Spatial Correlation Analysis on Physical Unclonable Functions

Florian Wilde, Berndt M. Gammel, and Michael Pehl, *Member, IEEE*

***Abstract*— The evaluation of the quality of physical unclonable functions (PUFs) is still under research. Most state-of-the-art metrics are found to measure only bias, while correlations, e.g., within the PUF response of a single device, remain unnoticed. In this paper, we introduce spatial autocorrelation analysis (SPACA), which is used in other fields of research, as a method to reveal correlations in the response of single-challenge PUFs. The presented statistics—Moran's I, Geary's c, and Join Count statistic—can be used to test the quality of an implementation using a set of sample devices as well as monitor ongoing production. Their results are also important for selecting an appropriate post-processing of the raw PUF response. Experiments on data sets from three different PUF implementations on field-programmable gate array and application-specific integrated circuit using SPACA show the capabilities of the introduced statistics. An efficient implementation of SPACA in MATLAB was developed to conduct the experiments. We discuss how SPACA can be used in practical testing and suggest to consider SPACA for a future test suite for PUFs.**

***Index Terms*— Physical unclonable function.**

I. Introduction

SECURED storage of a secret key and checking the authenticity of a device are two important tasks for hardware components used in security applications. PUFs are security primitives dedicated to solve these tasks: They derive a reproducible secret from measurements of hardware intrinsic manufacturing variations. Thus, the secret is inextricably linked with the device that contains the PUF, making it a candidate to base authentication upon it. In reasonable PUF designs, the secret should be protected from being measured if the PUF is not powered. This would permit to use PUFs for some scenarios, in the case secured non-volatile memory is not available.

Secret refers in this context to the secret output of the PUF – the PUF *responses*. A single response is provided by a PUF instance, i.e. by a circuit for the silicon-based PUFs in this work. Each response can be considered as a random variable that must be unpredictable from the outside but reproducible with low noise during the lifetime of a device.

Manuscript received July 13, 2017; revised October 20, 2017 and November 23, 2017; accepted December 18, 2017. Date of publication January 8, 2018; date of current version February 7, 2018. This work was supported in part by the German Federal Ministry of Education and Research through the Project SIBASE under Grant 01IS13020 and in part by the German Research Foundation (DFG) under Grant SI 2064/1-1. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Tanya Ignatenko. *(Corresponding author: Florian Wilde.)*

F. Wilde and M. Pehl are with the Technical University of Munich (TUM), 80333 Munich, Germany (e-mail: florian.wilde@tum.de).

B. M. Gammel is with Infineon Technologies, 85579 Neubiberg, Germany.

Digital Object Identifier 10.1109/TIFS.2018.2791341

This work focuses on evaluating the unpredictability of single-challenge PUFs. Compared to multi-challenge PUFs, which are PUFs that can be configured by a challenge, such as Arbiter PUFs [1] or Bistable-Ring PUFs [2], single-challenge PUFs refer to PUFs such as current ring-oscillator (RO) PUF implementations [3] or static random access memory (SRAM) PUFs [4], which cannot be reconfigured by a challenge. Also, multi-challenge PUFs with a fixed challenge can be considered single-challenge PUFs. Note that the terms multi- and single-challenge PUFs only takes into account if the intention is to configure a PUF by a challenge. This is a different perspective compared to the often used distinction Weak vs. Strong PUFs [5]: The latter is frequently used to classify PUFs according to the number of challenge-response pairs (CRPs) needed to predict further unknown responses and – in this context – implicitly needs knowledge about the PUF quality.

Two main flaws that reduce the guessing complexity for the response of a single-challenge PUF are bias of and correlation between the bits in the PUF response. The task to evaluate the quality of a random bit sequence derived from a PUF looks similar to the evaluation of a True Random Number Generator (TRNG) in the first place. However, TRNGs measure dynamic processes (noise) during run time to provide randomness, which allows to use a high number of bits (several million) for testing a single instance. PUFs provide randomness extracted from the manufacturing process and the number of available bits for testing a single implementation is the number of PUF responses generated on a device, which is for single-challenge PUFs usually only up to a few thousand. Thus, TRNG test suites cannot be applied directly to test the quality of PUFs. The goal of ongoing research must so be to establish a set of standardized tests that can be used to evaluate the quality of PUFs and allow for the certification of PUFs in the future.

Metrics to analyze bias in PUFs are already available, e.g. [6]–[8]. But these metrics do not take into account correlations between bits in the PUF response. Only few metrics such as joint entropy computed by Context Tree Weighting (CTW) [9] and the conditional entropy and min-entropy in [10] directly cover correlation effects as well as bias in PUFs. However, the CTW approach provides only an upper bound for the entropy of PUFs and requires a large number of devices to derive a sufficiently accurate bound. Furthermore, the results strongly depend on how data is preprocessed, e.g. how the PUF output bits are arranged for the compression. The conditional Shannon entropy and min-entropy in [10] are

metrics to estimate the dependency of the responses of PUFs at a certain position from its neighbours. However, the spatial correlations within a die are not explicitly covered by these conditional entropies and neither an estimate of the significance nor null-hypothesis testing is discussed for these metrics.

