IEEE TRANSACTIONS ON
EMERGING TOPICS
IN COMPUTING

Received 24 July 2014; revised 27 November 2014; accepted 16 December 2014.
Date of publication 23 December 2014; date of current version 26 February 2016.

Digital Object Identifier 10.1109/TETC.2014.2386137

Theoretical Limits of Helperless Stabilizers for Physically Unclonable Constants

RICCARDO BERNARDINI AND ROBERTO RINALDO

Department of Electrical, Management and Mechanical Engineering, University of Udine, Udine 33100, Italy
CORRESPONDING AUTHOR: R. BERNARDINI (riccardo.bernardini@uniud.it)

ABSTRACT Physically unclonable constants (PUCs) have recently been proposed for private ID generation. Because in most of the proposed PUC schemes the generated value is not stable (i.e., it can change at different turn-ons), a postprocessing with a stabilizer is usually required. Most of the proposed stabilizer schemes use auxiliary data (helper data) to overcome the inherent randomness of the generation process. However, this complicates the structure of the scheme and poses additional security problems (e.g., helper data can be vectors for attacks), so that there is some interest in stabilizers that do not use helpers (helperless stabilizers). In this paper, we begin the study of the theoretical limits of helperless stabilizers. We show three main results: 1) perfect stability is unachievable; 2) we can make as small as desired the probability that a PUC has low stability; and 3) we can reliably recognize the bad devices at production time and discard them. The proofs of the latter two results are constructive.

INDEX TERMS Security, physically unclonable functions, chip authentication, authentication protocol.

I. INTRODUCTION

The increasing requirements for security motivated a good amount of research in the last years. A problem that is currently analyzed is how to store a secret in a chip so that even an attacker who is able to physically open the chip and study it, cannot read the secret. This problem gave rise to the introduction of Physically Unclonable Functions (PUFs) [1]–[7].

A Physically Unclonable Function (PUF) is a function that (typically) maps binary words to binary words and whose behavior depends on the uncontrollable fine details of the integrated circuit (e.g., the dopant concentration in a MOSFET). This sensitivity should make the PUF practically impossible to reproduce, even for the original chip maker. In a sense, a PUF is like a fingerprint. As each person has a unique fingerprint, every chip has its own PUF; as the fingerprint minutiae are the result of casual variation during the fetal development, the PUF is the result of casual variation during chip production. As fingerprints, the ideal PUF is at the same time random and deterministic: random because it should be impossible to predict the PUF of a given chip, deterministic because a specific PUF should always give the same result when queried with a given input. In other words, an ideal PUF is a random oracle [8]–[10].

In the literature, PUFs are typically partitioned into two classes, *strong* and *weak* PUFs. *Strong* PUFs [2], [3] have a very large domain size (exponential with the required silicon area) and they are typically proposed for Challenge Response Pair (CRP) based chip authentication. Attacks to strong PUFs aim typically to obtain a physical model of the device to be simulated in software [3], [11], [12]. It is also worth citing [13] that proposes a different definition that add more constraints, such as the un-modelability for strong PUFs.

Weak PUFs [4], [5], [14]–[19] have a limited domain size that, in the extreme case, can reduce to the empty set, making the PUF a constant function. We propose for the latter case (the only one considered in this paper) the term Physically Unclonable Constant (PUC). The typical usage of a PUC is to provide the chip with a unique secret ID that can be used, for example, to generate private cryptographic keys. If the ideal PUF is a random oracle, the ideal PUC is a *random constant*: at construction time a random value is selected and the same value is returned every time the PUC is queried.

A major problem that both strong and weak PUFs must solve is the fact that most of the proposed PUFs are not strictly deterministic and their output to a given query can change. When strong PUFs are used for authentication [7], [11] the

problem is sometimes solved by accepting the response even if the match is not perfect, but this also helps the attackers, who will need to reproduce only a good approximation of the PUF. For the sake of completeness, it should be pointed out that the ''approximate match'' approach is not the only one; see, for example, the protocol list in [20].

PUCs used for private key creation are even more delicate. A single wrong bit in the key can make the whole system useless. In order to make the PUC output more stable, two-step schemes employing *helpers* based on error correcting codes are usually proposed.

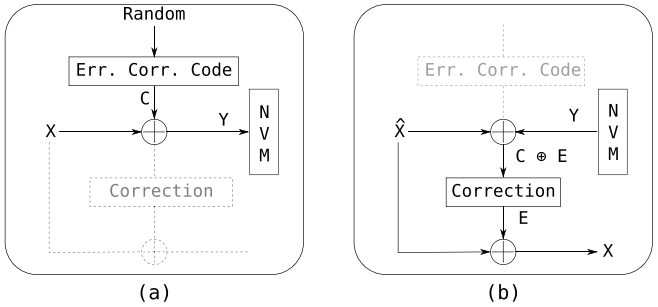

FIGURE 1. Example of helper-based PUF. (a) Enrollment, (b) Recovering X.

Example 1: Fig. 1 shows an example of stabilizer based on the *code offset secure sketch* approach [21], [22]. In the first step (*enrollment*, carried out only once, at the first turn-on, Fig. 1(a)) the PUF output X is used as noise to corrupt a randomly chosen codeword C in order to obtain $Y = X \oplus C$, which is saved in a Non Volatile Memory (NVM). In the second step (carried out every time the chip is turned on, Fig. 1(b)), the PUF is queried again and a possible ''noisy'' version of X, $\hat{X} = X \oplus E$, is obtained. Here E is a random noisy vector showing which bits of X changed from the first query and it has, hopefully, low Hamming weight. $\hat{X}$ is XOR-ed with Y to obtain $\hat{X} \oplus Y = C \oplus E$, from which C and E are computed via an error correction procedure. From E and $\hat{X}$ one obtains the original X.

Two drawbacks of helper-based stabilizers are the fact that the helper could leak information or be used for attacking the system (e.g, with a side-channel attack [23]–[25]) and that helper NVM are often expensive since they require special processes (different from the usual CMOS) for their production [26].

Given the drawbacks of helper-based stabilizers, there is some interest in developing *helper-less stabilizers*, that is, stabilizers that do not require auxiliary data. Developing an helper-less stabilizer is not trivial. For example, the direct usage of error correction codes does not work since there is no guarantee that the nominal output of the PUC will be a codeword (that is why a helper is used in Fig. 1). In order to understand if helper-less stabilization is feasible and which kind of performance we can expect, in this paper we determine some theoretical limits of helper-less stabilizers and show that the helper-less approach has the potential of providing good performance with limited complexity.

A. STATE OF THE ART

Most of the stabilizers developed so far make use of helpers. Among the most common proposed stabilizers one can cite *fuzzy extractors*, usually based on error correction codes [21], [22], stabilizer based on pattern matching [7], and *Index Based Syndrome* stabilizers [27]. The only helper-less stabilizer we are aware of is a brute-force exhaustive search for the error pattern proposed in [28]. For the sake of completeness, we observe that a preliminary version of this work has been presented at ICASSP [29].

B. OUR CONTRIBUTION

This paper has three main results. First, we prove that *global stability* is not achievable, that is, it is not possible to construct an helper-less stabilizer that stabilizes *every* possible device (see Theorem 1). The second result softens the first negative result showing that, although global stability cannot be achieved, one can achieve stability as good as desired over a *proper subset*, large as desired, of the possible devices (see Theorem 2). Finally, the third result shows that it is possible to recognize (and discard) at construction time the devices belonging to the ''bad'' subset (see Theorem 3), achieving *quasi-global stability*.

The proofs of Theorem 2 and Theorem 3 are constructive. Although the optimality of the procedures given in the proofs is still an open question, the proofs suggest an approach that is applicable in practical applications and whose complexity can be considered an upper bound of the complexity required by an helper-less stabilizer.

II. NOTATION

We recall here some of the less common notation used in this paper.

A. SETS

If A and B are sets, we will use $A \setminus B$ for their difference, that is, $A \setminus B := A \cap B^c$, we will use $|A|$ for the cardinality of A and we will use 2^A for the set of all subsets of A. If A is a subset of $\mathbb{R}$ of finite cardinality, we will define its *range* range (A) as the difference between its maximum and its minimum, that is range $(A) := \max A - \min A$.

If A is a metric space with distance $d : A \times A \to \mathbb{R}$, we will denote with $\mathfrak{B}_{c,r}$ the closed ball with center $c \in A$ and radius r, that is $\mathfrak{B}_{c,r} := \{x \in A : d(x, c) \leq r\}$. If A is endowed with a topology[1] [30] and $B \subseteq A$ we will denote with Int(B) the interior of B, with $\overline{B}$ its closure and with ∂B its boundary $\overline{B} \setminus \text{Int}(B)$. Also, we will suppose B implicitly endowed with the subspace topology inherited from A (that is, $C \subseteq B$ is open in B if C can be written as the intersection of an open set of A with B, see also Appendix A) [30].

[1] Appendix A gives brief summary of the topology concepts used in this paper.

We will denote with $\mathbb{R}_{\geq 0}$ and $\mathbb{R}_{>0}$ the sets of, respectively, non negative and positive reals.

B. RANDOM VARIABLES

We will write $X \sim \mathcal{N}(m, \sigma^2)$ to denote that X is a Gaussian random variable with mean m and variance σ^2. If A is a finite set, we will write $X \sim \mathcal{U}(A)$ to denote that X is a random variable uniformly distributed on A. We will use $X \sim \mathcal{B}(N, p)$ for a binomial r.v. with N trials and success probability p. We will sometimes commit a slight abuse of notation by using $\mathcal{N}(m, \sigma^2)$, $\mathcal{U}(A)$ and $\mathcal{B}(N, p)$ as anonymous r.v.; for example, we could write $P[\mathcal{N}(0, 1) > 1]$ for the probability that a zero mean, unit variance Gaussian r.v. is greater than 1.

C. VECTORS

If $\mathbf{A}$ and $\mathbf{B}$ are matrices, we will denote the block matrix obtained by placing $\mathbf{A}$ over $\mathbf{B}$ as $[\mathbf{A}\,;\,\mathbf{B}]$ (using a semi-colon as separator) rather than the more cumbersome $[\mathbf{A}^t, \mathbf{B}^t]^t$ or $\left[\begin{smallmatrix}\mathbf{A}\\ \mathbf{B}\end{smallmatrix}\right]$. If $\mathbf{v} \in \mathbb{R}^K$, we will denote its Euclidean norm as $\|\mathbf{v}\|$.

D. NOMENCLATURE

Depending on the context, ''PUC'' can mean both the abstract scheme (e.g., an electrical circuit) or a specific instance of it. In the former case, the PUC is an abstract object, while in the latter it is an actual physical object. This double use can make discussion difficult and sometimes it can introduce ambiguities. In order to avoid those difficulties, we will use the expression *PUC scheme* to refer to the abstract scheme and the expression *PUC instance* to refer to a specific physical implementation.

In this paper we are going to consider systems made of a PUC followed by a *stabilizer*. Note that the whole system, seen from the outside, behaves as a PUC. This introduces an ambiguity in the usage of the term PUC since it could refer both to the whole system ''PUC followed by the stabilizer'' or only to the ''internal'' PUC. In order to solve this ambiguity we introduce two new terms: when we will want to refer to the internal PUC we will talk about *Raw PUC (RPUC)*, while when we will want to refer to the whole system we will talk about *Stabilized PUC (SPUC)*. We reserve the term PUC for those concepts that can be applied both to RPUCs and SPUCs. (e.g., the *reliability distribution function* introduced in Remark 4.4).

III. THE MODEL

In our model a SPUC is made of two major components: the *RPUC* and the *stabilizer*.

A. THE RPUC

The RPUC is just a basic PUC, that is, a circuit whose behavior is very sensitive to variations in physical characteristics (e.g., dopant concentration). Every time the RPUC is queried, it produces a *raw output* (belonging to a suitable set $\mathcal{V}$) that is processed by the stabilizer to produce the SPUC output.

Remark 3.1: In most of the RPUCs proposed in the literature, $\mathcal{V}$ is a finite set of symbols, usually bit strings, i.e., $\mathcal{V} = \{0, 1\}^N$. The theory presented here, however, does not require $\mathcal{V}$ to be finite and it can handle even the case of RPUC with analog outputs, e.g., $\mathcal{V} = [0\text{ V}, 5\text{ V}]$.

