INTERNATIONAL JOURNAL OF CIRCUIT THEORY AND APPLICATIONS
Int. J. Circ. Theor. Appl. 2017; **45**:145–169
Published online 13 December 2016 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cta.2296

SPECIAL ISSUE PAPER

Embedded electronic circuits for cryptography, hardware security and true random number generation: an overview

Antonio J. Acosta[1], Tommaso Addabbo[2,*,†] and Erica Tena-Sánchez[1]

[1]*Instituto de Microelectrónica de Sevilla IMSE-CNM (CSIC/Universidad de Sevilla), Seville, Spain*
[2]*Department of Information Engineering and Mathematics, University of Siena, Siena, Italy*

SUMMARY

We provide an overview of selected crypto-hardware devices, with a special reference to the lightweight electronic implementation of encryption/decryption schemes, hash functions, and true random number generators. In detail, we discuss the hardware implementation of the chief algorithms used in private-key cryptography, public-key cryptography, and hash functions, discussing some important security issues in electronic crypto-devices, related to side-channel attacks (SCAs), fault injection attacks, and the corresponding design countermeasures that can be taken. Finally, we present an overview about the hardware implementation of true random number generators, discussing the chief electronic sources of randomness and the types of post-processing techniques used to improve the statistical characteristics of the generated random sequences. Copyright © 2016 John Wiley & Sons, Ltd.

Received 30 September 2016; Revised 24 October 2016; Accepted 25 October 2016

KEY WORDS: lightweight crypto-hardware; hash function; public key cryptography; private key cryptography; side channel attack; fault injection attack; random number generator

1. INTRODUCTION

Nowadays, cyber-security plays a key-role in everyday life, from business to the general public-safety. Cryptography is used for authentication and encryption (bank cards, wireless telephone, e-commerce, pay-TV), access control (car lock systems, restricted areas), and payment (prepaid telephone cards, e-cash) and may become the fundamental instrument of democracy with the advent of e-voting systems [1–3]. As described in a recent report, Gartner estimates endpoints of the Internet of Things will grow in the next years at a compound annual growth rate of 31.7% through 2020, reaching an installed base of 20.8 billion units. In year 2020, 6.6 billion 'things' will ship, with about two-thirds of them consumer applications; whereas, hardware spending on networked endpoints will reach $3 trillion [4, 5]. With such a background and forecast, it is expected that cryptographic hardware will pervade technologies with an increasing demand on energy efficiency, hardware reliability, system integration, portability, and security.

In this complex scenario, the involved computing power ranges within different orders of magnitude, from the foreseen computing capabilities of quantum computers to those of tiny devices like radio-frequency identification tags, industrial controllers, sensor nodes, and smart cards. In these latter devices, the implementation of approved conventional cryptographic NIST standards, like the advanced encryption standard (AES) block cipher and the Secure Hash Algorithm (SHA)-3 hash function, leads to unfeasible solutions in terms of timing performance, chip-area, power, and computing

*Correspondence to: Tommaso Addabbo, Department of Information Engineering and Mathematics, University of Siena, Siena, Italy.

†E-mail: addabbo@dii.unisi.it

resource consumption. This matter sets the point for the lightweight cryptography, that is, the subfield of cryptography aiming to provide solutions tailored for resource-constrained devices.

According to Elsevier Scopus, the largest database of research peer-reviewed literature, since 2010, about 40k documents are returned if searching the keyword ‘cryptography’ [6]. A huge subset of these papers deals with conceptual, algorithmic, software, hardware solutions that may be taken into account in lightweight cryptography. In the face of such a vast literature, in this work, we provide a brief overview of selected crypto-hardware devices, with a special reference to the lightweight electronic implementation of encryption/decryption schemes, hash functions, and true random number generators (TRNGs).

This paper is organized as in the following. In Section 2, we introduce some terminology and present an overview of the hardware implementation of the chief algorithms used in private-key cryptography, public-key cryptography (PKC), and hash functions. In Sections 3 and 4, we introduce some important security concerns about electronic crypto-devices, discussing SCAs, fault injection attacks, and the corresponding countermeasures that can be taken in the hardware design. Finally, Sections 5–8 are devised to provide an overview about the electronic implementation of TRNGs. In detail, in Sections 5 and 6, we discuss about security flaws, statistical defects, and predictability of TRNGs, presenting the chief sources of randomness used nowadays for their hardware implementation. In Sections 7 and 8, we discuss an overview of the different post-processing techniques aimed to improve the statistical characteristics of the generated random sequences and the evaluation methods used to assess the reliability of cryptographic TRNGs. Conclusion and References close the paper.

2. CRYPTOGRAPHIC ALGORITHMS

Cryptographic algorithms aim to convert secret data into an unreadable code for non authorized persons, protecting secret information from theft or alteration and also enabling authentication. For better understanding, in the next sections, we define the following terms. We refer to *plaintexts*(*pt*) as the input messages and *ciphertexts*(*ct*) are the output messages after encryption. Cryptographic algorithms are used in the encryption and decryption processes, where encryption transforms pt into ct using KeyA, and decryption retrieves pt using KeyB, as shown in Figure 1.

To accomplish these goals, cryptography makes use of different algorithms classified into three categories depending on the encrypt mechanism and the number of keys used in the encryption (one key, two keys, or none), see Figure 2:

- *Secret-key cryptography (SKC)*: also called symmetric key criptography, the same key is used for encryption and decryption (KeyA = KeyB). Both sender and receiver have to know the value of the key that, in practice, represents a shared secret between parties that is used to maintain a private information link.

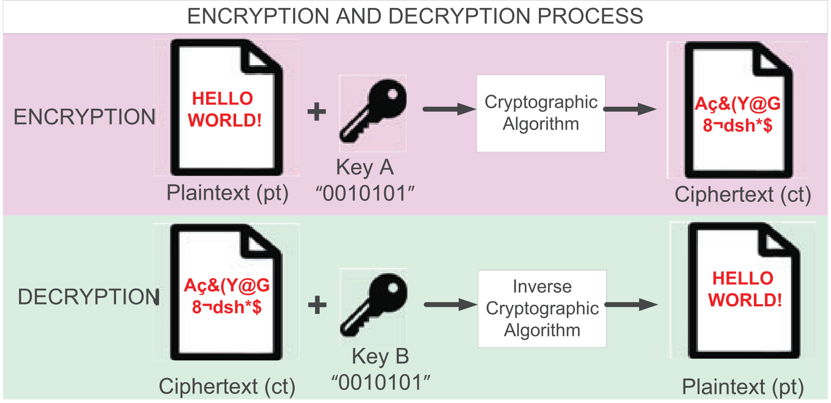

Figure 1. Simplified scheme of encryption and decryption processes. [Colour figure can be viewed at wileyonlinelibrary.com]

DOI: 10.1002/cta

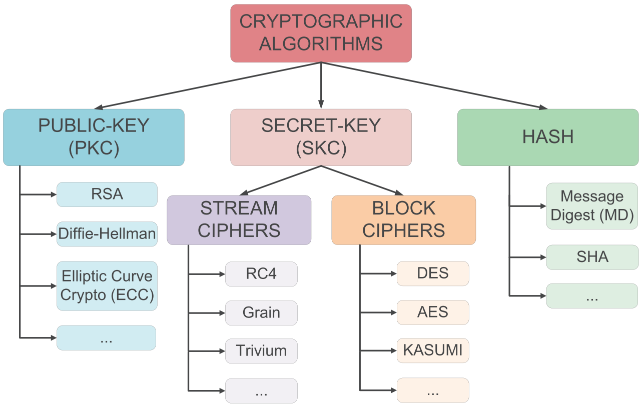

Figure 2. Cryptographic algorithm classification. [Colour figure can be viewed at wileyonlinelibrary.com]

- *PKC*: also called asymmetric key cryptography, two different paired keys are used for encryption and decryption (KeyA ≠ KeyB). KeyA is public, so any sender can use it to send private data that can only be decrypted by the owner of private key KeyB.
- *Hash functions*: uses a mathematical transformation to irreversibly encrypt the information without using any key.

The selection of an specific algorithm within these families depends on the application, security level desired, and related cost. Once selected, the next important issue is the way of implementing it physically. The algorithms can be implemented in different layers, from software down to specific hardware. The hardware implementation of cryptographic algorithms is closer to the hardware device itself, producing higher performance solutions than software, in terms of computational cost, power consumption, and speed.

In embedded crypto-hardware implementations, the cryptographic algorithm is included in an field-programmable gate array (FPGA) or application-specific integrated circuit (ASIC), as a part of the whole system. In many cases, to obtain the hardware implementation of a cryptographic algorithm, a digital synthesis of a hardware description language of the algorithm is made. However, the resulting hardware implementation may not be good enough in terms of performance or security.

For this reason, the designer is often forced to select lightweight hardware-oriented cryptographic algorithms, to be used in modern portable and wearable systems in the scenario of the Internet of Things. Also, special design techniques to increase the security of the algorithms against SCAs must be incorporated.

2.1. Secret-key/symmetric cryptography

Secret-key cryptography algorithms are classified into two groups depending on how the plaintext is encrypted: bit by bit in stream ciphers and through data blocks in block ciphers.

Stream ciphers generate a keystream that is XORed (XOR operation) with the plaintext bit by bit. They implement some kind of feedback mechanism so that the keystream is continuously changing producing different ciphertexts for the same plaintext in each encryption depending on the key, the initial value, and the encryption cycle [7–9].

There are several examples of used stream ciphers. For example, in the one-time pad [7], the plaintext is XORed with a truly random key bit by bit. Its main problem is that the key length has to be the same as the plaintext length, so it needs a huge amount of key bits. This cipher has been widely used but nowadays has been replaced because of its key length.

On demand of lightweight hardware implementations, the eSTREAM project [10] presented in 2004 the specific profile for hardware-oriented algorithms. Grain and Trivium ciphers were ones of the finalists. Grain [8] targets hardware environments where gate count, power consumption, and memory are very limited. It is based on two shift registers and a non-linear filter function as shown in Figure 3. An FPGA implementation of Grain is presented in [11]. Trivium [9] was designed to have the most

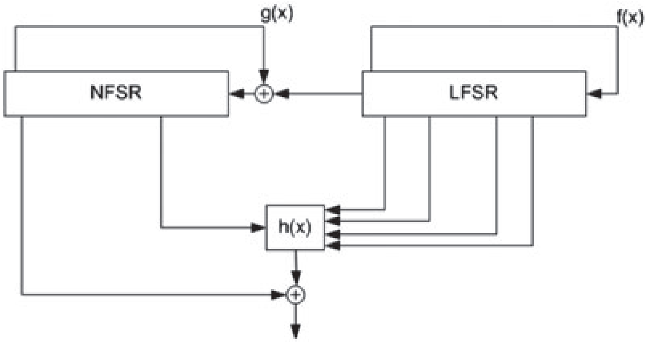

Figure 3. Simplified implementation of Grain algorithm [8].

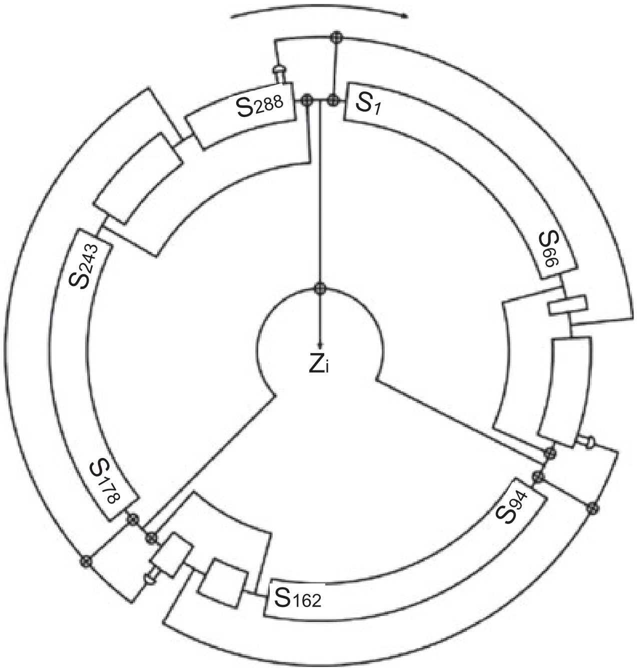

Figure 4. Simplified implementation of Trivium algorithm [9].

simplified structure without sacrificing security, speed, or flexibility. Trivium has a 80-bit secret key and 80-bit initial value, see Figure 4. Some hardware implementations of Trivium are presented in [11, 12]. Other hardware oriented stream ciphers submitted to eSTREAM project were Mickey, Moustique, and F-FCSR-H v2 among others [10].

Block ciphers encrypt one block of data at a time using the same key on each block. In general, the same plaintext block will always encrypt to the same ciphertext when using the same key in a block cipher. Some of the most commonly used block ciphers are the data encryption standard (DES) [13] and AES [14]. DES was designed in the 1970s and was adopted by the US government for commercial and unclassified government applications. DES is a block-cipher employing a 56-bit key that operates on 64-bit blocks. Some hardware implementations based on FPGA are presented in [15]. DES was abandoned because of its short key length.