Also, machine-learning approaches like in [11] and [12], which are used in modelling attacks on PUFs, consider dependencies like bias and correlation. However, machine-learning approaches are currently used on multi-challenge PUFs to predict unknown PUF responses of a given PUF instance. Therefore, they consider correlations and biases within the responses of a single PUF instance for multiple challenges. This predominantly focuses on correlations in the responses due to architectural weaknesses rather than due to correlations in the manufacturing variations. Machine-learning approaches are currently not used to identify correlations between instances of single-challenge PUFs within a device.

In contrast to the previously mentioned metrics, our focus is on methods that are suitable to find spatial correlations in a single-challenge PUF implementation on a single device and that allow for statistical hypothesis testing to receive a quantitative statement regarding the test result. Therefore, we introduce three metrics from SPACA, which are known from other fields of research, namely *Moran's I* [13], *Geary's c* [14], and the *Join Count statistic* [15]. We propose these methods to analyze spatial autocorrelations in PUFs and suggest SPACA as a candidate for a future test suite for the quality of single-challenge PUFs.

While we prepared this publication, two other papers about SPACA for PUFs appeared: In [16] we presented first results of our current work in an extended abstract. That abstract as well as the current work are based on unpublished initial research from 2011 on the application of all three SPACA methods to PUFs. Compared to this work, the abstract used the same datasets, but did neither contain mathematical nor implementation details and provided results only using Moran's I. Willsch *et al.* [17] also suggested the use of Moran's I, Geary's c, and Join Count statistic to analyze PUFs. But their work only briefly describes Moran's I and because the method is applied on a proprietary application specific integrated circuit (ASIC) PUF design, it is hard to reproduce their results or to generalize the findings onto other types of PUFs.

Other than the canonical autocorrelation function, SPACA tests whether the value of an observed variable at one location is independent of the observed variable values in neighboring locations. The methods support hypothesis testing for single devices and quantitative significance tests. We also sketch an efficient implementation for Matlab and discuss the complexity. Although the introduced metrics are applicable to PUF instances that are randomly spread on the chip, our implementation focuses on the important special case where the PUF cells are placed on a regular grid. We show the limitations of state-of-the-art methods w.r.t. correlations and the capabilities of the introduced metrics using three real-world data sets of PUFs field-programmable gate array (FPGA) and ASIC implementations. Our implementation of these tests and of other state-of-the-art tests for PUFs are available under [18]. Finally, in this work we discuss how the metrics derived with SPACA can be applied in practical tests for PUFs.

The remainder of the article is structured as follows: Section II shows the limitation of many canonical PUF metrics in finding spatial autocorrelations. Section III discusses the importance and definition of spatial proximity in the context of PUFs. Section IV introduces the mathematical background for SPACA followed by a sketch of the implementation in Section V. Section VI exemplifies the application of the methods and their capabilities. Section VII discusses how SPACA can be used in practical testing. The conclusion of the paper is drawn in Section VIII.

II. Sensitivity of Canonical Metrics to Intra-Die Correlations

The most prominent publications that explicitly define quality metrics for PUFs are those by Hori *et al.* [6] and Maiti *et al.* [7] (improved in [8]). The therein mentioned metrics are the most widely and most commonly used ones for the evaluation of PUFs and are therefore discussed in this section. Neither of these publications explicitly claims to target correlations, but their metrics might still be unintentionally sensitive to it, which we analyze below. Furthermore, some publications define dedicated quality indicators for the analysis of PUFs without explicitly proposing them as metrics. Indicators that consider correlations are the joint entropy computed by CTW [9] and the conditional min- and Shannon entropy in [10].

To cope with the dimensionality of PUF datasets, we introduce the following notation: We refer to a single measurement as $f_{s,p,d,c,e}$ where s iterates through samples (repeated measurements), p through positions on the device, d through devices, c through challenges, and e through environmental conditions. c is omitted if the device has only one challenge, i.e. it is a single-challenge PUF. e is omitted if data for different environmental conditions is not available for a dataset. We distinguish between challenges, used to configure a certain PUF such as an Arbiter PUF, and positions, which refer to an enumeration of the PUFs on a device such as the address space of an SRAM PUF. This distinction is especially important if multiple instances of a multi-challenge PUF are present on a device, e.g. two Arbiter PUFs that are combined into an XOR-Arbiter PUF or that run in parallel to double the size of the responses. In such applications correlations can exist within the responses of the same instance to multiple challenges or within the responses of multiple instances to a certain challenge or even both. The result of the digitization of the measurements is denoted as $b_{s,p,d,c,e} \in \{0, 1\}$, where the range of indices usually differs from f. The number of samples, positions, devices, challenges, and environmental conditions is denoted by the capitalized form of the corresponding index, e.g. S is the number of samples taken and $s \in \{1, 2, \ldots, S\}$. Environmental conditions are not considered for the metrics in this section, so e is omitted. For the remaining four dimensions, a visualization can be found in [6].

