IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Trivium stream cipher countermeasures against fault injection attacks and DFA

F.E. POTESTAD-ORDÓÑEZ[1, 2], E. TENA-SÁNCHEZ[1, 2], J.M. MORA-GUTIÉRREZ[1], M. VALENCIA-BARRERO[1, 2] AND C.J. JIMÉNEZ-FERNÁNDEZ[1, 2]

[1]Microelectronic Institute of Seville (IMSE-CNM-CSIC/US) (e-mail: potestad;erica;jmiguel;manolov;cjesus@imse-cnm.csic.es)
[2]Department of Electronic Technology, Escuela Politécnica Superior, University of Seville, Spain

Corresponding author: F.E. Potestad-Ordóñez (e-mail: potestad@imse-cnm.csic.es).

ABSTRACT Attacks on cryptocircuits are becoming increasingly sophisticated, requiring designers to include more and more countermeasures in the design to protect it against malicious attacks. Fault Injection Attacks and Differential Fault Analysis have proven to be very dangerous as they are able to retrieve the secret information contained in cryptocircuits. In this sense, Trivium cipher has been shown to be vulnerable to this type of attack. This paper presents four different fault detection schemes to protect Trivium stream cipher implementations against fault injection attacks and differential fault analysis. These countermeasures are based on the introduction of hardware redundancy and signature analysis to detect fault injections during encryption or decryption operations. This prevents the attacker from having access to the faulty key stream and performing differential fault analysis. In order to verify the correct operation and the effectiveness of the presented schemes, an experimental system of non-invasive active attacks using the clock signal in FPGA has been designed. This system allows to know the fault coverage for both multiple and single faults. In addition, the results of area consumption, frequency degradation, and fault detection latency for FPGA and ASIC implementations are presented. The results show that all proposed countermeasures are able to provide a fault coverage above 79% and one of them reaches a coverage of 99.99%. It has been tested that the number of cycles for fault detection is always lower than the number of cycles needed to apply the differential fault analysis reported in the literature for the Trivium cipher.

INDEX TERMS Countermeasure, DFA, Fault Attack, Fault Detection Schemes, Stream Cipher, Trivium.

I. INTRODUCTION

NOWADAYS the number of devices interconnected has grown exponentially, among other reasons, due to the great development of so-called internet of things (IoT). These connections bring with them the exchange of sensitive information that could be intercepted by external agents for malicious purposes.

In [1] the authors describe the importance of analysing security in the IoT, considering recent research work on the different stages of the IoT security solution. [2], [3] describe the importance of lightweight cryptography as a solution to the problem of securing resource-constrained devices in the IoT. Finally, [4] discusses the risks that exist if attacks on embedded applications used in this area are not considered. Due to this, the development of new cryptographic algorithms that try to protect the information and meet the strong constraints imposed by the applications is constant.

At the same time as new cryptographic algorithms appear, new attacks are being developed that try to break the security they offer. Attacks on the mathematical formulation of the algorithms themselves are currently useless due to the time consumption and resources needed, as they use keys whose lengths make them secure against brute force attacks. Because of this, the attackers are focusing not on attacking the algorithm itself, but the physical implementation where these algorithms are implemented. In this paper, we classify the attacks according to whether or not the normal operation of the algorithm is intentionally altered during encryption. On the one hand, Side Channel Analysis (SCA) attacks do not alter the normal operation of the device, they attack the circuit through leaked information such as power consumption [5] or electromagnetic radiation measurements during circuit operation. On the other hand, Active Fault Analysis attacks, such as Differential Fault Analysis (DFA), attack the circuit through modification of the operation conditions to inject faults during the circuit operation. These attacks

are theoretical, and in combination with Fault Injection Attack (FIA) mechanisms (laser beams, voltage peaks, clock glitches), it is possible to analyse the behaviour of the cipher and compromise its security [6].

Among the different algorithms proposed for cryptographic purposes, one of the best options for lightweight cryptography is the stream ciphers. These kinds of algorithms have a limited impact on the available resources and some of them are able to work with a low power consumption and high frequencies. One of these algorithms is the Trivium stream cipher [7]. Finalist in the eSTREAM project and accepted as ISO standard, Trivium is a cipher aimed at applications where resource consumption and power restrictions are very high, making it a very good candidate for IoT applications where the security of the exchanged data must be guaranteed. As a standard cipher and cryptosystem, Trivium has been subjected to numerous theoretical attacks to compromise its security. Different DFA techniques have been reported in the literature in a satisfactory and effective way [8]–[13], showing the possibilities of endangering the security of the data exchanged when this cipher is implemented as protection. These theoretical works, together with the experimental attacks presented in [14] and [15], have shown that this cipher must be protected to minimize its vulnerabilities against malicious attacks. Good examples of the continuous and recent interest in breaking this encryption algorithm, apart from the DFA, are the works presented in [16]–[20]. In [16], [17] a Boolean polynomial reduction technique and guess and determine attack in Trivium are presented. In [18], a conditional differential attack applied to the Trivium cipher is presented. The authors perform key recovery attacks on the 978-round, detecting non-randomness up to the 1108-round. With this, the authors offer an improvement to attackers implementing differential attacks. A study of cubic attacks against the cipher is proposed in [19], and a new method for finding non-linear superpolies using the linearity test principle is also described. Finally, in [20], a cube attack is performed, showing that it is able to recover the key in the 781-round of Trivium.

Because Trivium is a cipher oriented to low resource consumption applications, the development of countermeasures is a challenge, since in order to increase the security of the cipher, it is necessary to add countermeasures and at the same time ensure that these countermeasures have the lowest possible resource consumption to keep the cipher lightweight. As far as we know, all versions of this cipher have been presented without protections against active attacks by fault injections and therefore there is a need to carry out the development of countermeasures aimed at minimizing the vulnerabilities of this cipher. Taking this into account, in this paper, we focus on the development of protections against the active attacks and DFA for the standard Trivium stream cipher.

A. PREVIOUS WORKS

The theoretical vulnerability of the Trivium stream cipher against DFA has been reported in the literature. [8]–[13], [16]–[20], where theoretical attacks were modelled. Faults were injected into the internal state of the cipher to retrieve secret information from the device. The main assumption that all these works take is that if an attacker is able to inject only one faulty bit into the internal state of the cipher during its operation, it is possible to endanger its security using DFA. With a mathematical analysis of the correct and the faulty key streams obtained after the fault injections, it is possible to determine the internal state at the moment of attack. But none of these works tried to experimentally prove the feasibility of these assumptions and the scenario in an experimental mode.

