Floorplanning as a practical countermeasure against clock fault attack in Trivium stream cipher

F.E. Potestad-Ordóñez, C.J. Jiménez-Fernández, C. Baena-Oliva, P. Parra-Fernández, M. Valencia-Barrero Instituto de Microelectrónica de Sevilla/Universidad de Sevilla (CSIC-CNM)
Sevilla, España
Email: {potestad,cjesus,parra}@imse-cnm.csic.es,{baena,manolov}@dte.us.es

***Abstract*—The fault injection in ciphers operation is a very successful mechanism to attack them. The inclusion of elements of protection against this kind of attacks is more and more necessary. These mechanisms are usually based on introducing redundancy, which leads to a greater consumption of resources or a longer processing time. This article presents how the introduction of placement restrictions on ciphers can make it difficult to inject faults by altering the clock signal. It is therefore a countermeasure that neither increases the consumption of resources nor the processing time. This mechanism has been tested on FPGA implementations of the Trivium cipher. Several tests have been performed on a Spartan 3E device from Xilinx and the experimental measurements have been carried out with ChipScope Pro. The tests showed that an adequate floorplanning is a good countermeasure against these kind of attacks.**

***Index Terms*—Fault Attack, Stream Cipher, Trivium, FPGA implementation, countermeasure.**

I. Introduction

The amount of information exchanged daily in communication networks is growing in an exponential way. A substantial percentage of this information is sensitive. It must therefore be protected in order to avoid malicious use. Cryptography provides techniques, mechanisms and tools for confidentiality, authentication, integrity and non-repudiation. It has been applied for data exchange across large systems. However, nowadays there is an increasing number of devices with scarce resources and low power consumption, e.g. those suitable for the Internet of Things (IoT). In [1] and [2], the authors pointed out the importance of challenges related to security in the IoT, showing the relevance of secure implementations under strong constraints of power consumption and area, among other aspects. Indeed, the number of lightweight cryptography implementations has significantly increased in recent years. In this context, the use of stream ciphers is one of the best options because of their limited impact on the available resources. However, the behaviour of a device must be first comprehensively studied and tested against different attacks in order to know its security level.

With the aim of carrying out security improvements in crypto systems, like the stream cipher Trivium [3], it is necessary to test them and study their vulnerabilities. Since the work presented by Boneh $et\ al.$ [4], one of the most widely used methods for studying vulnerabilities of crypto systems is the use of side channel attacks. Boneh $et\ al.$ presented a theoretical model to break cryptographic circuits based on random fault injection on hardware implementations. They proved that some encryption algorithms are vulnerable against fault injection attacks. Street $et\ al.$ [5] presented a review of low-cost attacks based on fault analysis for Smart Cards as well as for different cryptographic systems (RSA, DES and other block ciphers). By applying Differential Fault Analysis, they retrieved the key of the DES cipher by using between one and ten ciphertexts. These works showed the fault attack effectiveness over cryptographic systems through the study of faulty system outputs.

After these first works, the study of the vulnerabilities of the cryptographic devices is performed in theoretical mode using mathematical formulations known as Differential Fault Analysis (DFA). The works [6]–[8] applies DFA for the Trivium stream cipher. The possibility of retrieving the secret key from a transient fault injection in the cipher inner state is studied in these works. In the attack scenario, the main assumption is the possibility of injecting one faulty bit in the internal register of the cipher, from which the attacker is able to determine the cipher inner state in a specific clock cycle thanks to the difference between the proper and faulty key stream.

In order to perform these kind of fault injections there are different techniques previously presented in the literature. In [9] and [10] a guide of the main techniques is presented. One of the most used is the manipulation of the system clock signal [5], [11], [12]. This low cost technique allows to introduce transient faults in crypto systems in an effective way through the increase of the clock frequency in a specific clock cycle with a short clock pulse, which causes that the circuits of the design do not work properly, producing setup and hold violations.

Once that the cryptographic device vulnerabilities are determined, the next point of interest is the countermeasure. The works [9], [10], [13] present different hardware countermeasures like component redundancy, but in these works the possibility of using a correct design and implementation of the crypto circuit as a countermeasure to reduce the vulnerability it is not considered. Due to that, a study of strategic floorplanning as a practical countermeasure is presented in this paper. This countermeasure increases the robustness of the crypto system against fault attacks.