In 1997, NIST initiated a public process to develop a new secure block cipher for US government applications. The result, the AES, became the official successor to DES and 3-DES in November 2001. AES encrypts data of a fixed block length (128 bits) under a key, which can either have 128, 192, or 256 bits. Currently, it is considered secure enough for critical applications. The first reported ASIC implementation of AES is in [16].

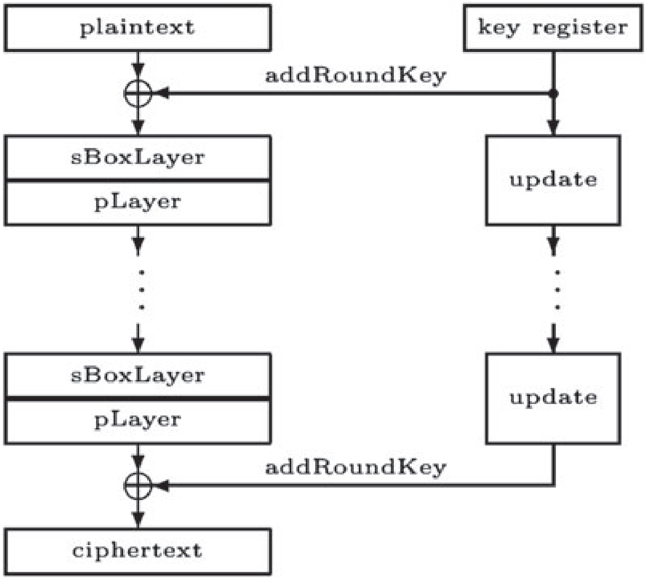

Figure 5. Top-level description of Present [17].

As in case of stream ciphers, because of demand of lightweight hardware implementations, new lightweight block ciphers have been presented. An example is Present [17], an ultra-lightweight block cipher notable for its compact size (about 2.5 times smaller than AES) with block size of 64 bits, and the key size can be 80 bit or 128 bit, see Figure 5.

2.2. *Public-key/asymmetric cryptography*

Secret-key cryptography needs a secure channel to exchange the key between sender and receiver, being this a serious drawback in many cases. So, in 1976, a novel branch of cryptography called PKC was introduced [18]. This method allows, with use of symmetric ciphers, the key exchange in a secure way even though making the communication in insecure/public channels.

Public-key cryptography uses a pair of keys: the public key *Key A* and the private key *Key B* that belongs only to the owner. Two functions can be achieved: using a public key to authenticate that a message originated with a holder of the paired private key or encrypting a message with a public key to ensure that only the holder of the paired private key can decrypt it [3].

Public-key cryptography algorithms that are in use today for key exchange or digital signatures include RSA (the well-known public-key cryptosystem developed by R. Rivest, A. Shamir, and L. Adleman) [19] and those based on elliptic curve cryptography (ECC) among others. RSA is the most used PKC implementation, with keys from 1024 to 4096 bits typically, preventing practical attacks. A hardware implementation in FPGA of RSA is presented in [20]. PKC algorithms based upon ECCs were initially proposed independently in [21, 22]. ECC is an approach to PKC based on the algebraic structure of elliptic curves over finite fields. It requires smaller keys compared with non-ECC cryptography, based on plain Galois fields, to provide equivalent security. Some FPGA and ASIC implementations can be found in [23, 24].

2.3. *Hash functions*

Hash algorithms take input plaintexts of arbitrary length and translate them to short fixed-length output strings without using any key. These algorithms are one way encryption algorithms because once the plaintext is computed it is impossible to recover neither the plaintext nor the length of it.

Hash algorithms are typically used to provide a digital fingerprint of a file's contents, often used to ensure that the file has not been altered by an intruder or virus. Also, they are commonly employed by many operating systems to encrypt passwords, providing a measure of the integrity of a file.

Some of the most used Hash algorithms are as follows:

- *Message digest (MD) algorithms* [25]: A series of byte-oriented algorithms that produce a 128-bit hash value from an arbitrary-length message (MD2, MD4, MD5). An FPGA implementation of MD5 is in [26].
- *SHA* [27]: family of cryptographic hash functions published by NIST. Some hardware implementations in FPGA of SHA-256 [28] and SHA-384 and SHA-512 are presented in [29]. Keccak [30] was selected in 2012 to become the new SHA-3 hash algorithm because it has higher performance in hardware implementations than SHA-2 or any of the other finalists. Some lightweight hardware implementations of Keccak are presented in [31, 32].

3. SECURITY OF CRYPTOGRAPHIC DEVICES

A cryptographic algorithm is considered to be secure in practice if there is no attack known that can break it within a reasonable amount of time and with reasonable amount of computing power. But although algorithms are mathematically safe, their physical implementations on hardware can leak side-channel information that can be used by third parties to reveal critical data, usually the secret key, through SCAs or by fault injection attacks [33]. The main objective of cryptohardware is the design of secure cryptographic devices onto electronic platforms to implement cryptographic algorithms and store cryptographic keys in a secure way, resisting any kind of malicious attack.

There exist different attack strategies that differ significantly in terms of cost, time, equipment, and expertise needed. They can be classified depending on two characteristics: if they are active/passive or if they are *invasive/non-invasive* [33]. Figure 6 summarizes the attack classification.

Invasive attacks manipulate the device, usually depackaging the chip and accessing to internal layers, while non-invasive attacks collect information provided by the device (accessible I/O, power consumption, execution time, …) without modifying it.

In a *passive attack*, the secret key is revealed while the cryptographic device operates in a correct way during encryption, analyzing side channel information as power consumption, timing, acoustic, or electromagnetic radiation. On the other hand, an *active attack* changes the device functionality during encryption manipulating its inputs, power supply, or environment, among others. This malfunction during encryption and the outputs provided by that operation can be used to reveal the secret key.

The most powerful attacks are invasive ones, being either passive or active, but they are very expensive in terms of time, cost, and effort, making in most cases an irreversible damage in the crypto-device. On the other hand, the non-invasive attacks are a big threat to cryptographic community because they usually require minimal equipment, effort, and cost, and they are very effective.

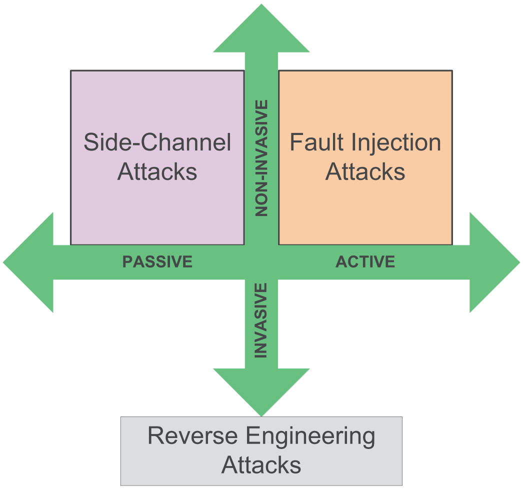

Figure 6. Attack classification. [Colour figure can be viewed at wileyonlinelibrary.com]

We will focus on non-invasive attacks, mainly *fault injection attacks*, where the normal operation of the device is changed injecting a fault, and SCAs, where the secret key is retrieved by monitoring the leakaged information during normal operation of the cryptographic device.

In next subsections, SCAs and fault injection attack techniques are exposed.

3.1. Side-channel attacks

Side-channel attacks on cryptographic devices use certain physical information such as power consumption, time delay, or electromagnetic radiation to find the secret key. SCAs usually require minimal equipment; hence, they are easy to carry out [33].

The most known SCAs are as follows:

- *Timing attacks* [34]: the secret key can be obtained by carefully measuring the time involved in cryptographic operations, exploiting the timing variance in the operation. A practical timing attack against an actual smart card implementation of the RSA was conducted in [35]
- *Power Analysis attacks* [36]: it takes advantage on the dependence of power consumption in cryptocircuits on data being processed. This dependency can be exploited to retrieve secret data from electronic devices conducting simple power analysis or differential power analysis (DPA) attacks. Simple power analysis takes information using a small number of power traces or even one single key, being quite challenging in practice because the attacker needs a detailed knowledge of the attacked device, so only are useful when few traces are available to the attacker. More powerful and effective are DPA attacks, being the most popular type of power analysis attack. Although it needs a huge amount of power traces, the attacker do not require detailed knowledge of the device, but power models, and can operate in a very noisy environment [33, 37–39].
- *Electromagnetic attacks (EM)* [40, 41]: are very similar to those based on the power consumption, but using the electromagnetic radiation of the device. Simple (SEMA) and differential electromagnetic analysis attacks use few or a huge amount of electromagnetic traces, respectively. There are a lot of works presenting EM attacks in cryptographic hardware implementations as in [42, 43].
- *Acoustic attacks*: the acoustic emanations of the electronic devices during encryption have correlation with the processed data. A first work using this technique was presented in 2004 [44].

There are many SCAs in the literature for SKC, PKC, and hashing. DPA attacks on block ciphers has received a lot of attention, for instance DES in [36] and AES in [45]. There is less work related to side channel vulnerability analysis on stream ciphers, but not less important. Theoretical DPA attacks on A5/1 and E0 stream ciphers are presented in [46], and on Trivium and Grain in [47].

3.2. Fault injection attacks

Fault injection attacks insert any kind of malfunction on the operation during encryption, using this wrong result to retrieve the secret key of a device.

Fault injection attacks were introduced in 1997 [48] where a fault in a computation was used to attack an RSA implementation using the Chinese Remainder Theorem. Since then, a huge amount of works have been presented in literature presenting different kinds of fault injection attacks to retrieve the secret key of cryptocircuits, making a big deal to protect devices against all kind of attacks. Fault injection technique overview can be found in [49–51]. Here is a brief summary of fault injection techniques:

- *Power supply variations*: a cheap and simple way to inject a fault is to under-power or insert a power spike in the power supply of a cryptographic device. This supply voltage variation causes malfunction on the device that can be used to reveal critical data [52].
- *Variation in the external clock*: they may cause malfunction in the cryptographic device. An example of this attack is presented theoretically in [53] and experimentally in [54] on Trivium stream cipher, injecting a glitch in the clock signal. There are also some fault attacks presented in block ciphers as the work in [55], where the block ciphers AES, DES, Camellia, CAST-128, SEED, and MISTY1 are attacked by injecting faults into any desired round by supplying a clock signal with a glitch.

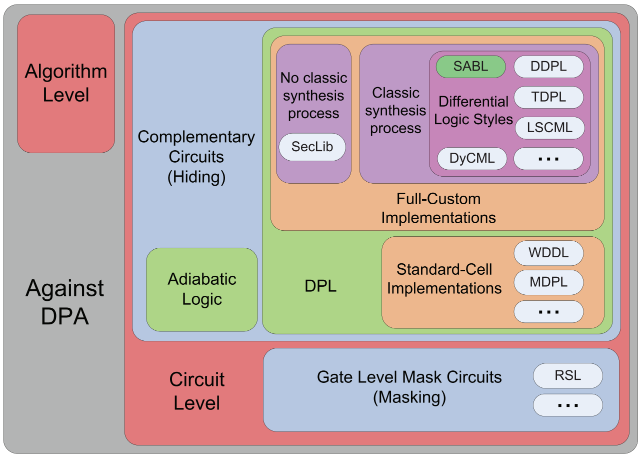
Algorithm Level
Against DPA
Complementary Circuits (Hiding)
No classic synthesis process
SecLib
Classic synthesis process
Differential Logic Styles
SABL
DDPL
TDPL
LSCML
DyCML
...
Full-Custom Implementations
Adiabatic Logic
DPL
Standard-Cell Implementations
WDDL
MDPL
...
Circuit Level
Gate Level Mask Circuits (Masking)
RSL
...

Figure 7. Countermeasures classification [56]. [Colour figure can be viewed at wileyonlinelibrary.com]

- *Temperature variations*: raising or decreasing the temperature of the cryptographic device during encryption to produce an error [49].
- *Electromagnetic pulses*: an external electromagnetic field is applied near the device to cause malfunction and retrieve secret information from it [49].

4. COUNTERMEASURES FOR CRYPTOGRAPHIC CIRCUITS

All the aforementioned attacks are performed on physical-hardware implementation of the algorithms. There is not a unique solution to prevent side-channel and fault injection attacks, but the solutions are forced to be developed at a hardware level, being the countermeasures designed physically on silicon. In this section, a brief survey of the countermeasures against different kinds of attacks is performed.

4.1. *Countermeasures against SCAs*

Since the first SCAs presented in [34, 36, 40], dozens of countermeasures have been proposed to deal with this type of intrusion. There are different kinds of countermeasures against effective PA/EMA attacks to be applied depending on the abstraction level, from algorithm to layout [56–58], see Figure 7.