On the one hand, experimental attacks on the Trivium cipher, such as FIA, were presented in [14] and [15]. In these works, different attack systems were designed to achieve the assumptions of theoretical attacks and to prove the experimental vulnerability of the Trivium cipher. In [14], where the attack is performed in FPGA implementations, it is shown that it is possible to achieve the fault injection within the internal state of the cipher with a high percentage of effectiveness and efficiency. On the other hand, in [15], the authors show the possibility of retrieving the secret key of a standard Trivium cipher implemented in ASIC. This attack is closer to a real scenario than [14] because it uses an ASIC implementation, fault injections are performed externally to the circuit and the key is recovered experimentally by combining fault attacks with DFA.

Nevertheless, the state of the art in the design of specific countermeasures for stream ciphers to counteract active non invasive attacks is not deeply studied and there are no hardware specific implementations to our knowledge presented in the literature. They are usually generally designed countermeasures based on complete redundancy of stream ciphers that implies an unacceptable area overhead.

B. OUR CONTRIBUTION

This paper presents four different countermeasure proposals to significantly reduce the vulnerabilities of the Trivium cipher against active attacks by fault injection. These countermeasures are: a total hardware redundancy, the use of LFSR as a signature generator, feedback protection using XOR gates, and the combination of the LFSR signature scheme and feedback protection scheme. To this end:

1) An extensive analysis of the requirements necessary to perform the DFA and the weak points of the Trivium cipher has been carried out. The number of faults needed by DFAs and the number of faulty bits of key stream needed to compromise the security of this type of cipher are analysed.
2) Four different countermeasure schemes are presented in detail that allow to provide different levels of security considering different transient types of fault that can be exploited in this cipher. Both single and multiple faults are considered, in feedback positions and within the cipher internal register.
3) A comparison of the implementations of the proposed schemes has been made in different technologies such

as FPGA (Xilinx Virtex-7) and ASIC (TSMC 90 nm).

4) In order to experimentally verify the effectiveness of the proposed countermeasures, a system of attacks by modifying the clock signal, namely an active non-invasive attack system, has been designed and implemented in FPGA.
5) An extensive analysis of the injected and detected faults, both multiple and single, has been carried out using the experimental attack system.
6) The fault detection latency of each of the countermeasures is analysed. This analysis makes it possible to determine whether the fault detection latency can be exploited by the DFAs.
7) The analysis of resource consumption and fault coverage of each scheme is presented, carrying out a comparison between them to determine which presents a better trade-off.

This paper provides four countermeasures, showing their design and implementation in two technologies, FPGA and ASIC, cost performance analysis, fault coverage analysis and a comparative analysis between security and resource overhead. We offer different solutions depending on the security requirements and resource consumption constraints, being able to detect fault injections that have been proven in different works which endanger the security of this cipher.

C. PAPER ORGANIZATION

The rest of the paper is organized as follows. Section II presents a brief description of the Trivium stream cipher architecture and its characteristics, the analysis of the theoretical DFA assumptions needed to endanger the cipher and the vulnerabilities of this cipher previously reported in the literature. In Section III, the countermeasure proposals for Trivium against fault attacks are presented. Section IV presents the results obtained after applying the experimental attack system on FPGA to validate each countermeasure, showing the fault coverage for multiple and single faults, the implementation costs, both in FPGA and ASIC technology, the trade-off between area and frequency degradation of each countermeasure and the comparative with other schemes. Finally, the conclusions are given in Section V.

II. TRIVIUM STREAM CIPHER VULNERABILITIES

Before introducing the countermeasures that improve the security of the Trivium stream cipher, let us describe its vulnerabilities. To do that, firstly we are going to present briefly the structure of this cipher and after that the theoretical, and experimental vulnerabilities reported in the literature.

A. TRIVIUM STREAM CIPHER

The Trivium stream cipher [7] is one of the finalists of the eSTREAM project and was adopted as ISO/IEC 29192-3 standard. It is a synchronous cipher designed to generate up to 2^{64} bits of key stream from an 80-bit secret key (K) and an 80-bit initialization vector (IV). Fig.1 shows the schematic

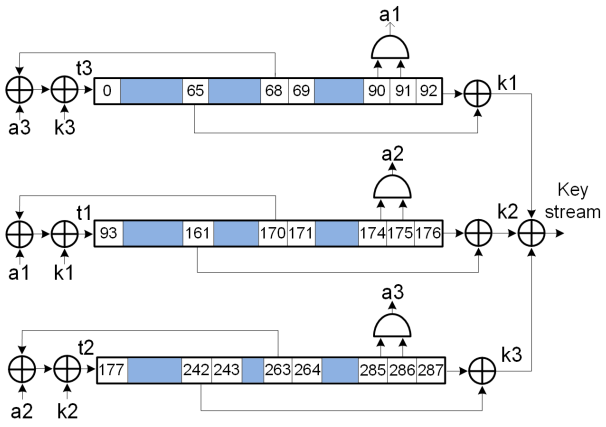

FIGURE 1. Schematic representation of the Trivium stream cipher internal structure.

representation of the internal structure of Trivium. The cipher architecture is based on three shift registers comprising 288 bits in total, as well as combinational logic to provide feedback. 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 is made up of 84 bits, and the third comprises 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.

Just like other synchronous stream ciphers, the underlying algorithm needs to be initialized with the load of 288 bits into the shift register (internal state) comprising one secret K, one IV, and a stream of zeros and ones. Before generating a valid key stream, the cipher must run for 1152 clock cycles. From then on, it generates a valid pseudo-random bit sequence.

B. THEORETICAL VULNERABILITIES OF TRIVIUM

In the case of the Trivium stream cipher, there are different works in which DFA is applied [8]–[13]. Depending on the work, the DFA needs a greater or lesser number of fault injections into the internal state of the Trivium. In [8] two different mathematical formulations are presented, where they need 388 fault injections for the first technique and being the second technique more efficient, allowing to retrieve the secret internal state with an average of 43 fault injections and 280 faulty key stream bits. The same authors present [9], where the number of fault injections needed was reduced to an average of 3.2 but increased the bits of the faulty key stream up to 800. On the basis of these two previous works, in [10] a more relaxed scenario of the attack is presented. The secret internal state could be retrieved with an average of 3.7 fault injections and 195 faulty key stream bits. In [11] the authors are able to retrieve the secret internal state with two fault injections and 420 faulty key stream bits. In [12], the authors state that with a new formulation, they only need one fault injection and 800 faulty key stream bits to retrieve the secret internal state. In [13] is presented a new

TABLE 1. DFAs on Trivium

Reference	Required number of fault injections (average)	Required number of key stream bits
[8]	43	280
[9]	3.2	800
[10]	3.7	195
[11]	2	420
[12]	1	800
[13]	2	450

analysis where the system constraints of [10] are improved. Using different fault models and injecting different faults into an unknown cycle, they affirm that it is possible to retrieve the secret internal state with an average of four fault injections and 450 faulty key stream bits. All these works have the same main assumption. In order to retrieve the secret key, an attacker must be able to change only one bit of the internal state injecting one transient faulty bit, namely flip one bit from 0 to 1 or from 1 to 0, and capture the key stream produced by this faulty internal state. Note that these differential analyses are based on the comparison between correct and faulty key streams, establishing linear equations between them and thereby revealing the secret information contained by the cipher.