A. Previous works

In previous works different studies have been presented about the Trivium stream cipher vulnerability to fault attacks. In [14]–[16], the vulnerability of FPGA implementations of the Trivium stream cipher against the fault injection by the manipulation of the clock signal is studied in a practical way. These studies conclude that the most vulnerable elements of the internal state for the introduction of faults are the flip-flops whose input comes from feedback signals. These signals are generated by combinational functions and therefore have a greater delay. In addition, the vulnerability is independent of the key and the initialization vector.

B. Our contribution

This paper presents a study of the relationship between the placement of the cryptographic circuit and its vulnerability to fault attacks. If some positioning constraints are imposed on the place & route tool, it is possible to reduce the delays in the feedback signals, increase the maximum operating frequency and, therefore, make it difficult to inject faults in the circuit. Due to that, in this paper the correct floorplanning of the circuit components is considered as an active countermeasure against fault attack. These conclusions are verified by experimental measurements of FPGA implementations of the cryptographic circuit.

C. Paper organization

The rest of the paper is organized as follows. Section II presents briefly the Trivium stream cipher. Section III introduces fault attacks and vulnerabilities of Trivium cipher against those attacks. Section IV presents the floorplanning as a practical countermeasure and why it is a good option in this scenario. Section V presents experimental results, with several floorplanning options and comparisons between results obtained from timing simulations. Finally, some conclusions are presented in Section VI.

II. Trivium Stream Cipher description

The Trivium stream cipher [3] is one of the eSTREAM project finalist. It is a synchronous cipher designed to generate up to 2^{64} bits of key stream from an 80-bit secret key and an 80-bit initialization vector (IV). The cipher architecture is based on three shift registers with 288 bits in total, as well as combinational logic to provide feedback. Like in other synchronous stream ciphers, the underlying algorithm begins with the load of 288 bits into the shift register (internal state) including one secret key, one initialization vector and a stream of zeros and ones. Before generating a valid key stream, the cipher needs to run during 1152 clock cycles. From that moment on, it generates a valid pseudo-random bit sequence.

The 288 bits of the internal state are distributed along three shift registers with different lengths. The first shift register has 93 bits, the second one is made up of 83 bits and the third has 111 bits. The feedback for each shift register is generated with AND and XOR operations. The key stream is the result of XOR operations on some bits in the shift register. Fig. 1 shows the schematic of the Trivium internal structure.

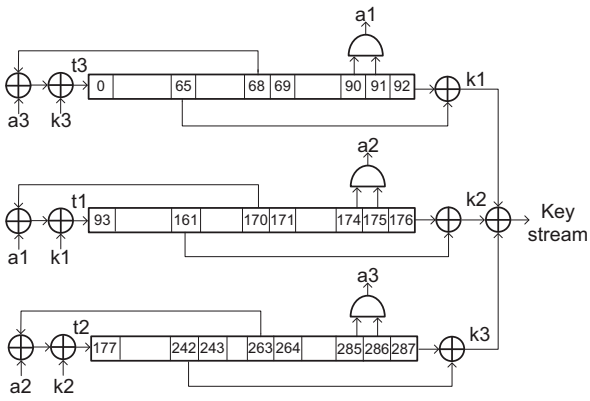

Fig. 1. Schematic representation of the Trivium stream cipher.

III. Fault attacks and cipher vulnerabilities

As explained in Section I, one of the most useful methods in order to study the cipher vulnerabilities is to perform side channel attacks. In works [14]–[16] the Trivium stream cipher vulnerabilities against clock signal attacks are studied. This technique allows the fault injection inside the inner state of the cipher Trivium, thanks to that it is possible to retrieve secret information that endanger the system security. This technique consists on introducing short pulses in the clock signal of the cipher. These short pulses in the clock imply a frequency above the maximum operating frequency of the cipher. With these attacks, errors in the transmission of data between the flip-flops are produced.