The use of countermeasures at algorithmic level is a hard issue because of the high dependency of the secure implementation on the specific cryptographic algorithm. This means that this technique is very specific and difficult to automate. Some approaches are presented in [59, 60].

At circuit level, there are two main options that are independent of the specific algorithm used, being valid for SKC, PKC, and hashing. The first one is the use of gate level mask circuits (*masking*) studied in [61, 62], where the designer tries to remove the data dependency with power consumption by using a mask mixed with an intermediate value of the processed data. The other alternative at circuit level is *hiding* [63, 64], where the implementation of a logic circuit achieves theoretically the same power consumption independently of the data being processed.

Between hiding techniques, those using dual-rail precharge logic (DPL) styles with standard-cell libraries or full-custom implementations are the most effective ones. DPL gates compute always the output and its complementary, having in all clock cycles a transition in the output node, achieving thus in all clock cycles the same power consumption independent on the data being processed (Figure 8).

In DPL techniques, the ones using standard cells are wave dynamic differential logic (WDDL) [65] and masked dual-rail pre-charged logic (MDPL) [66], among others. Those using full-custom techniques show the best results in terms of security and performance if they are correctly designed, also at layout level [56]. Some example of full-custom DPL techniques are DyCML [67], LSCML [68], SABL [63], and DDPL [69]. All these techniques use differential circuits to perform the logic operation in a pull-down circuit, alternating precharge, and evaluation phases; thanks to the action of pull-up circuitry. The success lies on full symmetry and lack of memory effect. Some improvements can be found

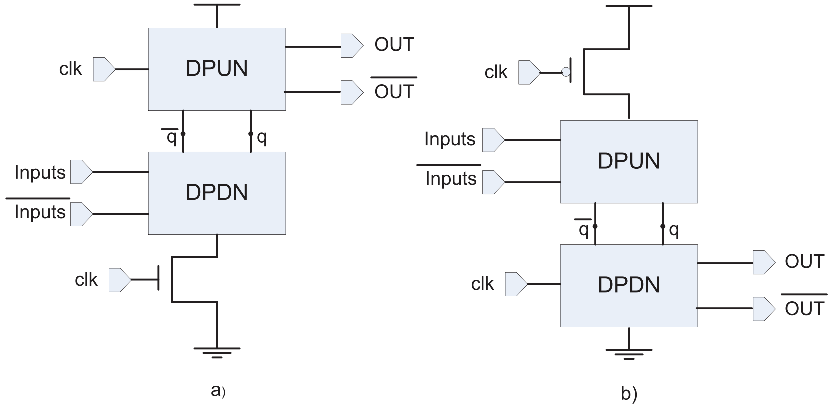

Figure 8. Dynamic and dual-rail gate logic style. (a) Using NMOS transistors to implement the DPDN block logic function. (b) Logic function implemented with PMOS transistors (DPUN). [Colour figure can be viewed at wileyonlinelibrary.com]

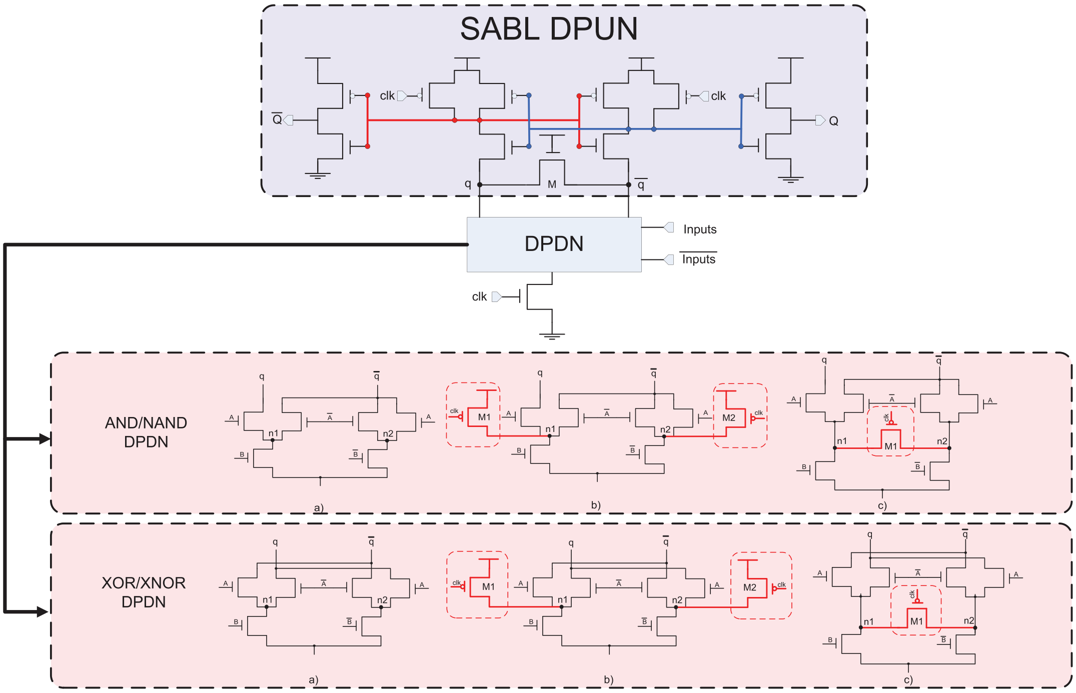

Figure 9. SABL, DPUN, and different AND/XOR DPDN combinations: (a) classic, (b) double-switch solution, and (c) single-switch solution. [Colour figure can be viewed at wileyonlinelibrary.com]

in [56]. In Figure 9, the widely used SABL logic style is shown as DPUN, with the differential DPDN network for the AND and XOR implementations, and two optimization techniques in the DPDN, to remove the memory effect for the XOR gate in (b) and (c) [56].

There are some novel countermeasures presented in [70–72]. In [70], a data-dependent delay assignment, where a reduction in the dependency of power consumption and the processed data is achieved by introducing delays in the data paths, is presented. In [71], a so-called process obfuscation, which can be used as a countermeasure for SCAs on sensor nodes, is presented. Finally, in [72], a novel circuit concept, which decouples the main power supply from an internal power supply that is used to drive a single logic gate, is presented.

DOI: 10.1002/cta

4.2. Countermeasures against fault injection attacks

There are two ways of protecting a cryptographic device against fault attacks [51, 73]:

- *Hardware countermeasures*: Using prevention mechanisms, as for instance metal shields over the ASIC to prevent it from illumination and external access. There are also reported mechanisms to detect light, under-powering, clock glitch injections, or optical fault injection attacks [51].
- *Design driven*: The cryptographic device is made secure against fault injection attacks, by adding redundancy in the design to check and report faults, or designing the implementation to be immune to fault injection.

Some hardware implementations of cryptographic devices to counteract fault attacks are presented in [74, 75]. In [74], an AES implementation is protected from suffering from differential fault attacks, by using the error detection technique to detect the errors forced during encryption or decryption and then providing the information for taking further action, such as interrupting or redoing the AES process. In [75], a novel concurrent error detection scheme is proposed to counter fault-based attack against RSA by exploiting its multiplicative homomorphic property.

5. RANDOM NUMBER GENERATORS

In cryptographic applications, the scope of a random number generator (RNG) is to provide sequences of random integers that are deemed to be *unpredictable*. RNGs represent a fundamental class of cryptographic hardware primitives, and in most cases, the overall theoretical security of a cryptographic protocol relies on the effectiveness of the random numbers used to set up and carry out the communication process [1–3, 76].

Nowadays, circuits and systems proposed to implement RNGs are divided in two intimately related categories, that is, TRNGs and pseudo RNGs (PRNGs), both playing fundamental roles in cryptography. As it is made clear in the following sections, TRNGs are devised to issue random numbers exploiting the measurement of truly stochastic physical processes. On the other hand, PRNGs are deterministic finite state-machines, eventually periodic, capable to generate, within their period, binary sequences that *appear as if* they are truly random [76, 77]. In few words, from a conceptual point of view, a PRNG is a device issuing and repeating indefinitely a finite random sequence, stored in its memory or generated according to different calculations. A number of efficient, interesting, and advanced methods to implement PRNGs have been proposed in the literature, concerning the research areas of number theory, discrete mathematics, and digital circuits [2, 76–90]. Linear and nonlinear congruential generators or feedback shift registers are well-known PRNGs used in a wide set of engineering fields. A basic text introducing to the subject is the book of D. Knuth, the Art of Computer Programming [91]. A review of different PRNGs can be found in [76, 77, 81].

In this paper, the discussion mainly focuses on TRNGs, whereas PRNGs are mentioned throughout the text only when useful.

5.1. A theoretical sketch for TRNGs

To make clear the fundamental properties of TRNGs, it is convenient to introduce some formal definitions taken from Information Theory [92]. From a theoretical point of view, a TRNG is an *information source* typically modeled as an ergodic stochastic process $\mathbf{S} = \{s_n\}, n \in \mathbb{N}$, whose realizations are infinite sequences of symbols, chosen among the elements of a finite set (alphabet) $\mathcal{M} = \{0, 1, \dots, m-1\} \subset \mathbb{N}$. In most cases, the alphabet is made of numbers represented by groups of bits (e.g., binary words), or in the simplest case ($m = 2$), the binary symbols $\{0, 1\}$. In the latter case, TRNGs are often referred to as a true random bit generators (TRBGs) [76].

In the following, we write $P(s_n = a)$ to denote the probability for the TRNG to issue the symbol $a \in \mathcal{M}$ at the time-step n. When considering joint probabilities, it is convenient to use the compact notation $P(\bigwedge_{i=1}^{k} s_{n_i} = a_i)$ to denote the probability for the TRNG to issue the symbols $a_1, \dots, a_k$ at the time steps $n_1, \dots, n_k$. Finally, we write $P(A|B)$ to denote the conditional probability for the event A to take place once assuming the event B occurred, that is, $P(A|B) = P(A \cap B)/P(B)$, with $P(B) > 0$. Referring to the introduced notation, we can provide a theoretical definition for an unpredicatble TRNG.

Definition 1
The stochastic process $\mathbf{S} = \{s_n\}, n \in \mathbb{N}$, is an ideal TRNG with alphabet $\mathcal{M} = \{0, \dots, m-1\}$ if and only if

1. $\forall n \in \mathbb{N}$ and $\forall a \in \mathcal{M}$, $P(s_n = a) = \frac{1}{m}$;
2. $\forall k \in \mathbb{N}$, $k > 1$, for all k-tuples of distinct natural numbers $(n_1, \dots, n_k)$ and for all k-tuples $(a_1, a_2, \dots, a_k) \in \mathcal{M}^k$ of symbols in $\mathcal{M}$, it results

$$P\left(s_{n_k} = a_k | \bigwedge_{i=1}^{k-1} s_{n_i} = a_i \right) = P(s_{n_k} = a_k). \tag{1}$$

An ideal TRNG is also referred to as an unpredictable TRNG.

From the earlier definition, it follows that an unpredictable TRNG is an ergodic stochastic process issuing a sequence of statistically independent and identically distributed (i.i.d.) discrete random variables, uniformly distributed among the first m natural numbers. As a theoretical consequence, because in (1), the n-tuple $(n_1, \dots, n_k)$ is given without any ordering, an unpredictable TRNG has no memory of the past generated symbols, and, reversing the time axis, the source has no memory of the future symbols, as well.

5.2. *Predictability of non-ideal TRNGs*

Given the Def. 1, it is important to stress the resulting concept that *a not-ideal TRNG is predictable in some sense*. The security of a cryptographic protocol (e.g., an encryption/decryption scheme) can be analyzed from different sides, but at its very root level, there always lies an RNG. If the numbers used in the protocol, deemed to be truly random, have instead some degrees of predictability, the security of the entire scheme may be compromised, for example, by exponentially decreasing the *average number of trials* that an attacker is expected to perform to break the encryption, using the so-called *brute-force attack*.

Accordingly, the aim of any hardware TRNG is to *approximate* an ideal TRNG at its best. Information Theory provides useful theoretical tools to express how well this approximation is achieved.

Definition 2
The average Shannon entropy(ASE) of a TRNG $\mathbf{S}$ is equal to

$$\text{ASE}(\mathbf{S}) = \lim_{k \to \infty} -\frac{1}{k} \sum_{\beta_k \in \mathcal{M}^k} P(\beta_k) \cdot \log_2 P(\beta_k) \tag{2}$$

where the summation extends to the finite set collecting all binary k-tuples β_k with positive generation probability.

Because in (2) logarithms with base 2 are used, the result is expressed in bits/symbol. The ASE indicates, for a given TRNG, the average amount of information issued at each time-step. From the earlier definition, it is immediate to check that for the ideal TRBG ($m = 2$ in Def. 1) the ASE is equal to 1 bit/time-step. Indeed, from the i.i.d. property of the binary output, for any $k \in \mathbb{N}, k > 0$, it results

$$-\sum_{\beta_k \in \{0,1\}^k} P(\beta_k) \cdot \log_2 P(\beta_k) = 2^k \cdot \frac{1}{2^k} \log_2 2^k = k \log_2 2 = k, \tag{3}$$

and the limit (2) is equal to 1 bit/time-step. In such case, the ASE agrees with the maximum classical Shannon entropy for a binary source [92]. For most hardware TRNGs, an adequate estimation of (2) can result almost unfeasible, because it involves statistical distributions of any order.