Table 1 summarizes the requirements for each DFA model for retrieving the secret internal state, and therefore the secret K and IV, of the cipher, showing the reference, the number of fault injections and the number of faulty key stream bits. Note that the attacks presented in [16]–[20] are not included in this analysis, since these attacks are against the mathematical algorithm itself and not against the physical implementations, therefore they are out of our scope.

None of these papers make experimental measurements of the fault injection possibilities or Trivium vulnerabilities. In the following subsection, we describe the works that performs experimental attacks on the Trivium cipher and where the weak points and vulnerabilities of this cipher are determined.

C. EXPERIMENTAL VULNERABILITIES OF TRIVIUM

In [14] and [15] the experimental vulnerability analysis was presented through the manipulation of the clock signal. In [14] the authors analyse the vulnerabilities of different FPGA implementations of the Trivium stream cipher against fault attacks. They analyse the standard implementation, the serial charge implementations and a low power consumption implementation, showing the possibility of injecting one faulty bit into the internal register of each cipher implementation with low cost tools. In [15] the authors present an experimental attack system that allows recovering the secret key by combining FIA and DFA. In contrast to [14], in [15] the attacks are performed externally on ASIC implementations with a non-invasive active attack system by manipulating the clock signal.

These works show that the main points of vulnerabilities are the feedback positions or their neighbour cells, these are the internal state positions 0, 93, and 177 (Fig. 1). These positions are those with greater delay and thus greater vulnerabilities against clock signal manipulations, being the most critical points in terms of timing. The attack systems show that it is possible to achieve the main assumption of the DFA, to inject one faulty bit in the internal state and retrieve the faulty key stream. In addition, these vulnerabilities do not have any dependency on the key and IV used due to the fact that effective fault injections were achieved using different injection clock cycles and different secret pair key/IV.

It has been demonstrated that in the real world this type of attack constitutes a real threat to the trivium cipher. Therefore, it is necessary to develop countermeasures to protect the cipher against them and to cover the detected vulnerabilities.

III. COUNTERMEASURE PROPOSALS

In this section, four different countermeasure schemes for the Trivium stream cipher are presented. These countermeasures allow to detect the fault injections performed by an attacker, being the main goal to avoid the possibility that the attacker can sample the necessary faulty key stream bits needed by the DFA models.

A. HARDWARE REDUNDANCY

The first countermeasure scheme is based on the introduction of hardware redundancy. This is a well-known countermeasure scheme and can be applied in many different ways [6]. Depending on the way, the number of blocks or functions duplicated and the operations performed in parallel, it is possible to obtain different levels of protections. Considering the general scheme of the Trivium presented in [6], our proposal belongs to the group of Simple Duplication with Comparison (SDC), with the duplication of hardware blocks followed by a comparator. The fact that the Trivium cipher is made up of shift registers means that hardware redundancy, to be effective, has to be applied to all the registers and therefore duplicate all the hardware. However, simple duplication can cause the fault to be injected in the same relative position of the shift register of both instances of the cipher (for example, in attacks by altering the clock signal). To improve this countermeasure, in addition to the duplication of the cipher, we propose an additional modification that consists of making the second instance (the redundant cipher) to operate with a delay of one clock cycle. Fig. 2 shows a schematic representation of the proposal. In addition to the redundant Trivium, this scheme adds two additional flip-flops. The first one is used to delay the control signals of the redundant cipher one clock cycle. The second register is used to sample the output of the original cipher and delay the key stream output one clock cycle. A comparator (an XOR gate) checks if both key streams are the same. To avoid this countermeasure, the attacker has to inject the fault into the two ciphers in the same relative position at the same time, and take into account the delay of a clock cycle that exists between them.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3136609, IEEE Access

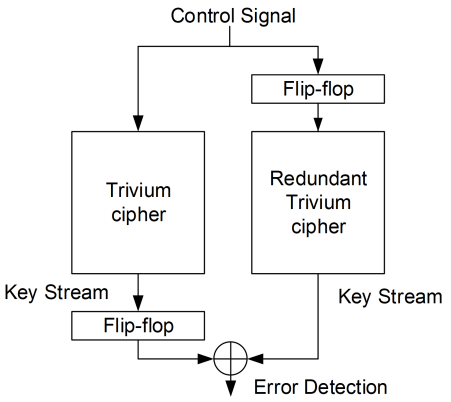

FIGURE 2. Schematic representation of the hardware redundancy of Trivium.

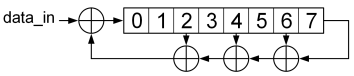

FIGURE 3. Schematic of the 8-bit LFSR used.

B. SIGNATURE USING LFSR

The second countermeasure we propose is based on the use of linear feedback shift registers (LFSR) for signature analysis. The signature analysis was presented by Hewlett-Packard in [21] for testing boards. LFSR schemes have been used because they are based on shift registers, as well as the structure of the cipher, and they are implemented with flip-flops and XOR cells, which are standard cells, whose resource consumption is minimal and whose operating frequency is high. This technique allows us to detect if faults have been introduced in a shift register, comparing the signature generated by the data at the input of the shift register with the signature generated by the data at the output of the shift register. The hardware implementation of LFSRs is very simple because it only requires a shift register and XOR gates to generate the feedback bits. The number of bits in the shift register determine the maximum number of different signatures that can be generated. For this countermeasure, we have used an 8-bit LFSR whose polynomial is given by $D = x^8 + x^7 + x^5 + x^3 + 1$, which can generate up to 255 different signatures, $(2^n - 1)$ where n is the number of LFSR bits, and can detect not only single faults, but also multiple fault injections. Fig. 3 shows the schematic of this LFSR. The signal $data\_in$ is the input value of the LFSR and it is connected to the input signal or to the output signal of one of the Trivium shift registers.