In [16] it is shown that it is possible to produce fault injections in experimental mode. This system consists of a state machine that allows the selection of a random secret key and initialization vector, insertion clock cycles and result sampling after the attacks. The creation of the short clock pulses is achieved through the combination of two clock signals, one of them fast and the other slow, where multiplexing both signals allows to introduce short pulses. The generation of both signals is done by using a device provided by Xilinx called Digital Clock Manager (DCM), which allows to produce clocks with different frequencies from an input clock. The frequencies of the generated clocks can be lower but also higher than the frequency of the input clock.

On the other hand, in order to multiplexing the clock signals, another component provided by Xilinx is used, this component is called $BUFGMUX$. It is a special multiplexer used to switch between different clock signals and guarantees the generation of a clock signal without glitches.

One important factor when the fault attacks are carried out on Trivium stream cipher is its own internal structure. Implemented by nonlinear shift registers, this cipher is able to work at frequencies very near to the maximum frequency of the technology where is implemented. This involves the problem of having to generate short pulses with frequencies faster than the maximum supported by the device where the

TABLE I
FAULTS POSITIONS ON THE CIPHERS FOR EACH KEY/IV PAIR AND INSERTION CYCLE.

	Key 1 IV 1		Key 1 IV 2		Key 2 IV 1		Key 2 IV 2	
Insertion cycles	Trivium 1	Trivium 2	Trivium 1	Trivium 2	Trivium 1	Trivium 2	Trivium 1	Trivium 2
1200	0	-	1	1	93	-	-	-
1300	1/0	1	1	1	1	-	-	-
1500	93/1/0	1/0	0	-	1	1	1	1
1750	0	0	1	-	0	-	0	-

cipher is implemented. In this case, transitions with times slightly faster than the maximum times allowed by the FPGA are necessary. Therefore the fault injection must be done very carefully, ensuring that small pulses are not filtered by the FPGA. In the experimental tests, the maximum frequency of the Spartan-3E FPGA is 311 MHz while the frequency used to inject faults is 316.66 MHz.

Table I shows the positions of the internal state in which faults have been introduced after the attacks carried out in four different insertion cycles. This Table shows the results obtained from two copies of the Trivium (Trivium 1 and Trivium 2) implemented using the defaults options of the place & route tool. Both Triviums are subjected to the same attacks, under the same conditions. The cases where a $''-''$ appears mean that after the attacks no errors were injected into the internal state. On the other hand, when more than one number appear indicates that after an attack, the faults were introduced in several positions at the same time.

As it can be seen, for different pairs of key and IV and different clock cycles, it is possible to inject faults in the internal state of the Trivium cipher. The results show that the flip-flops that tend to fail are those whose input signal comes from feedback lines or the next flip-flop. This is because combinational operations introduce a greater delay to those inputs. These flip-flops are at positions 0, 93 and 177, although in the tests performed this last flip-flop did not fail. In the FPGA, the clock signals are distributed by special lines designed for low skew and high fanout, but the signals for data transmission have a bigger delay they could even have a quite different delay between them. If the route of the signals is not balanced or well distributed, the length of the signal that connects the output of one flip-flop with the next one in the shift register could be an important factor from the vulnerability point of view.

If it becomes more difficult to insert faults in flip-flop with inputs from feedback, then the cipher vulnerability to this type of attack can be reduced. One way to achieve this is to introduce placement restrictions, both on the cipher and on the other circuits of the system.

IV. FLOORPLANNING AS A COUNTERMEASURE

The floorplanning of a circuit affects its timing behaviour. Therefore, as it has been mentioned above, floorplanning options could reduce delay times making the fault injection more difficult, and thus serve as a countermeasure against fault attack. This countermeasure is based on reducing the routing distances between flip-flops with critical paths reducing the delay time of that path.

The constraints and rules used to carry out the implementation have a great importance when a system is implemented on FPGA. So it is very important to know the guidelines to be followed in order to drive the synthesis and place & route tools towards the target. In most cases, the hardware designer does not impose good constraints, which results in a bad placement from the security point of view. The tools try to implement the system complying with timing constraints and using as few resources as possible. However, for FPGA technologies, placement tools tend to extend the design over the entire device. The hardware designer of crypto-circuits must take into account the place & route not only in terms of efficiency, but also in terms of cryptographic security.