In many cases the RPUC will be a collection of smaller generators (e.g., an RPUC with $\mathcal{V} = \{0, 1\}^N$ can be obtained by implementing N single bit generators). In this case we will call the smaller generators *cells*. If a cell produces a single bit output, we will call it a *binary cell*. Binary cells are maybe the most common examples of RPUC [5], [6], [16], [31].

As said in the introduction, the ideal PUC is a *random constant* (i.e., a random oracle [8]–[10] with no inputs): at construction time a random value is selected and the same value is returned every time the PUC is queried. However, most of the RPUC schemes proposed in the literature do not satisfy this requirement in the sense that, although a single instance can produce the same value most of the times, there is a non negligible probability that other values will be produced. The duty of the *stabilizer* is to process the RPUC output to convert it into a more stable SPUC output.

Example 2: It is worth to illustrate the concepts introduced above in the concrete case of a RPUC based on uninitialized Static RAM (SRAM) [5], [6], [31].

At turn-on an SRAM cell sets itself at random to one of the two possible values (0 or 1). If the structure of the cell is perfectly symmetric, both values are equiprobable; however, since the random variations during manufacturing will make the cell asymmetric, we expect that, for a single instance, one of the values will be more probable than the other [5], [6], [31]. It is intuitive (and it can be shown) that a cell with a strong asymmetry will have a marked preference for one of the two values, actually acting as a random constant; if the cell is almost symmetric it will have no preference and every query will be almost as a coin toss.

A number that measures the stability of a given instance is, for example, the probability Q of getting ''1'' as outcome: the best cells have $Q \approx 0$ or $Q \approx 1$. Note that Q is itself a random variable whose ''experiment'' is carried out at construction time.

B. THE STABILIZER

As explained above, the duty of the stabilizer is to reduce the run-time randomness of the RPUC in order to make the behavior of the SPUC more similar to a random constant. Formally, a stabilizer will be defined as a function $\mathcal{S} : \mathcal{V} \to \mathbb{I}$, where $\mathbb{I}$ is a finite set representing the set of the possible SPUC outputs. In the typical case we expect $\mathbb{I} = \{0, 1\}^M$, but this is not necessary for the theory given here, as long as $\mathbb{I}$ is finite and $|\mathbb{I}| > 1$.

Most of the stabilizers proposed in the literature require some auxiliary value (an *helper*) usually generated at the first turn-on [7], [21], [22], [27]. In this paper we are interested, however, in *helper-less stabilizers*, that is, stabilizers that do not require any auxiliary value. More precisely, we are

interested in finding out the theoretical limits to the stability of a SPUC that employs a helper-less stabilizer.

C. WHAT ARE GOOD SPUCs MADE OF?

It is worth pointing out the two most important characteristics that a SPUC should have: *stability* (or *noiseless*) and *uniformity* (or *maximum entropy*). Informally, the stability of a SPUC scheme is related to the fact that a given instance will always produce the same output value every time it is queried, while uniformity is related to the fact that the probability of producing an instance whose preferred output is '1' is equal to the probability of producing an instance whose preferred output is '0'.

Remark 3.2: A very informal example that clarifies the concepts of stability and uniformity is the following. A binary PUC scheme is like a bag full of coins, where the coins are not necessarily well-balanced. The act of constructing a PUC instance corresponds to extracting a coin at random from the bag, while the act of quering the PUC instance corresponds to throwing the coin. Note that the probability of getting "head" is a random variable since it depends on the result of the first part of the experiment (extracting the coin from the bag).

A *stable* PUC scheme corresponds to a bag full of coins that are maximally unbalanced, that is, every coin in the bag always lands on the same side. A *uniform* scheme corresponds to a bag having half coins that prefer "head" and half coins that prefer "tail."

In this paper we are mostly interested in stability, although even uniformity is considered (see, for example, Section IV-A).

IV. MEASURING THE STABILITY OF A SPUC

In order to make precise the meaning of *stability of a SPUC* it is necessary to introduce some concepts. First we give a formal definition of an RPUC that takes into account its double randomness nature.

Definition 1: A parametrized raw generator *is a four-tuple* $(\mathscr{V}, \mathscr{Q}, V, Q)$ *where* $\mathscr{V}$ *is any set representing the* raw alphabet*,* V *is a random variable assuming values in* $\mathscr{V}$*,* $\mathscr{Q}$ *is the set of* statistical parameters *and* Q *is a random variable assuming values in* $\mathscr{Q}$*. We will assume that* $\mathscr{Q}$ *is endowed with a topology* $\mathscr{T}_{\mathscr{Q}}$ *such that for every event* $A \subset \mathscr{V}$ *the map*

$$\mathscr{Q} \ni q \mapsto P[V \in A \mid Q = q] \tag{1}$$

is continuous.

Remark 4.1: Definition 1 formalizes the idea of the "double randomness" in a RPUC. At construction time, a value $q \in \mathscr{Q}$ is drawn; successively, at every turn-on, a value of $\mathscr{V}$ is drawn according to the conditional probability $P[\cdot \mid Q = q]$. Note that no special property is required for $\mathscr{V}$ and $\mathscr{Q}$; requiring that the parameter set $\mathscr{Q}$ is endowed with a topology $\mathscr{T}_{\mathscr{Q}}$ is a technicality necessary to talk about the continuity of (1) and, in the following, about the connectedness of $\mathscr{Q}$.

Example 3: It is worth giving few examples of models of raw generators.

- In the case of a binary cell, the raw alphabet is $\mathscr{V} = \{0, 1\}$ (the cell produces one bit) while, as explained in Example 2, a natural choice for Q is the probability that the cell produces "1", so that

$$P[V = 1 \mid Q = q] = q$$
$$P[V = 0 \mid Q = q] = 1 - q \tag{2}$$

  Clearly, $\mathscr{Q} = [0, 1]$. See Fig. 2(a).

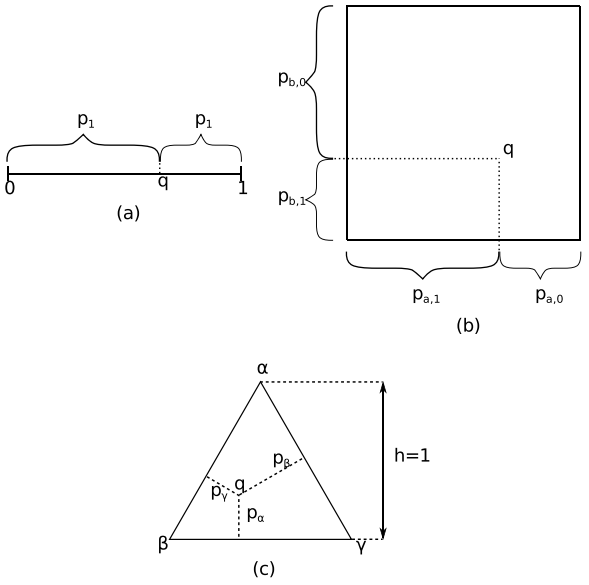

FIGURE 2. Examples of parameter sets for some RPUC. (a) Parameter set for a binary cell. (b) Parameter set for a pair of independent binary cells. (c) Parameter set for an RPUC with a three-symbol alphabet $\{\alpha, \beta, \gamma\}$.

- In the case of a RPUC made of two independent (but not necessarily identically distributed) cells, the raw alphabet is $\mathscr{V} = \{0, 1\}^2 = \{00, 01, 10, 11\}$, while the statistical parameter Q is a two-dimensional random vector assuming values in the square $\mathscr{Q} = [0, 1]^2$. See Fig. 2(b).
- In order to enrich the set of examples, we will also consider the fictitious case of a RPUC with three output symbols. In this case the alphabet is, for example, $\mathscr{V} = \{\alpha, \beta, \gamma\}$, while the natural choice for Q is the vector $[Q_\alpha, Q_\beta, Q_\gamma]$, where Q_v is the probability of outcome $v \in \mathscr{V}$. Therefore

$$P[V = \alpha \mid Q = [q_\alpha, q_\beta, q_\gamma]] = q_\alpha$$
$$P[V = \beta \mid Q = [q_\alpha, q_\beta, q_\gamma]] = q_\beta$$
$$P[V = \gamma \mid Q = [q_\alpha, q_\beta, q_\gamma]] = q_\gamma \tag{3}$$

  and the parameter set $\mathscr{Q}$ is the two-dimensional simplex

$$\mathscr{Q} = \{[q_\alpha, q_\beta, q_\gamma] \in \mathbb{R}^3_{\geq 0} : q_\beta + q_\alpha + q_\gamma = 1\} \tag{4}$$

  A convenient way to represent the parameter set (4) is as an equilateral triangle where the symbol probabilities can be read as barycentric coordinates. See Fig. 2(c).

Now we can refine the idea of stabilizer. More precisely,

Definition 2: If $\mathbb{I}$ *is any finite set and* $\mathscr{G} = (\mathscr{V}, \mathscr{Q}, V, Q)$ *is a parametrized generator, a* stabilizer for $\mathscr{G}$ with output in $\mathbb{I}$ *is a measurable function* $\mathscr{S} : \mathscr{V} \to \mathbb{I}$.

For the sake of notational convenience we define, for every $A \subseteq \mathbb{I}$

$$P_q[A] := P[\mathscr{S}(V) \in A \mid Q = q] \tag{5}$$

Moreover, if $A = \{i\}$, we will write $P_q[i]$ rather than $P_q[\{i\}]$. In other words, $P_q[i]$ is the probability of getting the output symbol i when the statistical parameter Q is equal to q.

For a given SPUC instance with statistical parameter equal to $q \in \mathscr{Q}$, we will denote with $\overrightarrow{P}_q\ (n)$, $n = 1, \ldots, |\mathbb{I}|$, the n-th largest output probability. More precisely,

$$\overrightarrow{P}_q\ (1) \geq \overrightarrow{P}_q\ (2) \geq \cdots \geq \overrightarrow{P}_q\ (|\mathbb{I}|) \tag{6}$$

and there is a bijection $n \mapsto i_n$ from $\{1, \ldots, L\}$ to $\mathbb{I}$ such that $\overrightarrow{P}_q\ (n) = P_q[i_n]$. Note that $\overrightarrow{P}_q\ (1)$ is the probability of the most probable output symbol when $Q = q$. It is convenient to give a special name to the most probable output symbol for a given $q \in \mathscr{Q}$.

Definition 3: We will say that $i \in \mathbb{I}$ *is* a winner in $q \in \mathscr{Q}$ *if*

$$P_q[i] = \overrightarrow{P}_q\ (1) \tag{7}$$

Moreover, we define the winning set *of q, denoted as* $w(q)$*, as the set of winners in q, that is*,

$$w(q) := \{i \in \mathbb{I} : P_q[i] = \overrightarrow{P}_q\ (1)\} \tag{8}$$

Remark 4.2: Note that although the winner is not necessarily unique, all the winners share the same probability.

It is convenient also to have a name to refer to the set of statistical parameters q that make a given output symbol i a winner.

Definition 4: The *winning region* of $i \in \mathbb{I}$ is defined as

$$\mathscr{W}_i := \{q \in \mathscr{Q} : P_q[i] = \overrightarrow{P}_q\ (1)\} \tag{9}$$

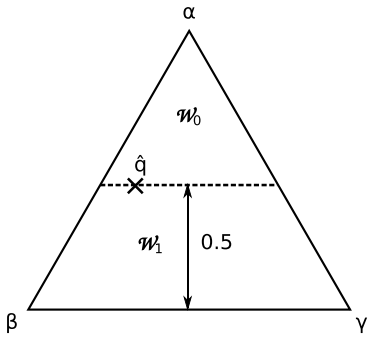

FIGURE 3. Partitioning of the parameter sets in winning regions for a stabilizer with ternary input and binary output. The cross marks a *tie point* where the stability cannot be larger than 1/2.

Example 4: Fig. 3 shows the division of the parameter set $\mathscr{Q}$ in winning regions for the case of a stabilizer with input alphabet $\mathscr{V} = \{\alpha, \beta, \gamma\}$ and output alphabet $\mathbb{I} = \{0, 1\}$. Note that, as explained in Example 3, $\mathscr{Q}$ is represented in Fig. 3 as an equilateral triangle.

In the specific case of Fig. 3, the stabilizer maps α to 0 and both β and γ to 1. It follows that the winning region of 0 is given by those q such that $q_\alpha \geq 1/2$, that is, the points of the triangles in Fig. 3 whose distance from the horizontal edge is larger than $1/2$.