5.3. *Statistical defects in non-ideal TRNGs*

A non-ideal TRNG fails to satisfy at least one of the two given conditions given in Def. 1. In most cases, any hardware TRNG fails both of the conditions at the same time, exhibiting *statistical defects* in its output that can be exploited to guess the most probable expected forthcoming symbols.

Statistical defects in the output sequence of TRNGs can be classified in stationary, related to the specific TRNG nominal design, or non-stationary, that may depend on the device aging or the environment (due to, e.g., external tampering, electromagnetic couplings, temperature, and electronic supply voltage variations).

From a theoretical point of view, statistical defects in TRNGs originate from its *statistical bias* (i.e., symbols are not evenly distributed in probability) and from its *memory* (i.e., the probability for a symbol to be generated in the future, depends on the past generated symbols). The statistical bias provides a direct advantage to an adversary to predict the TRNG, because some symbols are simply more probable than others (intuitively, the device is similar to an unfair dice). Similarly, TRNGs affected by memory suffer from correlation between the generated symbols. Also in this case, the autocorrelation function r_{xx} of a TRNG can be exploited to predict its future symbols, as it can be easily shown, without loss of generality, focusing on the special case of a TRBG ($m = 2$ in Def. 1). Indeed, for all $n \in \mathbb{N}, k \in \mathbb{Z}$, $k \geq -n$,

$$r_{xx}(k) = \sum_{b_1=0}^{1} \sum_{b_2=0}^{1} b_1 \cdot b_2 \cdot P(s_n = b_1, s_{n+k} = b_2) = P(s_n = 1, s_{n+k} = 1), \tag{4}$$

and

$$r_{xx}(0) = P(s_n = 1). \tag{5}$$

Recalling that $P(s_n = 1, s_{n+k} = 1) = P(s_{n+k} = 1|s_n = 1)P(s_n = 1)$, it directly result

$$P(s_{n+k} = 1|s_n = 1) = \frac{r_{xx}(k)}{r_{xx}(0)}; \tag{6}$$

that is, the probability to have the symbol $s_{n+k} = 1$ given the symbol s_n equal to 1 can be determined directly from the knowledge of the autocorrelation function r_{xx}, that can be easily estimated using the well-known estimator

$$\langle r_{xx}(k) \rangle = \frac{1}{N-k} \sum_{n=0}^{N-k-1} s_n \cdot s_{n+k}. \tag{7}$$

The earlier discussion can be easily extended to more complex systems to show that, in general, in a TRNG statistical biasing and memory decrease its ASE. As a countermeasure, to mitigate the deterioration of the statistical characteristics of the generated sequence, in cryptographic TRNGs, the last stage is a fully digital post-processor unit as shown in Figure 10. The post-processing is based on two different approaches, widely investigated in Cryptography and Information Theory: compression and diffusion/confusion. This topic is discussed in Section 7.

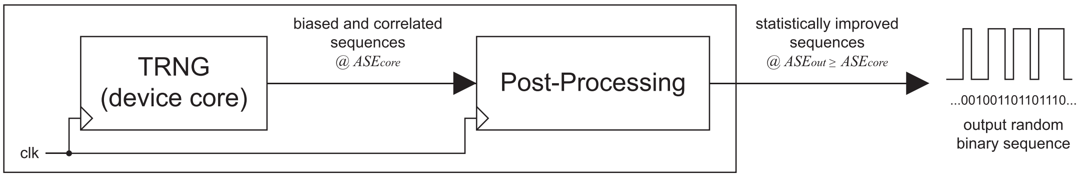

Figure 10. A generic scheme representing a cryptographic TRNG.

6. SOURCE OF RANDOMNESS IN TRNGS

A TRNG outputs random numbers exploiting a truly stochastic physical phenomenon. For the sake of our outline, hardware TRNGs are defined as *mixed-signal circuits* that can be classified depending on the source of randomness taken into account for their *conceptual design*, that is, based on the following:

- chaotic circuits;
- high-jitter oscillators;
- circuits to measure other stochastic physical processes.

Different authors have successfully proposed TRNGs exploiting each of the above approaches, and a combination is sometime used [93–95]. In the following subsections, we provide a brief review of these techniques.

6.1. Chaotic circuits

A chaotic circuit is an analog or, more often, a mixed-signal circuit in which currents and voltages changes in time, according to a deterministic evolution rule satisfying special mathematical properties [96]. In these circuits, the time evolution of currents and voltages is theoretically described as the state evolution of a nonlinear dynamical system exhibiting chaotic behavior.

A formal definition of chaos involves mathematical concepts introduced by Ergodic Theory, like topological transitivity, mixing, and measure preserving transformations [99–101]. For the sake of our outline, adopting a qualitative point of view, chaotic dynamical systems can be described as *deterministic aperiodic systems displaying sensitive dependence on initial conditions* [98, 100]. Furthermore, let us stress that the state evolution of a $n-$dimension chaotic system describes a moving point in $\mathbb{R}^n$, defining a so-called chaotic trajectory. Well-known chaotic systems are the Lorenz system, the Logistic map, the Hénon map, the Rössler system, and the double rod pendulum [98, 100]. Other chaotic dynamical systems, like the well-known Chua's system, the Tent map, or the Sawtooth map, have been particularly investigated in literature for their specific electronic hardware implementation [96, 102–110].

Chaotic systems can be classified in continuous-time or discrete-time systems. In the former case, the state evolution defines a signal $x(t)$, $x : \mathbb{R} \to \mathbb{R}^n$, being the state evolution ruled by a set of nonlinear differential equations, typically of the form $\dot{x} = f(x)$. In the discrete-time case, the state evolution defines a sequence $\{x(t_n)\}$, $x : \mathbb{N} \to \mathbb{R}^n$, being the state evolution ruled by a set of difference (recurrence) equations, typically of the form $x_{n+1} = f(x_n)$.

Ergodic and Information Theories provide the theoretical tools to design a TRNG exploiting a chaotic dynamical system. The result, often referred as *symbolic dynamics*, is achieved by construction, defining a process devised to sample, measure, and code the state of the chaotic system, adopting different strategies. The symbolic dynamics is typically obtained sampling and quantizing the projection of the system state in lower-dimensional subspaces, or coding the intersection of the chaotic trajectory with specific submanifolds, called Poincaré sections [99–101, 103, 112]. Even if the time-evolution of the system state is ruled by deterministic equations, proper symbolic dynamics can be obtained using coding techniques, discarding some information related to the system state, defining an information source ruled by an actual stochastic process [101, 103].

In TRNG design, an important class of chaotic systems is the family of discrete-time one-dimension piecewise linear maps, in which the recurrence equation $x_{n+1} = f(x_n)$ is given by piecewise linear functions (see, e.g., the Sawtooth map in Figure11). The importance of these maps comes from both the specific theoretical tools provided by Ergodic Theory for their investigation and the specific electronic design involved for their hardware implementation [96, 104–110, 113–121]. Among the cited literature, it is worth to mention the seminal papers, in chronological order, [105] (hardware implementation of the Tent map), [106] (hardware implementation of different discrete maps, including the Hénon Map), [107] (hardware implementation of the Sawtooth map), [108] (hardware implementation of a zigzag map for flicker noise generation), and [110] (hardware implementation of a truly mixed-signal discrete map).

For the sake of an example, it has been theoretically proved that the Sawtooth map $x_{n+1} = 2x_n \bmod 1$ and the Tent map $x_{n+1} = 1 - 2|x_n - 0.5|$ can be used to obtain the *ideal* TRNG, once the symbolic

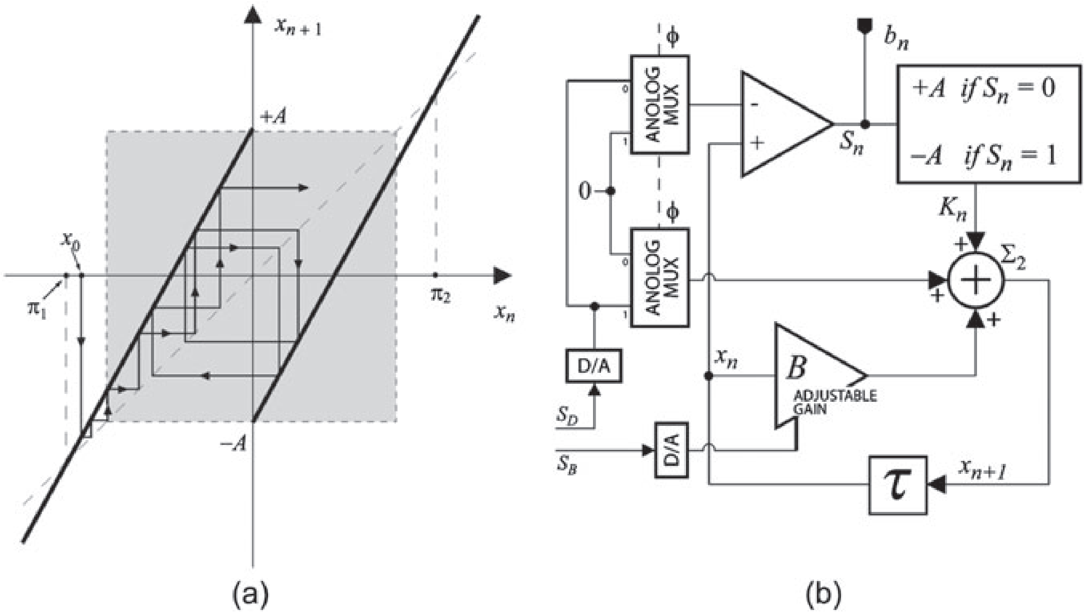

Figure 11. The Sawtooth discrete-time one-dimension piecewise linear map (a), and a TRNG exploiting this map with control signals to finely adjust the chaotic system parameters (b) [97]. In (a), the chaotic trajectory triggered by the initial condition x_0 is shown using a cobweb plot [98]. The true random binary sequence is collected at the net b_n in (b).

dynamics is designed to issue the sequence of binary symbols $\{b_n\}$, where b_n = '1' if $x_n > 0.5$, b_n = '0' otherwise. Nevertheless, the statistical characteristics of the sequence generated by these systems are highly sensitive to the chaotic system parameters perturbations, causing an issue that must be carefully taken into account when designing the hardware implementation of these TRNGs [96, 97, 103, 122–125]. The electronic design of piecewise linear chaotic maps has been investigated following different approaches and targeting different applications, including true random numbers generation, secure communication, and colored noise generation [96, 104–111, 120, 121].

In Figure 12, a CMOS circuit to implement the algebraic calculation of the Sawtooth nonlinear function shown in Figure 11(a) is reported, using cascode current mirrors [111]. The circuit calculates $f(I_n)$, being the chaotic state variable represented by a current. The complete iterated execution of the computation $I_{n+1} = f(I_n)$ is obtained by means of a delay block realized with track-and-hold switched-current stages [111, 126].

6.2. High-jitter oscillators

Jitter noise can be defined as the deviation of an oscillator output from its true periodicity, causing uncertainty in the low–high/high–low transition times [127–129]. The operation of TRNGs exploiting high-jitter oscillators is typically based on the interaction of independent free-running oscillators, expressively designed to exhibit high-jitter noise and having a relatively large difference between the nominal frequencies [130–134]. As shown in Figure 13, in the simplest solution, the slow oscillator is used to trigger the sampling of the fast oscillator. The frequency of the fast oscillator is typically greater than up to two order of magnitude of the slower one, being the oscillators obtained using ring-oscillators or similar structures. A further latch can be used to synchronize the digital stream to a master clock signal.

Differently from other kind of TRNGs, some solutions of this type can be implemented in fully digital processes, even in FPGAs or using micro-controllers, and this can be an advantage in several applications [135–137]. On the other hand, as discussed in the following, these TRNGs may exhibit correlation among symbols and instability of the statistical characteristics of the generated sequences, depending on the ratio between the two oscillator frequencies, on the jitter noise level, and on the sensitivity of the oscillation frequencies to aging, temperature, and voltage variations [134, 137–140].