Since Trivium is composed of three shift registers, this countermeasure requires six LFSRs, placed at the beginning and at the end of each of these three shift registers. Fig. 4 shows a schematic of the Trivium with the six LFSRs. In addition to the six LFSR, three counters and three additional registers are needed to store and compare LFSR signatures at

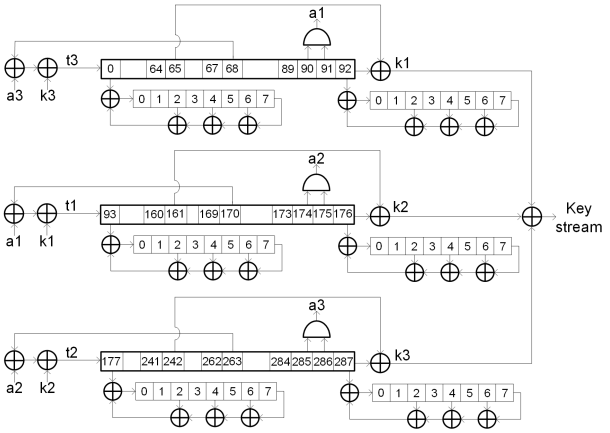

FIGURE 4. Schematic representation of the Trivium cipher with the LFSRs for signature analysis.

the appropriate times. It should be noted that for simplicity in the schematic representation the counters and comparators have not been included in the schematic in Fig. 4. The counter counts the number of cycles since a bit reaches the LFSR placed at the beginning of the shift register and the same bit reaches the LFSR placed at the end. Each additional register stores the contents of the LFSR located at the beginning of the shift register, so that it can be compared with the contents of the LFSR placed at the end. The number of cycles to wait for a new comparison is the number of bits in the shift register. A control circuit synchronizes the load and the comparison.

With the results of the comparisons of the LFSRs of the three shift registers, it is known if a fault has been introduced in the Trivium internal state register between the clock cycles from the store of the contents of the LFSR connected to the input, to the clock cycle in which it is compared with the contents of the LFSR connected to the output. Note that this protection scheme is very versatile since there are many possible configurations. LFSRs of larger or smaller number of bits can be used, or a greater number of LFSRs can be used to generate signatures. The 8-bit LFSR is a very useful option due to its high fault coverage versus minimum area penalty. It is possible to use a higher or lower number of bits, but it should be noted that with a lower number of bits the fault coverage will decrease (two different inputs will be more likely to generate the same signature) and with a higher number of bits, the resource penalty will be higher. In this case, we have selected the configuration that we consider to be more appropriate because it is capable of providing a high level of security without a high area penalty.

C. FEEDBACK BIT PROTECTION USING XOR OPERATIONS

Among the most vulnerable positions for fault injection are the bits of the state register to which the feedback signals are connected. This countermeasure uses partial redundancy to calculate twice the value of each feedback to detect the injec-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3136609, IEEE Access

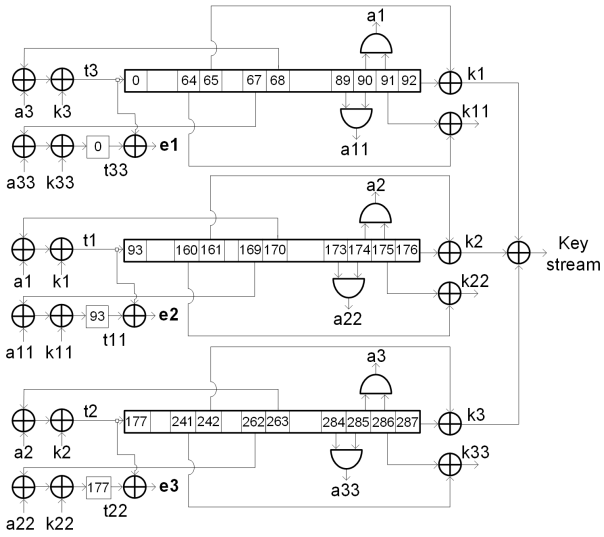

FIGURE 5. Schematic representation of the Trivium cipher with partial redundancy included.

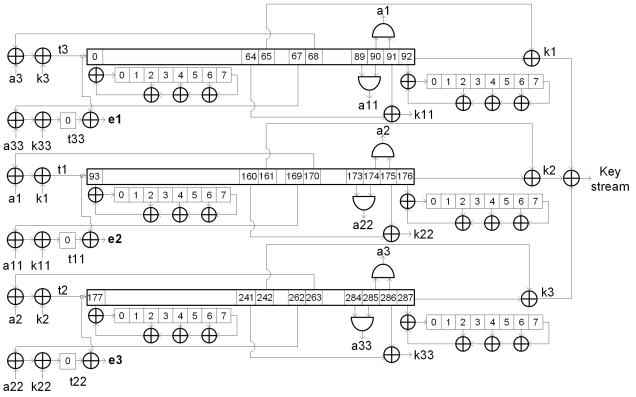

FIGURE 6. Schematic representation of the Trivium cipher with the combination of schemes included.

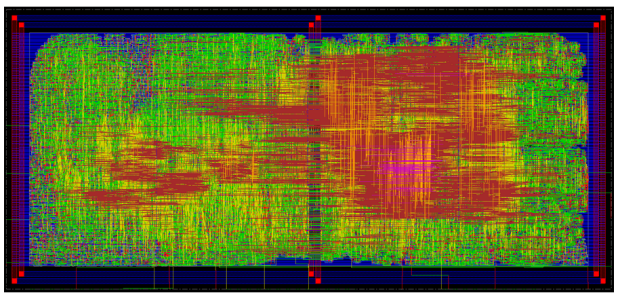

FIGURE 7. ASIC layout.

tion of a fault. The countermeasure consists of generating, in each clock cycle, the feedback signals of that cycle and also those to be generated in the next cycle. This is done, as it is shown in Fig. 5, by carrying out the combinational operations with the immediately preceding positions in the state register. This advanced value is stored in a flip-flop to be compared, in the next clock cycle, with the one generated at that time. This countermeasure adds to the Trivium cipher three flip-flops to store the value of the next feedback bits, three AND gates and nine XOR gates. Using XOR operations it is possible to know if the two bits are different, for example $t3$ and $t33$ (Fig. 5), and therefore if any fault has been injected (activating the error signals $e1$, $e2$ or $e3$).

This countermeasure protects against fault injections over the feedback positions, the positions used to calculate the feedback bits, or any fault that changes the correct value of the compared bits. An attack by manipulation of the clock signal like the one presented in [14], [15] injects most of the faults into the feedback bits, so this countermeasure fits very well in that type of attacks, because its implementation is very simple and consumes very fewer resources.

D. LFSR AND FEEDBACK BIT PROTECTION

This countermeasure adds two of the protection schemes previously presented: the signature protection using LFSR and the protection of the feedback bit using XOR. The LFSR protection detects fault injections inside of the register and the protection of the feedback bits adds protection if the fault is injected in the feedback logic. Fig. 6 shows a scheme of the Trivium cipher with both countermeasures included. This combination offers double security since it allows a signature analysis before and after each register and also detects possible faults in the feedbacks. Error detection is performed in parallel between the two countermeasures, i.e. each LFSR compares with each other while at the same time the XORs check the feedbacks positions.