A. NON-REDUNDANT STABILIZERS AND BALANCED STABILIZERS

It is clear that the winner represents the most probable outcome of the stabilizer. In this paper we will focus only on *non-redundant* stabilizers, i.e., stabilizers such that $\mathscr{W}_i \neq \emptyset$ for every $i \in \mathbb{I}$. In other words, with a non-redundant stabilizer every symbol i has a chance to be a winner.

Note that the fact that a stabilizer is non-redundant depends only on the map implemented by the stabilizer itself and not on the statistical properties of Q. Note that a non-redundant stabilizer can still have bad statistical properties. As an extreme example, one can observe that the stabilizer represented in Fig. 3 is clearly non redundant, but it could happen that for a specific RPUC scheme $P[Q \in \mathscr{W}_0] = 0$, so that when the stabilizer is used with such an RPUC, only "1" can be a winner.

In order to take into account this kind of cases it is convenient to introduce the concept of *unbalance* of a stabilizer. Let

$$P_{\text{win}}(i) := P[w(Q) = \{i\}] \tag{10}$$

denote the probability that $i \in \mathbb{I}$ is the *only* winner.

Example 5: $P_{\text{win}}(i)$ can be interpreted as the probability of constructing a PUC instance whose preferred output is i. For example, with reference to Remark 3.2, $P_{\text{win}}(\text{Tail})$ is the probability of extracting a coin that prefers to land on "tail."

Definition 5: The unbalance $\mathfrak{u}$ *of a stabilizer is defined as the range of set* $\{P_{win}(i)\}_{i\in\mathbb{I}}$*, that is,*

$$\mathfrak{u} := \text{range}\,(\{P_{\text{win}}(i)\}_{i\in\mathbb{I}}) = \max_{i\in\mathbb{I}} P_{\text{win}}(i) - \min_{i\in\mathbb{I}} P_{\text{win}}(i) \tag{11}$$

A stabilizer with $\mathfrak{u} = 0$ *will be said* balanced *or* unbiased.

It is easy to verify that $0 \leq \mathfrak{u} \leq 1$, that $\mathfrak{u} = 0$ if and only if $P_{\text{win}}(i) = 1/|\mathbb{I}|$ for all $i \in \mathbb{I}$ and $\mathfrak{u} = 1$ if and only if $P_{\text{win}}(i) = 1$, for some $i \in \mathbb{I}$. It is clear that the unbalance depends on the employed RPUC scheme via the distribution of Q and that the unbalance of a good stabilizer will be small and almost independent on the details of the distribution of Q. It is clear that we prefer balanced stabilizers since the output will have maximum entropy.

Remark 4.3: Note that in (10) we write $P[w(Q) = \{i\}]$ and not $P[w(Q) = i]$ because, according to (8), $w(Q)$ is a set. Also note that events $w(Q) = \{i\}$ are not a partition of the probability space since the subset of $\mathscr{Q}$ that has more than one winner (e.g., the dashed line in Fig. 3) is excluded. However, in most cases of practical interest the excluded subset has zero probability.

B. LOCAL AND GLOBAL STABILITY

Since we would like that the output of the stabilizer be always the same at every turn-on, we would like the probability of the winner being very close to 1. This suggests to define the stability of a stabilizer as follows.

Definition 6: If $q \in \mathcal{Q}$, we define $\epsilon(q)$, the local stability in *q, as the probability of a winner in q, that is*,

$$\epsilon(q) := \vec{P}_q(1) \tag{12}$$

The global stability *η of a stabilizer is defined as the worst-case stability, that is*,

$$\eta := \inf_{q \in \mathcal{Q}} \epsilon(q). \tag{13}$$

Remark 4.4: Another measure of quality of a SPUC instance is given by its *reliability* $R(q)$ defined as [16]

$$R(q) := \vec{P}_q(1) - \vec{P}_q(2) \tag{14}$$

Note that $R(q) \in [0, 1]$ and that $R(q) = 1$ if and only if $\vec{P}_q(1) = 1$, that is, if and only if the PUC instance produces always the same symbol, that is, it behaves as a random constant. We will say in this case that the instance is *perfectly reliable*. If instead $R(q) = 0$, it follows that $\vec{P}_q(1) = \vec{P}_q(2)$ which implies that there are at least two winners $i, j \in \mathbb{I}$ with the same probability. In this case the behavior of the SPUC is more similar to a coin toss and the SPUC is *totally unreliable*.

If function $R(q)$ in (14) is applied to random variable Q, one obtains random variable R_Q. The distribution F_R of R_Q will be called the *Reliability Distribution Function* (RDF) of the PUC scheme. In an ideal PUC scheme every instance is perfectly reliable, so that the RDF is a step function centered in 1.

Finally, note that since

$$\epsilon(q) \geq R(q) \geq 2\epsilon(q) - 1 \tag{15}$$

both stability and reliability can be considered two equivalent measures of quality for stabilizers. In this paper we will mainly refer to the stability $\epsilon(q)$.

C. THE INFLUENCE OF THE ENVIRONMENT

The behavior of the RPUC (and, therefore, of the whole SPUC) can change as environment variables (e.g., temperature, electromagnetic fields, aging effects) change. From the viewpoint of our model, the effect of the environment $\mathcal{E}$ can be represented as a change in the actual value of Q. The theory given here can be easily adapted to the case of changing environment.

V. FIRST RESULT: PERFECT RELIABILITY IS UNACHIEVABLE

Now we can give one of the main results of this paper.

Theorem 1: Let $\mathcal{S} : \mathcal{V} \to \mathbb{I}$ be a non-redundant stabilizer for the parametrized generator $(\mathcal{V}, \mathcal{Q}, V, Q)$. Let η be the global stability of $\mathcal{S}$. If $\mathcal{Q}$ is connected and $|\mathbb{I}| > 1$, then $\eta \leq 1/2$ and $R(q) = 0$ for some $q \in \mathcal{Q}$.

The proof of Theorem 1 is quite technical and it is given in Appendix C-A. It is possible, however, to give an intuitive motivation.

With reference to Example 4 and Fig. 3, consider point $\hat{q}$ marked with a cross in Fig. 3. Since $\hat{q}$ lies on the boundary between sets $\mathcal{W}_0$ and $\mathcal{W}_1$, we deduce that outputs value 0 and 1 have equal probability (that is, $\hat{q}$ is a *tie point*) and this implies that $P_{\hat{q}}[0] = P_{\hat{q}}[1] = 1/2$ which in turn implies $\epsilon(\hat{q}) = 1/2$ and $R(\hat{q}) = 0$, proving Theorem 1 in this very special case.

Note that the cornerstone of the just outlined reasoning is the existence of a *tie point*. The key part of the proof of Theorem 1 is to show that a tie point always exists as soon as $\mathcal{Q}$ is connected.

It is worth emphasizing the generality of Theorem 1. For example, it holds independently on the nature of $\mathcal{V}$ that can be, for example, discrete (e.g. $\{0, 1\}^N$) or continuous (e.g., $\mathbb{R}^N$). This means that Theorem 1 holds also, for example, for "soft-decoding style" stabilizers.

Theorem 1 holds also if one does $K > 1$ queries to the RPUC by turning it on and off K times [32]. Indeed, in this case one can consider a "virtual" RPUC made of K copies of the same RPUC. The virtual RPUC will produce values in $\mathcal{V}^K$, it will have the same parameter set $\mathcal{Q}$ and the probability density function of the new RPUC will be obtained as the K-times product of the density of the original RPUC. Since even the new RPUC has a connected parameter set, Theorem 1 still applies.

The concept of global stability is clearly a very strong one. Actually, it suffices a single value of q having $\epsilon(q)$ small for making the whole global stability η small, even if that "bad case" is very unlikely. This observation suggests a weaker form of stability where we accept to ignore bad cases, as long as they do not happen too often. This idea is formalized in the concept of (η, δ)-*stability* introduced in the next section.

VI. SECOND RESULT: (η, δ)-STABILITY IS ACHIEVABLE

Let us first define formally the concept of (η, δ)-stability.

Definition 7: Stabilizer $\mathcal{S}$ will be said to be (η, δ)- stable *if*

$$F_\epsilon(\eta) = P[\epsilon(Q) \leq \eta] = P\left[Q \in \epsilon^{-1}([0, \eta])\right] \leq \delta. \tag{16}$$

The second important result that we are going to prove is that it is possible to design an (η, δ)-stable stabilizer for every choice of $\eta < 1$ and $\delta > 0$; in other words, we can make an SPUC as stable as desired over a portion as large as desired of the parameter set. Of course, we expect that as η gets larger and δ smaller, the corresponding stabilizer will be more expensive.

The proof will be given for the case $\mathcal{V} \subseteq \mathbb{R}^N$. (This is a very weak hypothesis verified in every case of practical interest.) Since $\mathcal{V} \subseteq \mathbb{R}^N$, V is a N-dimensional random vector. Let its k-th component, $k = 1, \ldots, N$, be denoted with V_k and let the mean and variance of V_k be denoted as

$$m_k(q) := \mathbb{E}[V_k \mid Q = q] \tag{17a}$$

$$\sigma_k^2(q) := \mathbb{E}\left[(V_k - m_k(q))^2 \mid Q = q\right] \tag{17b}$$

Of course, we will suppose that expected values (17) exist for every $k \in \{1, \ldots, N\}$ and $q \in \mathcal{Q}$.

Let $\mathbf{m} : \mathcal{Q} \to \mathbb{R}^N$ be defined as $\mathbf{m}_q := [m_1(q), \ldots, m_N(q)]$. Similarly, let $\Sigma : \mathcal{Q} \to \mathbb{R}^N$ be the function defined

as $\Sigma_q := [\sigma_1(q), \ldots, \sigma_N(q)]$.[2] Finally, we will denote with $\mathscr{M} := \mathrm{Im}(\mathbf{m})$ the set of possible mean vectors.

Theorem 2: With reference to the just introduced notation, suppose function $\mathbf{m}$ *continuous and* Σ *and* $\mathbf{m}$ *bounded (in the sense that one can find* $\Sigma_{max} < \infty$ *and* $M < \infty$ *such that* $\Sigma_{max} \geq \|\Sigma_q\|$ *and* $M \geq \|\mathbf{m}_q\|$ *for all* $q \in \mathscr{Q}$*).*

If it is possible to find a partition $\{U_i\}_{i\in\mathbb{I}}$ *of* $\mathscr{M}$ *indexed by* $\mathbb{I}$ *such that*

$$P\left[\mathbf{m}_Q \in \bigcup_{i\in\mathbb{I}} \partial U_i\right] = P\left[Q \in \mathbf{m}^{-1}(\bigcup_{i\in\mathbb{I}} \partial U_i)\right] = 0 \qquad (18)$$

then for every $\eta < 1$ *and* $\delta > 0$ *it is possible to find an* (η, δ)*-stable stabilizer for* $(\mathscr{V}, \mathscr{Q}, V, Q)$*. Moreover, the unbalance of the stabilizer can be made as close as desired to* range $(\{P[\mathbf{m}_Q \in U_i]\}_{i\in\mathbb{I}})$.

Remark 6.1: It is worth commenting about the technicalities of the claim of Theorem 2. Condition (18) will be used to show that one can find, for every $\delta > 0$, a small "strip" around $\cup_i \partial U_i$ so that the probability of $\mathbf{m}$ belonging to such a strip is smaller than δ. Note that condition (18) is very weak. For example, if sets U_i are "non pathological" sets with an $N-1$-dimensional boundary (e.g., they are homeomorphic images of a sphere) and the density of $\mathbf{m}$ is not singular, then (18) is automatically satisfied.

We will prove this result constructively. The basic idea is a generalization of the stabilizer proposed in [32]. More precisely, we first estimate $\mathbf{m}_q$ by querying the RPUC K times and computing the average of the raw outputs, successively we "quantize" the computed estimate by using partition $\{U_i\}_{i\in\mathbb{I}}$ (see Algorithm 1). By a careful application of the law of large numbers and hypothesis (18), it will be proved that one can choose K large enough to make this stabilizer as stable as desired.

Algorithm 1 The Algorithm Used to Prove Theorem 2

1 Query the RPUC K times; let v_k be the outcome of V at the k-th query;
*/*K is decided at design time, see Lemma 5 */*
2 Compute the average $\mu = (1/K)\sum_{k=1}^{K} v_k$
3 Output the symbol $i \in \mathbb{I}$ such that $\mu \in U_i$.;
*/*Note that i exists and it is unique since* $\{U_i\}_{i\in\mathbb{I}}$ *is a partition of* $\mathscr{M}$ *and* $\mu \in \mathscr{M}$*. */*

Remark 6.2: It is worth emphasizing that the main reason for introducing Algorithm 1 is to prove the existence of arbitrarily good stabilizers. Although Algorithm 1 can be used in practice, no claim is done about its optimality. Actually, the problem of constructing the optimal helper-less stabilizer for a given RPUC is still open.

