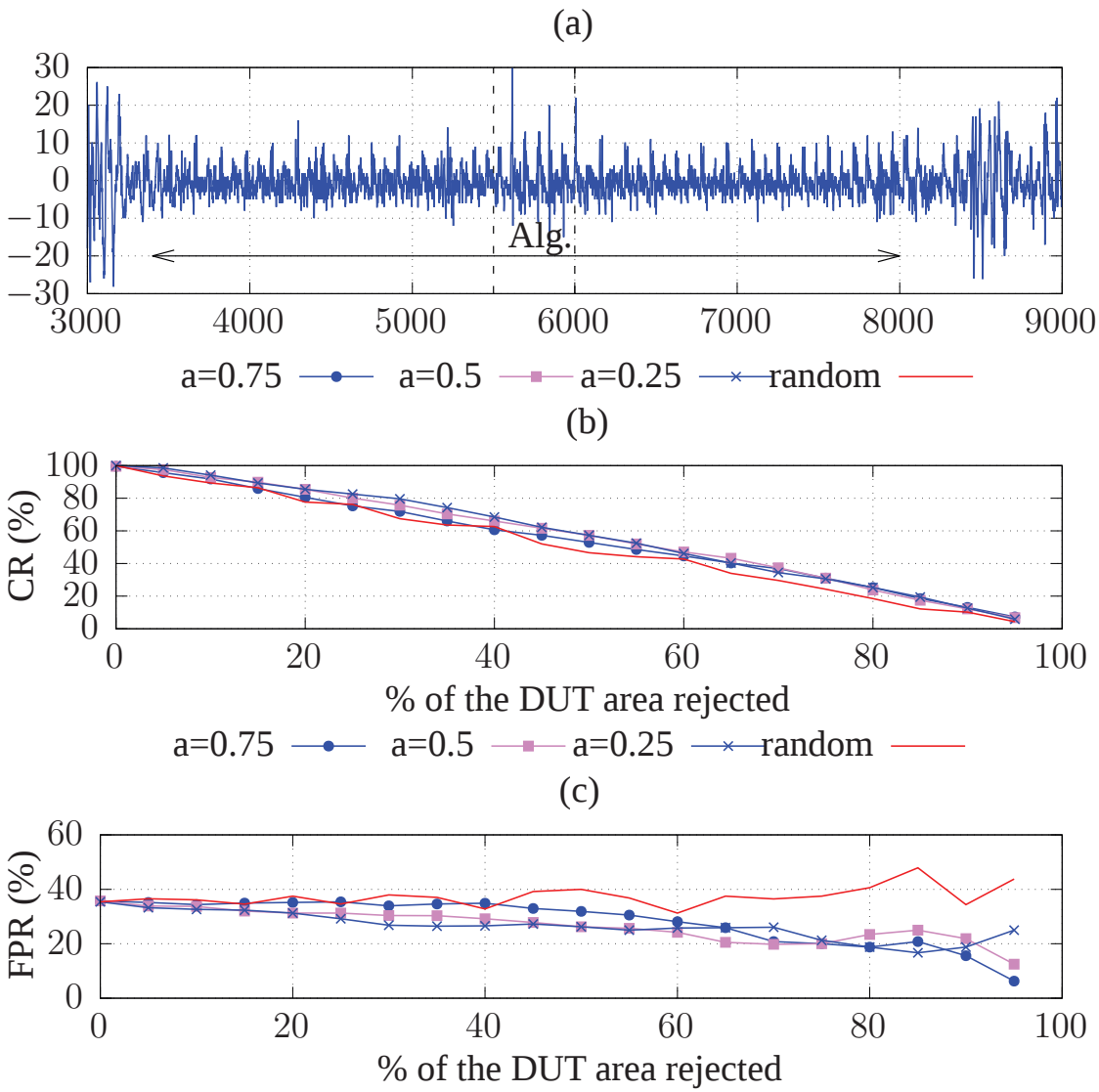


Figure 3.8: Testchip 2 (pulse amplitude $\pm 198V$) : (a) averaged EM traces at a given position, (b) evolution of CR with regard to α for $a = 0.25, 0.5, 0.75$ (c) evolution of FPR regarding α for $a = 0.25, 0.5, 0.75$, red curve = randomly selected point over the IC

As shown, if we do not privilege any guideline and thus consider $a = \frac{1}{2}$, the CR remains higher than 80% for $\alpha < 50\%$ for testchip1 which is a value significantly better than those obtained with a random selection of points. This means that more than 80% of positions leading to faults are captured by our method while keeping only 50% of the IC surface for testing. For $\alpha = 40\%$, CR remains higher than 60% values which is two to three times the values obtained with a random selection of points (for $a = \frac{1}{2}$).

In the case of the testchip2 the result are less spectacular still it has an almost constant 10% better Coverage Rate than the random approach (until we reject more than 60% of the DUT area). The difference being more important in terms of False Positive Rate which is discussed in the next paragraphs. The testchip2 has a huge percentage of surface that is faulted, namely 60%. This implies that for $\alpha > 60\%$ we start removing points of interest as our criterion is not perfect it is one reason why the result are less spectacular. Moreover, it shows that our criterion can be enhanced, this will be address in the next section. The fact, that for both testchips CR is greater than the random chosen method show the soundness of our approach.

Regarding the FPR, one can observe that from $\alpha = 50\%$ and $a = 0.5$, its value is close to 50% for testchip1 and 30% for testchip2. This value is significantly better than that obtained with a random selection of points. This difference between the two testchips being explained by the fact that we do not fault the same percentage of the chip area for both targets.i Indeed, in the case of testchip1 only 20% of its surface was faulted while on testchip2 I was able to fault 60% of its area. Therefore, the target with the higher amount of faulted point, i.e. testchips2, has obviously lesser false positive. One can finally observe that the FPR decrease while α increases contrarily to the random approach. This decrease demonstrates the soundness of EMFISC based method because we reject more points that do not lead to a fault than points leading to faults while increasing α.

Finally one can observe that CR and FPR values are much better for $a = 0.25$ for both testchips 'at least for CR since the FPR of testchip1 is quite bad). This indicates that antennas with incoherent EM emissions must be privileged over antennas with strong but coherent EM emissions related to the clock signal. This suggests that EMFI induces more easily faults on the data path of ICs than on the clock tree of processors (glue logic).

However, this points must be further investigated to definitively conclude.

3.4.5.1 Enhancing the criterion ?

To estimate the soundness of the proposed criterion a first approach was to compare the results it provides with another approach consisting in randomly selecting positions above the IC surface. The results of such a comparison have been presented in the preceding section. They show that the criterion is far better at discovering position leading to faults than a random walk. Yet, since we are working in white box it is possible to go further in the analysis. Indeed, we have at disposal both fault maps and criterion maps, i.e. incoherence and PSD maps. Therefore, in an enhancement perspective we can confront the two criteria to see which one enables to discover susceptible areas of the DUT. From the previous results, it can be sense that incoherence provides more valuable information than the power spectral density. Indeed, increasing the share given to incoherence (with the help of the a parameter) yields to better Fault Coverage and False Positive Rate.

To perform such a comparison the following protocol has been used. First all couples $(inc_n(f_{CK}), psd_n(f_{CK}))$ are reported in the plan as shown Fig.3.9. Then the information whether the position has been faulted and how it has been faulted are added to the plot using a colour code. With such a protocol, the expected result if our criterion is sound, is that most of the positions leading to a fault must have high psd_n and inc_n values or at least one of the two must be high. The result for the testchip1 are represented on Fig. 3.9.

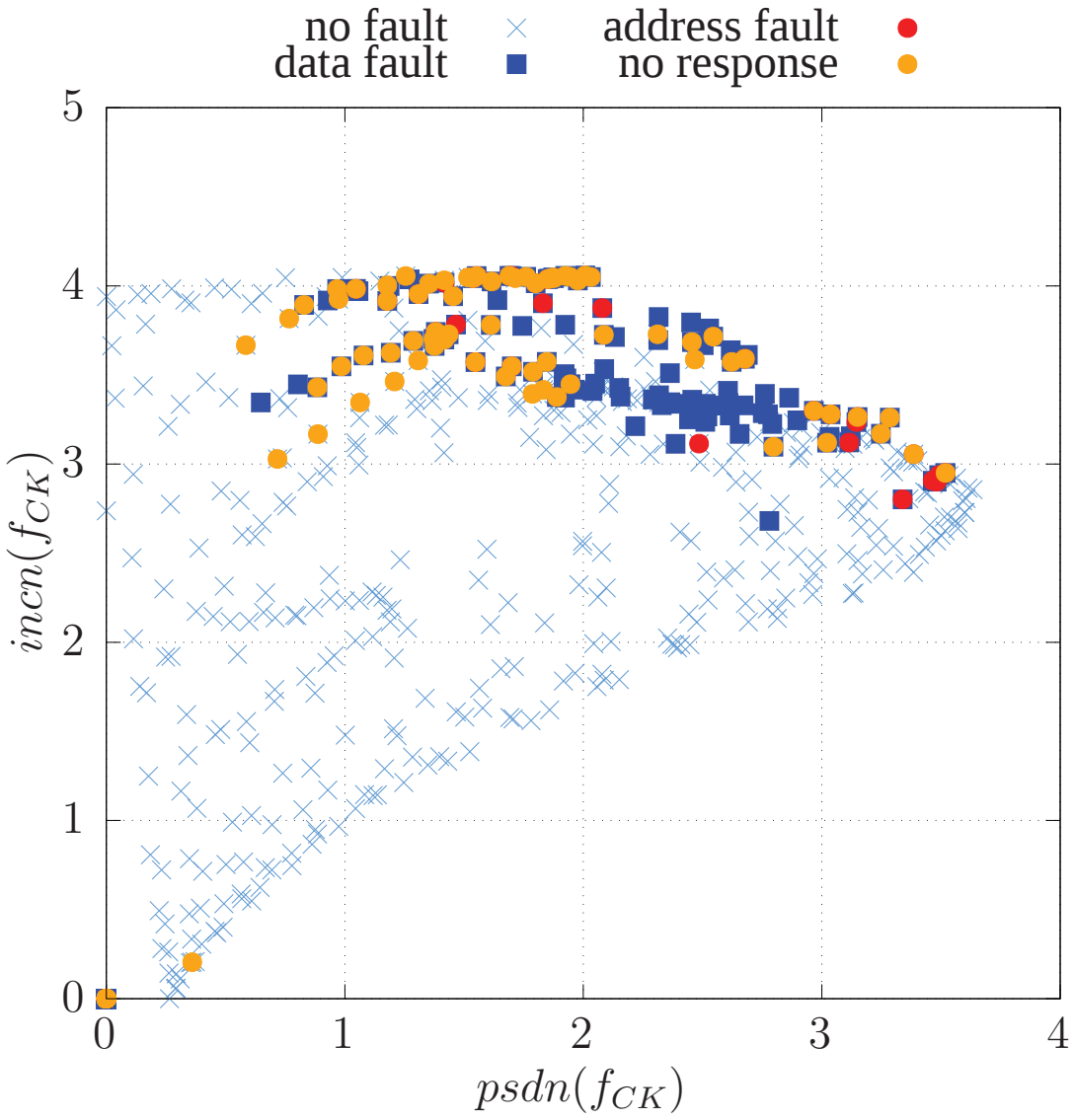


Figure 3.9: Fault repartition according to PSD and spectral incoherence value

The observation of Fig. 3.9 indicates that almost all positions leading to a fault are characterized by a strong incoherence value. This explains why giving more weight to incoherence enhances the Coverage Rate and reduce the rate of false positives. On the other hand, faults seems to be spread all along the $psdn$ axis. This means that this criterion is less discriminating even if there is no fault for really low values of $psdn$. This mean a certain threshold of $psdn$ should be reached to ensure fault. Two reasons could explain this observation:

- The value of the power spectral density at the clock frequency could be a too simple estimate of the coupling coefficient between the EM injector and the DUT.
- It might be too ambitious to assign EMFISC value to such reduced area. Indeed, EMFI injector are far larger than a pixel and the EM field spreads far away of the probe. In [67] authors have shown using measurements of the EM field distribution have shown than the EM field distribution has the shape of binormal distribution centered below the center of the EM injector. Thus, the EM field has a maximal value Vmax just below the center of the injector and experiments have indicated that the amplitude is roughly equal $\frac{1}{\sqrt{(2)}}$ to below the edge of the injector. The EM field is thus strong at a distance equal to 2 times the radius of the EM injector.

Regarding the first reason, a solution could be to directly look at the effect of EM perturbation on the power and ground network by monitoring the bias of the power and ground pads. However, such an approach also provides a rough estimate of the mutual inductance between the injector and the DUT. In addition, this implies additional preparation steps of the DUT and the DUT should enable to bypass its embedded voltage regulator. This approach has thus not be considered.

The second reason could potentially address by concatenating data from neighbouring pixels. This point is addressed by next sections.

3.4.6 Summary

In the preceding paragraphs, we have introduced a criterion to discover highly EMFI susceptible areas of ICs. The proposed criterion has been derived from physical considerations but also from the sampling fault model. Experimental results obtained for two

DUTs operating under with two different clock frequencies have demonstrated its soundness.

Still if we consider the criterion from an optimizing point of view the following problematic stands out. The $psdn$ criterion does not seem to be selective enough. By plotting the $incn$ value against $psdn$ one can see that faulted value are mostly high value of incoherence but this claim is only partially true for $psdn$. One potential reason could be the large spreading of the EM field over the IC surface that forbids analysis with a too accurate spatial resolution. This suggests the use of the EMFISC at coarser grain.