IV. COUNTERMEASURE ANALYSIS

After describing the vulnerabilities of the Trivium cipher and the proposed countermeasures, we present the analysis of the resources they require for FPGA and ASIC technologie and the fault coverage they provide. In the case of the FPGA implementation, an Artix-7 XC7A100T of Xilinx has been used and for the ASIC implementation, a TSMC 90nm technology has been used. In Fig. 7 it can be seen the ASIC layout. Each countermeasure applied to the standard Trivium has been implemented in both technologies. The fault coverage has been studied experimentally for FPGA implementations and in post-place and route simulations in the ASIC implementation. It is noteworthy that our countermeasures use a semi-custom design flow with automatic synthesis/place & route tools, they do not need to be routed manually, nor do they need any constraints. They are directly applicable using the synthesis tool in both FPGA and ASIC technologies.

A. IMPLEMENTATION COSTS

Table 2 shows the resources used by the FPGA implementation of the Trivium cipher, without countermeasures and with each of the countermeasures. All these versions have been compared with the unprotected Trivium cipher. The resources have been measured using the number of Slice flip-

TABLE 2. Obtained results of each countermeasure implemented on FPGA (Artix-7 XC7A100T of Xilinx)

Proposed Countermeasure	Resources	Resource Overhead	Frequency Degradation
Unprotected	Slices 288 LUTs 291	1 1	1
H. Redundancy	Slices 582 LUTs 583	2.02 2.00	0.98
LFSR Signature	Slices 387 LUTs 391	1.34 1.34	0.67
XOR	Slices 303 LUTs 309	1.05 1.06	0.83
XOR+LFSR	Slices 395 LUTs 403	1.37 1.38	0.64

TABLE 3. Obtained results of each countermeasure implemented on ASIC (TSMC 90nm)

Proposed Countermeasure	Resources (cells)	Resource Overhead	Frequency Degradation
Unprotected	634	1	1
H. Redundancy	1343	2.11	1.42
LFSR Signature	900	1.41	0.97
XOR	663	1.04	0.96
XOR+LFSR	923	1.45	0.93

flops and LUTs. As it can be seen, Hardware Redundancy has the highest cost in resources, near double the resources used by the Trivium without protection, while the XOR scheme has the lowest, with a 6%. For the LFSR countermeasure, the increase in resources is 34%, lower than the increase in resources required by the hardware redundancy. The XOR+LFSR countermeasure increases the resources required by Trivium by 38%, only 4% more than the LFSR countermeasure. In the case of frequency degradation, both the LFSR and the XOR+LFRS schemes have the highest penalty, while the XOR scheme has the lowest degradation, with just 17%.

Table 3 shows the costs of each ASIC implementation of the Trivium, without and with the proposed countermeasures. To determine the resource consumption, the number of cells occupied by each circuit was considered. The hardware redundancy countermeasure increases the resources required by 111%, which is slightly more than the increase in the FPGA implementation. The rest of the countermeasures also follow this trend. The LFSR countermeasure increases the resources used by the Trivium by 41%, the XOR countermeasure by 4% and the XOR+LFSR countermeasure by 45%. These results are very important since they show that protected implementations of the Trivium cipher in 90 nm technology for IoT applications would cost less than 45%, if the hardware redundancy countermeasure is not considered.

In the case of frequency degradation, two types of tests were performed to test the maximum operating frequency in ASIC. The first checks if the key stream output is correct as a function of the decrease of the operating clock period and the second checks if the error signal output works correctly as a function of the decrease of the operating clock period. It should be noted that in all cases, the key stream output fails with a clock period larger than the clock period, which causes the error output to fail. This means that the generation of the error signal has no impact on the maximum operating frequency. When the correct operation of the ciphers checking the key stream is carried out, the difference between maximum frequencies for the unprotected and the protected Trivium is minimal, except for the Hardware Redundancy scheme. In the case of Hardware Redundancy, the operating frequency is higher because the scheme samples the load and the enable signals with the additional flip-flops and can make its operating frequency higher, avoiding sample errors. The results show that in ASIC the degradation of the maximum operating frequency is lower than in FPGA implementations. In ASIC implementations, the maximum frequency is never below 93% of the maximum frequency without countermeasures.

B. FAULT COVERAGE

As described in section II, the main vulnerability of the Trivium cipher lies in the possibility of injecting a single fault into its internal register. However, to consider in more general terms the effectiveness of the presented countermeasures, in our fault detection analysis we have considered both single and multiple fault injections. To experimentally verify the effectiveness of the proposed countermeasures, we have designed and implemented in FPGA a system to carry out attacks by modifying the clock signal. It is important to note that the fault coverage analysis carried out in the FPGA is extensible to ASIC implementations. This is because the internal fault detection operation behaves in the same way in both implementations. The main difference is the difficulty of inserting faults in ASIC technology, where the clock frequency must be higher for fault injection. However, the study of fault injection experimental setup systems is out of the scope of this work, so we have developed this system only for FPGA. The developed system has the following characteristics:

1) It is capable of injecting a small pulse into the clock signal. The pulse period is automatically reduced until a fault is introduced into the cipher operation.
2) For each injected fault, it is stored whether the fault is single or multiple and whether it has been detected by the countermeasure.
3) The system instantiates 16 ciphers in parallel, in order to avoid the dependence of faults with the routing. In addition, two implementations with different synthesis options have been made, generating implementations with different routings. In total, fault injection is carried out on 32 different implementations of the same Trivium.
4) On each cipher, 65537 faults are injected in the same clock cycle, but each with a different key and IV (pseudo-randomly generated). This gives 2097184 faults injected for each of the cipher versions.