A. Maiti et al.

Maiti *et al*. define in [7] four metrics and none considers challenges, so c is omitted. In [8], they give a visualization and revise their definition of *Reliability*, but since Reliability tests are not of interest here, the following analysis still holds. The Reliability metric is also the only one considering environmental conditions, so e is also omitted. Further, all metrics operate on the *true* response of a device after averaging the data over the samples, so s is the third omitted index. Thus, the remaining metrics are *Bit-Alias* per position $\mathcal{B}_p$, *Uniformity* per device $\mathcal{J}_d$, and *Uniqueness* $\mathcal{V}$:

$$\mathcal{B}_p = \frac{100\%}{D}\sum_{d=1}^{D} b_{p,d},$$

$$\mathcal{J}_d = \frac{100\%}{P}\sum_{p=1}^{P} b_{p,d},$$

$$\mathcal{V} = \frac{100\% \cdot 2}{D(D-1)}\sum_{d=1}^{D-1}\sum_{d'=d+1}^{D}\frac{1}{P}\sum_{p=1}^{P}(b_{p,d} \oplus b_{p,d'}).$$

Uniformity $\mathcal{J}$ and Bit-Alias $\mathcal{B}$ simply measure the Hamming weight (HW) in percent – or probability of a logic 1 for that matter – in different dimensions, so they are pure bias metrics. Hence, they are not appropriate to detect correlations. Pehl *et al*. have shown in [19] that Uniqueness $\mathcal{V}$ can be expressed as a sum of Bit-Alias values:

$$\mathcal{V} = \frac{2\ D}{P(D-1)}\sum_{p=1}^{P}\mathcal{B}_p(1-\mathcal{B}_p)$$

This proves that Uniqueness $\mathcal{V}$ also cannot detect correlations, except if they already show up in Bit-Alias $\mathcal{B}$.

Although Uniqueness $\mathcal{V}$ is defined as the mean of the Hamming distances (HDs), many publications also provide a histogram of the HDs. The shape of this histogram is likely to be affected by correlations, but visually inspecting the histogram does not provide a quantitative statement and cannot serve as statistical hypothesis test on the probability of spatial correlations. Work that builds such a test upon the moments of the observed distribution has already been suggested by e.g. [20] but a properly defined metric is still missing.

B. Hori et al.

Hori *et al*. define in [6] five metrics and provide a figure to explain on which dimensions they operate. They do not consider environmental conditions, so e is omitted again. The first two metrics, Steadiness and Correctness, are unable to find correlations, because they target the reliability of the PUF response. The remaining three, *Randomness* $\mathcal{H}_d$, *Diffuseness* $\mathcal{D}_d$, and *Uniqueness* $\mathcal{U}$, all contain a sum over the positions:

$$\mathcal{H}_d = -\log_2 \max(p_d, 1-p_d)$$
$$\text{with}\quad p_d = \frac{1}{CSP}\sum_{c=1}^{C}\sum_{s=1}^{S}\sum_{p=1}^{P} b_{s,p,d,c},$$

$$\mathcal{D}_d = \frac{4}{C^2\ P}\sum_{c=1}^{C-1}\sum_{c'=c+1}^{C}\sum_{p=1}^{P}(\hat{b}_{p,d,c} \oplus \hat{b}_{p,d,c'}),$$

$$\mathcal{U} = \frac{4}{CD^2\ P}\sum_{c=1}^{C}\sum_{d=1}^{D-1}\sum_{d'=d+1}^{D}\sum_{p=1}^{P}(\hat{b}_{p,d,c} \oplus \hat{b}_{p,d',c})$$
$$\text{with}\quad \hat{b}_{p,d,c} = \lfloor 0.5 + \frac{1}{S}\sum_{s=1}^{S} b_{s,p,d,c}\rfloor.$$

The summation over positions leaves a lot of room for correlations to cancel out. In Randomness $\mathcal{H}_d$, the sum can be replaced for every challenge and every sample by the HW of that particular PUF response. Randomness $\mathcal{H}_d$ is therefore an extension of Uniformity $\mathcal{J}_d$ that additionally averages along samples and challenges. It detects only bias that is persistent among all these dimensions, but not correlations. Uniqueness $\mathcal{U}$ differs from Uniqueness $\mathcal{V}$ only by a coefficient and a summation over challenges, so it can be expressed as a sum of Bit-Alias values that is afterwards averaged among challenges. It thus cannot detect correlations. Diffuseness $\mathcal{D}_d$, compared to Uniqueness $\mathcal{V}$, considers the HD of PUF responses w.r.t. challenges while Uniqueness $\mathcal{V}$ uses the HD w.r.t. devices. Diffuseness $\mathcal{D}_d$ can therefore be expressed as a sum of – say *Challenge-Alias* – values $\mathcal{C}_{p,d}$ the same way Uniqueness $\mathcal{V}$ can be expressed as a sum of Bit-Alias values:

$$\mathcal{C}_{p,d} = \frac{100\%}{C}\sum_{c=1}^{C}\hat{b}_{p,d,c}$$

$$\mathcal{D}_d = \frac{4}{P}\sum_{p=1}^{P}\mathcal{C}_{p,d}(1-\mathcal{C}_{p,d})$$

This makes Diffuseness $\mathcal{D}_d$ a pure bias indicator, too.

C. Other Quality Indicators

Besides the metrics above, joint entropy is used in some works as an indicator for bias and correlation. In the area of PUFs, joint entropy is usually approximated by lossless compression via CTW [9]. The result can be interpreted as an upper bound for the joint entropy in the original dataset before compression. This approach has the drawbacks that the dataset must be large to get a reasonable approximation of the joint entropy and the result strongly depends on the ordering of the data within the dataset. Also, no confidence metric or hypothesis test is used in literature to evaluate how good the approximation is.