The first step in proving that the stabilizer constructed according to Algorithm 1 can be made (η, δ)-stable for a suitable choice of K is to choose the subset of "bad" q, that is, the subset of $\mathscr{Q}$ that will correspond to the devices with stability less than η. Since Algorithm 1 quantizes average μ using partition $\{U_i\}_{i\in\mathbb{I}}$, it is intuitive that the points belonging to $\mathbf{m}^{-1}(\cup_{i\in\mathbb{I}}\partial U_i)$ will be tie points and that the output of the stabilizer will be more stable for those q far from $\cup_{i\in\mathbb{I}}\partial U_i$. This suggests the following approach: we choose as bad set a "strip" S_Δ of width Δ (the exact meaning of "width" is given later) around $\cup_{i\in\mathbb{I}}\partial U_i$. The width Δ of the strip zone must be small enough to make the probability of $Q \in S_\Delta$ smaller than δ. For the RPUC whose parameter does not belong to S_Δ, we proceed to find a K large enough to make the variance of μ small enough to make the probability of crossing the buffer zone smaller than $1-\eta$.

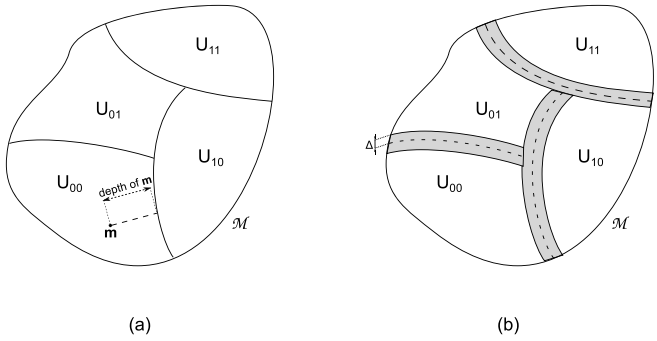

FIGURE 4. (a) Example of a partition of $\mathscr{M}$ and graphical representation of the depth of a point. (b) Example of strip S_Δ for the partition in (a).

In order to define precisely S_Δ and develop the outline of work, we need the concept of *depth* of $b \in \mathscr{M}$, which gives a measure of how far b is from a boundary. For the sake of notational convenience, if $b \in \mathscr{M}$, we will denote with ι_b the element of $\mathbb{I}$ such that $b \in U_{\iota_b}$. In other words, ι_b is the output of the stabilizer when the average computed by Algorithm 1 is b.

Definition 8: Given $b \in \mathscr{M}$*, its* depth $\mathfrak{d}_b$ *is defined as*

$$\mathfrak{d}_b := \inf_{x \notin \mathrm{Int}(U_{\iota_b})} \|x - b\| \qquad (19)$$

Remark 6.3: Informally, $\mathfrak{d}_b$ is the minimum distance that b must "travel" to exit from its partition U_{ι_b} (see Fig. 4(a)). The infimum is taken over the complement of $\mathrm{Int}(U_{\iota_b})$ and not ∂U_{ι_b} because if $\mathscr{M}$ is not connected, ∂U_{ι_b} could be empty (see Fig. 5).

In order to make the following analysis easier, it is worth giving some easy properties of the depth. The first property is that we can replace the inf in (19) with a min.

Lemma 1: The inf *in (19) is actually a* min*, that is, there exists* $c \in (\mathrm{Int}(U_{\iota_q}))^c$ *such that*

$$\mathfrak{d}_b = \|c - b\| \qquad (20)$$

Proof: Set $(\mathrm{Int}(U_{\iota_q}))^c$ is compact since it is closed (being the complement of an open set) and bounded, since it is a subset of bounded set $\mathscr{M}$. Therefore, continuous map $(\mathrm{Int}(U_{\iota_q}))^c \ni x \mapsto \|x - b\|$ has a minimum. □

The second result, whose proof is reported in Appendix C, claims that a point has null depth if and only if it is on a boundary.

[2]Note that for the sake of notational convenience we denote the value assumed by $\mathbf{m}$ and Σ in q as $\mathbf{m}_q$ and Σ_q rather than the usual $\mathbf{m}(q)$ and $\Sigma(q)$.

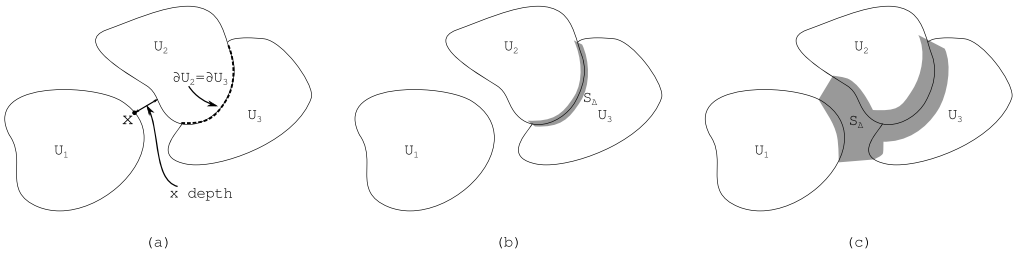

FIGURE 5. (a) Case of set $\mathscr{M}$ disconnected: Set U_1 is a connected component, so its boundary is empty. The only boundary is the separation between U_2 and U_3, shown with a dashed line. Note also that point x, although a "most external" point of U_1, has non null depth since it does not belong to a boundary. (b) and (c) Examples of strip $S_{\mathbf{\Delta}}$ for the partition in (a) and two different values of $\mathbf{\Delta}$.

Lemma 2: For every $b \in \mathscr{M}$, $\mathfrak{d}_b \geq 0$ and $\mathfrak{d}_b = 0$ if and only if $b \in \partial U_{\iota_b}$.

The third result, whose proof is reported in Appendix C, is a kind of triangular inequality.

Lemma 3: For every $a, b \in \mathscr{M}$, $\|a - b\| \geq |\mathfrak{d}_a - \mathfrak{d}_b|$.

By using the concept of depth, now we can give a formal definition of the "buffer zone" S_Δ.

Definition 9: For every $x \geq 0$, define $S_x \subseteq \mathscr{M}$ as

$$S_x := \{b \in \mathscr{M} : \mathfrak{d}_b \leq x\} \tag{21}$$

Example 6: Fig. 4(b) shows an example of S_Δ for the partitioning of Fig. 4(a). Fig. 5(b) and Fig. 5(c) show two different example of S_Δ for the partitioning of Fig. 5(a); note that in Fig. 5(b), Δ is small enough that only the border between U_2 and U_3 is involved.

The next step is to show that we can find a strip S_Δ thin enough so that the probability of having $\mathbf{m}_Q \in S_\Delta$ is as small as desired.

Lemma 4: For every $\delta > 0$ there is $\Delta > 0$ such that

$$P[\mathbf{m}_Q \in S_\Delta] \leq \delta \tag{22}$$

The proof, that exploits hypothesis (18), is reported in Appendix C.

The second step toward the proof of Theorem 2 is to show that it is always possible to find K such that $\epsilon(q) \geq \eta$ as soon as $\mathbf{m}_q$ does not belong to the bad set S_Δ.

Lemma 5: Let $\Delta > 0$ and $\eta < 1$. If

$$K \geq \frac{\Sigma_{\max}^2}{\Delta^2(1-\eta)} \tag{23}$$

then

$$\mathbf{m}_q \notin S_\Delta \Rightarrow P_q[\iota_{\mathbf{m}_q}] \geq \eta \tag{24}$$

The proof is given in Appendix C, and it uses the generalized Chebyshev's inequality shown in Appendix B.

Remark 6.4: Note that if $\eta > 1/2$, (24) implies that no output symbol can have probability larger than $P_q[\iota_{\mathbf{m}_q}]$, so that $P_q[\iota_{\mathbf{m}_q}] = \epsilon(q)$. Therefore, in this case, (24) can be read as the fact that any q not belonging to the bad set has stability at least equal to η.

Note also that if the distribution of Q changes because of aging or enviromental variations, the right hand side of (23) will be likely to change too, since $\Sigma_{\max}$ and Δ depend on the distribution of Q. If (23) is going to be used in a real design, a possible approach is to consider the worst case and take for K the maximum of (23), taken over all the environment conditions.

Finally, note that the right hand side of (23) is likely to be a "pessimistic" bound, since it is derived from Chebyshev's inequality. A less pessimistic bound can be derived by approximating $\mathbf{m}_q$ with a Gaussian variable. Although such an approximation is likely to be good enough for practical purposes, its use in a formal proof is not straightforward.

Now, in order to prove Theorem 2, it suffices to put together Lemma 4 and Lemma 5.

Proof of Theorem 2: In the proof we are going to suppose without loss of generality $\eta > 1/2$. If this was not true we can replace η with any value in $(1/2, 1)$ since if $\eta_1 > \eta_2$ a stabilizer that is (η_1, δ)-stable is also (η_2, δ)-stable.

Given $\delta > 0$, use Lemma 4 to find Δ such that $P[\mathbf{m}_Q \in S_\Delta] < \delta$. Successively use Δ and η with Lemma 5 in order to find K such that (24) is satisfied. Note that since $\eta > 1/2$, (24) implies

$$\mathbf{m}_q \notin S_\Delta \Rightarrow \epsilon(q) \geq \eta \tag{25}$$

Now we can verify that the proposed stabilizer is (η, δ)-stable

$$\begin{aligned} P[\epsilon(Q) < \eta] &\leq P[\mathbf{m}_q \in S_\Delta] \quad \text{Contrappositive of (25)} \\ &\leq \delta \qquad\qquad\quad \Delta \text{ obtained via Lemma 4.} \end{aligned} \tag{26}$$

□

VII. THIRD RESULT: FROM (η, δ)-STABILITY TO QUASI-GLOBAL STABILITY

The discussion in Section VI about the construction of an (η, δ)-stable stabilizer suggests a way to achieve global stability, despite of Theorem 1. The idea is very simple: if we were able to check at construction time which devices have a value of q such that $\mathbf{m}_q \in S_\Delta$, we could discard them, keeping only the ones with stability larger than η.

Remark 7.1: Note that discarding the devices such that $\mathbf{m}_q \in S_\Delta$ is equivalent to remove from $\mathscr{Q}$ the anti-image $\mathbf{m}^{-1}(S_\Delta)$ of S_Δ. The remaining set is typically disconnected, as claimed by the following property whose proof is in Appendix C.

For every $i \in \mathbb{I}$, let $V_i := U_i \cap S_\Delta^c$. If $\mathbf{m} : \mathscr{Q} \to \mathscr{M}$ is continuous and there are at least two $i, j \in \mathbb{I}$, $i \neq j$ such that $V_i \neq \emptyset$, $V_j \neq \emptyset$, then $\mathscr{Q} \setminus \mathbf{m}^{-1}(S_\Delta)$ is not connected.

The fact that $\mathscr{Q}\backslash\mathbf{m}^{-1}(S_\Delta)$ is disconnected is what allows us to escape the claim of Theorem 1 and achieve global stability. Note the condition that at least two sets V_i must be not empty; if Δ is too large it could happen that only one V_i "survives" and that set could be connected. However, in this case the stabilizer will output always the same symbol, so it is not of practical interest.

A drawback of this idea is that the actual value of $\mathbf{m}_q$ is not known. The natural solution is to estimate it by using a technique similar to Algorithm 1.

We will say that a device is *unreliable* if its statistical parameter q is such that $\mathbf{m}_q \in S_\Delta$. Let UNREL denote the event that a device is unreliable (that is, $\mathbf{m}_q \in S_\Delta$) and $\overline{\text{UNREL}}$ the event a device is reliable (that is, $\mathbf{m}_q \notin S_\Delta$).

We will call any procedure designed to check if a device is reliable or not a *verifier*. Let REJECT denote the event when the device is declared unreliable and $\overline{\text{REJECT}}$ when a device is declared reliable by a given verifier.