3.5 Coarse grain EMFISC analysis

To tackle the issue related to the large spreading of the EM field during EMFI, one should gather information collected at different positions during the EM near field scan. The key question is how points should be gathered ?

One could have tried to exploit the binormal distribution in space of the EM field during an EMFI. However, this is not a reasonable approach because we are trying to gather EM traces generated by the ICs in normal operation. In addition, this binormal distribution is guaranteed only when the injector is in not in proximity with other electrical devices. The true distribution could be different in the close vicinity of the DUT.

After having dismiss the above approaches and considered the following:

- the fineness of geometry characteristics of the power and ground networks routing,
- the compactness of the IC floorplan when one consider functional blocks one by one,
- the propagation of current along the power and ground networks

We decided to gather positions according to the similitude of the EM waves (and thus of their EMFISC value) collected above it; similar EM waves being likely to be induced by the same logical activity in the IC. But also because the propagation along power and ground rails should be the same for all traces in the same groups.

3.5.1 Clustering using k-means

K-means clustering is a very common clustering algorithm, the term was coined by James MacQueen in 1967, the paternity of the algorithm is given to Hugo Steinhaus. It is an unsupervised learning method. This means it does not need a training datasets to be used. This algorithm aims at partitioning the observation (EM traces) into k groups where k is set by the user. At the end the algorithm returns the label of each data, i.e. the groups at which belong the data and the centroid. The centroid is the mean of all curves in a cluster. This implies that the centroid does not belong to the datasets.

The k-means algorithm works as follows. First, it takes k random curves from the dataset, as initial centroids. Then it computes the distance between each of these centroids and all the remaining data. Usually, the distance used is the euclidean distance with the dimension of the trace length. According to all computed distances, each trace is assigned to the cluster with the closest centroid. After complete assignation of all data, centroids are updated. This procedure is repeated until no change occurs in the assignment or a certain number of iterations is reached. Since this algorithm is an heuristic, it is likely that it converges toward a local optimum. To ensure the best solution has been found it is common to run multiple times the algorithm (5000 times in our case) as it computes really fast (some minutes for traces of 700 samples long) on the EM traces. The algorithm we used during our work is the algorithm provided by python scikit learn version 0.20.

The interest of clustering is that one can reason with centroids instead of the whole dataset since the centroids are assumed representative in terms of behaviour of all vectors (traces) in the corresponding cluster. This, thus usually simplify the considered problem. In our case we decided, for the sake of simplicity, to stick with real data and therefore the notation centroid is abused in the rest of the paper. Indeed, in the rest of this document the centroid are the curves minimizing the distance with the mathematical centroid.

3.5.2 Enhanced EMFI Susceptibility Criterion

From the previous discussion the enhanced EMFISC can be derived straightforwardly. First, the set of EM traces are clustered according to their shape using the k-means. The cluster are then sorted according to their centroid's emfisc value $(psdn(f_{clk}), incn(f_{clk}))$.

This results in assigning to each trace the EMFI susceptibility of the cluster centroid to which it belongs. Spatially speaking, this resumes in partitioning the DUT surface into k regions with a specific EMFI susceptibility.

There is a key point when applying the clustering approach : the choice of k value. It defines the number of regions into which the DUT area has to be split. This value could be decided mathematically using different approaches among which there is the Akaike information criterion [3]. The value of k can also be derived from more practical considerations. Indeed, the larger the value of k the more positions leading to faults are discovered (a higher coverage rate) at the cost of keeping a larger part of the DUT to explore during EMFI campaigns. Thus, the value of k acts similarly to the quantile value q in the previously proposed technique but in a discrete manner.

3.5.3 Effectiveness of the Enhanced EMFI Susceptibility Criterion

To evaluate the Effectiveness of the enhanced EMFI the coverage and false positive rates were used as figures of merit. Doing so enables to compare the two criteria one with another. However, to further establish the effectiveness of the new approach the Internal Coverage Rate (ICR) should be introduced. The latter simply consists for each cluster in the ratio between the number of points that leads to a fault in a cluster and the total number of points within the cluster. It is computed as follows:

$$ICR(cluster_n) = \frac{card((pointsincluster) \cap (faultedpoints))}{card(pointsincluster)} \tag{3.11}$$

If the proposed approach is sound, this ratio must high for cluster with a centroid characterized by a high EMFISC score and should be low for other clusters.

3.5.3.1 Analyses of the spatial distribution and waveform of clusters

The experimental tests were performed on the same targets as before namely testchip1 and testchip2.

Before getting to the comparison with the previously defined figures of merit let us analyse the waveforms of the centroids obtained using the k-means as well as the spatial distribution of clusters. Figures 3.12 and 3.13 give the waveforms of the centroids for testchip1 and testchip2 respectively. Figures 3.10 and 3.11 give the spatial distributions.

One observation that can be made is that for both testchips the points belonging to a cluster are spatially grouped. This result was expected because of the way circuits are placed and routed and of the way currents propagate in ICs.
Indeed if we look at both figure 3.12 and 3.13 we can directly see that the different cluster, i.e. the different color, are quite packed.

Considering the waveforms of centroids, one can observe the k-means splits the DUT according to tiny details in the EM traces. For instance, at first sight, centroids associated to clusters 1 and 2 in Fig. 3.12 appear really similar because of the two main peaks around sample=210 and 350. However looking in more details clearly shows they have significant differences around samples 390, 450, 600 or 680. It seems that those clusters share some common characteristics in time but have their own specifies. Similar observations can be done with clusters of Fig. 3.13. It seems that the k-means is able to split the EM traces according to differences between traces which are limited in time i.e. time difference in the electrical activity below the EM probe which are time limited.

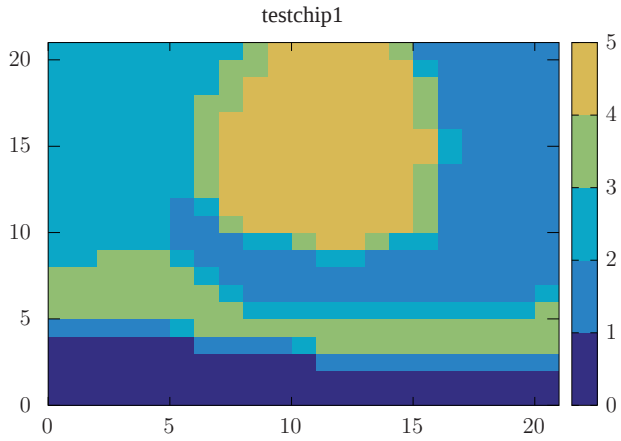


Figure 3.10: Spatial distribution of clusters or testchip1 (points of same color belong to the same cluster)

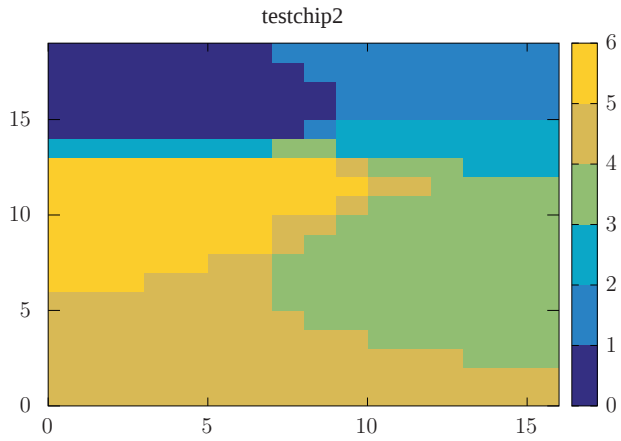


Figure 3.11: Spatial distribution of clusters for testchip2 (points of same color belong to the same cluster)

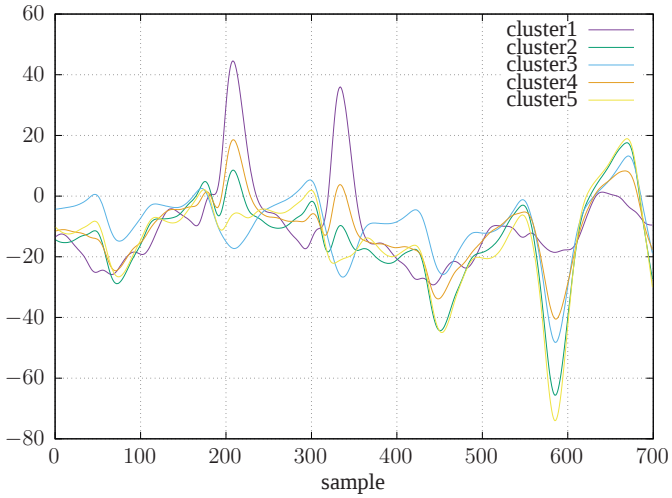


Figure 3.12: Centroid waveforms for testchip1

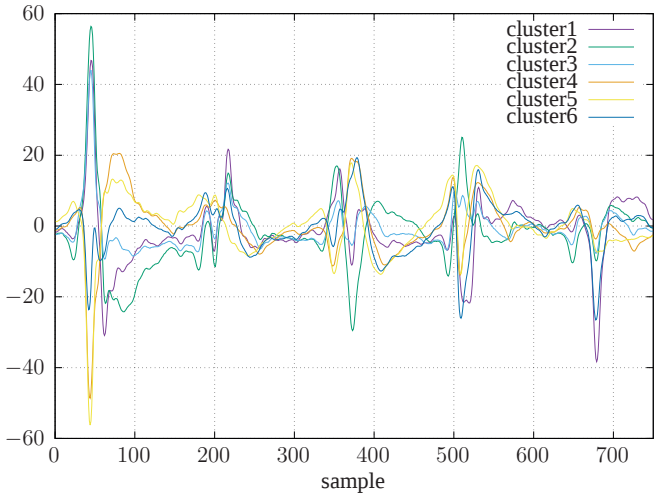


Figure 3.13: Centroid Waveforms for testchip2

3.5.3.2 Analysis of faults in clusters.

To check the soundness of the enhanced EMFI susceptibility criterion the different positions belonging to each cluster are plotted Fig. 3.14 and 3.15 in the psd_n, inc_n space. As for Fig. 3.9, a colour code has been used to highlight faults. Fig.3.14 and Fig.3.15, one can observe that for both testchips, clusters with the highest EMFISC value have the highest Coverage Rate i.e. gather lot of positions leading to faults. This demonstrates the soundness of enhanced EMFISC approach.

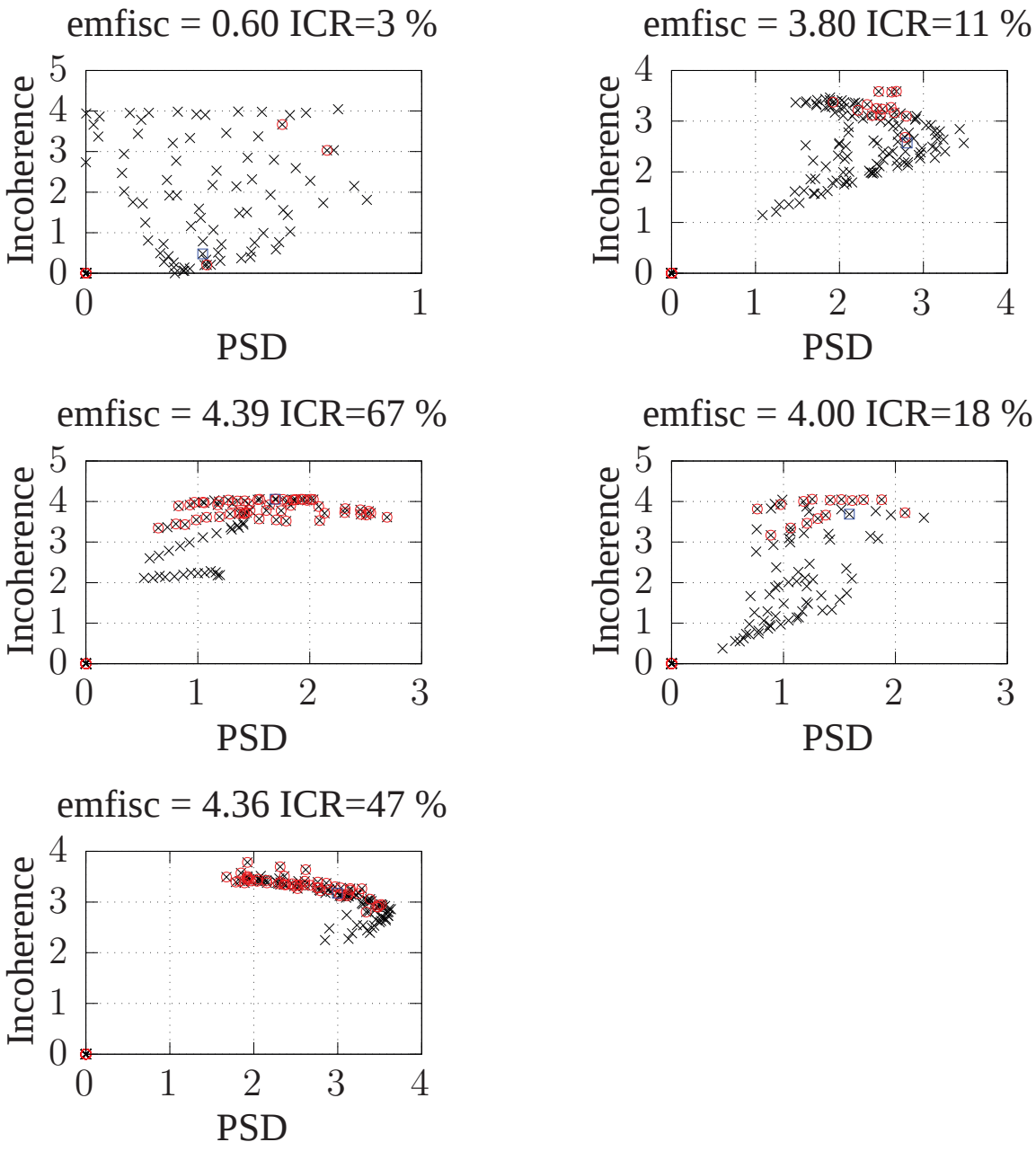


Figure 3.14: Faults and no fault distribution regarding PSD and spectral incoherence values for each cluster (clusters are plot in the following order top 1 2 | middle 3 4 | bottom 5)