In order to improve cryptographic security, it is possible to use timing and/or floorplanning constraints to guide the place & route tool. Timing restrictions are the most common mechanism to achieve a reduction in delays on critical paths. In this article we focus on how placement restrictions can improve performance. As an example, we have studied the position of flip-flop 0 in the FPGA device. As it can be seen in Fig. 1 the input value of flip-flop 0 is a logic operation that depends on the output of flip-flops 287, 285, 286, 242 and 68. Fig. 2 (a) shows how these flip-flops are placed in a dispersed area on the FPGA when floorplanning is free (there is not placement restrictions). On the contrary, Fig. 2 (b) shows that these flip-flops are placed closer on the FPGA when the aim is to protect the system against attacks. For this particular example, the PlanAhead tool has been used to define an area for the position of these critical flip-flops.

As it can be seen, both placements are different, being in the case (a) a design with big data transmission delays at the input of the flip-flop 0. In the case (b) having guided the floorplanning, the flip-flops are placed in a smaller area and closer to each other. In this case the timing delay is smaller than in the case (a) and the fault attack is more difficult. The next section presents different floorplanning strategies and explores the effectiveness of some of them to avoid attacks by faults injections.

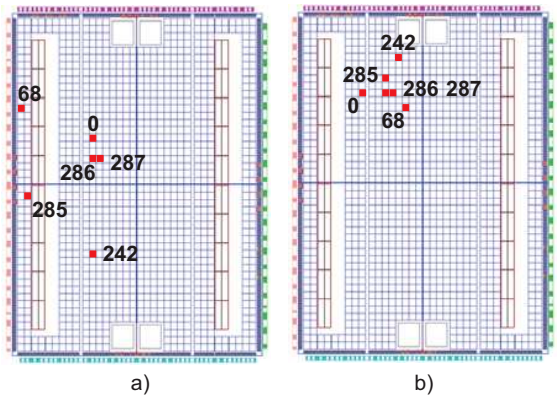

Fig. 2. Example detail of the positions between flip-flop 0 and its critical feedback flip-flops.

V. Results

This section presents the results of four FPGA floorplanning strategies for the Trivium stream cipher. The implementations have been done in a Spartan 3E FPGA. The implemented design includes, in addition to the Trivium, a state machine that controls the load of the key and IV, the Trivium operation and the injection of the fault by inserting a small pulse into the clock signal. The generation of floorplanning restrictions has been done with the Xilinx PlanAhead tool, which allows to select the area of the device where the design is placed. The place & route has been made within the Xilinx ISE environment and to check the positions in which the designs have been placed, the FPGA Editor tool from Xilinx has been used. The analysis of the injected faults, and the positions in which they have been introduced, have been done by sampling the data from the Trivium state register using ChipScope Pro. The timing simulations have been carried out with ISIM.

The FPGA implementation of the Trivium stream cipher can work at the maximum frequency of the device. Therefore, to inject faults in the cipher internal state could be necessary to exceed the frequency limitations of the device. Although difficult, it is possible to inject faults if the maximum operation frequency of the cipher is slightly above the maximum frequency of the device. But if the maximum frequency of operation of the cipher is well above the maximum frequency of the device, then it is not possible to inject faults.

Implementations have been made with four different floorplanning restrictions, which have been called free floorplan, bad floorplan, forced floorplan and strategic floorplan. The results obtained from these restrictions have allowed to determine the guidelines to follow for a correct floorplanning. The results obtained are presented in five subsections. Subsections A to D describe each FPGA floorplan strategies, while subsection E presents a comparison of maximum frequencies obtained by timing simulation.

For each floorplanning restriction, 100 tests have been carried out experimentally. In those tests, a small pulse has been introduced into the clock signal of the cipher and it has been checked whether faults have been introduced and their position.

A. Free floorplan

The first floorplanning strategy is to impose no placement constraint to the design. Fig. 3 shows a representation of the area of the device on the left and the placement obtained by the free floorplanning strategy on the right. White dots indicate a used CLB.

As it can be seen in this Figure, the design has been placed using most of the available area of the device. As it has been explained above, this placement causes that critical components are separated by large distances. The distance between all components is also big. The tool does not reduce more the delays because the circuit operates at the maximum frequency of the device in which it has been implemented. The experimental tests show that this implementation is vulnerable to clock signal attacks, because it is possible to inject faults in the Trivium cipher.