Any verifier can make two different types of misclassification: (i) *false negative*, when an unreliable device is accepted and (ii) *false positive*, when a reliable device is discarded. Two natural indices of performance for a verifier are the probabilities

$$P_- := P[\overline{\text{REJECT}} \mid \text{UNREL}]\text{False negative rate} \quad (27a)$$

$$P_+ := P[\text{REJECT} \mid \overline{\text{UNREL}}]\text{False positive rate} \quad (27b)$$

Note that P_- represents the fraction on shipped RPUC that are unreliable while P_+ represents the fraction of reliable RPUC that are discarded. Clearly, one wants to keep both probabilities low.

Definition 10: We will say that a verifier is (r_-, r_+)-reliable *if* $P_- \leq r_-$ *and* $P_+ \leq r_+$.

Theorem 3: If $P[\mathfrak{d}_{\mathbf{m}_Q} = \Delta] = 0$, *then for every choice of* $r_- > 0$ *and* $r_+ > 0$ *it is possible to find a verifier that is* (r_-, r_+)-reliable.

The proof of Theorem 3 is constructive since it shows the existence of the desired verifier by showing how to construct one. The algorithm used in the proof of Theorem 3 is Algorithm 2 which is parametrized by the two parameters $D > \Delta$ and N that are to be decided at design time. In the proof it is shown that one can find D and N such that the verifier represented by Algorithm 2 has the desired performance.

Algorithm 2 Proposed Verifier

1 Query the RPUC N times; let v_k be the outcome of V of the k-th query;
/* N is decided at design time and it can differ from K*/
2 Compute the average $\omega = (1/N)\sum_{k=1}^{N} v_k$;
3 If $\omega \in S_D$ declare the device unreliable, otherwise declare it reliable;
/* D is selected at design time. In every case of practical interest it will be $D > \Delta$*/

The proof is not difficult, but quite long, and is given in Appendix C-B. However, it is possible to give an informal description of the ideas behind the proof of Theorem 3. Fig. 6 shows the forbidden band S_Δ, the band S_D and $\mathbf{m}_q$ for three reliable devices (labeled r, s and t) and an unreliable one (labeled u). The average ω computed by Algorithm 2 can be approximated with a Gaussian variable with mean equal to $\mathbf{m}_q$ and variance proportional to $1/N$. The variance of ω in the cases of the three devices are symbolically shown as little circles around the points representing the three devices. One can think the circles as some kind of "effective support" of the density of ω.

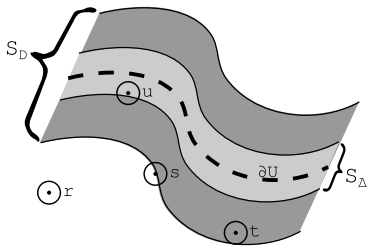

FIGURE 6. Graphical representation of reliable/unreliable devices and strips S_Δ and S_D. Device r is a reliable device whose false positive probability can be made as small as possible; devices s and t are reliable devices whose false positive probability cannot be made arbitrarily small; device u is an unreliable device whose false positive probability can be made arbitrarily small.

From Fig. 6 it is clear how one can make P_- (that is, the probability of accepting u) as small as desired: it suffices to choose N large enough so that every element of S_Δ has its effective support inside S_D. Note that this can be done, whatever the value of D, as long as $D > \Delta$.

The procedure to make P_+ small is not so direct. While it is clear that by choosing N large enough one can make the probability of discarding r as small as desired, we expect that devices like the one marked with s will always have approximately a 50% chance of being discarded, independently on N. Moreover, devices like t, belonging to the "belt" $S_D\backslash S_\Delta$, have a probability larger than 50% of being discarded (and that probability increases with N). Because of this, the only way to keep P_+ under control is by making the belt $S_D\backslash S_\Delta$ thin enough to make that the probability of a device belonging to $S_D\backslash S_\Delta$ small as desired.

VIII. EXAMPLES OF STABILIZER DESIGNS

In order to give some intuition about the results given above, we are going to show an example about how to specialize them to the case where RPUC $(\mathscr{V}, \mathscr{Q}, V, Q)$ is just a binary cell. The same procedure can be easily adapted to others RPUC. Remember that if the RPUC is a binary cell, then $\mathscr{V} = \{0, 1\} \subset \mathbb{R}$, the statistical parameter Q is equal to the probability of getting "1" and the parameter set is the interval $\mathscr{Q} = [0, 1]$. Note that $\mathbf{m} \in [0, 1] = \mathscr{M} \neq \mathscr{V}$ and that, in this very particular case, $\mathbf{m}_q = q$. The stabilizer will output a single bit, so $\mathbb{I} = \{0, 1\}$.

A. THE (η, δ)-STABILITY APPROACH

First we show how to construct a (η, δ)-stable stabilizer. As shown above, the first step is choosing a partition of $\mathscr{M}$,

possibly one with small range (so that the final stabilizer will be balanced). The most natural partition is

$$\mathscr{M} = [0, 1] = \underbrace{[0, 1/2]}_{U_0} \cup \underbrace{(1/2, 1]}_{U_1} \tag{28}$$

This partition results also to be balanced as soon as the density of Q is symmetric around $1/2$, as it happens, for example, with SRAMs [6]. It will be clear that with this specific setup, the constructive proof given before reduces itself to the stabilizer proposed in [32]. This is not surprising since, as said before, the proof is just a wide generalization of the approach of [32].

More into details, the idea is to query the cell instance K times in order to get an estimate $\hat{Q}$ of $\mathbf{m}_q = q$ and output "0" if $\hat{Q} \in U_0$ or "1" if $\hat{Q} \in U_1$. Intuitively, if K is large enough, the estimate $\hat{Q}$ will be good so that if q is far enough from $1/2$, the value of $\hat{Q}$ will always belong to the same set and the reliability of the stabilizer will be large. However, as q gets closer to $1/2$ the probability of getting the "wrong" output increases and the reliability decreases. At $q = 1/2$ we will always have a tie point, independently on K, therefore $\epsilon(1/2) = 1/2$ and $R(1/2) = 0$. Remember that, according to Theorem 1, tie points like this are unavoidable and this prevents achieving global stability.

The next step in the design of (η, δ)-stable stabilizer is choosing the width Δ of the "forbidden strip" S_Δ. In the specific case at hand the border of sets U_i is the single point $q = 1/2$, so that the forbidden strip is a segment around $1/2$, namely $S_\Delta = 1/2 + [-\Delta, \Delta]$. According to the procedure of Section VI, Δ must be chosen small enough so that the probability of $Q \in S_\Delta$ is not larger than δ. Since we expect Δ small, we can approximate $P[Q \in S_\Delta]$ with $2\Delta f_Q(1/2)$ and obtain

$$\Delta = \frac{\delta}{2 f_Q(1/2)} \tag{29}$$

Equation (29) shows that if $f_Q(1/2)$ is small (e.g.., if f_Q is concentrated around 0 and 1), we can use, for the same δ, a larger forbidden zone S_Δ around $1/2$. This will reflect itself in a smaller value of K.

Remark 8.1: In order to gain concreteness, it is worth specializing (29) to a real case. We will refer to the case of SRAM RPUC since its statistical description is derived in [6]. More precisely, in [6] it is shown that the density f_Q and distribution F_Q are

$$f_Q(x; \lambda) = \frac{\lambda \cdot \phi(\lambda \Phi^{-1}(x))}{\phi(\Phi^{-1}(x))} \tag{30a}$$

$$F_Q(x; \lambda) = \Phi(\lambda \Phi^{-1}(x)) \tag{30b}$$

where λ is a parameter, and ϕ and Φ are, respectively, the probability density function and distribution of $\mathscr{N}(0, 1)$.[3]

The parameter λ changes the shape of the density, smaller values of λ lower the part around $x = 1/2$ and make f_q more concentrated around 0 and 1. See Fig. 7 for few examples. In [6] the value $\lambda = 0.065$ provides a good model for experimental measurements. Since $f_q(1/2; \lambda) = \lambda$, (29) specializes to

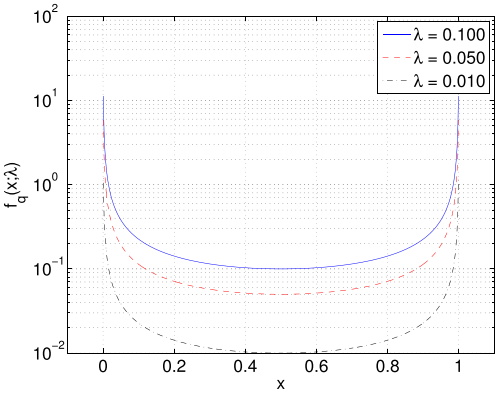

FIGURE 7. Examples of density $f_q(x; \lambda)$ for some values of λ.

$$\Delta = \delta/(2\lambda) \approx 7.7 \cdot \delta \tag{31}$$

Finally, it is worth mentioning that the Gaussian-based model used here for SRAM turns out to be very accurate for other PUFs as well [33], and therefore the considerations made here can be repeated in a wider context.

The last step is choosing K so that the stability $\epsilon(q)$ is at least η for those q that do not belong to the forbidden strip S_Δ, that is,

$$q \notin S_\Delta \Rightarrow \epsilon(q) \geq \eta \tag{32}$$

It is clear that the most critical case is when q is on the border of S_Δ, that is, $q = 1/2 \pm \Delta$. Therefore, if we choose K so that $\epsilon(1/2 \pm \Delta) \geq \eta$, condition (32) will be automatically satisfied.

Equation (23) in Lemma 5 shows a possible choice for K. Actually, (23) is a very pessimistic estimate since it uses the generalized Chebyshev's inequality (see Appendix B) that provides a pessimistic bound. Since, in the very specific case at hand, $\hat{Q}$ can be approximated as a Gaussian random variable $\mathscr{N}(q, q(1-q)/K)$, one can exploit this approximation to get an approximate, but more realistic value of K. By exploiting the Gaussian approximation, it is not difficult to deduce

$$\epsilon(q) \approx \Phi\left(\sqrt{K} \frac{|q - 1/2|}{\sqrt{q(1-q)}}\right) \tag{33}$$

From (33) it is easy to obtain a constraint for K that grants that the stabilizer is (η, δ)-stable

$$\begin{aligned} K &\geq \left\lceil \left(\frac{1}{4\Delta^2} - 1\right) \left(\Phi^{-1}(\eta)\right)^2 \right\rceil \\ &= \left\lceil \left[\left(\frac{f_Q(1/2)}{\delta}\right)^2 - 1 \right] \left(\Phi^{-1}(\eta)\right)^2 \right\rceil \end{aligned} \tag{34}$$

Remark 8.2: Note that (34) shows the three main "ingredients" of the stabilizer: the required performance

[3]Functions (30) has been obtained from the ones in [6] with the substitutions $\lambda_1 = \lambda$ and $\lambda_2 = 0$.

(represented by δ and η) and the behavior of the RPUC (represented by $f_Q(1/2)$). Note also that K grows quadratically with $f_Q(1/2)$.

Example 7: Fig. 8(a) shows $\epsilon(q)$ for a stabilizer (0.99, 0.005)-stable for an RPUC based on an SRAM whose statistical parameter q is distributed according to (30a) with $\lambda = 0.05$. The number of required iterations is $K = 536$ and $\Delta \approx \delta/(2\lambda) = 0.05$. The horizontal dashed line in the top plot corresponds to a stability equal to $\eta = 0.99$; the probability that $\epsilon(q)$ is lower than η is the probability that q belongs to the shadowed area in the bottom plot of Fig. 8(a).

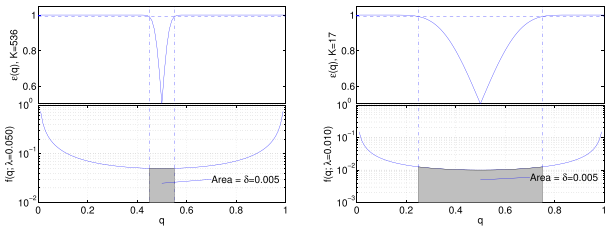

FIGURE 8. Stability vs. q of a (0.99, 0.005)-stable stabilizer for (a) $\lambda = 0.05$ and (b) $\lambda = 0.01$.

The required number of queries is very sensitive to the value of λ. For example, if it was $\lambda = 0.01$ a stabilizer with the same performance would require $\Delta = 0.25$ and $K = 17$. See Fig. 8(b).