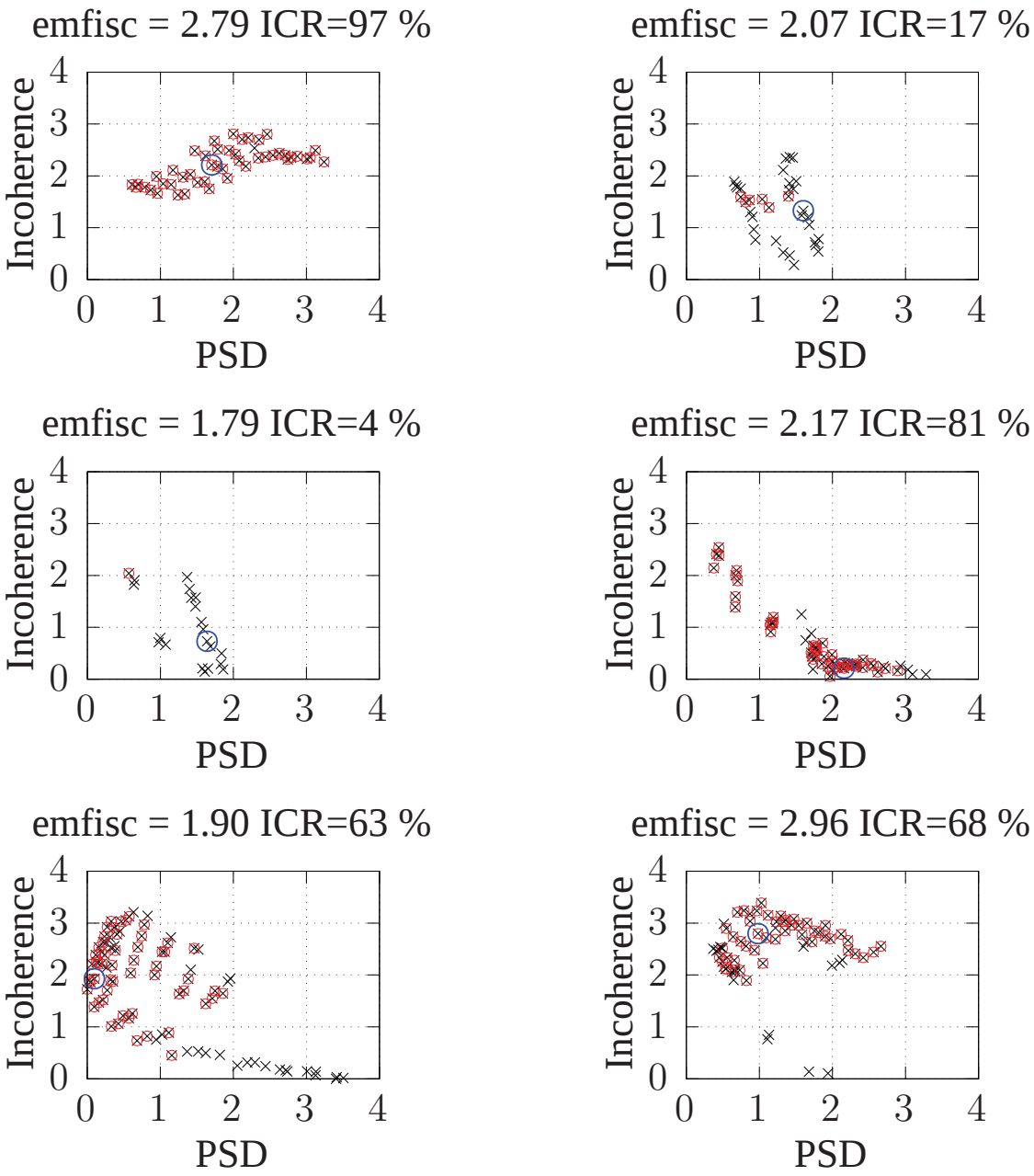


Figure 3.15: Faults and no faults distribution w.r.t PSD and spectral incoherence values for each cluster (cluster are plot in the following order top 1 2 | middle 3 4 | bottom 5 6)

3.5.3.3 Coverage Rate Analysis.

If the enhanced EMFISC seems efficient, one can wonder if it allows significantly reducing the area to be inspected during EMFI campaigns. To provide insight on this point, Tables and give for each cluster:

- the EMFISC value,
- the percentage of the IC surface occupied by each cluster (ICS Coverage),
- the internal coverage rate (ICR),
- the coverage rate obtained when considering each cluster.

By looking at the size of the cluster, one can notice that for both testchip the cluster with the highest EMFISC value occupies around 18% of the chip area which is quite low and therefore acceptable for EMFI characterisation since their ICR are respectively equal to 67% and 68%. Thus, by inspecting only 18% of the chip area one can discover about 40% and 22% of faults that would have being obtained with a full EMFI scan of the IC.

Keeping the two best clusters requires inspecting 42% and 37% of the IC surface and ensures CR equal to 77% and 44%. This clearly demonstrates the soundness of the proposed approach.

cluster id	EMFISC	ICS coverage (%)	ICR (%)	CR(%)
cluster 1	0.6	17.15	3	3
cluster 2	3.8	25.41	11	10
cluster 3	4.39	17.56	67	40
cluster 4	4.00	16.32	18	10
cluster 5	4.36	23.55	47	37

Table 3.1: EMFISC, ICS Coverage, ICR and CR for testchip1

cluster id	EMFISC	ICS coverage (%)	ICR(%)	CR(%)
cluster 1	2.79	14.6	97	21
cluster 2	2.07	10.6	17	27
cluster 3	1.79	6.5	4	1
cluster 4	2.17	18.7	81	26
cluster 5	1.90	27.5	63	3
cluster 6	2.96	21.8	68	22

Table 3.2: EMFISC, ICS Coverage, ICR and CR for testchip2

3.5.4 Comparing the EMFISC and enhanced EMFISC

To perform the fairest possible comparison the same metric should be applied to both criterion. The coverage and false positive rates have thus been computed with the two approaches. If it can be done for all real values of $q \in [0, 1]$ for the standard EMFI susceptibility criterion, this is not the case for the enhanced criterion. In the latter case, the calculus is only possible for a reduced set of values for q.

Indeed, with the enhanced approach, clusters are progressively involved in the analysis starting from the one with the highest EMFI susceptibility toward the one with the lowest value. Proceeding so, there is k possible values for q. For instance, considering testchip 1, the possible q values are:

- $q = 0.82$ when only the cluster 3 is kept for the EMFI campaign,
- $q = 0.59$ when only clusters 3 and 5 are kept for the EMFI campaign,
- $q = 0.43$ when only clusters 3, 5 and 4 and are kept for the EMFI campaign,
- $q = 0.17$ when only clusters 3, 5, 4 and 2 and are kept for the EMFI campaign,
- $q = 0.00$ when all clusters are conserved.

The results one gets by applying these two approaches to both testchips are plotted Fig. 3.16 and 3.17. From both figures it is clear that the enhanced EMFISC or cluster based approach gives the standard approach. For instance, if one consider testchip1, the CR is equal to 40% for $q = 82\%$ with the cluster based approach while it is equal to 35% with

the standard approach. Similarly, the FPR is equal to 33% with the enhanced approach instead of 43%.

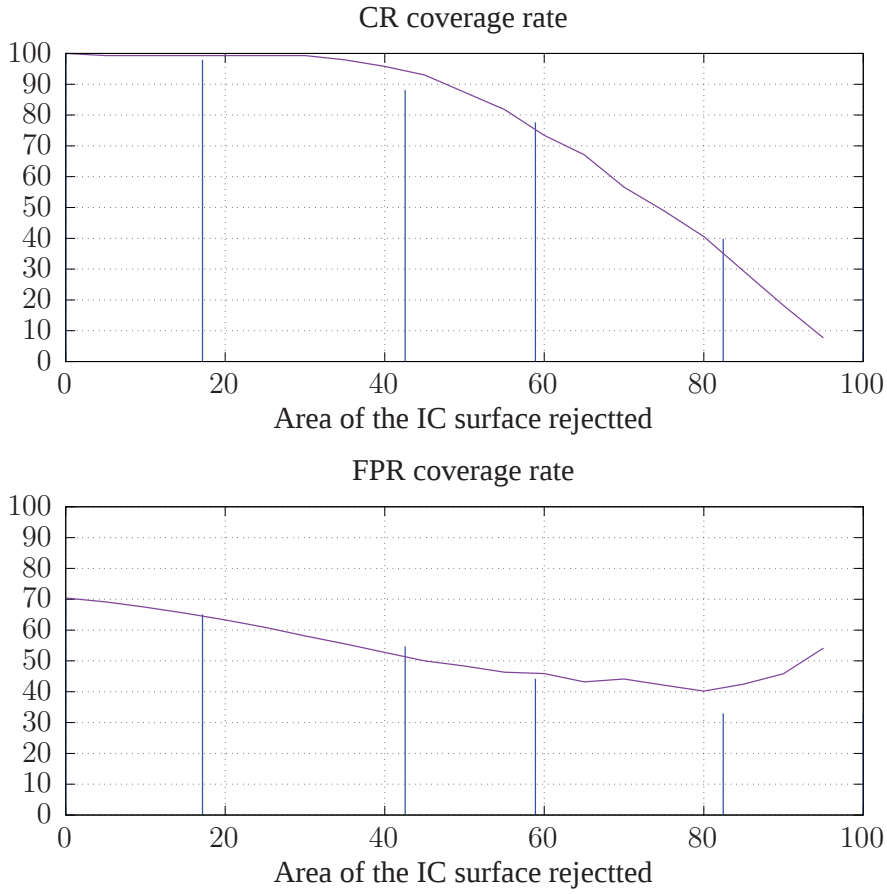


Figure 3.16: CR and FPR evolutions with q obtained for testchip1 with the standard (magenta) and cluster based (blue) approaches.

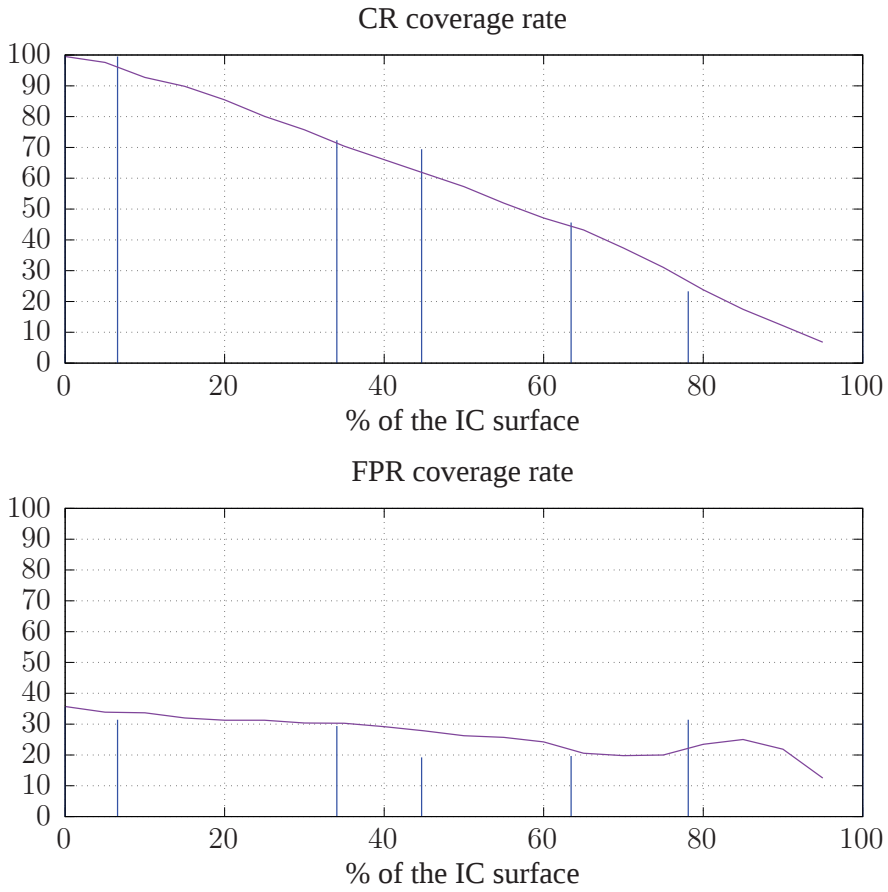


Figure 3.17: CR and FPR evolutions with q obtained for testchip2 with the standard (magenta) and cluster based (blue) approaches.