Machine-learning is currently used for modeling attacks on PUFs. It might also be seen as a quality indicator for PUFs, since – by the number of CRPs used to learn a PUF – it provides information on how hard it is to learn a PUF. Thus, this number of required CRPs can be seen as a metric for the entropy in a PUF considering bias and correlations, cf. e.g. [10], [20]. However, machine-learning currently requires CRPs from the same PUF instance and is not used to attack or evaluate single-challenge PUFs. Also, the result is only related to the upper bound for the entropy, since it usually cannot be ruled out that there is another, smaller set of CRPs that – maybe including a better learning strategy – allows to learn the PUF with less effort. This makes it also hard to define some confidence interval for the minimum required number of CRPs, limiting the usability

of machine-learning as a statistically substantiated metric for PUFs.

Another indicator that is sensitive to correlations is the conditional entropy in [10]. Although the corresponding formula in [10] lacks some explanation details, it most likely focuses on the correlation of a certain position with its neighbours rather than on the spatial correlations per device like the statistics computed by SPACA below. It also comes without hypothesis test or confidence evaluation of the result. This last metric from [10] as well as other non-mentioned metrics that are provided in certain PUF papers are not frequently used. However, including the quality indicators in this section, no metric can be found that focuses on spatial autocorrelations of PUF instances within a device and that supports the statistical hypothesis testing required to provide meaningful statistical evaluation.

III. Spatial Proximity in PUFs

The focus of this work is on single-challenge PUFs. We define such a PUF as a two-dimensional composition of identical instances of circuits built to measure manufacturing variations as an entropy source. While the metrics introduced in the next section are applicable to any scenario with randomly spread circuits on a device, for the sake of efficiency we limit our implementation to the for PUFs important case of array-like structures.

A two-dimensional PUF array consists of $n_x \times n_y$ nodes indexed by i (or j), where $1 \leq i \leq n = n_x n_y$. The observed output of a circuit is either a binary-valued PUF response, e.g. for SRAM PUFs [4], or a digitized analog measurement value, e.g. the frequency of an RO in an RO PUF [21]. The observed output value of a circuit at a certain position i on a device is considered as an outcome x_i of a random variable X_i.

Independently from the fact if the PUF cells are placed in an array-like structure or randomly spread on a device, for two circuits represented by X_i and X_j, a certain proximity can be defined. The proximity can be based on either actual spatial proximity, i.e. circuit i is close to circuit j on the silicon, or electrical proximity, e.g. circuit i and j are coupled by the supply line or some other wire. This can be modeled by a weighted graph $G(V, E), w_{i,j} \mapsto \mathbb{R}_+ \cup \{0\}$ where $V = \{1, ..., n\}$ corresponds to the circuits, $E = \{(i, j) \,|\, i \neq j \wedge i, j \in V\}$ describes a neighborhood of circuits, and the weights $w_{i,j}$ define the proximity of the circuits. A high proximity (low distance) of two circuits is modeled by a large weight and a low weight corresponds to a low proximity. The weights together with the adjacency information of the edges can be encoded by an $n \times n$ *weighting matrix* (also weighted *adjacency* or *connection matrix*)

$$\{w_{i,j}\} = \begin{bmatrix} 0 & w_{1,2} & w_{1,3} & \dots & w_{1,n} \\ w_{2,1} & 0 & w_{2,3} & \dots & w_{2,n} \\ w_{3,1} & w_{3,2} & 0 & \ddots & w_{3,n} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ w_{n,1} & w_{n,2} & w_{n,3} & \dots & 0 \end{bmatrix} \tag{1}$$

Zeros in $\{w_{i,j}\}$ denote the absence of a neighborhood. The diagonal of $\{w_{i,j}\}$ is zero, since a circuit is not adjacent to itself by definition.

Directed graphs correspond to a matrix with $w_{i,j} \neq w_{j,i}$ for some (i, j). This most general form of a weighting matrix is useful, e.g. if the model describes a directed flow. If the distance of any two nodes in the graph i and j is symmetric, like it is the case for the spatial proximity of PUF circuits, also $\{w_{i,j}\}$ is symmetric, i.e. $w_{i,j} = w_{j,i}$.

For SPACA in the context of PUFs, all those pairs of nodes that correspond to circuits with sufficient proximity are considered adjacent. This is technically motivated by the fact that correlations are expected for the values measured at the output of circuits with low spatial or electrical distance. Also, the performance of the tests can be improved if only relevant pairs of nodes are considered (cf. Section V). The distance of nodes is defined by a metric such as the Euclidean or the taxicab metric. The maximum distance upto which pairs are considered relevant is called *neighborhood radius*. The weights for pairs of circuits (i, j) beyond the neighborhood radius are set to zero, i.e. they are considered as non-adjacent.

IV. Spatial Autocorrelation Statistics

Spatial autocorrelation can always be positive or negative. To grasp spatial autocorrelation intuitively, we consider perfect positive and perfect negative spatial autocorrelation in two classes: black and white. Perfect negative spatial autocorrelation then corresponds to a chess board, where black and white fields are interleaved. Perfect positive spatial autocorrelation on the other hand would mean a bisection of the board with all white fields in the left half and all black fields in the right half side of the board (or vice versa). More general, while positive spatial autocorrelation corresponds to a case where neighboring positions (cf. Section III) tend to have the same value (clustering), negative spatial autocorrelation indicates a case where neighboring positions tend to have alternating values (dispersion).

Using the weights introduced in Section III, three statistics are described in the following to quantify spatial autocorrelation: *Moran's I* [13], *Geary's c* [14], and the *Join Count statistic* [15], cf. also [22]–[24]. The metrics are used afterwards to reveal spatial autocorrelations in PUFs.