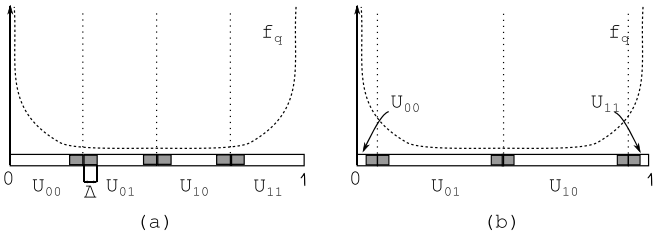

FIGURE 9. (a) Example of partition that allows to extract two bits from a single binary cell, together with the density f_q of q associated with the specific case of SRAM [6]. Note that the probability of getting "00" or "11" is larger than the probability of getting "01" or "10," so that the outcomes are not equiprobable. (b) Different partitioning that gives rise to equiprobable outputs.

1) GETTING 2 BITS FROM A SINGLE BINARY CELL

Note that Theorem 2 poses no restriction on the cardinality of the partition, so that one can split $\mathscr{M} = [0, 1]$ into four pieces and label them with bit pairs "00," "01," "10" and "11." Fig. 9 shows an example where the four sets have been chosen of equal size, together with a possible S_Δ. Note, however that the resulting stabilizer is not balanced since the pairs "00" and "11" are more probable. A different, balanced partition is shown in Fig. 9(b). Note, however, that while in the single bit case one has a balanced stabilizer as soon as the density is symmetric, in this case a change in the parameter λ could cause an unbalance in the stabilizer.

2) THE CASE OF TERNARY CELLS

In order to emphasize the generality of the results given here, it is worth to consider briefly an example with a two-dimensional non-separable parameter set. Fig. 10 shows the parameter set $\mathscr{Q}$ for a ternary cell, together with a partitioning of $\mathscr{Q}$ in four sets that will allow to extract two-bit words from the ternary cell. The forbidden band S_Δ is shown too.

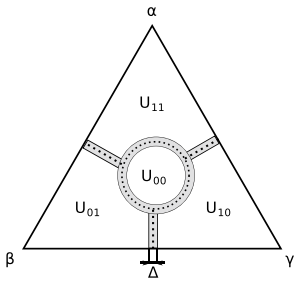

FIGURE 10. Example of partition associated with stabilizer that produces two bits from a ternary RPUC.

In this case the stabilizer would query the RPUC K times and compute the relative frequencies of the three symbols. The vector of the computed frequencies will correspond to a point of the triangle in Fig. 10 and the output of the stabilizer will be the binary word associated with the corresponding set U_{ij}. The determination of Δ and K from δ and η can be done similarly to the binary cell case.

B. THE QUASI-GLOBAL STABILITY APPROACH

As shown in Example 7, depending on the characteristic of the RPUC and the reliability required to the stabilizer, the number of queries to the RPUC can be quite large. An alternative approach that requires a smaller number of queries is the approach described in Section VII, that is, testing the cells at the first turn-on and disabling those that are not reliable enough. This approach has the drawback that it requires a surplus of cells in order to get a working device even with some cells disabled.

A possible approach to implementing this solution is as follows. Design parameters are: the number of queries K that we are willing to do at run-time, the desired minimal stability η, the ID size L in bits and the the probability P_{disc} of discarding a device because too many cells are unreliable (so that the yield is $1 - P_{\text{disc}}$). Note that in this case there is no parameter δ since we are considering the quasi-global stability case of Section VII. Our goal is to find the number of additional cells G and the parameters N and D used in Algorithm 2.

The first step is to determine the value of Δ. In the current case Δ is not determined by δ, but by the fact that we are willing to query the device at most K times. This implies that a reliable cell is a cell whose q is far enough from 1/2 so that both q and its estimate $\hat{Q}$ (obtained averaging K results) will belong to the same set U_i with large probability. A suitable value of Δ can be easily obtained by inverting (34).

From the knowledge of Δ we can determine the probability that a single cell is unreliable

$$p = P[\text{UNREL}] = \int_{1/2-\Delta}^{1/2+\Delta} f_Q(x)dx \approx 2\Delta f_Q(1/2) \qquad (35)$$

where the approximation is valid if Δ is small.

In order to determine the number of surplus cells G, observe that a device is discarded if more than G out of $L+G$ cells are unreliable and that the probability of this event must be smaller than P_{disc}. Since the number of unreliable cells is a binomial r.v. with $L+G$ trials and success probability p in (35) it follows that G must satisfy

$$P[\mathscr{B}(L+G,p) > G] \leq P_{\text{disc}} \qquad (36)$$

The law of large numbers implies that the left hand of (36) goes to zero when G goes to infinity, so a value of G that satisfies (36) exists and it can be found numerically.

Remark 8.3: Note that also with this approach the cost (in this case in terms of surplus cells) depends mainly on the behavior of f_Q around $1/2$: if f_Q is small around $1/2$, probability (35) is small and the number of required surplus cells G is small as well. The opposite reasoning holds for $f_Q(1/2)$ large.

IX. CONCLUSIONS

In this paper, we analyzed the theoretical limits of helper-less stabilizers. We proved three major results: (i) perfect global stability is unachievable, however (ii) we can make as small as desired the probability of a device with bad stability and (iii) quasi-global stability can be achieved by recognizing, with arbitrary reliability, the bad devices at production time and discarding them. The proofs of the second and third result are given in a constructive way. Future research direction will explore the minimum complexity that an helper-less stabilizer requires to guarantee a given performance.

APPENDIX A
BASIC SUMMARY OF TOPOLOGY

In order to make the results in this paper as general as possible, we use some notions of topology. In order to improve the paper accessibility, in this section we give a brief recall of few basic notions of topology. It is not our intention to write a tutorial, for a more detailed exposition the reader is referred to one of the many books available (e.g., [30]).

The basic idea in topology is to extend concepts like open/closed sets, compactness, limits, continuity, and so on, to a setting much more general than the usual $\mathbb{R}^N$.

The cornerstone concept in topology is the idea of *open set*. While in $\mathbb{R}^N$ a set S is said to be open if every point $x \in S$ has a neighborhood contained in S, in a general topological context one defines the open sets S by specifying their collection.

More precisely, if S is any set, a *topology* $\mathscr{T}_S$ for S is a class of subsets of S (the *open sets* of S) that satisfies the following properties: (i) both $\emptyset$ and S are open (i.e., they belong to $\mathscr{T}_S$,) (ii) the *arbitrary* (even non-numerable) union of open sets is open and (iii) the *finite* intersection of open sets is open. Note that the usual definition of open sets of $\mathbb{R}^N$ satisfies those conditions. A set with a topology is usually called a *topological space* [30].

A known result in real analysis is that the complement of an open set is closed, in topology this results is used as a definition, so that a set $A \subseteq S$ is *closed* if its complement belongs to $\mathscr{T}_S$. Concepts like interior, closure and border are readily derived from the concepts of open and closed set.

If $R \subseteq S$ and $\mathscr{T}_S$ is a topology for S, one can define the *subspace topology* $\mathscr{T}_R$ for R by taking the intersection of R with all the elements of S, that is,

$$\mathscr{T}_R = \{A \cap R : A \in \mathscr{T}_S\} \qquad (37)$$

Subspace topologies can be surprising at first. For example, if $S = \mathbb{R}$ and $R = [0, 1]$, the set $[0, 1/2)$ is *open in* R (but not in S!) since it is the intersection of R with (for example) $(-1, 1/2)$ which is open in S.

The generalization of the idea of open set allows us to generalize the idea of *continuous function*. If S and X are two topological spaces, a function $f : S \to X$ is said to be *continuous* if the anti-image of any set open in X is open in S, that is, if $A \in \mathscr{T}_X \Rightarrow f^{-1}(A) \in \mathscr{T}_S$. It is easy to check that this definition reduces itself to the usual notion of continuity when working in $\mathbb{R}^N$.

Finally, we will need the concept of *connected set*. A set S is said to be connected if it *cannot* be written as $S = A \cup B$ where A and B are *disjoint non-empty open sets*. An equivalent definition is that S is connected if the only subsets of S that are both open and closed *in* S are $\emptyset$ and S itself.[4] This definition is slightly weaker than the more common definition of connectedness that requires that there is a path between any pair of points of S (*path connectedness*). A set can be connected without being path connected, see, [30] for some examples.

APPENDIX B
GENERALIZED CHEBYSHEV'S INEQUALITIES

Lemma 6: Let $\mathscr{V}$ be a set with a distance $d : \mathscr{V} \times \mathscr{V} \to \mathbb{R}$ and let V be a random variable assuming values in $\mathscr{V}$. For every $\epsilon > 0$, $v_0 \in \mathscr{V}$ and $k \in \mathbb{N}$ the following inequality holds

$$P[d(V, v_0) \geq \epsilon] = P[V \notin \mathfrak{B}_{v_0,\epsilon}] \leq \frac{\mathbb{E}\left[d(V, v_0)^k\right]}{\epsilon^k} \qquad (38)$$

where $\mathfrak{B}_{v_0,\epsilon}$ denotes the ball of center v_0 and radius ϵ.

Proof: The proof is similar to the proof of Chebyshev's inequality. Define, for notational convenience, $B = \mathfrak{B}^c_{v_0,\epsilon}$. Let χ_B be the characteristic function[5] of B and observe that

$$\forall x \in \mathscr{V} \qquad \chi_B(x) \leq d^k(x, v_0)/\epsilon^k \qquad (39)$$

[4]For example, interval $S = (0, 1)$ is open in $\mathbb{R}$, but it is both open and close in its subspace topology.

[5]$\chi_B(x) = 1$ if $x \in B$ and $\chi_B(x) = 0$ otherwise.

By taking expectations of both sides of (39) and remembering $P[V \notin \mathfrak{B}_{v_0,\epsilon}] = P[V \in B] = \mathbb{E}[\chi_B(V)]$ the thesis follows. □

Corollary 1: Let $\mathscr{V} = \mathbb{R}^N$ and let $\|\cdot\| : \mathscr{V} \to \mathbb{R}$ be the Euclidean norm. Let V be a random variable assuming values in $\mathscr{V}$ and let $\mathbf{m} = [m_1, \ldots, m_N]$ and $\Sigma = [\sigma_1, \ldots, \sigma_N]$ be the vectors with the means and the standard deviations of the components of V. The following inequality holds

$$P[\|X - \mathbf{m}\| \geq \epsilon] \leq \frac{\|\Sigma\|^2}{\epsilon^2} = \frac{\sum_{i=1}^N \sigma_i^2}{\epsilon^2} \tag{40}$$

Proof: Use Lemma 6 with $v_0 = \mathbf{m}$, $d(x, y) = \|x - y\|$, $k = 2$ and observe that $\mathbb{E}\left[d(X, \mathbf{m})^2\right]/\epsilon^2 = \|\Sigma\|^2/\epsilon^2$. □

APPENDIX C
PROOFS

Proof of Lemma 2: The fact that $\mathfrak{d}_b \geq 0$ is obvious. It remains to prove that $\mathfrak{d}_b = 0$ if and only if $b \in \partial U_{\iota_b}$. One implication (If $b \in \partial U_{\iota_b}$ then $\mathfrak{d}_b = 0$) follows at once from $\partial U_{\iota_b} \subset (\mathrm{Int}(U_{\iota_b}))^c$; in order to prove the other one, we will prove the contrappositive, that is, if $b \notin \partial U_{\iota_b}$, then $\mathfrak{d}_b > 0$. If b belongs to the interior of U_{ι_b} there is, by definition of interior, $r > 0$ such that $\mathfrak{B}_{b,r} \subseteq \mathrm{Int}(U_{\iota_b})$ and this implies $\mathfrak{d}_b \geq r > 0$. □

Proof of Lemma 3: Suppose first that $U_{\iota_a} \neq U_{\iota_b}$, that is, a and b belong to different components. Since $\|a - b\| \geq \inf_{x \notin \mathrm{Int}(U_{\iota_a})} \|a - x\| = A \geq A - B$, the thesis is true if $U_{\iota_a} \neq U_{\iota_b}$. Suppose now $U_{\iota_a} = U_{\iota_b}$ and let $c \notin \mathrm{Int}(U_{\iota_b})$ such that $\|b - c\| = \mathfrak{d}_b = B$. If $\|a - b\| < A - B$, then

$$\|a - c\| \leq \|a - b\| + \|b - c\| < (A - B) + B = A \tag{41}$$

Note also that

$$\mathfrak{d}_a = \min_{x \notin \mathrm{Int}(U_{\iota_a})} \|a - x\| \leq \|a - c\| \tag{42}$$