3.5.5 Summary

From all the above results showing that the cluster based approach is superior to the standard one, one may conclude that:

- the EMFI susceptible criterion, even if one can improve it, is sound. The way for enhancing has been identified. It consists in better estimating from EM measurements the mutual inductance between the probe and the DUT. These points are still at that point an open problem. Maybe a better model of EMFI effects could open the door to a solution ?

- a coarse grain approach is preferable to a fine grain one because of the reduced

spatial resolution of EMFI but also because of the propagation of currents (either consumed by the DUT or induced by an EMFI) in the power / ground networks.

3.6 Conclusion

In this chapter, a criterion called EMFI susceptibility criterion has been introduced to speed up DUT characterisations with regard to EMFI by reducing its complexity. It aims at taking advantage of the reduced time required to acquire of EM near field maps regarding the time required to get equivalent EMFI maps. To that end, the developed criterion addresses the problematic of finding positions of EMFI injectors leading to fault from EM near field measurements. This criterion has been derived from the EM sampling fault models and considerations related to the EM coupling between a probe and DUT. Two ways for applying this criterion has been set up : one with a high spatial resolution and one at lower spatial resolution.

Application, at both fine and coarse spatial resolutions, of the proposed criterion to two different testchips have been given. Obtained results have demonstrated the soundness of the approach : the use of such criterion can help evaluators at quickly finding EM hotspots in a reduced time. The saved time can dedicated to explore additional settings of EMFI and thus at the enhancement of EMFI characterisations quality.

Even if the obtained results are satisfactory, there is room for further enhancement. Among them, a better and more precise estimation technique of the EM coupling between an EM probe and a DUT seems to be the most promising one. This enhancement way has been left aside during my thesis due to constraint related to Hector research program in which my team was involved. Part of this program has focused on TRNG design and their evaluation. Next chapter is dedicated to the estimate of the robustness of a TRNG and of a random number based countermeasure, namely masking, against EMFI.

Chapter 4

Pulse EMFI effect on a TRNG: A case study on the Delay Chain TRNG

To be robust against cryptanalysis or side channel attacks, cryptographic algorithms or their countermeasure mostly rely on random number generation. This shifts the attacker problematic to first disrupt random number generation before being able to target the cryptographic operation. Again, fault injection shines by its efficiency against such devices and in the particular case of EM fault injection one can cite the following paper [12] where authors were able to bias random number generation.

In 2015 the European project so-called Hardware Enabled Crypto And Randomness (HECTOR) has been initiated to study the generation of random number and evaluate their robustness. As my thesis was done within STMicroelectronics (ST) and that ST was part of the project I took part to the research on the robustness of RNG against fault attack.

As said before using EM fault injection against TRNG has been proven efficient but only in the case of harmonic platform. To fill this lack, we [1] *decided to focus on the use of pulse EMFI to test the target and therefore being able to state whether it can be a threat or not. To be as relevant as possible the targeted TRNG relies on one the most used random generator structure which is Ring oscillators.*

[1] (HECTOR project)

The result of this work has been published at the Fault Diagnostic and Tolerance in Cryptography (FDTC) taking place in 2018 [53]. *Those results showed that as of now pulsed EMFI is not a threat because of the slowness of repetition between two EM shoots. Yet, in the near future platform might be able to reach the throughput of TRNG and in this case care should be taken mostly on the storage of the random value. Indeed, our study showed that it is more relevant to fault the processing of the random value than the physical phenomenon behind the randomness. Where, by relevant we mean that it leads to exploitable faults.*

Contents

4.1 Introduction

This thesis was done as part of the HECTOR project, this acronym stands for Hardware Enabled Crypto And Randomness. This project is an EUROPEAN Horizon 2020 research and innovative program, which started in April 2015 and ended in September 2018. The project aims at studying, designing and implementing Random Number Generator (RNGs) and Physically Uncloneable Functions (PUFs) with demonstrable entropy guarantees and quality metrics. To that end, the project bring together experts from a carefully selected mix of 3 industry (STMicroelectronics, Micronic, Technikon), 3 academia (Katholieshe universität of Leuven, TU Gratz and Université Jean Monnet Saint Etienne) and 2 evaluation lab (Thales, Brightside) to work together. In the context of this thesis the PuF side of the project won't be address. Moreover, the project also focus on the robustness against physical attack of random number generator as well as PUF.

This demand of security regarding random number generator being motivated by the fact that most cryptographic algorithm security relies on random number generation. This either to be robust against cryptanalysis or against side channel attacks, i.e. based on listening to the target channel (current consumption, EM, …). As a reminder by cryptanalysis, I mean the science that aim at breaking cryptographic algorithm without relying on internal state variable as opposed to side channel attacks. In other words, cryptanalysis only relies on input text or cipher text and the mathematics behind the cryptographic algorithm to try to derive the secret part of the algorithm.

Random number generation is therefore a crucial operation for the security of cryptographic functions which makes it a good entry point for potential attacker. In terms of fault injection, almost all medium have been used from LASER to glitch and of course EM field to evaluate the robustness of TRNG. However, in the specific case of EM only harmonic EM effect seems to have been study in the literature. This can be explained by the fact that most random number generators are relying on non locking oscillators to extract randomness. Such a structure being a target of choice for harmonic EMFI as it has been already discussed in the state of the art.

In this chapter we propose to study the use of pulse EMFI on true random generator.

The target will be the delay chain TRNG (*DC-TRNG*) both because it is based on a classical entropy source namely ring oscillators and because it has been designed as part of the HECTOR project. From this study, design guidelines to prevent EMFI fault injection on TRNG will be drawn.

This chapter is organized as follows, first we will give a definition of random number generation. Then the role of random number generation in security will be presented both for side channel and cryptanalysis. In a third time, the TRNG under study will be presented as well as the different experimental protocols it undergoes. To conclude the various results will be introduced and different attack scenarios will be proposed as well as how to circumvent them in the case of our EMFI injection platform.

4.2 Random Number Generation

4.2.1 Definition

Random number generation can be classified in two categories namely the pseudo random generator (*PRNG*) and the true random number generator (*TRNG*). This thesis focuses on true random number generator but such a generator is often not used as is but rather as a seed for a pseudo random generator.

In the case of pseudo random generation, the numbers that constitute the random number sequences are computed using an algorithm. Such sequences have therefore the following properties, there are fast to compute as opposed to true random number generator, they are periodic and deterministic. Moreover, such algorithm use a initialiser often named *seed* to generate the random sequences. Here deterministic means that if the same seed is given twice to the same generator it will output the same random sequence in both cases.

On the other hand, true random number generator should be neither deterministic nor periodic. To that end, they rely at least on one unpredictable physical phenomenon. However, such type of phenomenons being very slow, this impacts the responsiveness of an application using such components. Therefore, to balance these drawbacks TRNG are not

used as is. Instead, they are used as seed of PRNG which are fast to compute. The seed being random, it enables to avoid replaying identical sequence and finding a trade-off between speed and randomness.

Hereafter are listed the most used physical phenomena to generate random number sequences:

1. Jitter: short time deviation of a signal with regard to a time reference. For instance, the phase signal between two oscillators running at the same frequency [83].

2. Metastability ([40]): metastability could occur when timing constraints of latches or DFF are not met. In such a case, the output state of these elements becomes metastable : the output is neither a '0' nor a '1'. Such a state is resolved by noise that randomly forces the output to go to '0' or '1' after an unbounded period.

3. Oscillatory metastability: this phenomenon is really similar to metastability but refers to the situation in which a ring oscillator [41] (or any other oscillating) structure is turned into a latch. In such a case, according to the random start-up of such structures or according to when the stopping signal occurs, the ring oscillator stops randomly in one of the two possible states.

4. Initialization of flip-flops: Random start-up of memory elements : this phenomenon occurs in memory elements which are designed to be symmetrical such as core cells of embedded RAM. Indeed, at the powering of embedded RAM, all its core cells are expected to take a random value due to noise. However, because of process mismatches and power routing unbalances, only part of them have a truly random response. Nevertheless, because process mismatches are unpredictable nor reproducible this phenomenon is commonly used to design physically unclonable functions [39].

5. Chaos theory ?

The work I did during my phd thesis was only focused on a TRNG with jitter as source of randomness. Most of the formerly proposed TRNGs which extract random numbers from the jitter between signals are designed around one or several ring oscillators and of a digitizer as depicted Fig. 4.1. The digitizer usually takes as input a periodic signal uses

a time reference to regularly sample in time the output(s) of the ring oscillator(s) (ROs). Because, the ROs are freely oscillating with regard to the time reference and are characterized by a random drift of their performance, their output are asynchronous regarding the digitizer operation. Hence, when the digitizer samples the RO's outputs the results is expected to be random. The unpredictable elements in such TRNGs are thus the positions of the RO's output edges with regard to the time reference as shown Fig. 4.2.

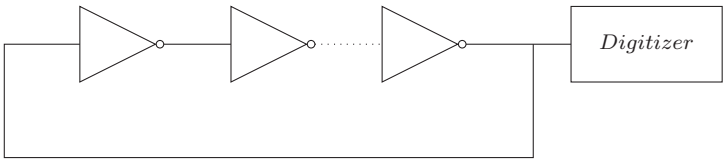


Figure 4.1: RO based TRNG typical architecture

Figure 4.2: Representation of the jitter on the edges of Ring Oscillator's output

If this type of TRNG has been proven efficient to generate random numbers and quite easy to implement, its operation is unfortunately susceptible to harmonic EMFI. Indeed, it has been shown [57] [12] that using this injection method one can lock the phases between several RO's outputs at constant values thus reducing drastically the randomness of the output sequence of numbers. This results in increasing the bias of the TRNG i.e. for instance the ratio between the number of '0' and '1'. This results in increasing the bias of the TRNG i.e. for instance in increasing or decreasing the ratio between the numbers of '0' and '1', ratio which is ideally equal to 1.

4.2.2 Security Use-case

Random number generation is the foundation brick of the security of most signature schemes and in particular the Digital Signing Algorithm (DSA) and its elliptic curves based variant named ECDSA. This algorithm was proposed by the NIST to be used in their Digital Signature Standard. Algorithm parameter are p, q, g where p is prime, q is a prime that divides p-1 and g has order q ($g^q = 1 \, (mod \, p)$). Originally p length was a multiple of 64 between 64 and 512. The private key is denoted as $private$ and the public

key follows $public = g^p \, (mod \, p)$. To sign a message m the signer generates a random value k (called the nonce) where $k < q$ and computes:

$$r = (g^k \, (mod \, p)) mod \, q$$

,

$$s = \frac{SHA(m) + private * r}{k} \, (mod \, q)$$

The verification is then perform according to

$$r = ((g * public)^{w * SHA(m)} \, (mod \, p))(mod \, q)$$

where $w = \frac{1}{s} \, (mod \, q)$.

Many articles warn about the nonce value which, if biased, can become a major flaw for the security of the signature scheme. Maybe the most well-known case of DSA signature broken because of the nonce being not truly random is the Playstation 3 case. In December 2010 a group named *failOverflow* announced the recovery of the private key used by Sony to sign software. This attack being successful because the same random value k was used for all signatures. However, the whole reused of k is not necessary to mount attacks and only knowing some bits of the nonce k can be sufficient as presented in both [65] and [66] for both DSA and ECDSA. Regarding fault injection only voltage spikes seems to have been tested in [63]. In this paper authors show how using voltage spikes they were able to stick some bits provided by the TRNG and finally recover the private key.

Nonetheless random number generation is also used in Side channel analysis countermeasure As a reminder side channel analysis efficiency relies on the possibility to link physical information leaked by the target with intermediate values used by the algorithm. One of the most deployed countermeasure against SCA is masking [19], [36]. The goal of masking is to eradicate this dependency by splitting intermediate values (denoted as V) into $d+1$ shares ($m_1, ..., m_{d+1}$) so that $V = m_1 \odot m_2 \odot ...m_{d+1}$, where $\odot$ denotes a group operation. Those shares $m_1, ..., m_d$ are usually provided by a random number generator, and the share m_{d+1} is then computed as $m_{d+1} = V \odot m_1 \odot ...m_d$. Therefore, either stuck the whole TRNG outputs or biasing its outputs should enable to lower the efficiency of

masking and allowed some combined attacks.

From the two cases presented just before, I decided to consider the three following threat models and to study their implications:

1. *Model 1:* it corresponds to the case of an attacker being of injecting *known* sequences of numbers passing the embedded statistical tests used to internally check the correct operation of the random number generation chain.

2. *Model 2:* it consists in an adversary able to temporarily stuck at least one bit in the sequence of random numbers. Such a situation being hardly detected by embedded tests.

3. *Model 3:* it consists in an adversary able to inject a controlled bias in the random numbers sequence.

The first model has implications on the security level of DSA/ECDSA but also on the efficiency of the masking countermeasure. Indeed, if the adversary is able to force the TRNG/PRNG to provide a known sequence then there are two types of implications according to the considered target. In case it targets the masking countermeasure, forcing the random numbers sequence forces the values of the masks. This is equivalent of suppressing the countermeasure.

In case it targets the DSA, it forces the use of the same sequence of random numbers and this has, as discussed above and in [63], dramatic consequences for the security.