A. Moran's I Statistic

In 1950 Moran [13] described the first measure of spatial autocorrelation for phenomena distributed in two-dimensional or higher dimensional space. The coefficient for the test statistic is called *Moran's I*. It is analogous to Pearson's correlation coefficient having a product moment term in the numerator. The coefficient can be cast into a form with a generalized weighting matrix $\{w_{i,j}\}$

$$I = \frac{n}{\sum_i \sum_j w_{i,j}} \cdot \frac{\sum_i \sum_j w_{i,j} (x_i - \overline{x})(x_j - \overline{x})}{\sum_i (x_i - \overline{x})^2},$$

where $\overline{x}$ is the sample mean over the set of n observations $\{x_i\}$, cf. [22], [23]. Like Pearson's linear correlation coefficient it has range $-1 \leq I \leq 1$, where -1 indicates strong

negative spatial autocorrelation, 0 means a random pattern, and $+1$ proves strong positive spatial autocorrelation. The weights between the different points $w_{i,j}$ are chosen like described above. To perform a significance test, a model for the observations under a null hypothesis H_0 is needed. The expected values for the moments of I can be calculated under either of the following two assumptions for H_0:

Normality Assumption (N): The observations $\{x_i\}$ are the result of n independent drawings from one (or several identical) normal populations.

Randomization Assumption (R): The observations $\{x_i\}$ are the result of random independent drawings from one (or several identical) populations with unknown distribution functions. So, the observed value of I is considered relative to all possible outcomes, if the $\{x_i\}$ were randomly permuted.

We introduce the following sums over the weight coefficients to achieve a concise and unified notation for all considered spatial statistics:

$$S_0 = \sum_i \sum_j w_{i,j},$$
$$S_1 = \frac{1}{2} \sum_i \sum_j (w_{i,j} + w_{j,i})^2,$$
$$S_2 = \sum_i \left(\sum_j w_{i,j} + \sum_j w_{j,i} \right)^2.$$

Furthermore, we need some central and standardized sample moments

$$\overline{x} = \mu_1 = \frac{1}{n} \sum_i x_i \qquad \text{(mean)},$$
$$\sigma^2 = \mu_2 = \frac{1}{n} \sum_i (x_i - \overline{x})^2 \qquad \text{(variance)},$$
$$\mu_4 = \frac{1}{n} \sum_i (x_i - \overline{x})^4 \qquad \text{(4th central moment)},$$
$$g_2 = \frac{\mu_4}{\mu_2^2} \qquad \text{(kurtosis)}.$$

Under the normality assumption (N) the moments of I are obtained as

$$E_N[I] = \frac{-1}{n-1},$$
$$E_N[I^2] = \frac{n^2\ S_1 - nS_2 + 3S_0^2}{(n^2-1)S_0^2},$$
$$\mathrm{var}_N[I] = E_N[I^2] - E_N[I]^2.$$

Under the randomization assumption (R) the moments of I are

$$E_R[I] = \frac{-1}{n-1},$$
$$E_R[I^2] = \frac{\begin{pmatrix} n[(n^2-3n+3)S_1 - nS_2 + 3S_0^2] \\ \quad - g_2[(n^2-n)S_1 - 2nS_2 + 6S_0^2] \end{pmatrix}}{(n-1)(n-2)(n-3)S_0^2},$$
$$\mathrm{var}_R[I] = E_R[I^2] - E_R[I]^2.$$

The null hypothesis H_0 for the significance test is that each observation $\{x_i\}$ is equally probable and independent from all others. The statistics for I is asymptotically normal distributed for the assumptions (N) and (R) [22]. The standard normal deviates (z-scores) for the assumptions (N) and (R) can be calculated by

$$z_N(I) = \frac{I - E_N[I]}{\sqrt{\mathrm{var}_N[I]}} \quad \text{and} \quad z_R(I) = \frac{I - E_R[I]}{\sqrt{\mathrm{var}_R[I]}},$$

respectively. The null hypothesis H_0 can be accepted (or rejected) based on some *acceptance level* α, say $\alpha = 0.05$. In a two-sided test H_0 is accepted, if the p-value

$$p(z) = \mathrm{erfc}\left(\frac{|z(I)|}{\sqrt{2}} \right)$$

is in the interval $\frac{\alpha}{2} \leq p(z) \leq 1 - \frac{\alpha}{2}$.

B. Geary's c Statistic

In 1954 Geary [14] and Griffith [24] described a statistic to quantify spatial autocorrelation, which is called *Geary's Contiguity Ratio* or short *Geary's c*. The coefficient can also be cast into a form with a generalized weighting matrix $\{w_{i,j}\}$, cf. [22], [23]:

$$c = \frac{n-1}{2 \sum_i \sum_j w_{i,j}} \frac{\sum_i \sum_j w_{i,j} (x_i - x_j)^2}{\sum_i (x_i - \overline{x})^2}.$$