Table 4 shows, for each countermeasure implemented in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3136609, IEEE Access

TABLE 4. Experimentally obtained results applying the attack system to each of the implementations. Multiple and single faults are considered

Proposed Countermeasure	Total number of faults injected	Multiple faults	Single Faults	Number of faults detected	Fault Coverage (%)	Efficiency (%)
Unprotected	2097184	80350	2016834	0	0.00	96.17
H. Redundancy	2097184	386296	1710888	2064078	98.4214	81.58
LFSR Signature	2097184	84396	2012788	1848488	88.1414	95.98
XOR	2097184	115210	1981974	1493009	71.1911	94.51
XOR+LFSR	2097184	194888	1902296	2097148	99.9983	90.71

TABLE 5. Experimentally obtained results applying the attack system to each of the implementations. Multiple and single faults are considered

Proposed Countermeasure	Total number of faults injected	Number of faults detected	Fault Coverage (%)	Fault detection Latency (bits)
H. Redundancy	2005823	1973103	98.3687	2 to 169
LFSR Signature	2007004	1763332	87.8589	19 to 169
XOR	2008032	1596068	79.4842	1 to 69
XOR+LFSR	1425701	1425648	99.9963	2

Trivium, the $Total\ number\ of\ faults\ injected$ by the attack system, the number of $Multiple$ and $Single\ faults$, the $Number\ of\ faults\ detected$, the $Fault\ Coverage$, and the $Efficiency$. The number of faults detected and the fault coverage consider both single and multiple faults. Efficiency represents the percentage of single faults with respect to the total number of faults. The results show that all versions of the ciphers are highly vulnerable to fault injection attacks by the clock signal. The lowest efficiency of the system fault injection is obtained for the cipher with hardware redundancy countermeasure and is above 81%. For the rest of the countermeasures, the efficiency of the attack system is above 90%.

Regarding fault coverage, all countermeasures reach a high level of fault detection (above 71%). The one that detects the least faults is the XOR countermeasure with just over 71% of faults detected (that is, because it is oriented to detect faults in the cipher feedback flip-flops). The LFSR signature countermeasure is able to detect just over 88% of the injected faults. But it is Hardware redundancy and XOR+LFSR countermeasures that prove to be the most effective for fault detection. For the Hardware redundancy countermeasure, only 1.57% of more than two million injected faults were not detected and the XOR+LFSR countermeasure was able to detect 99.9983% of injected faults.

Since only fault injections that introduce a single fault in the internal register of the cipher are effective, a second independent and new group of fault injections has been performed in which only the effective faults are analysed. The total number of injected faults is still 2097184, but ignoring multiple faults, the number of effective faults differs for each cipher. The results are shown in Table 5. In addition, in this table the $fault\ detection\ latency$ is considered. This latency represents the number of key stream bits required for the countermeasure to detect that a fault has been introduced.

The first thing that can be observed in Table 5 is that the percentage of fault coverage does not vary significantly with respect to fault coverage when no distinction is made between single and multiple faults. Only for Trivium with the XOR countermeasure the percentage rises significantly, from 71% to 79%. If the latency is considered, in the case of the Hardware Redundancy, the scheme has between 2 and 169 bits of fault detection latency, while the LFSR signature has between 19 and 169 bits of latency and 1 to 69 bits in the case of XOR countermeasure. If XOR+LFSR is considered, the countermeasure has only two bits of latency until it detects that a fault has been inserted. A very important aspect in the fault detection latency of each countermeasure is that, in all cases, these latency cycles do not imply a faulty key stream at the output. All the key stream bits until the fault injection is detected are correct. Therefore, all countermeasures prevent the application of DFA attacks on detected faults, because they prevent the generation of faulty key streams by the cipher.

It should be noted that changing the external environment (temperature or supply voltage variations) will affect the critical paths of the cipher, these changes could slow down the circuit and make it possible to introduce faults more easily. Nevertheless, in our studies we have not detected any degradation in the critical paths when including our countermeasures. Therefore, although we have not tested the influence of changing the cipher environment with the included countermeasures, we can expect them to work correctly as the faults occur in the cipher feedbacks (main critical paths in our system).

C. TRADE-OFF

Depending on the desired level of security and the implementation constraints, one countermeasure or another may be used. Fig. 8 shows a comparison between the frequency and the resources of each countermeasure (in FPGA). In this figure, the Trivium with countermeasures is positioned in three very clear zones. In the first zone is the XOR countermeasure, with a very small resource consumption and low frequency degradation. The second zone corresponds to the LFSR and XOR+LFSR countermeasures, whose frequency degradation is higher, but whose resource consumption is medium. Finally, in a third zone is the Hardware redundancy

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3136609, IEEE Access

TABLE 6. Comparison with different protection schemes

Countermeasure	HW oriented	SW oriented	Fault Coverage (%)	Resource Overhead	Frequency Degradation
Unprotected	✔	✔	0	1	1
[22]	✔	✘	–	1	0.72 - 0.96
[23]	✘	✔	–	–	0.60-0.62
[25]	✔	✘	–	1.33	<1
H. Redundancy	✔	✘	98.3687	2.00	0.98
LFSR Signature	✔	✘	87.8589	1.34	0.67
XOR	✔	✘	79.4842	1.06	0.83
XOR+LFSR	✔	✘	99.9963	1.38	0.64

– = not information available.
✔ = the scheme is applicable.
✘ = the scheme is not applicable.

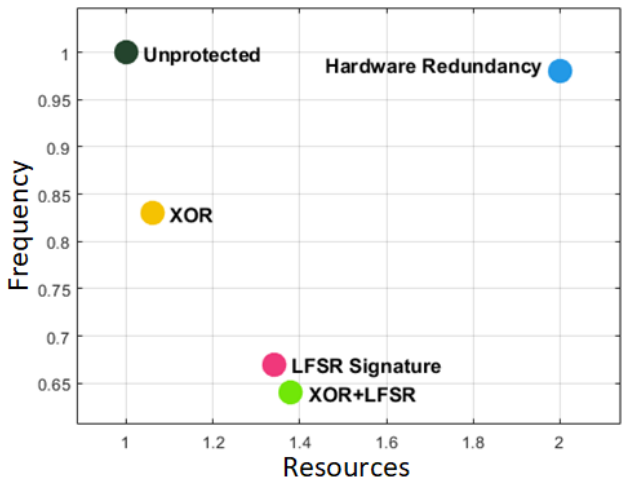

FIGURE 8. Frequency versus Resources representation for each countermeasure scheme.

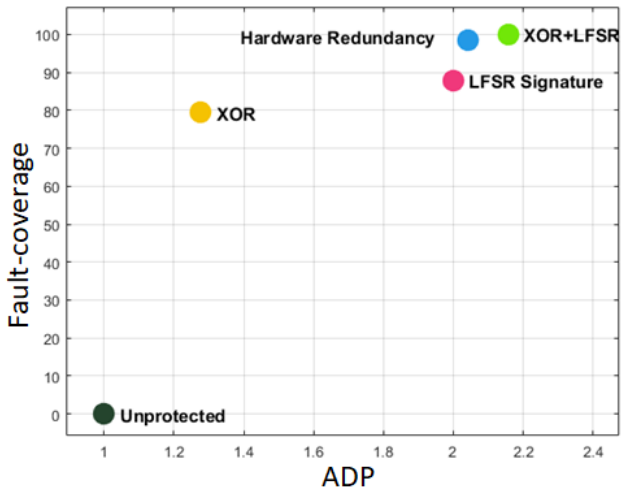

FIGURE 9. Fault-coverage versus Area Delay Product (ADP) for each countermeasure scheme.

countermeasure, with a medium frequency degradation but with the highest resource consumption.