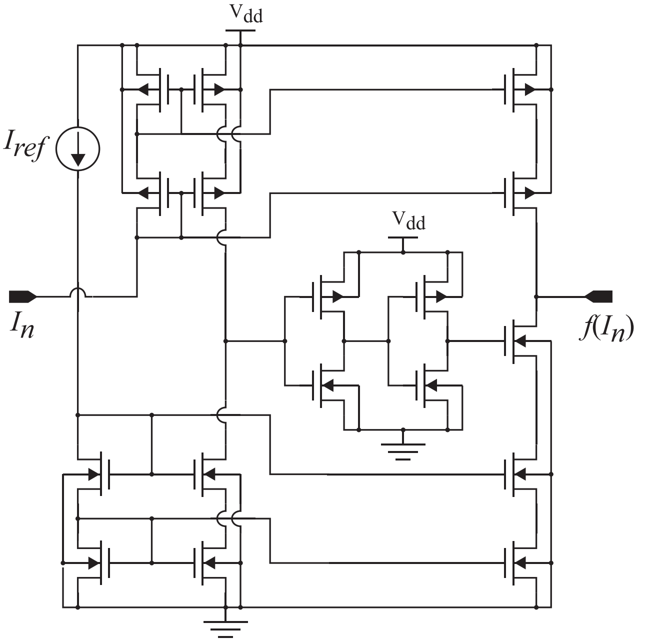

Figure 12. CMOS circuit to implement the algebraic calculation of the Sawtooth nonlinear function shown in Figure 11(a), using cascode current mirrors [111]. The chaotic state variable is represented by the current I_n.

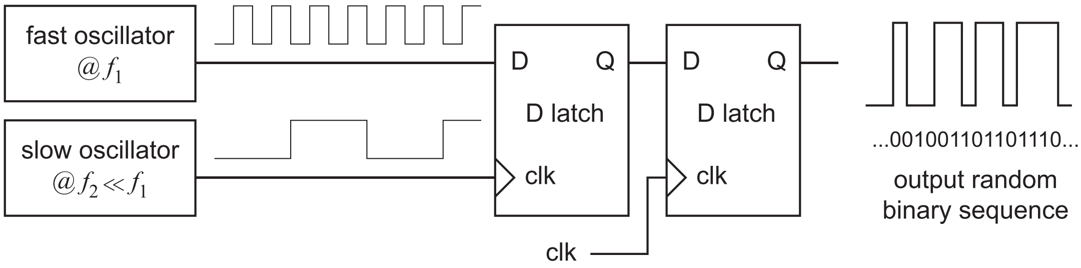

Figure 13. A schematic representation of the core structure of a TRNG exploiting high-jitter oscillators.

Jitter noise has been deeply investigated in literature, mainly due to its effects, for example, in sampling devices and clock distribution in digital circuits. Most authors proposing this kind of TRNGs assume jitter noise to be completely random and Gaussian distributed, whereas in practice important deterministic components may arise due to different factors, among which the presence of deterministic variations in the supply voltage, the crosstalk between the involved oscillators, between the whole TRNG section and the neighbor circuitry or other external electromagnetic sources [129, 134, 139, 140].

Starting from the structure shown in Figure 13, several solutions have been proposed in literature, using voltage controlled oscillators, chaotic systems (as in Figure 14), and free running digital loops with circuit topologies inspired to hardware PRNGs, mixing the two paradigms of randomness and pseudorandomness (the Fibonacci and Galois Ring Oscillators shown in Figure 15) [94, 141]. Other authors proposed fully digital circuits capable to operate in alternating conditions between oscillations and metastability [136, 142–144].

6.3. Circuits to measure other stochastic physical processes

In this class of TRNGs, the source of randomness is obtained from the measurement of intrinsically random physical phenomena including radioactive decay, photon detection, and various sources of

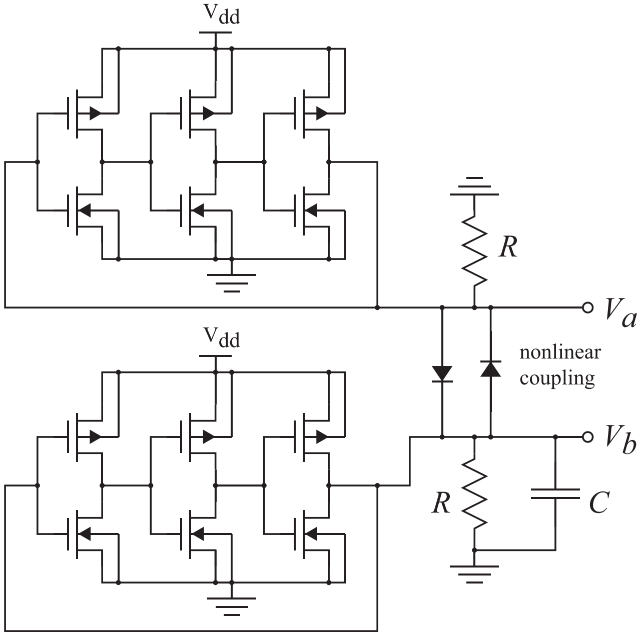

Figure 14. A nonlinear network coupling two ring oscillators have been proposed in [94] to implement a chaotic 'oscillator', to substitute the conventional fast oscillator in Figure 13.

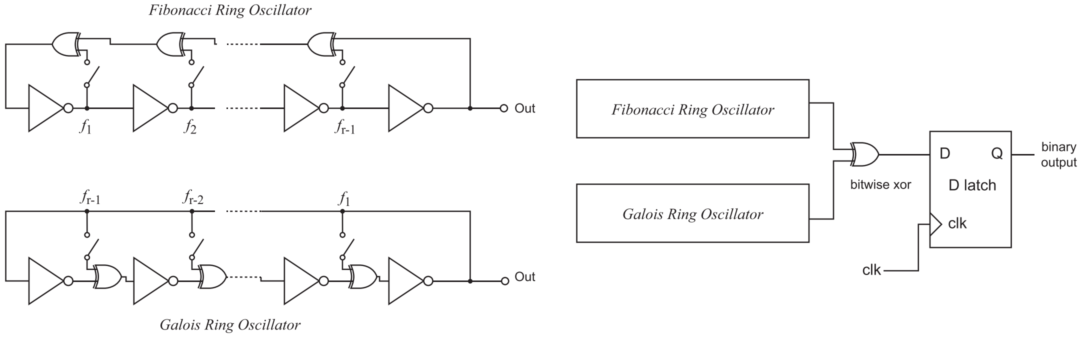

Figure 15. A TRNG exploiting Fibonacci and Galois ring oscillators [141].

electronic noise in semiconductor devices (e.g., thermal, diffusion, shot, avalanche, flicker, and generation/recombination noises) [95, 145, 145–151]. In the same class of TRNGs, we can also include other approaches proposed in literature, using antennas, sensors, and transducers to retrieve stochastic signals from different sources like, for example, lasers, noisy images taken with digital cameras, the Sun radiation, or the atmosphere dynamics [152–154].

Depending on the exploited physical phenomenon, the implementation of these TRNGs involves the design of custom solutions expressively devised to process the stochastic signal, from the source to the output, differing case by case. A generic scheme describing this kind of systems is shown in Figure 16.

Even if the exploited physical phenomenon offers ideal theoretical statistical properties for the task, like, for example, the Gaussian white thermal noise in resistors, hardware implementations of TRNGs result affected by statistical bias and memory mainly due to offset, gain and nonlinearity errors in the band-limited signal conditioning stages, and A/D conversion. Furthermore, in these TRNGs, the stochastic signal at the source can have equivalent amplitudes as lower as few tens of microvolts, and a special care has to be taken in the design to make the device robust with respect to circuit mismatches, electromagnetic couplings with the neighbor circuitry, unforeseen aging effects, temperature, and supply-voltage variations.

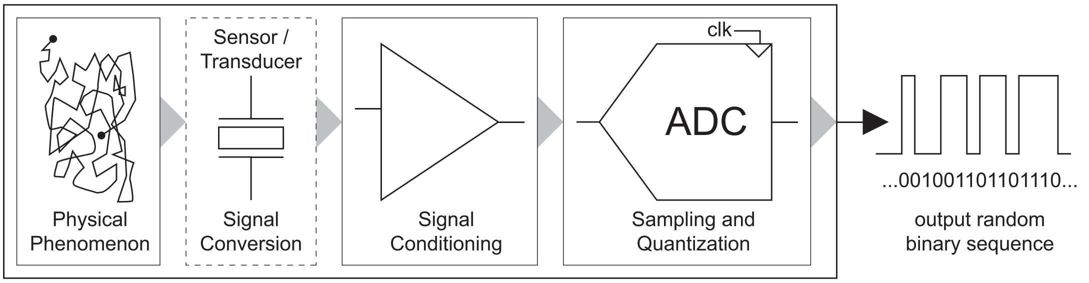

Figure 16. Schematic representation of the core structure of a TRNG exploiting the measurement of a stochastic physical process. When the stochastic source itself issues electric signals, as in the case of TRNGs based on electronic noise, the sensor/transducer is not necessary.

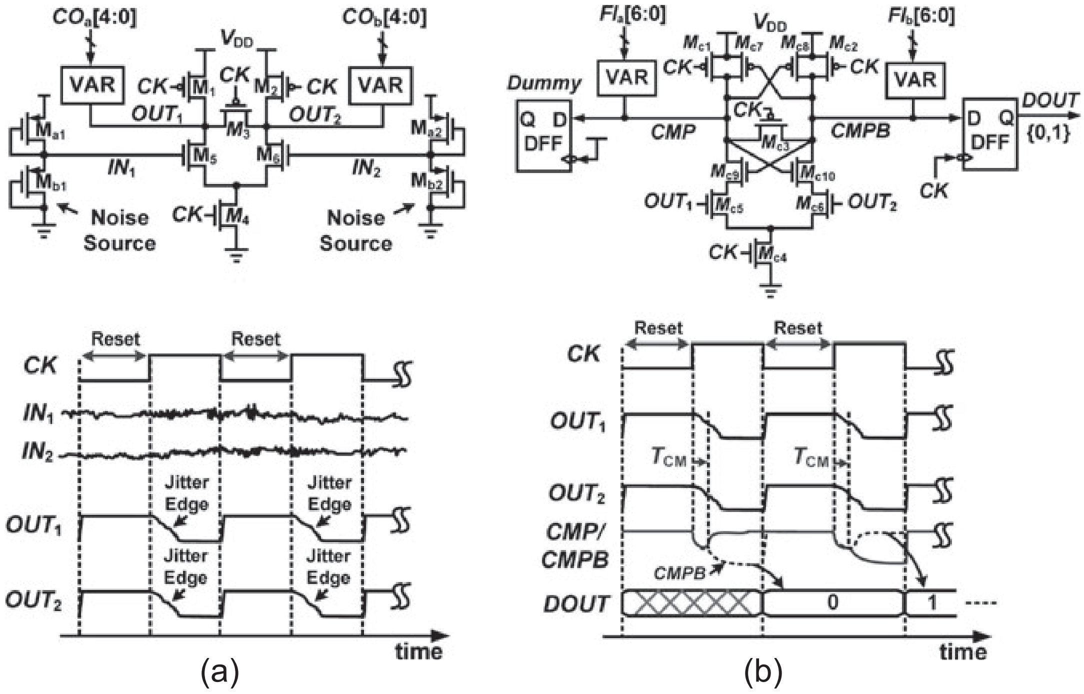

Figure 17. A TRNG exploiting electronic noise and metastability, generating one random bit DOUT each clock period (phase (a) and phase (b)) [145]. The VAR blocks are digitally controlled networks of varactors, to finely adjust the statistical biasing of the generated random sequences.

For the sake of an example, in Figure 17, the core subcircuits of a TRNG exploiting both electronic noise and metastability are shown [145]. Furthermore, Figure 18 shows the block diagram of a TRNG exploiting a mixture of the three sources of randomness mentioned in Section 6: electronic noise, chaos, and oscillators sampling [93].

7. POST-PROCESSING UNITS

As mentioned in Section 5.3, in cryptographic TRNGs, the last stage issuing the random sequence is a fully digital post-processor unit based on two different ideas, widely used in Cryptography and Information Theory: compression and diffusion/confusion [1–3, 155]. A scheme using both the approaches is shown in Figure 19.

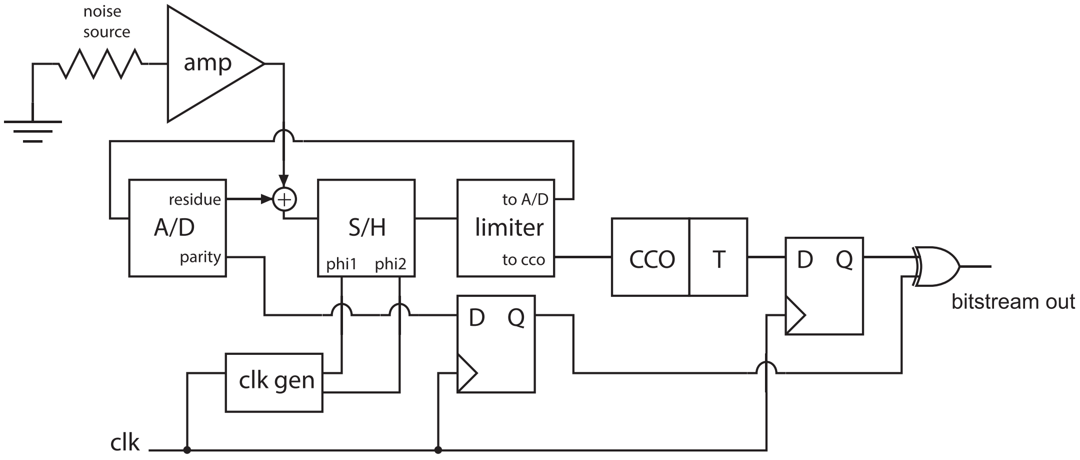

Figure 18. A TRNG exploiting a mixture of the three sources of randomness mentioned in Section 6: electronic noise, chaos, and oscillators sampling [93]. The A/D block, with analog residuals, is actually used to implement the Sawtooth chaotic map of Figure 11.