B. Bad floorplan

The design to be implemented has three main modules: the Trivium, the State Machine that controls the Trivium and the clock glitch generator. In this second option, a placement restriction for each component is imposed, as it is shown on the left in Fig. 4. The constraints force the tool to place the circuit in positions that do not comply timing restrictions because they are too far away from each other. The result is a bad distribution of the components because the tool does not follow the placement rules as it can be seen on the right in Fig. 4.

Despite of achieving a reduction in the area, the tool produces a bad design. The ciphers may not work properly. The experimental tests show synchronization errors, resulting in bad operation.

C. Forced floorplan

In the third floorplan strategy the designer uses constraints on the component placement too strong (the area for the components is very small) and therefore the tool will be unable to achieve them. In Fig. 5 it can be seen on the left the area restriction for each module and on the right the placement obtained by the tool.

When the designer uses such too strong constraints the tool assumes that it is not possible to achieve them and therefore tries to place the components in the indicated areas but ignores the area constraint imposed by the designer. This kind of strategy leads to a correct placement but the designer can not control whether the tool decision is correct or not. Hence it is possible to achieve the same result that in the bad floorplan case. In this case it is not possible to guarantee the operation of the device since the tool tries to achieve the constraints but does not get it. Due to this, this kind of practice is not recommended from the point of view of vulnerability to the fault injection because, as in the case that no floorplanning restrictions were imposed, the tool decides the placement of the different components. The experimental tests show that it is possible to introduce faults.

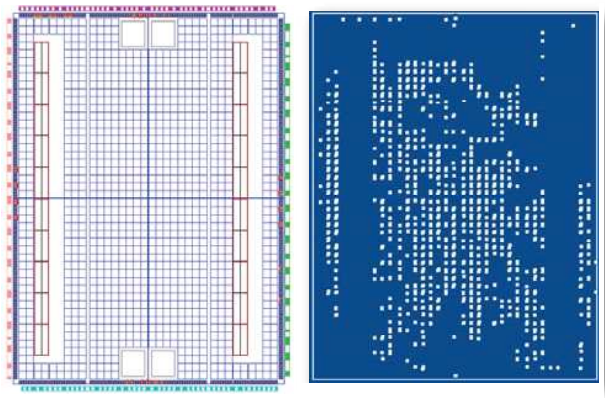

Fig. 3. FPGA area and free floorplanning without constraints.

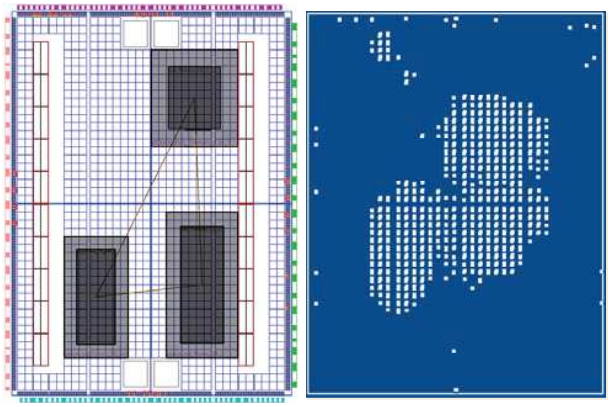

Fig. 4. FPGA area with constraints and bad floorplanning.

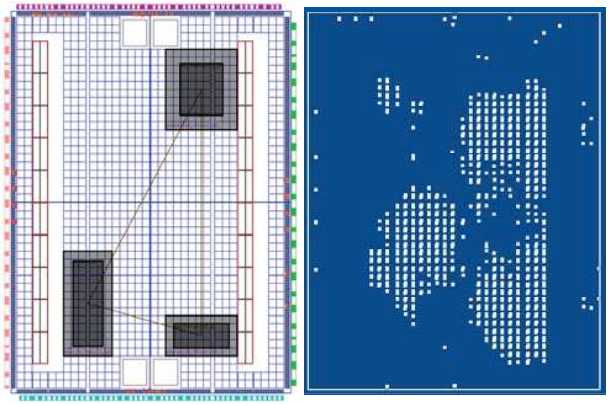

Fig. 5. FPGA area with constraints and forced floorplanning.