Geary's c ranges between 0 and 2. Positive spatial autocorrelation is indicated by values between 0 and 1 and negative spatial autocorrelation is found for values between 1 and 2. Sokal and Oden [25], [26] note that empirical results computed by both Moran's I and Geary's c are similar, but not identical. The numerator of Geary's c will tend to be sensitive to absolute differences between neighboring variates due to the squared term in the numerator. The weights are treated the same way as in Moran's I. Similarly, the significance test can be performed either under the normality and the randomization assumption. Under the normality assumption (N) the moments of c are obtained as

$$E_N[c] = 1,$$
$$\mathrm{var}_N[c] = \frac{(n-1)(2S_1 + S_2) - 4S_0^2}{2(n+1)S_0^2}.$$

Under the randomization assumption (R) the moments of c are

$$E_R[c] = 1,$$
$$\mathrm{var}_R[c] = \frac{\begin{pmatrix} (n-1)\left[n^2 - 3n + 3 - (n-1)g_2\right]S_1 \\ \quad - \frac{1}{4}(n-1)\left[n^2 + 3n - 6 - (n^2 - n + 2)g_2\right]S_2 \\ \quad + \left[n^2 - 3 - (n-1)^2 g_2\right]S_0^2 \end{pmatrix}}{n(n-2)(n-3)S_0^2}.$$

The statistics for c exhibits asymptotic normality under the assumptions (N) and (R), cf. [22]. Hence, an acceptance test for the null hypothesis H_0 can be applied just as with Moran's I.

C. Join Count Statistic

For nominal data the *Join Count* statistical test can be applied [15], [22]. In a binary classification scheme we denote the presence and absence of a certain property as black B (1) and white W (0), respectively. To determine spatial correlations the numbers of neighboring pairs (*joins*) BB $(1-1)$, BW $(1-0)$, and WW $(0-0)$ are counted and compared with the expected numbers under the null hypothesis of a random distribution. A clustering of the values (positive autocorrelation) is indicated, if the number of BW joins is lower than expected by chance. A dispersion (negative autocorrelation) is given, if the number of BW joins is higher than expected. The Join Count statistics for BB and BW are defined by

$$J_{BB} = \frac{1}{2}\sum_i \sum_j w_{i,j} x_i x_j,$$
$$J_{BW} = \frac{1}{2}\sum_i \sum_j w_{i,j} (x_i - x_j)^2.$$

Note the similarity of J_{BW} to Geary's c. The statistic for WW can be calculated from these by $J_{WW} = \frac{S_0}{2} - J_{BB} - J_{BW}$.

Under the *free sampling* assumption (sampling with replacement) it is supposed that B and W are independently distributed with probabilities p and $q = 1 - p$, respectively. Under this assumption the moments of the observables BB, BW, and WW are obtained as

$$\begin{aligned}
E[BB] &= \frac{1}{2} S_0 p^2, \\
E[WW] &= \frac{1}{2} S_0 q^2, \\
E[BW] &= S_0 pq, \\
\mathrm{var}[BB] &= \frac{1}{4}[S_1 p^2 + (S_2 - 2S_1)p^3 + (S_1 - S_2)p^4], \\
\mathrm{var}[WW] &= \frac{1}{4}[S_1 q^2 + (S_2 - 2S_1)q^3 + (S_1 - S_2)q^4], \\
\mathrm{var}[BW] &= \frac{1}{4} S_2 pq + (S_1 - S_2)p^2 q^2.
\end{aligned}$$

In the *nonfree sampling* model (sampling without replacement) it is assumed that each cell has the same a priory probability of being B or W, but the overall constraint of fixed numbers B and W with $B + W = n$ is imposed. Under this assumption the moments are given by [22]

$$E[XX] = \frac{1}{2} S_0 \frac{\binom{X}{2}}{\binom{n}{2}} \text{ for } X \in \{B, W\},$$
$$E[BW] = \frac{1}{2} S_0 \frac{B \cdot W}{\binom{n}{2}},$$
$$E[XX^2] = \frac{1}{4}\left[S_1 \frac{\binom{X}{2}}{\binom{n}{2}} + (S_2 - 2\ S_1)\frac{\binom{X}{3}}{\binom{n}{3}} + (S_0^2 + S_1 - S_2)\frac{\binom{X}{4}}{\binom{n}{4}} \right] \text{ for } X \in \{B, W\},$$
$$E[BW^2] = \frac{1}{4}\left[S_1 \frac{B \cdot W}{\binom{n}{2}} + (S_2 - 2\ S_1)\frac{B(B+W-2)W}{n(n-1)(n-2)} + 4(S_0^2 + S_1 - S_2)\frac{B(B-1)W(W-1)}{n(n-1)(n-2)(n-3)} \right].$$

The statistics of BB, BW, and WW approach asymptotic normality [22], such that an acceptance test for the null hypothesis H_0 can easily be set up, e.g. using z-scores. For generalizations to more than two classes cf. [22].

D. Interpretation of Metrics in the PUF Context

A distinguished advantage of SPACA is that the calculated z-scores or p-values can be easily used for single device tests: If the p-value is below a predefined acceptance level α, the device is considered flawed, because the probability that such a low p-value occurs by chance, although the device is free of spatial autocorrelation (Type I Error), is considered too low.

The z-scores are interesting for the analysis of complete PUF designs: Because the z-scores approach normality under both assumptions as outlined in the previous section, a representative set of devices with this design can be investigated regarding its distribution of z-scores. If the set of z-scores passes a test for standard normal distribution with a certain confidence, the quality of the design regarding spatial autocorrelation can be confirmed for this confidence level.

Note that this does not state anything about non-linear dependencies. However, detecting and thus being able to remove linear dependencies is already an important first step.