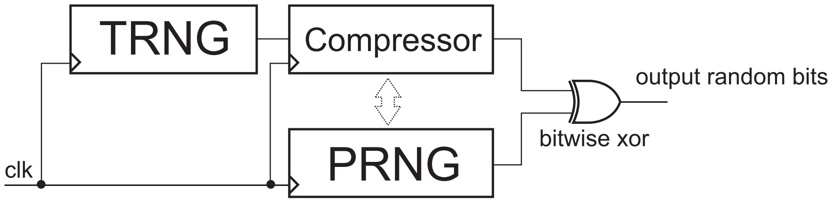

Figure 19. A possible scheme representing a cryptographic RNG. The kind of interaction between the true random sequence and the PRNG (dotted arrow) prior to the xoring may depend on the RNG design.

7.1. *Compressors*

The aim of compressors is to encode the information issued by the TRNG using fewer bits than the original representation, increasing the ASE. In literature, compression algorithms have been distinguished between lossy or lossless algorithms, depending whenever the information coding is reversible (in the lossless case) or not [155]. Efficient lossless compressors require large computation resources, and in TRNGs, lossy compressors are typically preferred, admitting a decrease of the throughput in return for a much less hardware complexity [156, 157].

The simplest lossy compressor proposed for a random source is the well-known Von–Neumann algorithm, capable of theoretically eliminating the statistical bias among the binary symbols $0, 1$ of a TRBG. The generalization of this method, proposed in [157], requires high-complexity implementations, whereas other approaches, based on pseudo-chaotic processors or hash functions, are devised to maintain a restrained hardware complexity [156, 158–160].

7.2. *Diffusion/confusion processors*

Even if an optimized compression algorithm can turn a poor TRNG in a cryptographically strong device, it is worth recalling that any given coding cannot protect against the hardware failure of the TRNG. Furthermore, residual statistical defects may still be present at the output of a suboptimal compressor.

The aim of diffusion/confusion processors is to mask the residual statistical defects properly scrambling and encrypting the generated sequences. The simplest approach in cryptographic applications is to perform a bit-by-bit XOR-operation of the compressed sequence with a sequence generated by a cryptographically strong pseudorandom bit generator, as shown in Figure19. The use of a cryptographically strong PRNG represents also a last resort against the hardware failure of the TRNG.

Copyright © 2016 John Wiley & Sons, Ltd.

DOI: 10.1002/cta

8. ASSESSMENT OF CRYPTOGRAPHIC TRNGS

From a theoretical point of view, the assessment of a cryptographic TRNG passes through the estimation of its ASE. Unfortunately, as previously mentioned, for most TRNGs, an adequate estimation of (2) is unfeasible, because it involves statistical distributions of any order. Moreover, it is not possible to evaluate a priori the effects of possible non-stationary statistical defects. To overcome this drawback, cryptographic TRNGs are evaluated by means procedures based on standard statistical hypothesis testing [76, 161], as discussed in the following.

Statistical testing of TRNGs is an intriguing topic that would require too much text in this paper for its detailed presentation, and we limit the discussion to a conceptual sketch. In TRNG assessment, the statistical hypothesis to be tested is the null hypothesis H_0: 'the generator under evaluation is unpredictable'. The task is accomplished focusing on one specific statistical feature of the sequences at a time (e.g., the frequency of occurrence of certain symbolic patterns) examining a finite set of finite sequences generated by the TRNG under inspection. The outcome of the test is probabilistic; that is, it expresses the probability that the collected sequences were actually generated by a TRNG. This probability is then compared with a given threshold to determine the acceptance/rejection of the statistical hypothesis H_0.

About the TRNG statistical testing, it is worth highlighting the following remarks:

- The number of possible statistical tests is infinite, as infinite are the different statistical features to be analyzed in a random sequence. This means that any statistical test suite cannot be deemed 'complete' to assess a TRNG;
- for any given setup of statistical test, it is possible to build a non-random device capable to obtain the acceptance of the null hypothesis H_0.
- as a result of the aforementioned remarks, performing well in statistical testing is a necessary condition for cryptographic TRNGs; nevertheless, it is not sufficient to assure their cryptographic security (i.e., their unpredictability).

Well-known statistical test suites for TRNGs are the Marsaglia's DIEHARD tests and the NIST SP88.22 standard [76, 161]. These tests are complex software routines to be executed by a processor and are not suitable for being implemented in digital hardware. A low-complexity set of statistical tests designed to be implemented in digital hardware is the FIPS 140.2 test suite [162]. These latter tests are only recommended to monitor possible critical hardware failure of the TRNG, because they are too simple to assess its cryptographic reliability.

9. CONCLUSION

We have provided an overview of selected crypto-hardware devices, with a special reference to the lightweight electronic implementation of encryption/decryption schemes, hash functions, and TRNGs. In detail, we have discussed about the hardware implementation of the chief algorithms used in private-key cryptography, PKC, and hash functions, discussing some important security issues in electronic crypto-devices related to SCAs, fault injection attacks, and the corresponding design countermeasures that can be taken. Finally, we have provided an overview about the hardware implementation of TRNGs, presenting the chief electronic sources of randomness and the types of post-processing techniques used to improve the statistical characteristics of the generated random sequences.

ACKNOWLEDGEMENTS

This work has been partially supported by CSIC under Projects MISAL 201450E034 and LACRE 201550E039, and the Spanish Government under Projects CESAR TEC2013-45523-R and INTERVALO TEC2016-80549-R.

REFERENCES

1. Vaudenay S. *A Classical Introduction to Cryptography*. Springer: New York, US, 2006.

2. Katz J, Lindell K. *Introduction to Modern Cryptography*. CRC Press: Boca Raton, 2015.
3. Stinson DR. *Cryptography: Theory and Practice*. CRC Press: Boca Raton, 2005.
4. Goldman Sachs. *The internet of things: Making sense of the next mega-trend*, 2014. Available: www.goldmansachs.com [Online] [Accessed on November 14, 2016].
5. Gartner Inc. *Gartner forecast: Internet of things – endpoints and associated services, worldwide, 2015*, 2015. Available: www.gartner.com [Online] [Accessed on November 14, 2016].
6. *Scopus, abstract and citation database of peer-reviewed literature. bibliometric tools track, analyze and visualize research. by Elsevier*. Available: www.scopus.com [Online] [Accessed on November 14, 2016].
7. Vernam GS, inventor. *Secret signaling system*. US Patent 1,310,719. Jul. 22 1919.
8. Hell M, Johansson T, Meier W. Grain: a stream cipher for constrained environments. *International Journal of Wireless and Mobile Computing* 2007; **2**(1):86–93.
9. De Cannière C. Trivium: A stream cipher construction inspired by block cipher design principles. In *International Conference on Information Security,* Samos Island, Greece, 2006; 171–186.
10. ECRYPT. *The eSTREAM project*. http://http:www.ecrypt.eu.org/stream/ [Accessed on November 10, 2016].
11. Marmolejo-Tejada J, Trujillo-Olaya V, Velasco-Medina J. Hardware implementation of grain-128, mickey-128, decim-128 and trivium. In *IEEE ANDESCON,* Bogota, Colombia, 2010; 1–6.
12. Mora-Gutiérrez J, Jiménez-Fernández CJ, Valencia-Barrero M. Low power implementation of trivium stream cipher. In *International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS12),* Newcastle upon Tyne, UK, 2012; 113–120.
13. FIPS PUB 46. *Data Encryption Standard*. NTIS: Springfield, VA, USA, 1977.
14. N. F. Pub. 197: Advanced encryption standard (aes). *Federal Information Processing Standards Publication* 2001; **197**:441–0311.
15. Taherkhani S, Ever E, Gemikonakli O. Implementation of non-pipelined and pipelined data encryption standard (des) using xilinx virtex-6 fpga technology. In *IEEE 10th International Conference on Computer and Information Technology (CIT10),* Bradford, UK, 2010; 1257–1262.
16. Verbauwhede I, Schaumont P, Kuo H. Design and performance testing of a 2.29-gb/s rijndael processor. *IEEE Journal of Solid-State Circuits* 2003; **38**(3):569–572.
17. Bogdanov A, Knudsen LR, Leander G, Paar C, Poschmann A, Robshaw MJ, Seurin Y, Vikkelsoe C. Present: An ultra-lightweight block cipher. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES07),* Vienna, Austria, 2007; 450–466.
18. Diffie W, Hellman M. New directions in cryptography. *IEEE Transactions on Information Theory* 1976; **22**(6): 644–654.
19. Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM* 1978; **21**(2):120–126.
20. Iana GV, Anghelescu P, Serban G. RSA encryption algorithm implemented on FPGA. In *International Conference on Applied Electronics*, Vol. 1, Pilsen, Czech Republic, 2011; 1–4.
21. Koblitz N. Elliptic curve cryptosystems. *Mathematics of computation* 1987; **48**(177):203–209.
22. Miller VS. Use of elliptic curves in cryptography. In *Proceedings on Advances in Cryptology (CRYPTO85),* Santa Barbara, USA, 1985; 417–426.
23. Park J, Hwang J-T, Kim Y-C. FPGA and ASIC implementation of ECC processor for security on medical embedded system. *Proceedings - 3rd International Conference on Information Technology and Applications, ICITA 2005*, Vol. II, 2005; 547–551.
24. Guitouni Z, Chotin-Avot R, Machhout M, Mehrez H, Tourki R. High performances ASIC based elliptic curve cryptographic processor over GF(2m). *IJCA Special Issue on Network Security and Cryptography* 2011; **NSC**(4): 1–10.
25. Rivest RL et al. Rfc 1321: The md5 message-digest algorithm. *Internet Activities Board* 1992; 143. In https://www.ietf.org/rfc/rfc1321.txt. [Accessed on November 10, 2016].
26. Jarvinen K, Tommiska M, Skytta J. Hardware implementation analysis of the md5 hash algorithm. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences,* Big Island, USA, 2005; 298a.
27. FIPS 180-4. *Secure Hash Standard (SHS)*. National Institute of Standards and Technology: Gaithersburg, MD, 2015.
28. Ting KK, Yuen SC, Lee K-H, Leong PH. An fpga based sha-256 processor. *International Conference on Field-Programmable Logic and Applications: Reconfigurable Computing Is Going Mainstream,* Montpellier, France, 2002; 577–585.
29. McLoone M, McCanny JV. Efficient single-chip implementation of sha-384 and sha-512. In *Proceedings of the IEEE International Conference on Field-Programmable Technology (FPT02),* Hong Kong, China, 2002; 311–314.
30. Bertoni G, Daemen J, Peeters M, Van Assche G. *Keccak*. http://keccak.noekeon.org/ [Accessed on November 14, 2016].
31. Kavun EB, Yalcin T. A lightweight implementation of keccak hash function for radio-frequency identification applications. In *International Workshop on Radio Frequency Identification: Security and Privacy Issues*, Istanbul, Turkey, 2010; 258–269.
32. San I, At N. Compact keccak hardware architecture for data integrity and authentication on FPGAs. *Information Security Journal: A Global Perspective* 2012; **21**(5):231–242.
33. Mangard S, Oswald E, Popp T. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*, Vol. 31. Springer: New York, US, 2008.