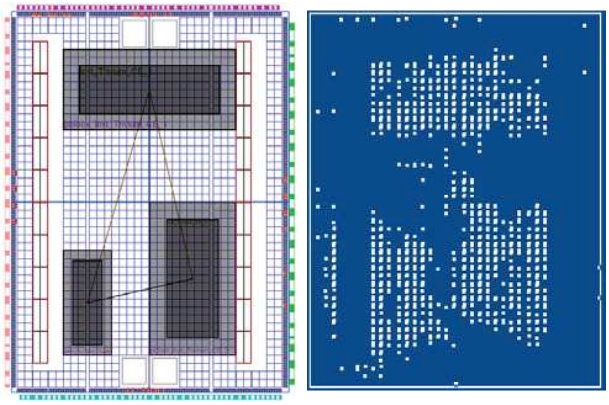

Fig. 6. FPGA area with constraints and strategic floorplanning.

D. Strategic floorplan

In this case, the designer knows the area for each component, so the modules can be fitted in the area imposed by the floorplan restrictions. The relative position of the modules is not too big. The floorplanning achieves the strategy of the designer perfectly. The result of this case can be seen in Fig. 6 where, on the left, it is the placement area imposed by the designer while, on the right, it is shown the result obtained after the placement. It can be observed how the tool achieves the designer aim, placing the components in their selected areas and occupying the desired area. Following this strategy, the designer is able to place the components according to their aims, reduce the distance between components and improve the timing behaviour. Therefore to minimize the vulnerabilities against fault attacks is possible because in the experimental tests it was impossible to introduce faults in the Trivium operation of this implementation.

The vulnerability reduction is achieved because of the reduction of the distance between the critical components, with the consequent reduction in the delay of critical paths. Thus, small pulses in the clock signal do not introduce errors. In order to inject faults it is necessary that the small pulses in the clock signal have higher frequencies. The frequency must be so high that the pulses are filtered by the FPGA, making the fault injections impossible.

E. Post-routed Simulations

Once the strategies to place the components in the FPGA have been analysed experimentally on the device, we will analyse the vulnerability of the four implementations using timing simulations and compare the results with those obtained experimentally.

The implementation with the highest maximum operating frequency will be the least vulnerable. Logic timing simulations have been used to test the maximum operating frequency, but it is necessary to take into account that logic timing simulations are usually very pessimistic in its timing results. The timing simulations will show bigger delays than the real ones of the FPGA device. The maximum operating frequency of the device is 311 Mhz.

In Table II it can be seen the maximum operating frequencies obtained by logic timing simulation for the four implemented floorplan strategies.

The strategic floorplan presents a maximum frequency around 298 MHz, which would make it less vulnerable to

TABLE II
MAXIMUM FREQUENCIES OBTAINED FOR EACH FLOORPLAN USING TIMING SIMULATIONS.

Floorplan	Maximum frequency (MHz)
Free	226,190
Bad	268.338
Forced	258,503
Strategic	298.743

attacks by manipulation of the clock signal. On the other hand, the forced and bad floorplans present a notable reduction of maximum frequency, being 258.503 MHz and 268.338 MHz respectively. This frequency reduction is due to the inadequate placement of the components of the ciphers. Finally, there is the case of timing simulation of the free floorplan strategy where the tool does not receive any constraints from the designer. In this case, the maximum frequency is 226,190 MHz. This result does not follow the trend of the previous results since it should work faster in simulation than the forced and bad designs but lower than the strategic one. This could be verified through the experimental tests on the FPGA, where this floorplan was able to operate at higher frequencies than the bad and the forced but at a lower frequency than the strategic one. This result through simulation is opposite to the results obtained in the experimental implementation in FPGA, where it could be seen that free floorplan works at higher frequency than the forced and bad strategies. This result is because, as previously explained, the Xilinx simulation tool is very pessimistic in terms of time delays and it is possible that it generates errors that do not appear in a real implementation. These results show that timing logic simulations cannot be used as a confident way to analyse the vulnerability of FPGA implementations against clock signal fault attacks.