V. Implementation Notes

The statistical tests introduced in the previous section were implemented in this work in Matlab R2016b based on the original publications[1] [13]–[15], [22], [24]. The implementation, which is available under [18], is optimized with respect to performance and it considers simplifications. E.g., for the weights in the weighting matrix (Eq. 1) $\forall(i, j) : w_{i,j} = w_{j,i}$ holds since the dependency between PUF cells (i.e. datapoints) is symmetric and undirected. All three tests iterate over all n points (or positions) in two nested loops and sum up the similarity values, e.g. $(x_i - \overline{x})(x_j - \overline{x})$ for Moran's I and $(x_i - x_j)^2$ for Geary's c. Hence, a naive implementation has complexity $\mathcal{O}(n^2)$. The neighborhood radius (cf. Section III) and thus the number of considered adjacent positions m is a matter of choice. This choice is independent of – and usually less than – the number of positions. Since non-adjacent positions have a weight of $w_{i,j} = 0$, only one of the nested loops needs to iterate over n while there are $m \ll n$ iterations for the other loop. I.e. for constant m the complexity class is $\mathcal{O}(n)$.

Vectorized calculations are typically much faster than loops in Matlab. Two assumptions are used to take advantage from this:

1) The positions form a regular two-dimensional structure and the samples are arranged in a corresponding $n_x \times n_y$ matrix $\{x_{k,l}\}$.

[1]Note that flaws can be found in many existing implementations of and introductions to SPACA. E.g.: In [27, Sec. 9.4.2], the table contains several erroneous formulae for the expected moments of J_{XY}; PySAL (Python Spatial Analysis Library) v.1.11.0 has errors in the implementation of the formulae for Moran's $\mathrm{var}_R[I]$ and Geary's $\mathrm{var}_R[c]$; [22, Table 1.2], the reference results of Moran's $E(I)$ for the city population example under the randomization and normality assumptions should read -0.067 instead of -0.67.

2) The weights $w_{i,j}$ are chosen such that for two positions the weight only depends on the difference in the indexes k and l in $\{x_{k,l}\}$.

On the one side, these assumptions limit the implementation to the important case where the PUF cells are arranged in an array-like structure, although the methods above could also handle PUFs where the cells are randomly spread on a device. On the other side, the assumptions allow for the calculation of the similarity values for all neighbors with a certain distance in k and l at once by shifting the entire matrix and applying a matrix operation. E.g. for Geary's c the comparison function $(x_i - x_j)^2$ for all the right neighbors is computed by

$$\mathtt{sum}\left(\begin{Bmatrix} x_{1,2} & \cdots & x_{1,n_x} \\ \vdots & \ddots & \vdots \\ x_{n_y,2} & \cdots & x_{n_x,n_y} \end{Bmatrix} - \begin{Bmatrix} x_{1,1} & \cdots & x_{1,n_x-1} \\ \vdots & \ddots & \vdots \\ x_{n_y,1} & \cdots & x_{n_x,n_y-1} \end{Bmatrix}\right)^2,$$

where the square is taken element-wise and sum is the sum over all matrix elements. Since the comparison functions for all three statistics are commutative, this immediately yields also the result for all left neighbors, so the result is taken twice. It is then multiplied with the weight corresponding to a distance of 1 in index k and 0 in index l.

The concept equally applies to all other metrics in Section IV and for positions with other differences in index k and l. Thus, if m points are considered adjacent, the approach allows to compute all similarity values with only $\frac{m}{2}$ loop iterations, which contain only matrix operations and a scalar multiply-add operation. To exemplify the benefit of this approach, one of the analyses of Section VI is used: For the SRAM-XMC dataset, $n \approx 1.3 \cdot 10^6$ is the number of SRAM cells, while for a neighborhood radius of $d = 1$ the neighborhood consists of only the upper, lower, left and right neighbor, i.e. $m = 4$. This way only two explicit and slow loop iterations are required, while the laborious iteration over $n \approx 1.3 \cdot 10^6$ positions is outsourced to the fast built-in methods of Matlab.

VI. Experimental Verification

After having introduced the methods and their mathematical background, we show that SPACA gives important insight in the field of PUFs. We apply various flavors of these tests to the following three datasets:

RO-Maiti Frequencies of an array of 512 ROs implemented on 193 Xilinx Spartan 3 FPGAs, sampled 100 times at uncontrolled room temperature [7].

SRAM-XMC Start-up values of 160 KiB of SRAM cells on 144 Infineon XMC4500 microcontrollers, sampled 101 times at uncontrolled room temperature [28].

TS-IFX Response bits of an array of 4096 Two-Stage ID PUFs [29] on 243 ASICs in a 90 nm technology, sampled 1001 times at different operating conditions.

We choose the taxicab norm as a distance metric. It reflects the model assumption that elements on a straight horizontal or vertical line are electrically closer than others. This is a reasonable assumption for both ROs on an FPGA and SRAM cells in a typical memory array. For the Two-Stage ID PUF an Euclidean norm would be reasonable as well, because its design reduces the risk of electrical coupling. In such a case, physical properties such as dopant concentration or oxide thickness remain as possible common factors that cause spatial autocorrelations. However, for small radii as we use them in our analysis, the difference between taxicab norm and Euclidean norm is marginal. So only the taxicab norm is used for all three datasets. The weights in $\{w_{i,j}\}$ are the reciprocal values of the distances, because they reflect proximity, e.g. $\frac{1}{3}$ if the distance is 3.