Equations (41) and (42) imply $\mathfrak{d}_a < A$ and this is impossible. □

Proof of Lemma 4: Let H denote the real-valued random variable defined as $H := \mathfrak{d}_{\mathbf{m}_Q}$ and observe that $P[\mathbf{m}_Q \in S_\Delta] = P[\mathfrak{d}_{\mathbf{m}_Q} \leq \Delta] = F_h(\Delta)$, where F_h is the distribution of H. Note that $F_h(0) = 0$ since

$$F_h(0) = P[\mathbf{m}_Q \in S_0] = P[\mathfrak{d}_{\mathbf{m}_Q} = 0] = P[\mathbf{m}_Q \in \cup_{i \in \mathbb{I}} \partial U_i] = 0$$

where Lemma 2 and hypothesis (18) have been used. F_h is a distribution, so it is right-continuous and for every δ there is Δ^+ such that $x \in (0, \Delta^+) \Rightarrow \delta > F_h(x) - F_h(0) = F_h(x) = P[\mathbf{m}_Q \in S_{\Delta^+}]$. Any $\Delta < \Delta^+$ satisfies constraint (22). □

Proof of Lemma 5: In order to prove Lemma 5 we need to find a lower bound for $P_q[\iota_{\mathbf{m}_q}]$. Observe that if $\|\mu - \mathbf{m}_q\| < \mathfrak{d}_{\mathbf{m}_q}$, then the stabilizer will produce the output symbol $\iota_{\mathbf{m}_q}$, since by Definition 8, $\|\mu - \mathbf{m}_q\| < \mathfrak{d}_{\mathbf{m}_q}$ implies $\mu \in U_{\iota_{\mathbf{m}_q}}$. It follows that

$$P\left[I = \iota_{\mathbf{m}_q} \mid Q = q\right] \geq 1 - P[\|\mu - \mathbf{m}_q\| \geq \mathfrak{d}_{\mathbf{m}_q}] \tag{43}$$

The last probability is something that we can upper bound by using the extended version of the Chebyshev's inequality given in Lemma 6 and Corollary 1. Suppose q is such that $\mathbf{m}_q \notin S_\Delta$. By applying Corollary 1 to the last term of (43), one obtains

$$P[\|\mu - \mathbf{m}_q\| \geq \mathfrak{d}_{\mathbf{m}_q}] \leq \frac{1}{K}\frac{\|\Sigma_q\|^2}{\Delta^2} \leq \frac{1}{K}\frac{\Sigma_{\max}^2}{\Delta^2} \tag{44}$$

By using (44) in (43) one obtains $P[I = \iota_q \mid Q = q] \geq 1 - \frac{1}{K}\frac{\Sigma_{\max}^2}{\Delta^2}$. Any $K \geq \frac{\Sigma_{\max}^2}{\Delta^2(1-\eta)}$ satisfies constraint $P[I = \iota_q \mid Q = q] \geq \eta$. □

Proof of Property 7.1: Since $\mathbf{m}$ is continuous, it suffices to show that $S_\Delta^c = \mathscr{M} \setminus S_\Delta$ is not connected since $\mathbf{m}(\mathscr{Q} \setminus \mathbf{m}^{-1}(S_\Delta)) = \mathscr{M} \setminus S_\Delta$ and the continuous image of a connected set is necessarily connected. Since

$$\mathscr{M} \cap S_\Delta^c = \bigcup_{n \in \mathbb{I}} V_n = V_i \cup \left[\left(\bigcup_{n \neq i,j} V_n\right) \cup V_j\right] = V_i \cup \mathfrak{U} \tag{45}$$

if we prove that every set V_n is open it will follow that $\mathscr{M} \cap S_\Delta^c$ is the union of the non-empty and disjoint open sets V_i and $\mathfrak{U} \supseteq V_j$, showing that $\mathscr{M} \cap S_\Delta^c$ is not connected.

In order to show that V_n is open we are going to show that

$$V_n = U_n \cap S_\Delta^c = \mathrm{Int}(U_n) \cap S_\Delta^c \tag{46}$$

by observing that $U_n = \mathrm{Int}(U_n) \cup B$ with $B \subseteq \partial U_n \subset S_\Delta$, where the latter inclusion descends from the fact that $x \in \partial U_n$ implies $\mathfrak{d}_x = 0$ which in turn implies $x \in S_\Delta$. It follows that $U_n \cap S_\Delta^c = (B \cup \mathrm{Int}(U_n)) \cap S_\Delta^c = (B \cap S_\Delta^c) \cup (\mathrm{Int}(U_n) \cap S_\Delta^c) = \mathrm{Int}(U_n) \cap S_\Delta^c$. □

A. PROOF OF THEOREM 1

Remember notation $P_q[A] = P[\mathscr{S}(V) \in A \mid Q = q]$, simplified to $P_q[i]$ when $A = \{i\}$. In order to prove Theorem 1 we will prove the existence of tie points by showing that there are at least two non-disjoint winning regions.

Given $i, j \in \mathbb{I}$, let function $d_{i,j} : \mathscr{Q} \to \mathbb{R}$ be defined as

$$d_{i,j}(q) := P_q[i] - P_q[j] \tag{47}$$

Lemma 7: For every $i, j \in \mathbb{I}$, function $d_{i,j}$ is continuous.

Proof: By hypothesis, for every event A, map $q \mapsto P_q[A]$ is continuous (see Definition 1). □

Lemma 8: For every $i \in \mathbb{I}$, the winning region can be expressed as

$$\mathscr{W}_i = \{q \in \mathscr{Q} : d_{i,j}(q) \geq 0 \quad \forall j \in \mathbb{I}\} \tag{48}$$

Set $\mathscr{W}_i$ is closed in $\mathscr{Q}$ and its interior $\mathrm{Int}(\mathscr{W}_i)$ is

$$\mathrm{Int}(\mathscr{W}_i) = \{q \in \mathscr{Q} : d_{i,j}(q) > 0 \quad \forall j \in \mathbb{I}\} \tag{49}$$

Proof: Equation (48) follows from definition (9) and (47). Set $\mathscr{W}_i$ is sclosed since, according to (48), $\mathscr{W}_i = \bigcap_{j \neq i} d_{i,j}^{-1}(\mathbb{R}_{\geq 0})$ and every term of the intersection is closed, being the inverse image of the closed set $\mathbb{R}_{\geq 0}$. Finally, (49) follows by observing that the right hand side of (49) is the inverse image of $\mathbb{R}_{>0}^N$. □

A fact that will be exploited in the proof is that the collection of the winning sets $\mathscr{W}_i$ is *almost* a partition of $\mathscr{Q}$.

Lemma 9: The following equalities hold

$$\mathscr{Q} = \cup_{i\in\mathbb{I}}\mathscr{W}_i \tag{50a}$$

$$i \neq j \Rightarrow \mathscr{W}_i \cap \mathrm{Int}(\mathscr{W}_j) = \emptyset, \quad i,j \in \mathbb{I} \tag{50b}$$

Proof: In order to prove (50a), consider $q \in \mathscr{Q}$, let $i \in w(q)$ and observe that $q \in \mathscr{W}_{w(i)}$, therefore q belongs to the right hand side of (50a), so $\mathscr{Q} \subseteq \bigcup_{i\in\mathbb{I}} \mathscr{W}_i$. Since, obviously, $\mathscr{W}_i \subseteq \mathscr{Q}$ for every $i \in \mathbb{I}$, (50a) follows.

In order to prove (50b) observe that if one could find $q \in \mathscr{W}_i \cap \mathrm{Int}(\mathscr{W}_j)$, then it would be $P_q[i] \geq P_q[j]$ (since $q \in \mathscr{W}_i$) and $P_q[j] > P_q[i]$ (since $q \in \mathrm{Int}(\mathscr{W}_j)$), which is absurd. □

Lemma 10: If $\mathscr{Q}$ is connected, $|\mathbb{I}| > 1$ and every $\mathscr{W}_i$, $i \in \mathbb{I}$ is not empty, then there are at least two output symbols $i, j \in \mathbb{I}$, $i \neq j$, such that their winning regions are not disjoint, that is, $\mathscr{W}_i \cap \mathscr{W}_j \neq \emptyset$.

Proof: We will show the contrappositive: if regions $\mathscr{W}_i$ are disjoint, then $\mathscr{Q}$ is not connected. Let i be any element of $\mathbb{I}$ and define set $R_i = \bigcup_{j\in\mathbb{I}\setminus\{i\}} \mathscr{W}_j$. Observe that $R_i \cup \mathscr{W}_i = \mathscr{Q}$ because of (50a) and that $R_i \cap \mathscr{W}_i = \emptyset$ because all the $\mathscr{W}_i$ are disjoint. Since R_i and $\mathscr{W}_i$ are closed, $\mathscr{Q}$ is disconnected.

Finally we can prove Theorem 1

Proof of Theorem 1: Since by hypothesis $\mathscr{Q}$ is connected and the conditioner is not redundant, there are at least two output symbols $i, j \in \mathbb{I}$, $i \neq j$, such that their winning regions are not disjoint.

Let $q \in \mathscr{W}_i \cap \mathscr{W}_j$. Since $q \in \mathscr{W}_i$ we know that $\epsilon(q) = P_q[i] \geq P_q[j]$; but also, since $q \in \mathscr{W}_j$ we know that $\epsilon(q) = P_q[j] \geq P_q[i]$. It follows that

$$\epsilon(q) = P_q[j] = P_q[i] \tag{51}$$

that is, q is a tie point. Equation (51) implies $\epsilon(q) \leq 1/2$ which in turn implies $\eta = \inf_{q\in\mathscr{Q}} \epsilon(q) \leq 1/2$. □

B. PROOF OF THEOREM 3

1) BOUNDING THE FALSE NEGATIVE RATE

Observe that

$$P[\omega \notin S_D \,|\, \mathbf{m}_Q \in S_\Delta] = \frac{\int_{\mathbf{m}^{-1}(S_\Delta)} P[\omega \notin S_D \,|\, Q=q]\, f_Q(q)\, dq}{\int_{\mathbf{m}^{-1}(S_\Delta)} f_Q(q)\, dq} \leq \max_{q\in\mathbf{m}^{-1}(S_\Delta)} P[\omega \notin S_D \mid Q=q] \tag{52}$$

Therefore, it suffices to show that $P[\omega \notin S_D \mid Q = q]$ can be upper bounded with a bound that can be made as small as desired.

Lemma 11: Let $r_- > 0$ and $D > \Delta$. If $N \geq N_0 := \lceil \Sigma^2_{\max}/[r_-(D-\Delta)^2] \rceil$ then $\forall q \in \mathbf{m}^{-1}(S_\Delta)$, $P[\omega \notin S_D \mid Q = q] < r_-$.

Proof of Lemma 11: Let q such that $\mathfrak{d}_{\mathbf{m}_q} \leq \Delta$ and observe that

$$\begin{aligned} P[\omega \notin S_D \mid Q=q] &= P[\mathfrak{d}_\omega > D \mid Q=q] \\ &\leq P[\|\omega - \mathbf{m}_q\| > D - \mathfrak{d}_{\mathbf{m}_q} \mid Q=q] \\ &\leq P[\|\omega - \mathbf{m}_q\| > D - \Delta \mid Q=q] \\ &\leq \frac{\|\Sigma_q\|^2/N}{(D-\Delta)^2} \leq \frac{\Sigma^2_{\max}}{N(D-\Delta)^2} \end{aligned}$$

where Lemma 3 has been exploited. □

2) BOUNDING THE FALSE POSITIVE RATE

Let $B = S_{2D-\Delta} \setminus S_\Delta$ and $C = S^c_\Delta \setminus B = S^c_{2D-\Delta}$ and observe that

$$\begin{aligned} P_+ &= P[\omega \in S_D \mid \mathbf{m}_Q \in S^c_\Delta] = \frac{P[\omega \in S_D, \mathbf{m}_Q \in S^c_\Delta]}{P[\mathbf{m}_Q \in S^c_\Delta]} \\ &= \frac{P[\omega \in S_D, \mathbf{m}_Q \in B]}{P[\mathbf{m}_Q \in S^c_\Delta]} + \frac{P[\omega \in S_D, \mathbf{m}_Q \in C]}{P[\mathbf{m}_Q \in S^c_\Delta]} \\ &\leq \frac{P[\mathbf{m}_Q \in B]}{P[\mathbf{m}_Q \in S^c_\Delta]} + \frac{P[\omega \in S_D, \mathbf{m}_Q \in C]}{P[\mathbf{m}_Q \in S^c_\Delta]} \\ &\leq P[\mathbf{m}_Q \in B \mid \mathbf{m}_Q \in S^c_\Delta] \\ &\quad + P[\omega \in S_D \mid \mathbf{m}_Q \in C] C, B \subseteq S^c_\Delta \\ &\leq P[\mathbf{m}_Q \in B \mid \mathbf{m}_Q \in S^c_\Delta] + \sup_{q\in\mathbf{m}^{-1}(C)} P[\omega \in S_D \mid Q=q] \end{aligned} \tag{53}$$

It suffices to show that both addends in the last part of (53) can be made small as desired. This is shown in Lemma 12 and Lemma 13.