The last threat model may to have a little impact of DSA/ECDSA especially if a PRNG is used in conjunction with the TRNG. However, according to the importance of the injected bias, the effect could be important on the efficiency of the masking countermeasure. This point should also be studied.

This threat as to be moderate since TRNG output are not used as is but often as a seed to PRNG. In the case of ECDSA biasing could also enable to retrieve the private key as referenced in [65] and [66] but those address old restriction of the algorithm considering key length.

4.3 DC-TRNG Case Study

The TRNG under study in this thesis is the Delay Chain TRNG (*DC-TRNG*) (Fig.4.3) which has been designed as part of the Horizon 2020 European project HECTOR. It is characterized by a high throughput a low silicon footprint and exploits the jitter as randomness source.

As shown on Fig. 4.3, the first element is indeed a ring oscillator as we are looking forward to extract randomness from the jitter of such structure. This element then feed a Carry Chain also referred to as Delay Chain. This element is the reason why this TRNG can achieve such a high speed. Carry-chain primitives are originally designed to improve the operation speeds of additions and multiplications on Field Programmable Gate Array (*FPGA*). As their name indicatesi, they are used to propagating the carry during mathematical operations. Therefore, carry-chain primitives are widely available on most FPGA devices from different vendors. They can be cascaded together to form a delay-chain used for sampling the jittery signal. This architecture enables an efficient entropy extraction without the requirement of a high jitter accumulation time. This leads to a high-throughput TRNG.

The digitizer part is composed of a tapped Delay Chain (*DC*), a Priority Encoder (*PE*) and a decimator. This jittered signal provided by the RO propagates through the DC and is sampled by D flip-flops, each of which is attached to a delay element. The expected content of the DFF will be $00...0011...11$ or $11...1100...00$ according to which edge has been captured (rising or falling edge). This imposes the constraint that the DC size has to be at least one period long to be able to capture an edge. Then using the PE this captured edge is encoded on into one bit which value changes according to the position of the edge in the DC chain. The PE filters the ”bubbles” in the code that can be created due to the intrinsic architecture of the carry primitives or/and violating the timing conditions of the flip-flops. The least significant bit of the position is generated as the output of the PE. To mitigate the impact of the low frequency noise, the parity of the distance between two consecutive sample positions is calculated by a decimator to generate a raw random bit.

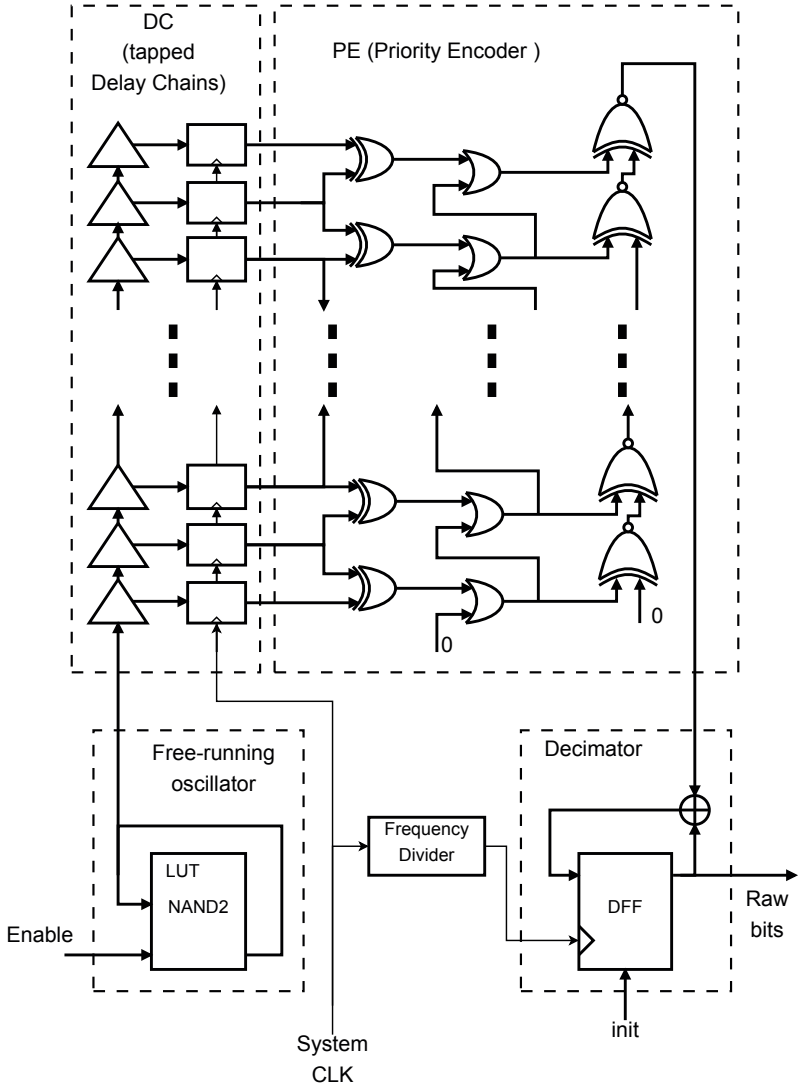


Figure 4.3: Implementation of DC-TRNG.

4.3.1 Experiments

4.3.1.1 Implementation

A DC-TRNG composed of ten carry chains and two decimation stages in cascade was implemented into a Xilinx Spartan-6 FPGA and operates at a clock frequency of $4.5MHz$. The entropy source was instantiated as one Lookup Table (*LUT*) and the delay chain is composed of CARRY4 primitives.

As a reminder our EMFI platform is composed of a pulse generator with a voltage am-

plitude in the range of $0V$ to $400V$ and a pulse width from $8ns$ to $100ns$. The pulse generator can shoot at a frequency of $2kHz$. The experimental campaigns were run with two different EM probes: an U-shape probe and a cylindrical probe with a flat end (Fig. 2.6). The research of the injection probe position leading to fault was done with the flat ended probe. Then in order to produce more localized faults (faults disrupting only one functional block of the DC-TRNG), the U-shape probe was used. Thus faults reported in the next tables were obtained with the U-shape probe.

The DC-TRNG tests were designed to provide an insight into the internal behaviour of its components (namely PE and DC) as well as computations involved in the random stream generation, i.e. decimator stage. Moreover, across all the different tests we ran the PE, DC and RO were placed at the same position. As a first test, the whole design was considered, to monitor PE and Decimator's streams their output were sent directly to the scope alongside their sampling clocks (which was used as a trigger signal for pulse EM generation). Then to be able to separate real faults on TRNG to side effect (i.e. perturbation of I/O bondings) on the design induced by EMFI, output of monitored signals were delayed by several clock cycles using FIFOs as depicted in Fig. 4.4. Therefore, a fault occurring on the DC-TRNG would appear some clock cycles (equal to the length of the FIFO) after pulse generation as illustrated on Fig. 4.5 and Fig. 4.6. By contrast perturbation occurring on IO bonding wires or on our FIFO would appear either instantly or with a smaller delay that of the FIFO.

etecting the occurrence of faults in the numbers stream provided by a TRNG is not straightforward. The way we proceeded is as follows. The DC-TRNG was launched one thousand times for each considered EMFI probe positions and injection time. Then the vectors of binary values collected at the output of the PE (synchronously with the clock) were averaged so that to detect the occurrence of stuck-at faults. Since we used AC coupling, the expected behaviour is an average trace close to 0, i.e. with an equal number of 1 and 0. Otherwise, if a stuck-at 0 fault is systematically induced at a given position and at a given time, a negative extremum will appear (resp. a positive extremum will appear for a stuck-at 1 fault). This procedure is illustrated on Fig. 4.5 in which the orange trace indicates the average vector. On this curve we can observe that EMFI produces a systematic stuck at '0' fault at time samples 25,000.

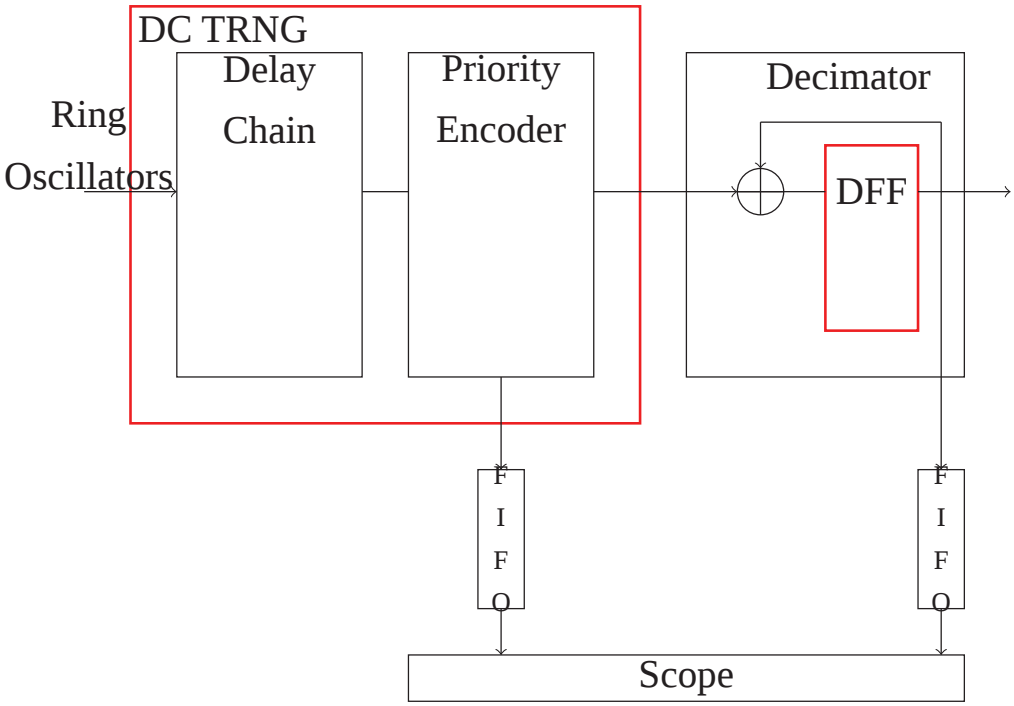


Figure 4.4: FPGA design for experiment 1

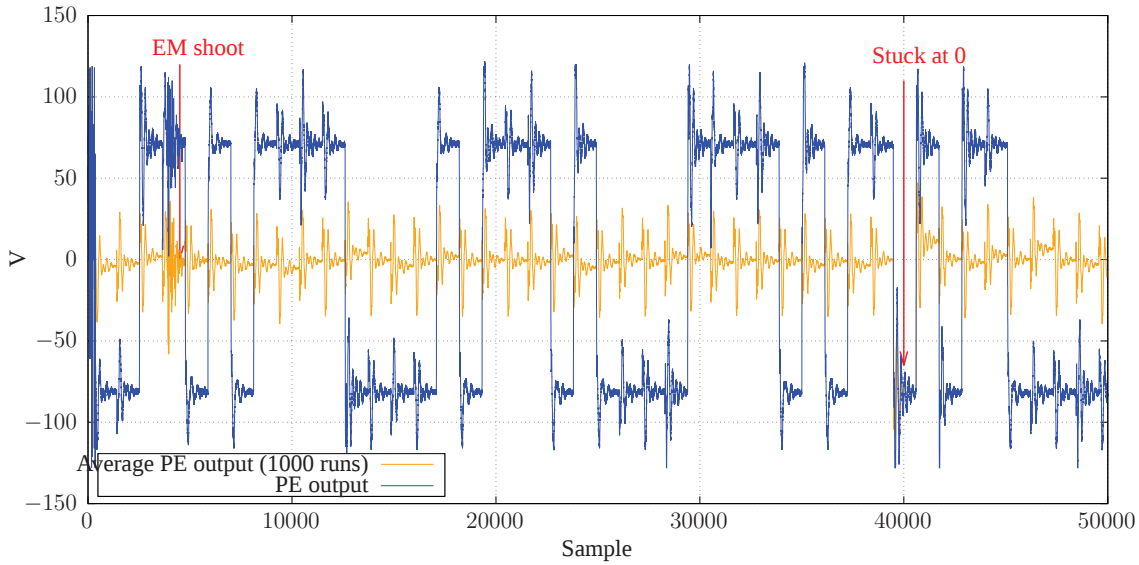


Figure 4.5: The blue curve is the output of one run of the PE and the orange curve is the mean value taken on 1000 curves (on normal behavior 0 is expected). Beware the scope is in AC coupling explaining the expected means of 0 for normal behavior.