VI. CONCLUSIONS

In this work, a countermeasure against clock fault attacks of Trivium stream cipher implemented in FPGA has been presented. The imposition of FPGA placement restrictions has been proven to be a way to reduce the vulnerability against fault attacks. After analysing the vulnerabilities of Trivium stream cipher implemented in FPGA, the results show that the weakest points are the flip-flops whose inputs are a combinational function of the output of several flip-flops.

With a floorplan strategy, it is possible to reduce the timing delays of critical paths and achieve an increase in the maximum frequency of the ciphers, making difficult the fault injections. The results obtained were carried out experimentally and using timing simulations.

It has been shown that the imposition of placement restrictions for cipher circuits in the place & route tool is a valid mechanism to reduce the vulnerability of ciphers against faults attacks through the clock signal. However, it is necessary that the selected areas are large enough for the placement of the device.

ACKNOWLEDGMENT

This work was supported by the Spanish Government projects: CESAR under Grant TEC2013-45523-R, INTER-VALO under Grant TEC2016-80549-R, and LACRE under Grant CSIC 201550E039.

REFERENCES

[1] R. Mahmoud, T. Yousuf, F. Aloul, and I. Zualkernan, "Internet of things (IoT) security: Current status, challenges and prospective measures," *International Conference for Internet Technology and Secured Transactions (ICITST'15)*, pp. 336–341, 2015.

[2] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of iot systems: Design challenges and opportunities," in *Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design*, pp. 417–423, 2014.

[3] C. D. Cannière, "Trivium: A stream cipher construction inspired by block cipher design principles," *Information Security*, pp. 171–186, 2006.

[4] D. Boneh, R. DeMillo, and R. Lipton, "On the importance of checking cryptographic protocols for faults," *Advances in Cryptology - EUROCRYPT'*, 1997.

[5] P. Street and W. Lafayette, "Low Cost Attacks on Tamper Resistant Devices," *Security Protocols*, pp. 125–136, 1998.

[6] M. Hojsík and B. Rudolf, "Differential Fault Analysis of Trivium," *Fast Software Encryption, Springer*, pp. 158–172, 2008.

[7] M. Hojsík and B. Rudolf, "Floating Fault Analysis of Trivium," *Progress in Cryptology, Springer*, pp. 239–250, 2008.

[8] Y. Hu, J. Gao, Q. Liu, and Y. Zhang, "Fault analysis of Trivium," *Designs, Codes and Cryptography, Springer*, vol. 62, no. 3, pp. 289–311, 2012.

[9] H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan, "The Sorcerer's Apprentice Guide to Fault Attacks," *Proceedings of the IEEE*, vol. 94, no. 2, pp. 370–382, 2006.

[10] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache, "Fault Injection Attacks on Cryptographic Devices : Theory, Practice, and Countermeasures," *Proceedings of the IEEE*, vol. 100, no. 11, pp. 3053–3076, 2012.

[11] M. Agoyan and J. Dutertre, "When clocks fail: On critical paths and clock faults," *Smart Card Research and Advanced Application*, pp. 182–193, 2010.

[12] Y. Ren, A. Wang, and L. Wu, "Transient-Steady Effect Attack on Block Ciphers," *Cryptographic Hardware and Embedded Systems, Springer*, vol. 2, pp. 433–450, 2015.

[13] T. G. Malkin, F.-X. Standaert, and M. Yung, "A comparative cost/security analysis of fault attack countermeasures," in *Fault Diagnosis and Tolerance in Cryptography*, pp. 159–172, 2006.

[14] F. E. Potestad-Ordóñez, C. J. Jiménez-Fernández, and M. Valencia-Barrero, "Fault Attack on FPGA implementations of Trivium Stream Cipher," *International Symposium on Circuits and Systems (ISCAS16)*, pp. 562–565, 2016.

[15] F. E. Potestad-Ordóñez, C. J. Jiménez-Fernández, and M. Valencia-Barrero, "Experimental and Timing Analysis Comparison of FPGA Trivium Implementations and their Vulnerability to Clock Fault Injection," *Design of Circuits and Integrated Systems (DCIS'16)*, 2016.

[16] F. E. Potestad-Ordóñez, C. J. Jiménez-Fernández, and M. Valencia-Barrero, "Vulnerability analysis of trivium fpga implementations," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 12, pp. 3380–3389, 2017.