34. Kocher PC. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *Advances in Cryptology (CRYPTO'96),* Santa Barbara, USA, 1996; 104–113.
35. Dhem J-F, Koeune F, Leroux P-A, Mestré P, Quisquater J-J, Willems J-L. A practical implementation of the timing attack. In *Smart Card Research and Applications*. Springer-Verlag: Berlin, 1998; 167–182.
36. Kocher P, Jaffe J, Jun B. Differential power analysis. *Advances in Cryptology (CRYPTO'99),* Santa Barbara, USA, 1999; 388–397.
37. Moradi A, Mischke O, Paar C. Practical evaluation of dpa countermeasures on reconfigurable hardware. In *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST11),* San Diego, USA, 2011; 154–160.
38. Lu Y, Boey K, O'Neill M, McCanny J. Practical comparison of differential power analysis techniques on an asic implementation of the aes algorithm. In *IET Irish Signals and Systems Conference (ISSC09),* IET, Dublin, Ireland, 2009; 57.
39. Reparaz O, Gierlichs B, Verbauwhede I. Generic dpa attacks: curse or blessing? *In Constructive Side-Channel Analysis and Secure Design (COSADE14),* Paris, France, 2014; 98–111.
40. Gandolfi K, Mourtel C, Olivier F. Electromagnetic analysis: Concrete results. In *Cryptographic Hardware and Embedded Systems (CHES'01),* Paris, France, 2001; 251–261.
41. Hayashi Y, Homma N, Mizuki T, Aoki T, Sone H, Sauvage L, Danger J-L. Analysis of electromagnetic information leakage from cryptographic devices with different physical structures. *IEEE Transactions on Electromagnetic Compatibility* 2013; **55**(3):571–580.
42. de Mulder E, Ors SB, Preneel B, Verbauwhede I. Differential electromagnetic attack on an FPGA implementation of elliptic curve cryptosystems. In *2006 World Automation Congress,* Budapest, Hungary, 2006; 1–6.
43. Yoshikawa M, Nozaki Y, Asahi K. Electromagnetic analysis attack for a lightweight block cipher twine. In *IEEE/ACES International Conference on Wireless Information Technology and Systems (ICWITS16) and Applied Computational Electromagnetics (ACES),* Honolulu, USA, 2016; 1–2.
44. Shamir A, Tromer E. *Acoustic cryptanalysis*, 2004. Presentation available from: http://www.wisdom.weizmann.ac.il/~tromer/acoustic/. [Accessed on November 14, 2016].
45. Mangard S, Pramstaller N, Oswald E. Successfully attacking masked aes hardware implementations. *In Cryptographic Hardware and Embedded Systems (CHES05),* Edinburgh, UK, 2005; 157–171.
46. Lano J, Mentens N, Preneel B, Verbauwhede I. Power analysis of synchronous stream ciphers with resynchronization mechanism. *In ECRYPT Workshop, SASC–The State of the Art of Stream Ciphers,* Brugge, Belgium, 2004; 327–333.
47. Fischer W, Gammel BM, Kniffler O, Velten J. Differential power analysis of stream ciphers. In *Topics in Cryptology–CT-RSA 2007,* San Francisco, USA, 2007; 257–270.
48. Boneh D, DeMillo R, Lipton R. On the importance of checking cryptographic protocols for faults. In *Advances in Cryptology – (EUROCRYPT97),* Konstanz, Germany, 1997; 37–51.
49. Joye M, Tunstall M. *Fault Analysis in Cryptography*, Vol. 7. Springer-Verlag: Berlin, 2012.
50. Bar-El H, Choukri H, Naccache D, Tunstall M, Whelan C. The sorcerer's apprentice guide to fault attacks. *Proceedings of the IEEE* 2006; **94**(2):370–382.
51. Karaklajic D, Schmidt J-M, Verbauwhede I. Hardware designer's guide to fault attacks. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 2013; **21**(12):2295–2306.
52. Kömmerling O, Kuhn MG. Design principles for tamper-resistant smartcard processors. *Smartcard* 1999; **99**:9–20.
53. Hojsik M, Rudolf B. Differential fault analysis of trivium. In *International Workshop on Fast Software Encryption,* Lausanne, Switzerland, 2008; 158–172.
54. Potestad-Ordóñez FE, Jiménez-Fernández CJ, Valencia-Barrero M. Fault attack on FPGA implementations of trivium stream cipher. In *IEEE International Symposium on Circuits and Sistems (ISCAS16),* Montreal, Canada, 2016; 562–565.
55. Fukunaga T, Takahashi J. Practical fault attack on a cryptographic LSI with ISO/IEC 18033-3 block ciphers. In *Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC09),* Lausanne, Switzerland, 2009; 84–92.
56. Tena-Sanchez E, Castro J, Acosta AJ. A methodology for optimized design of secure differential logic gates for DPA resistant circuits. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 2014; **4**(2):203–215.
57. Marzouqi H, Al-Qutayri M, Salah K. Review of gate-level differential power analysis and fault analysis countermeasures. *Information Security, IET* 2014; **8**(1):51–66.
58. Popp T, Mangard S, Oswald E. Power analysis attacks and countermeasures. *Design & Test of Computers, IEEE* 2007; **24**(6):535–543.
59. Goubin L, Patarin J. Des and differential power analysis the duplication method. In *Cryptographic Hardware and Embedded Systems (CHES99),* Worcester, USA, 1999; 158–172.
60. Akkar ML, Giraud C. An implementation of des and aes, secure against some attacks. *In Cryptographic Hardware and Embedded Systems (CHES'01),* Paris, France, 2001; 309–318.
61. Ishai Y, Sahai A, Wagner D. Private circuits: Securing hardware against probing attacks. In *Advances in Cryptology (CRYPTO'03),* Santa Barbara, US, 2003; 463–481.
62. Reparaz O, Bilgin B, Nikova S, Gierlichs B, Verbauwhede I. Consolidating masking schemes. *In Advances in Cryptology (CRYPTO'15),* Santa Barbara, USA, 2015; 764–783.
63. Tiri K, Akmal M, Verbauwhede I. A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards. In *Proceedings of the 28th European Solid-State Circuits Conference (ESSCIRC02),* Florence, Italy, 2002; 403–406.

64. Guilley S, Sauvage L, Flament F, Vong V-N, Hoogvorst P, Pacalet R. Evaluation of power constant dual-rail logics countermeasures against DPA with design time security metrics. *IEEE Transactions on Computers* 2010; **59**(9): 1250–1263.
65. Tiri K, Verbauwhede I. A logic level design methodology for a secure DPA resistant ASIC or FPGA implementation. In *Proceedings of the Design, Automation and Test in Europe-Conference, (DATE'04),*Volume 1, Paris, France, 2004; 246–251.
66. Popp T, Mangard S. Implementation aspects of the DPA-resistant logic style MDPL. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'06),* Kos, Greece, 2006; 2913–2916.
67. Allam MW, Elmasry MI. Dynamic current mode logic (dycml): a new low-power high-performance logic style. *IEEE Journal of Solid-State Circuits* 2001; **36**(3):550–558.
68. Hassoune I, Macé F, Flandre D, Legat J-D. Low-swing current mode logic (LSCML): a new logic style for secure and robust smart cards against power analysis attacks. *Microelectronics Journal* 2006; **37**(9):997–1006.
69. Bucci M, Giancane L, Luzzi R, Scotti G, Trifiletti A. Delay-based dual-rail precharge logic. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 2011; **19**(7):1147–1153.
70. Levi I, Keren O, Fish A. Data-dependent delays as a barrier against power attacks. *IEEE Transactions on Circuits and Systems I: Regular Papers* 2015; **62**(8):2069–2078.
71. Pongaliur K, Abraham Z, Liu AX, Xiao L, Kempel L. Securing sensor nodes against side channel attacks. In *11th IEEE High Assurance Systems Engineering Symposium (HASE08),* Nanjing, China, 2008; 353–361.
72. Gornik A, Moradi A, Oehm J, Paar C. A hardware-based countermeasure to reduce side-channel leakage: design, implementation, and evaluation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 2015; **34**(8):1308–1319.
73. Maistri P. Countermeasures against fault attacks: The good, the bad, and the ugly. In *IEEE 17th International On-Line Testing Symposium (IOLTS11),* Athens (Greece), 2011; 134–137.
74. Yen C-H, Wu B-F. Simple error detection methods for hardware implementation of advanced encryption standard. *IEEE Transactions on Computers* 2006; **55**(6):720–731.
75. Ma K, Liang H, Wu K. Homomorphic property-based concurrent error detection of rsa: a countermeasure to fault attack. *IEEE Transactions on Computers* 2012; **61**(7):1040–1049.
76. *NIST Special Publication 800-22 Rev.1a: A statistical test suite for random and pseudorandom number generators for cryptographic applications*, Apr. 2010.
77. Gentle J. *Random Numbers Generation and Monte Carlo Methods*, 2nd. Springer-Verlag: New York, 2003.
78. Addabbo T, Alioto M, Bernardi S, Fort A, Rocchi S, Vignoli V. Hardware-efficient PRBGs based on 1-D piecewise linear chaotic maps. In *11th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2004,* Tel-Aviv, Israel, 2004; 242–245.
79. Addabbo T, Alioto M, Bernardi S, Fort A, Rocchi S, Vignoli V. The digital Tent map: Performance analysis and optimized design as a source of pseudo-random bits. In *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, Vol. 2, Como, Italy, 2004; 1301–1304.
80. Addabbo T, Alioto M, Fort A, Rocchi S, Vignoli V. Long period pseudo random bit generators derived from a discretized chaotic map. In *Proceedings - IEEE International Symposium on Circuits and Systems,* Kobe, Japan, 2005; 892–895.
81. L'Ecuyer P. Random number generation with multiple streams for sequential and parallel computing. In *Proceedings - Winter Simulation Conference*, Vol. 2016, Huntington Beach, CA, USA, 2016; 31–44.
82. Addabbo T, Alioto M, Fort A, Rocchi S, Vignoli V. The digital Tent map: performance analysis and optimized design as a low-complexity source of pseudorandom bits. *IEEE Transactions on Instrumentation and Measurement* 2006; **55**(5):1451–1458.
83. Low-hardware complexity PRBGS based on a piecewise-linear chaotic map. *IEEE Transactions on Circuits and Systems II: Express Briefs* 2006; **53**(5):329–333.
84. Addabbo T, Alioto M, Fort A, Mugnaini M, Rocchi S, Vignoli V. Implementation-efficient maximum-period nonlinear congruential generators. In *Conference Record - IEEE Instrumentation and Measurement Technology Conference,* Warsaw, Poland, 2007.
85. Addabbo T, Alioto M, Fort A, Rocchi S, Vignoli V. Maximum-period PRNGs derived from a piecewise linear one-dimensional map. In *Proceedings - IEEE International Symposium on Circuits and Systems,* New Orleans, USA, 2007; 693–696.
86. Addabbo T, Alioto M, Fort A, Pasini A, Rocchi S, Vignoli V. A class of maximum-period nonlinear congruential generators derived from the Rényi chaotic map. *IEEE Transactions on Circuits and Systems I: Regular Papers* 2007; **54**(4):816–828.
87. Addabbo T, Fort A, Mugnaini M, Rocchi S, Vignoli V. On the efficient digital implementation of nonlinear congruential generators derived from the Rényi chaotic map. In *Conference Record - IEEE Instrumentation and Measurement Technology Conference,* Victoria, Canada, 2008; 1707–1711.
88. Addabbo T, Fort A, Rocchi S, Vignoli V. On the generation of pseudo-random sequences exploiting digitized chaotic systems. In *European Conference on Circuit Theory and Design 2007, ECCTD 2007,* Seville, Spain, 2008; 639–642.
89. Digitized chaos for pseudo-random number generation in cryptography. *Studies in Computational Intelligence* 2011; **354**:67–97.
90. Addabbo T, de Caro D, Fort A, Petra N, Rocchi S, Vignoli V. Efficient implementation of pseudochaotic piecewise linear maps with high digitization accuracies. *International Journal of Circuit Theory and Applications* 2012; **40**(1):1–14.