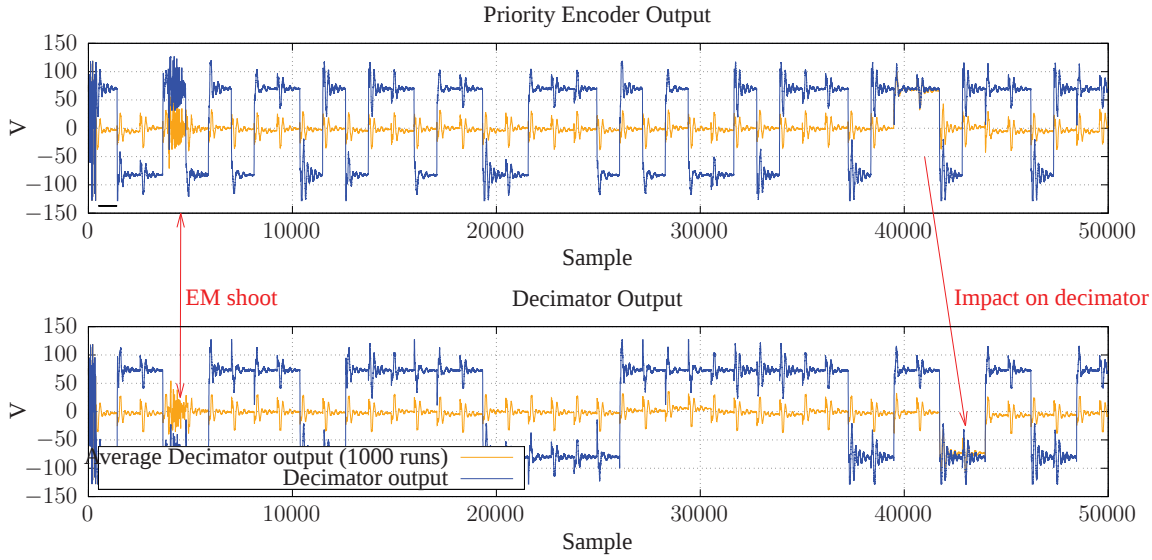


Figure 4.6: On the top graph the blue curve is the output of one run of the PE and the orange curve is the mean value taken on 1000 curves (on normal behavior 0 is expected). For the bottom graph the color code is the same but it is the decimator output. Beware the scope is in AC coupling explaining the expected means of 0 for normal behavior.

Using the above fault detection procedure, the impact of EMFI was analysed at the output of both PE and decimator. Table 4.2 indicates the pulse parameters required to obtained certain types of faults at a specific position and at any time instants. Being able to draw such tables is a direct illustration that an attacker can fully, but temporarily, control the output of TRNGs. This partially sustains the threat model introduced in the preceding section.

Table 4.2 gives the required settings of the EM pulse to obtain different types of faults at the PE's output. However, at that point, for these settings, it is impossible to state if faults occur in the PE or in the carry-chain. The content of the carry-chain was thus monitored in a last experiment (Fig. 4.7).

Fault types	Amplitude	Pulse width	Delay
stuck at 0:	171V	12.6ns	1.12ns
stuck at 1:	179V	12.6ns	1.12ns

Table 4.1: Pulse generator settings to induce faults in the Decimator

Fault types	Amplitude	Pulse width
stuck at 0:	292V	6.45ns
stuck at 1:	274V	6ns
stuck at 00:	356V	7.7ns
stuck at 01:	356V	7.4ns
stuck at 10:	314V	11ns
stuck at 11:	350V	11ns

Table 4.2: Pulse generator settings to induce faults in the PE

This last experiment (Fig. 4.7) focuses on the effect of pulsed EMFI on the DC and PE, thence their stream were output to a serial port. Since this test enables to get DC and PE values, PE's outputs were re-computed off-line using the obtained DC values to decide (by comparison) if a fault occurs in the DC or in the PE.

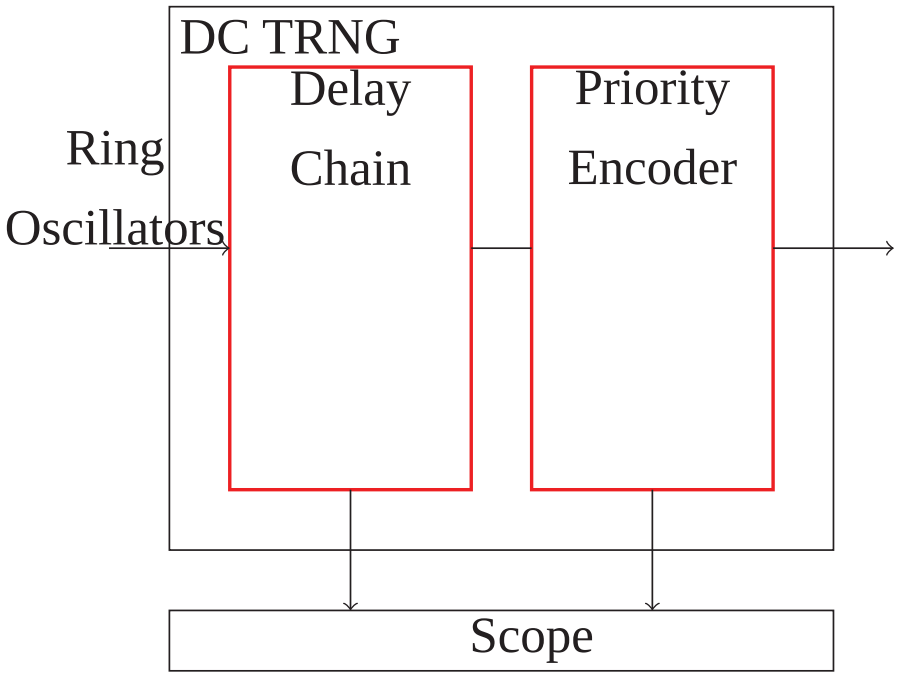


Figure 4.7: FPGA design for experiment 2

This last experiment have demonstrated that EMFI was able to induce stuck at faults in the PE for some probe positions without any effect on the carry chain, but also to disrupt

the carry-chain's content for other positions. However, the fault induced in the carry-chain were uncontrollable because of the asynchronous operation with regard to the master clock of the free-running RO. Indeed, instead of having a vector like ...0000011111... that corresponds to the correct operation of the DC-TRNG, we obtained vectors like ...01...10..11...0..10110...1111 but with the position of the first '1' changing at each run.

To conclude, all these experiments validate a fault model stating that pulse EM can stick one bit at a specific value. This fault model applies on both the PE or the decimator stages and has a bit accuracy. Yet in the particular case of the PE we were able to stuck-at a specific value up to two bits using one injection as illustrated on Fig. 4.6 with the example of stucking two bits at 1. Since the fault impacts two bits of PE's output, we can see its direct effect on the decimator stage.

4.3.2 Security guidelines

These experiments highlight the fact that entropy source is not the only entry point to induce bias in the random numbers flow delivered by a TRNG. Still, today's strategy to harden TRNGs mainly consists in monitoring the statistical properties of the entropy sources. Nonetheless, when dealing with pulsed EMFI, exploitable faults seem to be more easily induced in digital post-processing stages than in the entropy source. Therefore, it seems mandatory to also protect these processing blocks.

As of now lets re-consider our threat model:

1. being able to inject a known pseudo random sequence that can passed embedded statistical test.

2. being able to stick at least one bit.

3. inject bias to circumvent masking scheme.

At a first glance since we are able to stuck bits the first point could have been possible. However, in practice it is not possible since up to date injection platform are stick at some kilohertz compared to the Mega Hertz achieved by the DC-TRNG throughput. Moreover, when shooting at this rate it is more likely that the target device broke very fast. This

makes high speed TRNG a desirable feature against EMFI.

The second point seems more feasible. We are able to stick up to two bits after and before decimation. This makes possible attacks such as the one described in [63], [66] and [65]. However, such attacks should be moderate since they are targeting old implementation with shorter key than recommend now. Speed is not sufficient to cover all attacks and care should be taken with for example adding EMFI sensors to protect the TRNG.

To test whether we are able to inject sufficient bias on a Masking implementation to be able to circumvent this kind of countermeasure we choose the following approach. We first emulate a straightforward attack scenario on the AES Sbox output, and we use simulations based on the Hamming weight leakage model and additive Gaussian noise. And then we simulate the worst case injection scenario in which we were able to shoot at $2Khz$ to find the lowest operating frequency for this kind of TRNG to be protected against EMFI.

The analysis results of the 1st-order Boolean masking scheme supplied with biased random numbers are shown in Fig.4.8. The curves in this figure illustrate the minimum number of measurements needed for successful CPA attack as a function of different noise levels σ and different levels of bias.

In the case of the Inner Product (*IP*) masking scheme [7], the attack's success depends on the type of bias - the attack is more successful if the random numbers are biased towards 0. However, IP masking is more resilient than Boolean masking - the minimum number of measurements that enable the attack for low noise levels and a bias of 25% is 22,000. Moreover, the attack is only successful for a bias level of at least 10%, when it requires approximately 1,250,000 measurements.

We observe that in the case of Boolean masking, both types of biases - towards 0 and towards 1, have the same effect on security degradation, i.e. the curves have very similar shapes. Furthermore, for low noise levels and a bias of 25% towards 1, the number of measurements needed for successful attack is only 225. On the other hand, for high noise levels and bias of only 5% towards 1, the attack can still be successfully mounted, but now requires approximately 650,000 measurements.

From the above analysis, two scenarios can be identified. Firstly, the attacker is able to fully control the random bits flow by disrupting directly the entropy source or its post-processing blocks (the *PE* and the *DC* chain in our case). In such a case, the attacker is thus able to craft a sequence of numbers passing the embedded statistical tests and to insert it once or several times in the random numbers flow. Secondly, the attacker is able to introduce a small bias of at least 10% to be able to successfully run a CPA.

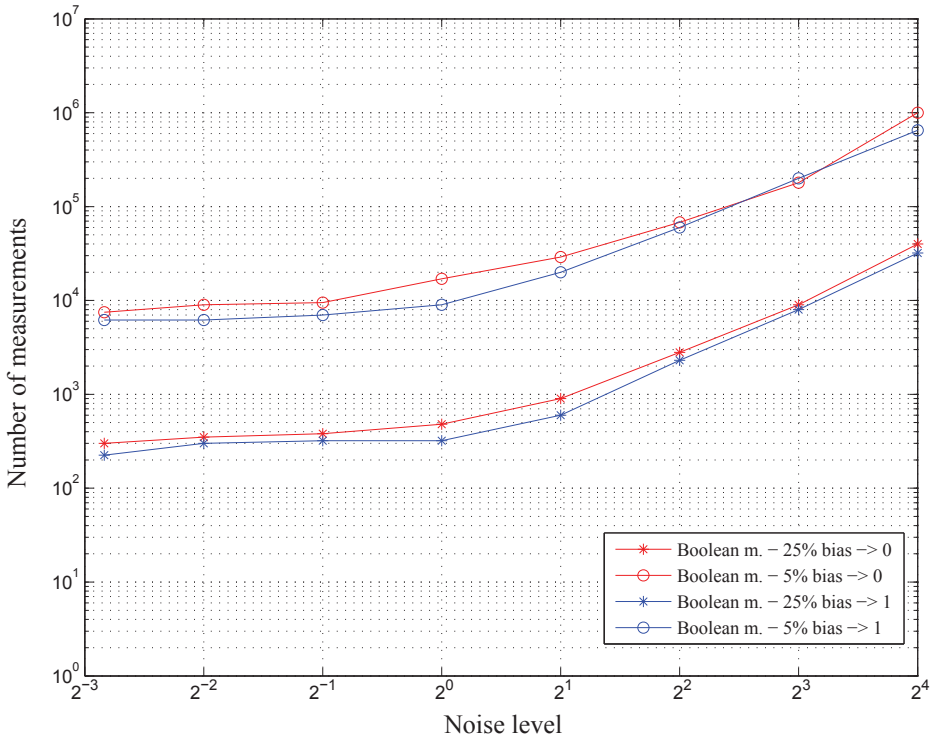


Figure 4.8: Univariate *CPA* attacks against AES Sbox protected by 1st-order Boolean masking scheme.

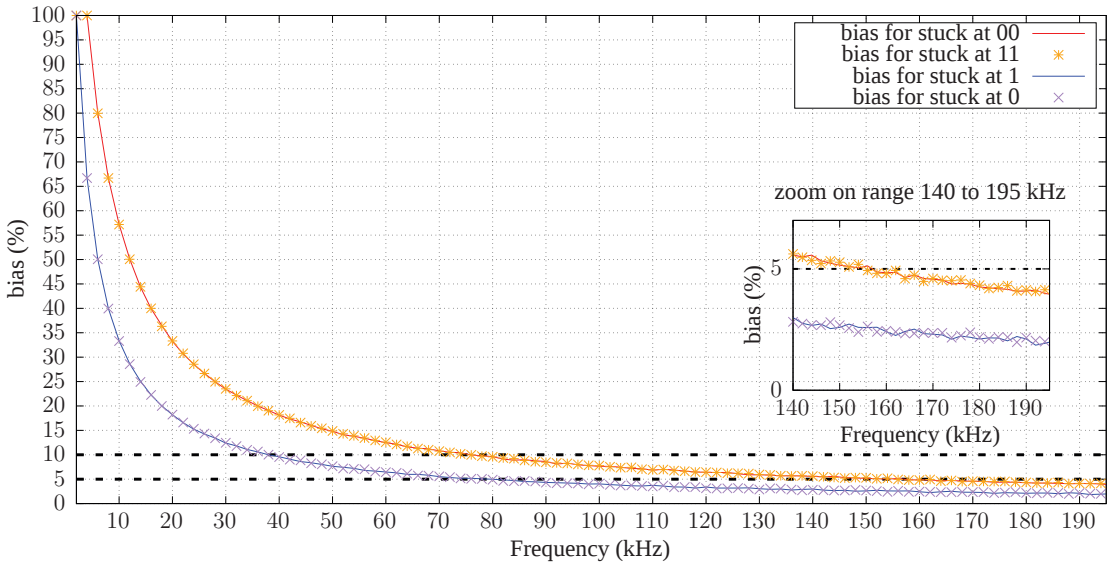


Figure 4.9: Bias on the TRNG random output vs its operating frequency.