A second comparison, this time between fault coverage and the figure of merit Area Delay Product (ADP), is shown in Fig. 9. The countermeasures can be grouped into two zones. In the first zone, there is the XOR countermeasure, with the lowest fault coverage, above 79%, and with an ADP value very close to the unprotected cipher. In the second zone, there are the other countermeasures, whose fault coverage is greater than 98% and whose ADP is greater than ADP of the unprotected cipher. As a conclusion of all these comparisons, the XOR scheme is ideal for applications where the cipher cannot increase its resources but needs a certain level of security, assuming a 79% of effective fault coverage. If the application has more relaxed resource constraints, and it is desired to increase the encryption protection, it is possible to use any of the other schemes that offer protection above 87%, but at a higher cost in terms of ADP.

D. COMPARATIVE WITH OTHER SCHEMES

In order to be able to compare the proposed schemes, we have carried out a search for stream cipher countermeasures proposed in the literature. For this type of ciphers, hardware redundancy is generally proposed as a solution; however, we have selected different countermeasures reported in the literature, focusing on the design level and countermeasures against power analysis. On the one hand, there is work [22], where an improved hardware design methodology is presented. This paper states that depending on the floorplanning used during the implementation process, it is possible to difficult fault injections, but does not constitute a countermeasure as such. In [23], a software countermeasure is presented where the so-called Single Instruction Multiple Data instructions are used, where redundancies are implemented in data processing to avoid injection of faults. The paper [24] presents two algorithm-level countermeasures focused on preventing DPA attacks on the Trivium cipher, but not against fault injections. Finally, in [25] a countermeasure to prevent DFA is presented using faulty ciphertext randomization and applied to the Grain-128 stream cipher. If no fault injection is detected, the correct cipher text is provided, otherwise a randomized ciphertext is given as output. The drawback of this countermeasure is the area overhead, around 33%, and the fact that the error detection module is not specified, without which the countermeasure would be neither implementable nor feasible.

Table 6 shows a comparison of the different schemes. In this table we have taken into account whether they are hardware or software oriented countermeasures, fault coverage, resource cost and frequency degradation. As it can be seen, in the case of [22], there are no fault coverage data since it is a design methodology to avoid non-invasive active faults on the Trivium cipher. The frequency costs of the different proposed designs range from 0.72 to 0.96. Regarding [23], we can see that it is a software-oriented scheme and therefore resource consumption is not applicable. In this proposal, no fault coverage is provided and the frequency degradation is below the schemes proposed in this paper. Finally, in [25], it can be seen that there are no fault coverage data is provided and the resource consumption is 33% above the unprotected cipher. As for frequency degradation, no data is provided, but since it is a check and randomisation scheme, the cost must be high and therefore less than 1. In this particular case, the LFSR and XOR countermeasures present a better trade-off than the one presented in [25]. Note that our countermeasures could be applied to other stream ciphers such as Grain, because of their internal structure, which is normally composed by a shift registers and feedback combinational operations.

V. CONCLUSIONS

In this paper a number of countermeasures have been proposed which allow to detect the fault injections on the Trivium stream cipher. In total, four different protection schemes have been presented: hardware redundancy, LFSR signature, XOR and XOR combined with LFSR signature. For the design of these countermeasures have been taken into account the main vulnerabilities of this cipher reported in the literature against fault injections. The designs of these countermeasures have been implemented in both FPGA and ASIC technologies. To test the effectiveness of the countermeasures, an experimental FPGA attack system has been designed based on active non-invasive attacks by manipulating the clock signal. This system has allowed to inject faults (single and multiple faults) into unprotected and protected Trivium ciphers and to analyse the fault coverage of each one. In addition, a complete analysis of experimental results of area consumption, frequency degradation and fault latency have been carried out.

With these schemes, it is possible to detect both single and multiple faults. It has been shown that the fault coverage in the case of the XOR countermeasure is higher than 79.48% while for the other countermeasures it is higher than 87% and reaching 99,99% for the XOR+LFSR countermeasure. In addition, the fault detection latency of the countermeasures, which can go from 1 to 169 cycles, never allow a faulty key stream in the output.

Among the different schemes, the so-called XOR scheme is the one that presents the lowest resource overhead, with low frequency degradation and fault latency, and is therefore one of the best options when resource consumption constraints are very high. However, if the security level is the main objective and resource consumption is not very restrictive, the XOR+LFSR countermeasure is the best option because it is able to detect 99.99% of the injected faults.

It is important to notice that our countermeasures could be applied to other stream ciphers such as Grain, because of their internal structure, which is normally composed by a shift registers and feedback combinational operations.

ACKNOWLEDGMENT

This work was partially funded by Grant PID2020-116664RB-I00 funded by MCIN/AEI/10.13039/501100011033 and by Programa Operativo FEDER 2014-2020 and Consejería de Economía, Conocimiento, Empresas y Universidad de la Junta de Andalucía under Project US-1380823. This work has received funding by the SPIRS (Secure Platform for ICT Systems Rooted at the Silicon Manfacturing Process) Project with Grant Agreement No. 952622 under the European Union's Horizon 2020 research and innovation programme.

REFERENCES

[1] I. K. Dutta, B. Ghosh and M. Bayoumi, "Lightweight Cryptography for Internet of Insecure Things: A Survey," in *IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC'19)*, pp. 475–481, 2019.

[2] N. A. Gunathilake, A. Al-Dubai and W. J. Buchana, "Recent Advances and Trends in Lightweight Cryptography for IoT Security," in *16th International Conference on Network and Service Management (CNSM'20)*, pp. 1–5, 2020.

[3] S. S. Dhanda, B. Singh and P. Jindal, "Lightweight cryptography: A solution to secure IoT," in *Wireless Personal Communications*, vol. 112, no. 3, pp. 1947–1980, 2020.

[4] Z. Kazemi, M. Fazeli, D. Hely and V. Beroulle, "Hardware Security Vulnerability Assessment to Identify the Potential Risks in a Critical Embedded Application," in *IEEE 26th International Symposium on On-Line Testing and Robust System Design (IOLTS'20)*, pp. 1–6, 2020.

[5] S. Mangard, E. Oswald, and T. Popp, "Power analysis attacks: Revealing the secrets of smart cards". *Springer-Verlag*, US, 2007.

[6] 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.

[7] C. De Cannière, "Trivium: A stream cipher construction inspired by block cipher design principles," in *International Conference on Information Security (ISC'06)*, pp. 171–186, 2006.