Lemma 12: If $P[\mathfrak{d}_{\mathbf{m}_Q} = \Delta] = 0$, then for every $u > 0$ it is possible to find $D > \Delta$ such that

$$P[\mathbf{m}_Q \in B \mid \mathbf{m}_Q \in S^c_\Delta] < u \tag{54}$$

Proof: Let $H := \mathfrak{d}_{\mathbf{m}_Q} - \Delta$ and observe that

$$P[\mathbf{m}_Q \in B \mid \mathbf{m}_Q \in S^c_\Delta] = P[H \leq 2D \mid H > 0] = F_H(2D) \tag{55}$$

where F_H denotes the distribution of H conditioned by $H > 0$. Note that $F_H(x) = 0$ for $x < 0$ and $F_H(0) = 0$ since

$$F_H(0) = P[H \leq 0 \mid H \geq 0] = P[H = 0] = P[\mathfrak{d}_{\mathbf{m}_Q} = \Delta] \tag{56}$$

and the last probability is zero by hypothesis. Since F_H is a distribution, it is right continuous so for every $u > 0$ it is possible to find $v > 0$ such that $F_H(v) - F_H(0) = F_H(v) < u$. Choosing $D = v/2$ proves the claim. □

Lemma 13: Let $D > \Delta$ and $u > 0$. If $N \geq N_1 := \lceil \Sigma^2_{max}/[N(D-\Delta)^2] \rceil$, then $\forall q \in \mathbf{m}^{-1}(C)$, $P[\omega \in S_D \mid Q = q] < u$ which implies $\sup_{q\in\mathbf{m}^{-1}(C)} P[\omega \in S_D \mid Q = q] < u$

Proof: Let $q \in C$, that is, $\mathfrak{d}_{\mathbf{m}_q} \geq 2D - \Delta$. Observe that

$$\begin{aligned} P[\omega \in S_D \mid Q=q] &= P[\mathfrak{d}_\omega \leq D \mid Q=q] \\ &\leq P[\|\omega - \mathbf{m}_q\| \geq \mathfrak{d}_{\mathbf{m}_q} - D \mid Q=q] \\ &\leq P[\|\omega - \mathbf{m}_q\| \geq D - \Delta \mid Q=q] \\ &\leq \frac{\|\Sigma_q\|^2}{N(D-\Delta)^2} \leq \frac{\Sigma^2_{\max}}{N(D-\Delta)^2} \end{aligned}$$

□

Proof of Theorem 3: Split r_+ as $r_+ = r^a_+ + r^b_+$, r^a_+, $r^b_+ > 0$ (for example, $r^a_+ = r^b_+ = r_+/2$). Using Lemma 12 find $D > \Delta$ such that $P[\mathbf{m}_Q \in B \mid \mathbf{m}_Q \in S^c_\Delta] < r^a_+$; successively, use Lemma 13 to find the minimum N_1 such that $\sup_{q\in\mathbf{m}^{-1}(C)} P[\omega \in S_D \mid Q = q] < r^b_+$. Finally, use Lemma 11 to find N_0 and choose $N = \max(N_0, N_1)$ to get the desired verifier. □

REFERENCES

[1] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. 9th ACM Conf. Comput. Commun. Secur. (CCS)*, New York, NY, USA, 2002, pp. 148–160.

[2] D. Lim, J. W. Lee, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, "Extracting secret keys from integrated circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 10, pp. 1200–1205, Oct. 2005.

[3] D. Lim, "Extracting secret keys from integrated circuits," M.S. thesis, Dept. Elect. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, USA, May 2004.

[4] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. 44th ACM/IEEE Design Autom. Conf. (DAC)*, Jun. 2007, pp. 9–14.

[5] D. E. Holcomb, W. P. Burleson, and K. Fu, "Initial SRAM state as a fingerprint and source of true random numbers for RFID tags," in *Proc. Conf. RFID Secur.*, 2007.

[6] R. Maes, P. Tuyls, and I. Verbauwhede, "A soft decision helper data algorithm for SRAM PUFs," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun./Jul. 2009, pp. 2101–2105.

[7] Z. Paral and S. Devadas, "Reliable and efficient PUF-based key generation using pattern matching," in *Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust (HOST)*, Jun. 2011, pp. 128–133.

[8] O. Goldreich, S. Goldwasser, and S. Micali, "How to construct random functions," *J. ACM*, vol. 33, no. 4, pp. 792–807, Aug. 1986.

[9] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *Proc. 1st ACM Conf. Comput. Commun. Secur. (CCS)*, New York, NY, USA, 1993, pp. 62–73.

[10] A. Yaglom. (2011). *Random Function*. [Online]. Available: http://www.encyclopediaofmath.org/index.php?title=Random_function&oldid=13830

[11] J. Delvaux and I. Verbauwhede, "Fault injection modeling attacks on 65nm arbiter and RO sum PUFs via environmental changes," International Association for Cryptologic Res., Tech. Rep. 2013/619, 2013. [Online]. Available: http://eprint.iacr.org/

[12] D. Merli, D. Schuster, F. Stumpf, and G. Sigl, "Semi-invasive EM attack on FPGA RO PUFs and countermeasures," in *Proc. Workshop Embedded Syst. Secur. (WESS)*, New York, NY, USA, 2011, pp. 2:1–2:9.

[13] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "FPGA intrinsic PUFs and their use for IP protection," in *Proc. Cryptograph. Hardw. Embedded Syst. (CHES)*, Sep. 2007, pp. 63–80.

[14] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Science*, vol. 297, no. 5589, pp. 2026–2030, Sep. 2002.

[15] R. Maes and I. Verbauwhede, "Physically unclonable functions: A study on the state of the art and future research directions," in *Proc. Towards Hardw.-Intrinsic Secur.-Found. Pract.*, 2010, pp. 3–37.

[16] R. Bernardini and R. Rinaldo, "A simple and reliable cell for single bit physically unclonable constants," in *Proc. Austrochip*, Graz, Austria, Oct. 2014, pp. 1–6.

[17] Y. Su, J. Holleman, and B. Otis, "A 1.6pJ/bit 96% stable chip-ID generating circuit using process variations," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2007, pp. 406–611.

[18] N. Beckmann and M. Potkonjak, "Hardware-based public-key cryptography with public physically unclonable functions," in *Proc. 11th Int. Workshop Inf. Hiding (IH)*, Darmstadt, Germany, Jun. 2009, pp. 206–220.

[19] M. Potkonjak, S. Meguerdichian, A. Nahapetian, and S. Wei, "Differential public physically unclonable functions: Architecture and applications," in *Proc. 48th ACM/EDAC/IEEE Design Autom. Conf. (DAC)*, San Diego, CA, USA, Jun. 2011, pp. 242–247.

[20] J. Delvaux, D. Gu, D. Schellekens, and I. Verbauwhede, "Secure lightweight entity authentication with strong PUFs: Mission impossible?" in *Proc. 16th Int. Workshop Cryptograph. Hardw. Embedded Syst. (CHES)*, Busan, Korea, Sep. 2014, pp. 451–475.

[21] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 3027, C. Cachin and J. L. Camenisch, Eds. Berlin, Germany: Springer-Verlag, 2004, pp. 523–540.

[22] B. Škorić and N. de Vreede, "The spammed code offset method," International Association for Cryptologic Res., Tech. Rep. 2013/527, 2013. [Online]. Available: http://eprint.iacr.org/

[23] J. Delvaux and I. Verbauwhede, "Attacking PUF-based pattern matching key generators via helper data manipulation," International Association for Cryptologic Res., Tech. Rep. 2013/566, 2013. [Online]. Available: http://eprint.iacr.org/

[24] J. Delvaux and I. Verbauwhede, "Key-recovery attacks on various RO PUF constructions via helper data manipulation," in *Proc. Design, Autom. Test Eur. Conf. Exhibit. (DATE)*, Dresden, Germany, Mar. 2014, pp. 1–6.

[25] D. Merli, D. Schuster, F. Stumpf, and G. Sigl, "Side-channel analysis of PUFs and fuzzy extractors," in *Proc. 4th Int. Conf. Trust Trustworthy Comput. (TRUST)*, Pittsburgh, PA, USA, Jun. 2011, pp. 33–47.

[26] P. Cappelletti, C. Golla, P. Olivo, and E. Zanoni, Eds., *Flash Memories*. Norwell, MA, USA: Kluwer, 1999.

[27] M.-D. Yu and S. Devadas, "Secure and robust error correction for physical unclonable functions," *IEEE Des. Test Comput.*, vol. 27, no. 1, pp. 48–65, Jan./Feb. 2010.

[28] E. Öztürk, G. Hammouri, and B. Sunar, "Towards robust low cost authentication for pervasive devices," in *Proc. 6th Annu. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom)*, Hong Kong, Mar. 2008, pp. 170–178.

[29] R. Bernardini and R. Rinaldo, "Helper-less physically unclonable functions and chip authentication," in *Proc. ICASSP*, Florence, Italy, May 2014, pp. 8193–8197.

[30] J. R. Munkres, *Topology: A First Course*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1974.

[31] B. Zhang, A. Arapostathis, S. Nassif, and M. Orshansky, "Analytical modeling of SRAM dynamic stability," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2006, pp. 315–322.

[32] F. Armknecht, R. Maes, A.-R. Sadeghi, B. Sunar, and P. Tuyls, "Memory leakage-resilient encryption based on physically unclonable functions," in *Proc. 15th Int. Conf. Theory Appl. Cryptol. Inf. Secur. Adv. Cryptol. (ASIACRYPT)*, Tokyo, Japan, Dec. 2009, pp. 685–702.

[33] R. Maes, "An accurate probabilistic reliability model for silicon PUFs," in *Proc. 15th Int. Workshop Cryptograph. Hardw. Embedded Syst. (CHES)*, Santa Barbara, CA, USA, Aug. 2013, pp. 73–89.

RICCARDO BERNARDINI was born in Genova, Italy, in 1964. He received the Laurea degree in electronics engineering from the University of Padua, Padua, Italy, in 1990, and the Ph.D. degree in filterbanks from AT&T Bell Labs, Murray Hill, NJ, USA, with Prof. Jelena Kovačević. From 1996 to 1997, he was with the École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, as a Post-Doctoral Fellow with Prof. Martin Vetterli. He is currently an Aggregate Professor with the Dipartimento di Ingegneria Elettrica, Gestionale e Meccanica, University of Udine, Udine, Italy. He worked in the areas of multidimensional signal processing, wavelets, filter banks, multimedia coding, robust transmission, bioengineering, chaotic systems, peer-to-peer streaming, and some security-related areas, such as random number generation, physical unclonable functions, and embedding random permutations on chips. He has been involved in tenths of projects, from regional ones up to European-level ones, sometimes as a Partner and Principal Investigator/Coordinator.

ROBERTO RINALDO received the Laurea degree in electronics engineering from the University of Padua, Padua, Italy, in 1987, the M.S. degree from the University of California at Berkeley, Berkeley, CA, USA, in 1992, and the Ph.D. degree in electrical engineering from the University of Padua. In 1992, he joined the Dipartimento di Elettronica e Informatica, University of Padova. Since 2001, he has been an Associate Professor with the Dipartimento di Ingegneria Elettrica, Gestionale e Meccanica, University of Udine, Udine, Italy, where he has been a Full Professor with the Dipartimento di Ingegneria Elettrica, Gestionale e Meccanica since 2003. He is currently the Director of the Dipartimento di Ingegneria Elettrica, Gestionale e Meccanica. He has authored approximately 100 publications, most of which in the IEEE journals and conferences. His interests are in the field of multidimensional signal processing, video signal coding, fractal theory, random number generation, and physically unclonable functions.