4.4 Summary

To conclude, from this analysis a bias level of at least 10% is necessary to lead a successful CPA against a masked implementation. To measure the criticality of modern pulse EMFI platforms against masking scheme relying on RO based TRNGs we computed the bias induced for different values of DC-TRNG's operating frequency and different induced fault types. Moreover, we consider our EMFI platform performing at full speed, i.e. $500\mu s$. From the results reported in Fig. 3.9 it is clear that pulsed EMFI does not constitute a major threat against masking unless the DC-TRNG runs with a frequency lower than 170kHz, However, there is room to increase the repetition rate of EMFI platforms and adding EMFI detectors [33], [89], [60] to protect TRNGs should not be viewed as a luxury solution in the future.

Chapter 5

Conclusion

As of today, more and more day to day objects are embedding cryptography either in hardware or software form. To ensure that once implemented those algorithms are secure, founders or evaluation laboratories perform penetration testing on the DUT.

Amongst the different methods used to recover secrets or gain privileges during those tests lie the fault injection methods. Almost all the members of this family despite their great efficiency suffered from a great flaw when it comes to evaluation. Indeed, such attacks enable many degrees of freedom to the evaluator, while the evaluation is bounded in time. This shifts the choice of determining which parameter to explore to the experience of the evaluator, instead of physical consideration. This might impede the detection of flaws in implementations.

Furthermore to protect cryptographic algorithms against the different families of attacks various countermeasures have been designed. Many countermeasures share a common element which is the use of a True Random Generator as the basis of the security. Therefore, if an attacker can bias the TRNG the whole security foundation will fall apart.

This why the thesis presented in the different chapters a novel way to speed up evaluations by reducing the area to test when using pulse electromagnetic fault injection. The second chapter explored the effect of EMFI on a TRNG and highlighted the best way to bias such structure.

Regarding EM fault injection evaluation, the criterion designed in this thesis address the problematic of performing a whole map of the chip for each set of pulse parameters. This operation being one of the root causes of long time evaluation. To that end the second chapter first defined EM susceptibility of a position as a (X, Y) position that emits strongly at the clock frequency and is bound to a flip-flop operation. This strong link with flip-flops aims at taking into account the sampling fault model which states that flip-flops are more likely to be faulted before the combinatorial part of ICs. On the other hand, looking at strong emissions enable to highlight parts of the circuit that are likely to transfer most of the power of the EM injection into the circuit thence inducing faults more easily. Indeed, from a basic model of the coupling between probes and circuit one can observe some symmetry between emission and reception of an antenna. To find those positions the criterion relies on the Power Spectral Density and the spectral incoherence. This criterion showed encouraging results on two different chips.

Yet, this criterion also showed some flaws which are the alignment between the EM analysis map and the injection one as well as the PSD being far less discriminant than spectral incoherence. In order to enhance our criterion we decide to reason with clusters of points instead of single point value. This having the assets of reducing the effect of bad alignment between maps and to better model EM emissions coupling. Indeed, the probes being quite big compared to the circuit, the emissions at one position are the result of the surrounding positions EM field combines with the position of interest. The efficiency of the two criteria are very similar and tends to state that using PSD as a measure of coupling between probes and circuit is maybe a too naive model.

In the last chapter, the evaluation of up to date TRNG against EMFI is done. The TRNG used for the test is the DC-TRNG which has been designed as part of the European HECTOR project. This evaluation work was also done as part of this project since the thesis was in collaboration with STMicroelectronics. This TRNG is based on a ring oscillator phase jitter as most of the today TRNG which makes it a good target for our analysis. From the evaluation several conclusions can be stated. First, pulse EMFI shoot interval is far too large to be a threat against ”fast” TRNG. Secondly, it seems, from an evaluator/attacker point of view, more efficient to try to disrupt the digitalisation of the random phenomenon used by the TRNG. One of the reasons is that sometimes the fre-

quency can be divided for instance because of decimator stage, the second reason is that the random value is often stored in flip-flop and those elements are known to be easy to fault. Therefore, if EMFI platforms become fast enough to strongly bias TRNG output, designers will have to first take care to protect the digitalisation part rather than focusing on the entropy sources.

Publications

- Maxime Madau, Michel Agoyan, and Philippe Maurine. An EM fault injection susceptibility criterion and its application to the localization of hotspots. In *Smart Card Research and Advanced Applications - 16th International Conference, CARDIS 2017, Lugano, Switzerland, November 13-15, 2017, Revised Selected Papers*, pages 180–195, 2017
- Maxime Madau, Michel Agoyan, Josep Balash, Milos Grujic, Patrick Haddad, Philippe Maurine, Vladimir Rozic, Dave Singelee, Ingrid Verbauwhede, and Bohan Yang. ”the impact of pulsed electromagnetic fault injection on true random number generators”. In *2018 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC*, 2018

Bibliography

[1] Martín Abadi, Mihai Budiu, Úlfar Erlingsson, and Jay Ligatti. Control-flow integrity principles, implementations, and applications. *ACM Trans. Inf. Syst. Secur.*, 13(1):4:1–4:40, 2009.

[2] Giovanni Agosta, Alessandro Barenghi, and Gerardo Pelosi. A code morphing methodology to automate power analysis countermeasures. 06 2012.

[3] Hirotugu Akaike. Akaike's information criterion. In *International Encyclopedia of Statistical Science*, page 25. 2011.

[4] Antoine Amarilli, Sascha Müller, David Naccache, Dan Page, Pablo Rauzy, and Michael Tunstall. Can code polymorphism limit information leakage? In *Information Security Theory and Practice. Security and Privacy of Mobile Devices in Wireless Communication - 5th IFIP WG 11.2 International Workshop, WISTP 2011, Heraklion, Crete, Greece, June 1-3, 2011. Proceedings*, pages 1–21, 2011.

[5] Zelalem Aweke, Salessawi Ferede Yitbarek, Rui Qiao, Reetuparna Das, Matthew Hicks, Yossi Oren, and Todd Austin. Anvil: Software-based protection against next-generation rowhammer attacks, 04 2016.

[6] J. Balasch, D. Arumí, and S. Manich. Design and validation of a platform for electromagnetic fault injection. In *2017 32nd Conference on Design of Circuits and Integrated Systems (DCIS)*, pages 1–6, Nov 2017.

[7] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, and Ingrid Verbauwhede. Theory and practice of a leakage resilient masking scheme. In *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, pages 758–775, 2012.

[8] Alessandro Barenghi, Luca Breveglieri, Israel Koren, Gerardo Pelosi, and Francesco Regazzoni. Countermeasures against fault attacks on software implemented AES: effectiveness and cost. In *Proceedings of the 5th Workshop on Embedded Systems Security, WESS 2010, Scottsdale, AZ, USA, October 24, 2010*, page 7, 2010.

[9] Thierno Barry, Damien Couroussé, and Bruno Robisson. Compilation of a countermeasure against instruction-skip fault attacks. In *Proceedings of the Third Workshop on Cryptography and Security in Computing Systems, CS2@HiPEAC, Prague, Czech Republic, January 20, 2016*, pages 1–6, 2016.

[10] Alberto Battistello and Christophe Giraud. Fault analysis of infective AES computations. In *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography, Los Alamitos, CA, USA, August 20, 2013*, pages 101–107, 2013.

[11] Pierre Bayon, Lilian Bossuet, Alain Aubert, Viktor Fischer, François Poucheret, Bruno Robisson, and Philippe Maurine. Contactless electromagnetic active attack on ring oscillator based true random number generator. In *COSADE*, pages 151–166, 2012.

[12] Pierre Bayon, Lilian Bossuet, Alain Aubert, Viktor Fischer, François Poucheret, Bruno Robisson, and Philippe Maurine. Contactless electromagnetic active attack on ring oscillator based true random number generator. In Werner Schindler and Sorin A. Huss, editors, *Constructive Side-Channel Analysis and Secure Design*, pages 151–166, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[13] Ali Galip Bayrak, Nikola Velickovic, Paolo Ienne, and Wayne Burleson. An architecture-independent instruction shuffler to protect against side-channel attacks. *ACM Trans. Archit. Code Optim.*, 8(4):20:1–20:19, January 2012.

[14] Guido Bertoni, Luca Breveglieri, Israel Koren, Paolo Maistri, and Vincenzo Piuri. Error analysis and detection procedures for a hardware implementation of the advanced encryption standard. *IEEE Trans. Computers*, 52(4):492–505, 2003.

[15] Shivam Bhasin, Jean-Luc Danger, Sylvain Guilley, and Zakaria Najm. NICV: normalized inter-class variance for detection of side-channel leakage. *IACR Cryptology ePrint Archive*, 2013:717, 2013.

[16] Sarani Bhattacharya and Debdeep Mukhopadhyay. Curious case of rowhammer: Flipping secret exponent bits using timing analysis. In *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, pages 602–624, 2016.

[17] Ferdinand Brasser, Lucas Davi, David Gens, Christopher Liebchen, and Ahmad-Reza Sadeghi. Can't touch this: Software-only mitigation against rowhammer attacks targeting kernel memory. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 117–130, Vancouver, BC, 2017. USENIX Association.

[18] J. Breier, S. Bhasin, and W. He. An electromagnetic fault injection sensor using hogge phase-detector. In *2017 18th International Symposium on Quality Electronic Design (ISQED)*, pages 307–312, March 2017.

[19] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards Sound Approaches to Counteract Power-Analysis Attacks. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 398–412. Springer, 1999.

[20] Zhi Chen, Junjie Shen, Alex Nicolau, Alexander V. Veidenbaum, Nahid Farhady Ghalaty, and Rosario Cammarota. CAMFAS: A compiler approach to mitigate fault attacks via enhanced simdization. *IACR Cryptology ePrint Archive*, 2017:1083, 2017.

[21] Yueqiang Cheng, Zhi Zhang, Surya Nepal, and Zhi Wang. Cattmew : Defeating software-only physical kernel isolation. 2019.

[22] L. Chusseau, R. Omarouayache, J. Raoult, S. Jarrix, P. Maurine, K. Tobich, A. Bover, B. Vrignon, J. Shepherd, T. Le, M. Berthier, L. Rivière, B. Robisson, and A. Ribotta. Electromagnetic analysis, deciphering and reverse engineering of integrated circuits (e-mata hari). In *2014 22nd International Conference on Very Large Scale Integration (VLSI-SoC)*, pages 1–6, Oct 2014.

[23] Christophe Clavier. Secret external encodings do not prevent transient fault analysis. In *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, pages 181–194, 2007.

[24] Lucian Cojocar, Kaveh Razavi, Cristiano Giuffrida, and Herbert Bos. Exploiting correcting codes : On the effectiveness of ecc memory against rowhammer attacks. 2018.

[25] Damien Couroussé, Bruno Robisson, Jean-Louis Lanet, Thierno Barry, Hassan Noura, Philippe Jaillon, and Philippe Lalevée. COGITO: code polymorphism to secure devices. In *SECRYPT 2014 - Proceedings of the 11th International Conference on Security and Cryptography, Vienna, Austria, 28-30 August, 2014*, pages 451–456, 2014.

[26] Ang Cui and Rick Housley. BADFET: Defeating modern secure boot using second-order pulsed electromagnetic fault injection. In *11th USENIX Workshop on Offensive Technologies (WOOT 17)*, Vancouver, BC, 2017. USENIX Association.

[27] Amine Dehbaoui, Jean-Max Dutertre, Bruno Robisson, and Assia Tria. Electromagnetic transient faults injection on a hardware and a software implementations of aes. In *FDTC*, pages 7–15, 2012.

[28] Amine Dehbaoui, Victor Lomné, Thomas Ordas, Lionel Torres, Michel Robert, and Philippe Maurine. Enhancing electromagnetic analysis using magnitude squared incoherence. *IEEE Trans. VLSI Syst.*, 20(3):573–577, 2012.

[29] Amine Dehbaoui, Victor Lomné, Philippe Maurine, Lionel Torres, and Michel Robert. Enhancing electromagnetic attacks using spectral coherence based cartography. pages 135–155, 10 2009.

[30] Amine Dehbaoui, Amir-Pasha Mirbaha, Nicolas Moro, Jean-Max Dutertre, and Assia Tria. Electromagnetic glitch on the AES round counter. In *Constructive Side-Channel Analysis and Secure Design - 4th International Workshop, COSADE 2013, Paris, France, March 6-8, 2013, Revised Selected Papers*, pages 17–31, 2013.

[31] I. Diop, P. Liardet, Y. Linge, and P. Maurine. Collision based attacks in practice. In *2015 Euromicro Conference on Digital System Design*, pages 367–374, Aug 2015.

[32] Christoph Dobraunig, Maria Eichlseder, Thomas Korak, Stefan Mangard, Florian Mendel, and Robert Primas. Sifa: Exploiting ineffective fault inductions on symmetric cryptography. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(3):547–572, Aug. 2018.

[33] David El-Baze, Jean-Baptiste Rigaud, and Philippe Maurine. An embedded digital sensor against EM and BB fault injection. In *2016 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2016, Santa Barbara, CA, USA, August 16, 2016*, pages 78–86, 2016.

[34] Benedikt Gierlichs, Jörn-Marc Schmidt, and Michael Tunstall. Infective computation and dummy rounds: Fault protection for block ciphers without check-before-output. In *Progress in Cryptology - LATINCRYPT 2012 - 2nd International Conference on Cryptology and Information Security in Latin America, Santiago, Chile, October 7-10, 2012. Proceedings*, pages 305–321, 2012.