[8] M. Hojsík and B. Rudolf, "Differential fault analysis of Trivium," in *International Workshop on Fast Software Encryption (FSE'08)*, pp. 158–172, 2008.

[9] M. Hojsík and B. Rudolf, "Floating fault analysis of Trivium," in *International Conference on Cryptology in India (INDOCRYPT'08)*, pp. 239–250, 2008.

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

[11] M. S. E. Mohamed, S. Bulygin, and J. Buchmann, "Improved differential fault analysis of Trivium," in *International Workshop on Constructive Side-Channel Analysis and Secure Design (COSADE'11)*, pp. 147–158, 2011.

[12] M. S. E. Mohamed and J. Buchmann, "Mutant Differential Fault Analysis of Trivium MDFA," in *International Conference on Information Security and Cryptology (ICISC'14)*, pp. 433–446, 2014.

[13] P. Dey and A. Adhikari, "Improved multi-bit differential fault analysis of Trivium," in *International Conference on Cryptology in India (INDOCRYPT'14)*, pp. 37–52, 2014.

[14] F. 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.

[15] F. E. Potestad-Ordonez, M. Valencia-Barrero, C. Baena-Oliva, P. Parra-Fernandez and C. J. Jimenez-Fernandez, "Breaking Trivium Stream Ci-

pher Implemented in ASIC Using Experimental Attacks and DFA," *Sensors*, vol. 20(23):6909, 2020.

[16] X. Fu, X. Wang, X. Dong, and W. Meier, "A key-recovery attack on 855-round trivium," in *Annual International Cryptology Conference*, pp. 160–184, 2018.

[17] L. Jiao, Y. Hao, and Y. Li, "Improved guess-and-determine attack on trivium," *IET Information Security*, vol.13, no. 5, pp. 411–419, 2019.

[18] C. D. Ye, T. Tian and F. Y. Zeng, "The MILP-aided conditional differential attack and its application to Trivium," *Designs, Codes and Cryptography*, vol. 89, no. 2, pp. 317–339, 2021.

[19] C. Ye and T. Tian, "A new framework for finding nonlinear superpolies in cube attacks against Trivium-like ciphers," *Australasian Conference on Information Security and Privacy*, pp. 172–187, 2018.

[20] M. Cianfriglia, S. Guarino, M. Bernaschi, F. Lombardi and M. Pedicini, "Kite attack: reshaping the cube attack for a flexible GPU-based maxterm search," *Journal of Cryptographic Engineering*, vol. 9, no. 4, pp. 375–392, 2019.

[21] R. A. Frohwerk, "Signature analysis: A new digital field service method," *Hewlett-Packard Journal*, vol. 28, no. 9, pp. 2–8, 1977.

[22] F. E. Potestad-Ordóñez, C. J. Jiménez-Fernández, C. Baena-Oliva, P. Parra-Fernández and M. Valencia-Barrero, "Floorplanning as a practical countermeasure against clock fault attack in Trivium stream cipher," in *2018 Conference on Design of Circuits and Integrated Systems (DCIS'2018)*, pp. 1-6, 2018.

[23] B. Lac, A. Canteaut, J. J. A. Fournier and R. Sirdey, "Thwarting Fault Attacks against Lightweight Cryptography using SIMD Instructions," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS'18)*, pp. 1-5, 2018.

[24] D. Shanmugam and S. Annadurai, "Secure Implementation of Stream Cipher: Trivium," in *Innovative Security Solutions for Information Technology and Communications. (SECITC'15), Lecture Notes in Computer Science*, vol. 9522. Springer, Cham, 2015.

[25] S. Ghosh and D. R. Chowdhury, "Preventing fault attack on stream cipher using randomization," in *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST'15)*, pp. 88-91, 2015.

F.E. POTESTAD-ORDÓÑEZ received a B.Sc. degree in Electronic Engineering, the M.Sc. degree in Electronic Products Design (with honors) and the Ph.D. degree from the University of Seville, Spain, in 2013, in 2015 and 2019 respectively. Since 2015, he has been with the Instituto de Microelectrónica de Sevilla (CSIC/US), and the Department of Technology Electronic, Escuela Politécnica Superior of Seville. His current research interests lie in the field of fault injection attacks of cryptographic circuits and the countermeasure study and design for secure systems.

E. TENA-SÁNCHEZ received a B. Sc. degree in Telecomunications in 2010 from the University of Cantabria, Spain, and Electronics Engineering (with honors), M.Sc. degree in Microelectronics and Ph. D. degree from the University of Seville, Spain, in 2012, 2013 and 2019 respectively. Since 2011, she has been with the Instituto de Microelectrónica de Sevilla (CSIC/US) and also joined in 2021 the Department of Technology Electronic, Escuela Politécnica Superior of Seville. Her current research interests lie in the field of CMOS Digital Design of secure cryptographic circuits.

J.M. MORA-GUTIÉRREZ received BSc. 6-year degree in Telecommunications Engineering (specialization microelectronic) at the Universidad Politécnica de Madrid, Spain, in 1990 and a Ph.D. degree in Engineer from the University of Seville, Spain, in 2017. Since April 1992, he has been a member of the Technical Staff at the Instituto de Microelectrónica de Sevilla (IMSE-CNM, CSIC/University of Seville), where he is currently an Assistant Manager in the Research and Development section. His main areas of interest include the design and testing of digital and mixed integrated circuits and field-programmable gate arrays, and the design and testing of IC for cryptography. He has been on the design teams of several integrated circuit and system research projects.

M. VALENCIA-BARRERO received the B.Sc. degree in 1976, the M.Sc. degree in 1977, and the Ph.D. degree in 1986 in Electronic Physics from the University of Seville, Spain. He is currently with the Institute of Microelectronics of Seville, IMSE-CNM-CSIC and also with the Department of Electronic Technology of the University of Seville, where he has been a Full Professor since 2000. His current research interests are in the areas of CMOS Digital VLSI Design, low-power CMOS, lightweight cryptographic circuits, timing in VLSI digital, and modelling and simulation at the logic-timing level.

C.J. JIMÉNEZ-FERNÁNDEZ Graduated in Physics at the University of Seville (Spain) in 1989 and PhD. in Physics at the same university in 2000. He is attached to the IMSE since 1990 and is assistant professor of the University of Seville since 2002. He has participated in 7 projects financed by the EC and in 12 projects financed in national calls. Co-author of more than 40 contributions to workshops, conferences and journals. He has supervised 5 Ph.D. theses. His current research interest includes the experimental vulnerability analysis of cryptocircuits against fault attack and the design of secure cryptocircuits.