DOI: 10.1002/cta

91. Knuth D. *The Art of Computer Programming*, 2nd, Vol. 2. Addison-Wesley: USA, 1981.
92. Gray RM. *Entropy and Information Theory*. Springer: New York, 2011.
93. Petrie CS, Connelly JA. A noise-based IC random number generator for applications in cryptography. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 2000; **47**(5):615–621.
94. Çiçek I. A chaos based integrated jitter booster circuit for true random number generators. *2013 European Conference on Circuit Theory and Design (ECCTD)*, 2013; 1–4.
95. Yamazaki T, Uchida A. Performance of random number generators using noise-based superluminescent diode and chaos-based semiconductor lasers. *IEEE Journal of Selected Topics in Quantum Electronics* 2013; **19**(4):0600309.
96. Delgado-Restituto M, Rodriguez-Vazquez A. Integrated chaos generators. *Proceedings of the IEEE* 2002; **90**(5):747–767.
97. Addabbo T, Alioto M, Fort A, Rocchi S, Vignoli V. A feedback strategy to improve the entropy of a chaos-based random bit generator. *IEEE Transactions on Circuits and Systems I: Regular Papers* 2006; **53**(2):326–337.
98. Hirsch S, Smale MW, Devaney R. *Differential Equations, Dynamical Systems, and an Introduction to Chaos*, 3rd. Academic Press: San Diego, 2013.
99. Walters P. *An Introduction to Ergodic Theory*. Springer-Verlag: New York, 2000.
100. Lasota A, Mackey MC. *Chaos, Fractals and Noise - Stochastic Aspects of Dynamics*, 2nd. Springer: New York, 1994.
101. Boyarsky A, Góra P. *Laws of Chaos*. Birkhäuser: Boston, 1997.
102. Chen D, Sun Z, Ma X, Chen L. Circuit implementation and model of a new multi-scroll chaotic system. *International Journal of Circuit Theory and Applications* 2014; **42**(4):407–424.
103. Addabbo T, Fort A, Rocchi S, Vignoli V. Chaos based generation of true random bits. *Studies in Computational Intelligence* 2009; **184**:355–377.
104. Rodriguez-Vazquez A, Huertas J, Chua L. Chaos in switched-capacitor circuit. *IEEE Transactions on Circuits and Systems* 1985; **32**(10):1083–1085.
105. Rodriguez-Vazquez A, Rueda A, Perez-Verdu B, Huertas JL. Chaos via a piecewise-linear switched-capacitor circuit. *Electronics Letters* 1987; **23**(12):662–663.
106. Rodriguez-Vazquez A, Huertas JL, Rueda A, Perez-Verdu B, Chua LO. Chaos from switched-capacitor circuits: discrete maps. *Proceedings of the IEEE* 1987; **75**(8):1090–1106.
107. Rodriguez-Vazquez A, Delgado M, Espejo S, Huertas JL. Switched-capacitor broadband noise generator for cmos vlsi. *Electronics Letters* 1991; **27**(21):1913–1915.
108. Delgado-Restituto M, Rodriguez-Vasquez A, Espejo S, Huertas JL. A chaotic switched-capacitor circuit for 1/f noise generation. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 1992; **39**(4): 325–328.
109. Rodríguez-Vázquez A, Domínguez-Castro R, Medeiro F, Delgado-Restituto M. High resolution cmos current comparators: design and applications to current-mode function generation. *Analog Integrated Circuits and Signal Processing* 1995; **7**(2):149–165.
110. Delgado-Restituto M, Rodriguez-Vazquez A. Mixed-signal map-configurable integrated chaos generator for chaotic communications. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 2001; **48**(12):1462–1474.
111. Çiçek I, Pusane A, Dündar G. An Integrated Dual Entropy Core True Random Number Generator. *IEEE Transactions on Circuits and Systems II: Express Briefs* 2016. DOI: 10.1109/TCSII.2016.2568181.
112. Ergün S, Özoguz S. Truly random number generators based on non-autonomous continuous-time chaos. *International Journal of Circuit Theory and Applications* 2010; **38**(1):1–24.
113. Addabbo T, Fort A, Rocchi S, Vignoli V. Histogram test of ADCs with chaotic samples. In *2010 IEEE International Instrumentation and Measurement Technology Conference, I2MTC 2010 - Proceedings,* Austin, TX, USA, 2010; 546–549.
114. Katz O, Ramon DA, Wagner IA. A robust random number generator based on a differential current-mode chaos. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 2008; **16**(12):1677–1686.
115. Addabbo T, Fort A, Papini D, Rocchi S, Vignoli V. An efficient and accurate method for the estimation of entropy and other dynamical invariants for piecewise affine chaotic maps. *International Journal of Bifurcation and Chaos* 2009; **19**(12):4175–4195.
116. Invariant measures of tunable chaotic sources: robustness analysis and efficient estimation. *IEEE Transactions on Circuits and Systems I: Regular Papers* 2009; **56**(4):806–819.
117. Addabbo T, Fort A, Mugnaini M, Rocchi S, Vignoli V. Statistical characterization of a chaotic piecewise linear map for uniform-distributed analog noise generation. In *Proceedings of the 16th IMEKO TC4 Int. Symp.: Exploring New Frontiers of Instrum. and Methods for Electrical and Electronic Measurements; 13th TC21 Int. Workshop on ADC Modelling and Testing - Joint Session,* Florence, Italy, 2008; 688–693.
118. Wang C-C, Huang J-M, Cheng H-C, Hu R. Switched-current 3-bit CMOS 4.0-MHz wideband random signal generator. *IEEE Journal of Solid-State Circuits* 2005; **40**(6):1360–1365.
119. Addabbo T, Fort A, Rocchi S, Vignoli V. An efficient and accurate method for computing the invariant measure of piecewise affine chaotic maps. In *Proceedings - IEEE International Symposium on Circuits and Systems,* Seattle, WA, USA, 2008; 760–763.
120. Callegari S, Rovatti R, Setti G. First direct implementation of a true random source on programmable hardware. *International Journal of Circuit Theory and Applications* 2005; **33**(1):1–16.

121. Addabbo T, Alioto M, Fort A, Rocchi S, Vignoli V. Uniform-distributed noise generator based on a chaotic circuit. *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, 2006; 1156–1160.
122. Addabbo T, Fort A, Rocchi S, Vignoli V. Exploiting chaotic dynamics for A-D converter testing. *International Journal of Bifurcation and Chaos* 2010; **20**(4):1099–1118.
123. Addabbo T, Alioto M, Fort A, Rocchi S, Vignoli V. A technique to design high entropy chaos-based true random bit generators. *Proceedings - IEEE International Symposium on Circuits and Systems*, 2006; 1183–1186.
124. Addabbo T, Alioto M, Fort A, Rocchi S, Vignoli V. A scalable low-entropy detector to counteract the parameter variability effects in TRBGs. In *I2MTC 2010 - Proceedings of the 2010 IEEE International Instrumentation and Measurement Technology Conference,* Austin , TX, USA, 2010; 605–609.
125. Addabbo T, Alioto M, Fort A, Rocchi S, Vignoli V. A variability-tolerant feedback technique for throughput maximization of TRBGs with predefined entropy. *Journal of Circuits, Systems and Computers* 2010; **19**(4): 879–895.
126. Degaldo-Restituto M, Medeiro F, Rodriguez-Vazquez A. Nonlinear switched-current cmos ic for random signal generation. *Electronics Letters* 1993; **29**(25):2190–2191.
127. Sui C, Bai S, Zhu T, Cheng C, Beetner D. New methods to characterize deterministic jitter and crosstalk-induced jitter from measurements. *IEEE Transactions on Electromagnetic Compatibility* 2015; **57**(4):877–884.
128. Xu L, Duan Y, Chen D. A low cost jitter separation and characterization method. In *Proceedings of the IEEE VLSI Test Symposium,* Napa, CA, 2015; 1–5.
129. Li MP. *Jitter, Noise, and Signal Integrity at High-speed*. Pearson Education, Inc: Boston, 2008.
130. Robson S, Leung B, Gong G. Truly random number generator based on a ring oscillator utilizing last passage time. *IEEE Transactions on Circuits and Systems II: Express Briefs* 2014; **61**(12):937–941.
131. Yang K, Fick D, Henry MB, Lee Y, Blaauw D, Sylvester D. 16.3 A 23Mb/s 23pJ/b fully synthesized true-random-number generator in 28nm and 65nm CMOS. In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC),* San Francisco, CA, 2014; 280–281.
132. Lubicz D, Bochard N. Towards an oscillator based TRNG with a certified entropy rate. *IEEE Transactions on Computers* 2015; **64**(4):1191–1200.
133. Güler U, Dündar GG. Modeling CMOS ring oscillator performance as a randomness source. *IEEE Transactions on Circuits and Systems I: Regular Papers* 2014; **61**(3):712–724.
134. Bayon P, Bossuet L, Aubert A, Fischer V, Poucheret F, Robisson B, Maurine P. Contactless Electromagnetic Active Attack on Ring Oscillator Based True Random Number Generator. In *Constructive Side-Channel Analysis and Secure Design*, Schindler W, Huss SA (eds). Springer: Berlin Heidelberg, 2012; 151–166.
135. Wold K, Tan C. Analysis and enhancement of random number generator in FPGA based on oscillator rings. In *Proceedings - 2008 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2008,* Cancun, Mexico, 2008; 385–390.
136. Epstein M, Hars L, Krasinski R, Rosner M, Zheng H. Design and Implementation of a True Random Number Generator Based on Digital Circuit Artifacts. In *Cryptographic Hardware and Embedded Systems - CHES 2003*, Walter CD, Koç ÇK, Paar C (eds). Springer: Berlin Heidelberg, 2003; 152–165.
137. Yang K, Blaauw D, Sylvester D. An all-digital edge racing true random number generator robust against pvt variations. *IEEE Journal of Solid-State Circuits* 2016; **51**(4):1022–1031.
138. Haddad P, Teglia Y, Bernard F, Fischer V. On the assumption of mutual independence of jitter realizations in P-TRNG stochastic models. In *2014 Design, Automation Test in Europe Conference Exhibition (DATE),* Dresden, Germany, 2014; 1–6.
139. Martín H, Korak T, Millán ES, Hutter M. Fault attacks on STRNGs: impact of glitches, temperature, and underpowering on randomness. *IEEE Transactions on Information Forensics and Security* 2015; **10**(2):266–277.
140. Baudet M, Lubicz D, Micolod J, Tassiaux A. On the security of oscillator-based random number generators. *Journal of Cryptology* 2011; **24**(2):398–425.
141. Golíc JD. New methods for digital generation and postprocessing of random data. *IEEE Transactions on Computers* 2006; **55**(10):1217–1229.
142. Bucci M, Luzzi R. Fully digital random bit generators for cryptographic applications. *IEEE Transactions on Circuits and Systems I: Regular Papers* 2008; **55**(3):861–875.
143. Suresh V, Burleson W. Entropy and energy bounds for metastability based TRNG with lightweight post-processing. *IEEE Transactions on Circuits and Systems I: Regular Papers* 2015; **62**(7):1785–1793.
144. Wieczorek PZ. An FPGA implementation of the resolve time-based true random number generator with quality control. *IEEE Transactions on Circuits and Systems I: Regular Papers* 2014; **61**(12):3450–3459.
145. Kuan TK, Chiang YH, Liu SI. A 0.43pJ/bit true random number generator. In *2014 IEEE Asian Solid-State Circuits Conference (A-SSCC),* KaoHsiung, 2014; 33–36.
146. Yasuda S, Satake H, Tanamoto T, Ohba R, Uchida K, Fujita S. Physical random number generator based on MOS structure after soft breakdown. *IEEE Journal of Solid-State Circuits* 2004; **39**(8):1375–1377.
147. Holleman J, Bridges S, Otis BP, Diorio C. A 3 uW CMOS true random number generator with adaptive floating-gate offset cancellation. *IEEE Journal of Solid-State Circuits* 2008; **43**(5):1324–1336.
148. Perić M, Milićević P, Banjac Z, Orlić V, Milićević S. High speed random number generator for section key generation in encryption devices. In *2013 21st Telecommunications Forum (TELFOR),* Belgrade, 2013; 117–120.
149. De Roover C, Steyaert M. A 500 mV 650 pW random number generator in 130 nm CMOS for a UWB localization system. In *2010 Proceedings of the ESSCIRC,* Seville, 2010; 278–281.
150. Zhun H, Hongyi C. A truly random number generator based on thermal noise. In *Proceedings of the 4th International Conference on ASIC,* Shanghai, 2001; 862–864.

151. Khanmohammadi A, Enne R, Hofbauer M, Zimmermanna H. A monolithic silicon quantum random number generator based on measurement of photon detection time. *IEEE Photonics Journal* 2015; **7**(5):1–13.
152. Li R. A true random number generator algorithm from digital camera image noise for varying lighting conditions. In *SoutheastCon 2015,* Fort Lauderdale, FL, 2015; 1–8.
153. Tanyer SG, Atalay KD, Inam SÇ. Goodness-of-fit and randomness tests for the sun's emissions true random number generator. In *2014 International Conference on Mathematics and Computers in Sciences and in Industry (MCSI),* Varna, 2014; 216–218.
154. Hennebert C, Hossayni H, Lauradoux C. Entropy harvesting from physical sensors. In *WiSec 2013 - Proceedings of the 6th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, Budapest, Hungary, 2013; 149–154.
155. Salomon D. *A Concise Introduction to Data Compression*. Springer: London, 2008.
156. Addabbo T, Fort A, Kocarev L, Rocchi S, Vignoli V. Pseudo-chaotic lossy compressors for true random number generation. *IEEE Transactions on Circuits and Systems I: Regular Papers* 2011; **58**(8):1897–1909.
157. Juels A, Jakobsson M, Shriver E, Hillyer BK. How to turn loaded dice into fair coins. *IEEE Transactions on Information Theory* 2000; **46**(3):911–921.
158. Addabbo T, Alioto M, Fort A, Rocchi S, Vignoli V. Efficient post-processing module for a chaos-based random bit generator. In *Proceedings of the IEEE International Conference on Electronics, Circuits, and Systems*, Nice, 2006; 1224–1227.
159. Sunar B, Martin WJ, Stinson DR. A provably secure true random number generator with built-in tolerance to active attacks. *IEEE Transactions on Computers* 2007; **56**(1):109–119.
160. Addabbo T, Fort A, Kocarev L, Rocchi S, Vignoli V. Pseudo-chaotic lossy compression of TRBGs. In *2011 IEEE International Symposium of Circuits and Systems (ISCAS),* Rio de Janeiro, 2011; 1980–1983.
161. Marsaglia G. *The marsaglia random number cdrom including the diehard battery of tests of randomness*, 1995. Available: http://stat.fsu.edu/pub/diehard/ [Online] [Accessed on November 14, 2016].
162. NIST. *Fips 140.2 - security requirements for cryptographic modules. Effective 15 Nov 2001*, 2001. Available: csrc.nist.gov [Online] [Accessed on November 14, 2016].

DOI: 10.1002/cta