[35] Christophe Giraud. DFA on AES. In *Advanced Encryption Standard - AES, 4th International Conference, AES 2004, Bonn, Germany, May 10-12, 2004, Revised Selected and Invited Papers*, pages 27–41, 2004.

[36] Louis Goubin and Jacques Patarin. DES and Differential Power Analysis (The "Duplication" Method). In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES'99*, volume 1717 of *Lecture Notes in Computer Science*, pages 158–172. Springer, 1999.

[37] Daniel Gruss, Moritz Lipp, Michael Schwarz, Daniel Genkin, Jonas Juffinger, Sioli O'Connell, Wolfgang Schoechl, and Yuval Yarom. Another flip in the wall of rowhammer defenses. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 245–261, 2018.

[38] Daniel Gruss, Clémentine Maurice, and Stefan Mangard. Rowhammer.js: A remote software-induced fault attack in javascript. In *Detection of Intrusions and Malware, and Vulnerability Assessment - 13th International Conference, DIMVA 2016, San Sebastián, Spain, July 7-8, 2016, Proceedings*, pages 300–321, 2016.

[39] Jorge Guajardo, Sandeep S. Kumar, Geert Jan Schrijen, and Pim Tuyls. FPGA intrinsic pufs and their use for IP protection. In *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, pages 63–80, 2007.

[40] Mike Hamburg, Paul Kocher, and Mark E Marson. Analysis of intel's ivy bridge digital random number generator prepared for intel by. 04 2019.

[41] Laszlo Hars. Random number generation based on oscillatory metastability in ring circuits. *IACR Cryptology ePrint Archive*, 2011:637, 2011.

[42] Naofumi Homma, Yu-ichi Hayashi, Noriyuki Miura, Daisuke Fujimoto, Daichi Tanaka, Makoto Nagata, and Takafumi Aoki. EM attack is non-invasive? - design methodology and validity verification of EM attack sensor. In *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings*, pages 1–16, 2014.

[43] Gorka Irazoqui, Thomas Eisenbarth, and Berk Sunar. MASCAT: stopping microarchitectural attacks before execution. *IACR Cryptology ePrint Archive*, 2016:1196, 2016.

[44] Michael Hutter Jörn-Marc Schmidt. Optical and em fault-attacks on crt-based rsa: Concrete results. In *Proceedings of the 15th Austrian Workhop on Microelectronics*, 2007.

[45] Ramesh Karri, Grigori Kuznetsov, and Michael Gössel. Parity-based concurrent error detection of substitution-permutation network block ciphers. pages 113–124, 09 2003.

[46] Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji-Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *ACM/IEEE 41st International Symposium on Computer Architecture, ISCA 2014, Minneapolis, MN, USA, June 14-18, 2014*, pages 361–372, 2014.

[47] Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *CRYPTO '96: Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, pages 104–113, 1996.

[48] Benjamin Lac, Anne Canteaut, Jacques J. A. Fournier, and Renaud Sirdey. Thwarting fault attacks against lightweight cryptography using SIMD instructions. In *IEEE International Symposium on Circuits and Systems, ISCAS 2018, 27-30 May 2018, Florence, Italy*, pages 1–5, 2018.

[49] Marc Lacruche and Philippe Maurine. Electromagnetic activity vs. logical activity: Near field scans for reverse engineering. In *Smart Card Research and Advanced Applications, 17th International Conference, CARDIS 2018, Montpellier, France, November 12-14, 2018, Revised Selected Papers.*, pages 140–155, 2018.

[50] Jean-François Lalande, Karine Heydemann, and Pascal Berthomé. Software countermeasures for control flow integrity of smart card C codes. In *Computer Security - ESORICS 2014 - 19th European Symposium on Research in Computer Security, Wroclaw, Poland, September 7-11, 2014. Proceedings, Part II*, pages 200–218, 2014.

[51] Moritz Lipp, Misiker Tadesse Aga, Michael Schwarz, Daniel Gruss, Clémentine Maurice, Lukas Raab, and Lukas Lamster. Nethammer: Inducing rowhammer faults through network requests. *CoRR*, abs/1805.04956, 2018.

[52] Victor Lomné, Thomas Roche, and Adrian Thillard. On the need of randomness in fault attack countermeasures - application to AES. In *2012 Workshop on Fault Diagnosis and Tolerance in Cryptography, Leuven, Belgium, September 9, 2012*, pages 85–94, 2012.

[53] Maxime Madau, Michel Agoyan, Josep Balash, Milos Grujic, Patrick Haddad, Philippe Maurine, Vladimir Rozic, Dave Singelee, Ingrid Verbauwhede, and Bohan Yang. "the impact of pulsed electromagnetic fault injection on true random number generators". In *2018 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC*, 2018.

[54] Maxime Madau, Michel Agoyan, and Philippe Maurine. An EM fault injection susceptibility criterion and its application to the localization of hotspots. In *Smart Card Research and Advanced Applications - 16th International Conference, CARDIS 2017, Lugano, Switzerland, November 13-15, 2017, Revised Selected Papers*, pages 180–195, 2017.

[55] Paolo Maistri and Régis Leveugle. Double-data-rate computation as a countermeasure against fault analysis. *IEEE Trans. Computers*, 57(11):1528–1539, 2008.

[56] Paolo Maistri, Pierre Vanhauwaert, and Régis Leveugle. A novel double-data-rate AES architecture resistant against fault injection. In *Fourth International Workshop*

on Fault Diagnosis and Tolerance in Cryptography, 2007, FDTC 2007: Vienna, Austria, 10 September 2007, pages 54–61, 2007.

[57] A. Theodore Markettos and Simon W. Moore. The frequency injection attack on ring-oscillator-based true random number generators. In Christophe Clavier and Kris Gaj, editors, *Cryptographic Hardware and Embedded Systems - CHES 2009*, pages 317–331, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[58] A. Theodore Markettos and Simon W. Moore. The frequency injection attack on ring-oscillator-based true random number generators. In *Cryptographic Hardware and Embedded Systems - CHES 2009, 11th International Workshop, Lausanne, Switzerland, September 6-9, 2009, Proceedings*, pages 317–331, 2009.

[59] Philippe Maurine. Techniques for em fault injection: Equipments and experimental results. pages 3–4, 09 2012.

[60] Noriyuki Miura, Zakaria Najm, Wei He, Shivam Bhasin, Xuan Thuy Ngo, Makoto Nagata, and Jean-Luc Danger. Pll to the rescue: a novel em fault countermeasure. pages 1–6, 06 2016.

[61] Nicolas Moro, Amine Dehbaoui, Karine Heydemann, Bruno Robisson, and Emmanuelle Encrenaz. Electromagnetic fault injection: towards a fault model on a 32-bit microcontroller. *CoRR*, abs/1402.6421, 2014.

[62] Nicolas Moro, Karine Heydemann, Emmanuelle Encrenaz, and Bruno Robisson. Formal verification of a software countermeasure against instruction skip attacks. *J. Cryptographic Engineering*, 4(3):145–156, 2014.

[63] David Naccache, Phong Q. Nguyen, Michael Tunstall, and Claire Whelan. Experimenting with faults, lattices and the DSA. *IACR Cryptology ePrint Archive*, 2004:277, 2004.

[64] Laurence W. Nagel and D.O. Pederson. Spice (simulation program with integrated circuit emphasis). Technical Report UCB/ERL M382, EECS Department, University of California, Berkeley, Apr 1973.

[65] Phong Q. Nguyen and Igor E. Shparlinski. The insecurity of the digital signature algorithm with partially known nonces. *J. Cryptology*, 15(3):151–176, 2002.

[66] Phong Q. Nguyen and Igor E. Shparlinski. The insecurity of the elliptic curve digital signature algorithm with partially known nonces. *Des. Codes Cryptography*, 30(2):201–217, 2003.

[67] R. Omarouayache, J. Raoult, S. Jarrix, L. Chusseau, and P. Maurine. Magnetic microprobe design for em fault attack. In *emceurope*, 2013.

[68] Rachid Omarouayache. Near-field electromagnetic imagery at millimeter waves and microwave near-field injection. Master's thesis, École Doctorale Information, Structures, Systèmes (Montpellier), 12 2015.

[69] Rachid Omarouayache, J Raoult, Sylvie Jarrix, Laurent Chusseau, and Philippe Maurine. Magnetic microprobe design for em fault attack. pages 949–954, 01 2013.

[70] Sébastien Ordas, Ludovic Guillaume-Sage, and Philippe Maurine. EM injection: Fault model and locality. In *2015 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2015, Saint Malo, France, September 13, 2015*, pages 3–13, 2015.

[71] Sébastien Ordas. Evaluation of low power methods against hardware attacks. Master's thesis, École Doctorale Information, Structures, Systèmes (Montpellier), 11 2015.

[72] Kyungbae Park, Chul Seung Lim, Donghyuk Yun, and Sanghyeon Baeg. Experiments and root cause analysis for active-precharge hammering fault in ddr3 sdram under 3x nm technology. *Microelectronics Reliability*, 57:39–46, 2016.

[73] Sikhar Patranabis, Abhishek Chakraborty, and Debdeep Mukhopadhyay. Fault tolerant infective countermeasure for AES. In *Security, Privacy, and Applied Cryptography Engineering - 5th International Conference, SPACE 2015, Jaipur, India, October 3-7, 2015, Proceedings*, pages 190–209, 2015.

[74] Stjepan Picek, Lejla Batina, Pieter Buzing, and Domagoj Jakobovic. Fault injection with a new flavor: Memetic algorithms make a difference. In *Constructive Side-Channel Analysis and Secure Design - 6th International Workshop, COSADE 2015, Berlin, Germany, April 13-14, 2015. Revised Selected Papers*, pages 159–173, 2015.

[75] Gilles Piret and Jean-Jacques Quisquater. A differential fault attack technique against SPN structures, with application to the AES and KHAZAD. In *Cryptographic Hardware and Embedded Systems - CHES 2003, 5th International Workshop, Cologne, Germany, September 8-10, 2003, Proceedings*, pages 77–88, 2003.

[76] François Poucheret, Karim Tobich, Mathieu Lisart, Laurent Chusseau, Bruno Robisson, and Philippe Maurine. Local and direct em injection of power into cmos integrated circuits. In *FDTC*, pages 100–104, 2011.

[77] Rui Qiao and Mark Seaborn. A new approach for rowhammer attacks. In *2016 IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2016, McLean, VA, USA, May 3-5, 2016*, pages 161–166, 2016.

[78] JJ Quisquater and D Samyde. Eddy current for magnetic analysis with active sensor. In *Proceedings of ESmart 2002*, page pp 185 – 194. Eurosmart, 2002.

[79] Denis Réal, Frédéric Valette, and M'hamed Drissi. Enhancing correlation electromagnetic attack using planar near-field cartography. In *Design, Automation and Test in Europe, DATE 2009, Nice, France, April 20-24, 2009*, pages 628–633, 2009.

[80] Laurent Sauvage, Sylvain Guilley, Florent Flament, Jean-Luc Danger, and Yves Mathieu. Blind cartography for side channel attacks: Cross-correlation cartography. *Int. J. Reconfig. Comp.*, 2012:360242:1–360242:9, 2012.

[81] Laurent Sauvage, Sylvain Guilley, Florent Flament, Jean-Luc Danger, and Yves Mathieu. Blind cartography for side channel attacks: Cross-correlation cartography. *Int. J. Reconfig. Comp.*, 2012:360242:1–360242:9, 2012.

[82] Tobias Schneider and Amir Moradi. Leakage assessment methodology - A clear roadmap for side-channel evaluations. In *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*, pages 495–513, 2015.

[83] B. Sunar, W. J. Martin, and D. R. Stinson. A provably secure true random number generator with built-in tolerance to active attacks. *IEEE Transactions on Computers*, 56(1):109–119, Jan 2007.

[84] Andrei Tatar, Radhesh Krishnan Konoth, Elias Athanasopoulos, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi. Throwhammer: Rowhammer attacks over the network and defenses. In *2018 USENIX Annual Technical Conference, USENIX ATC 2018, Boston, MA, USA, July 11-13, 2018.*, pages 213–226, 2018.

[85] Colin D. Walter. Data integrity in hardware for modular arithmetic. In *Cryptographic Hardware and Embedded Systems - CHES 2000, Second International Workshop, Worcester, MA, USA, August 17-18, 2000, Proceedings*, pages 204–215, 2000.

[86] K. Wilken and J. P. Shen. Continuous signature monitoring: efficient concurrent-detection of processor control errors. In *International Test Conference 1988 Proceeding New Frontiers in Testing*, pages 914–925, Sep. 1988.

[87] Chih-Hsu Yen and Bing-Fei Wu. Simple error detection methods for hardware implementation of advanced encryption standard. *IEEE Trans. Computers*, 55(6):720–731, 2006.

[88] Loic Zussa, Amine Dehbaoui, Karim Tobich, Jean-Max Dutertre, Philippe Maurine, Ludovic Guillaume-Sage, Jessy Clédière, and Assia Tria. Efficiency of a glitch detector against electromagnetic fault injection. In *DATE*, pages 1–6, 2014.

[89] Loic Zussa, Amine Dehbaoui, Karim Tobich, Jean-Max DUTERTRE, Philippe Maurine, Ludovic Guillaume-Sage, Jessy Clédière, and Assia Tria. Efficiency of a glitch detector against electromagnetic fault injection. In *DATE: Design, Automation and Test in Europe*, Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014, pages 1–6, Dresden, Germany, March 2014. IEEE.