To determine the expected moments for H_0, we choose the randomization approach and correspondingly the non-free sampling approach for the Join Count statistic. The normalization approach would require the data to be drawn from the same normal distribution, which cannot be assumed in many cases.

All histograms in this section come with a kernel density estimation (KDE) for easier comparison to the overlaid standard normal distribution (SND), which is the expectation for H_0. Scott's rule was used to infer the number of containers to avoid overfitting. Vertical bars indicate the acceptance interval for $\alpha = 5\%$. The two y-axes for probability density (left) and event count (right) are scaled such that the area covered by the histogram bars matches the area under the SND.

For all datasets, we perform a preprocessing to approximate the noiseless response. For binary data, we apply majority voting among samples; for interval data, we take the mean

$$f_{p,d,c,e} = \frac{1}{S}\sum_{s=1}^{S} f_{s,p,d,c,e}. \tag{2}$$

A. RO-Maiti

1) Frequency Data: The RO-Maiti dataset [30] has already been investigated in several publications, e.g. [3], [31]–[33]. Because data for different environmental conditions is not available for all field programmable gate arrays (FPGAs), and ROs do not feature challenges, indices e and c are omitted for brevity. All publications found noteworthy spatial patterns in the mean frequencies of the ROs

$$f_p = \frac{1}{D}\sum_{d=1}^{D} f_{p,d}. \tag{3}$$

Fig. 1 shows the means together with the corresponding standard deviations for reference. The figure shows a distribution of the frequencies in four segments, each overlaid by some curvatures, minimum frequencies at the boarders, and maximum frequencies in the middle of the segments. However, the observations in all publications at least partially relied on manual inspection of plots and therefore can only provide qualitative results.

The methods proposed herein instead make quantitative statements on the probability that the frequencies of the ROs are truly randomly spread over a device. The statement is provided in form of a z-score – which can be converted into a p-value – for each device individually (cf. Section IV-D). This has several benefits, e.g. more efficient parallelization of the analysis of large datasets. Fig. 2 shows histograms of z-scores

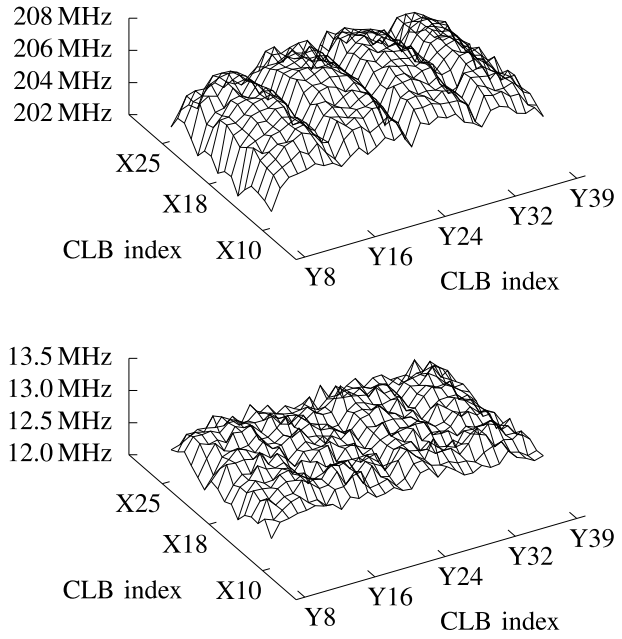

Fig. 1. Location preserving plot of mean (top) and standard deviation (bottom) over devices of RO frequencies (i.e. $f_{p,d}$) in dataset RO-Maiti.

of Moran's I and Geary's c applied to the raw frequencies, for a neighborhood radius of 1. All scores are far to the right of the SND, which indicates strong positive spatial autocorrelation.

This result can partially be explained by the pattern of mean frequencies in Fig. 1. Because the SPACA tests consider each device individually, they cannot distinguish whether a certain pattern on a device comes from unequal mean values or true spatial autocorrelation. This cross-sensitivity is not obstructive, though, because unequal mean frequencies also impair the unpredictability of the response. So if the SPACA tests flag a device, the question is rather on the exact reason. If the mean frequencies differ sufficiently from one position to the other, most devices will also be flagged regarding spatial autocorrelation. If mean frequencies are equal for all positions, true spatial autocorrelation is causative. Thus, the question if the problem flagged by SPACA is caused by bias or correlations can be answered if SPACA is combined with appropriate bias tests.

Another way to distinguish the reason for a peculiar SPACA result is to repeat the test after aligning the mean frequencies:

$$g_{p,d} = f_{p,d} - f_p \tag{4}$$

Fig. 3 shows the results on $g_{p,d}$ for the same tests as Fig. 2 on $f_{p,d}$. Comparing the two figures, the indicated positive spatial autocorrelation decreases, but does not vanish. So there is true spatial autocorrelation in the dataset. A finding impossible to draw from Fig. 1 alone.

Without SPACA, it takes at least a principal component analysis (PCA) to detect this [33]. A PCA, however, requires a sufficiently large number of devices and thus cannot be applied to a single device. [33] concluded from the PCA results that most devices are subject to a speed gradient over the die, whose direction and slope varies. This matches well with the positive spatial autocorrelation in Fig. 3: The stronger the speed gradient, the stronger the measured autocorrelation.

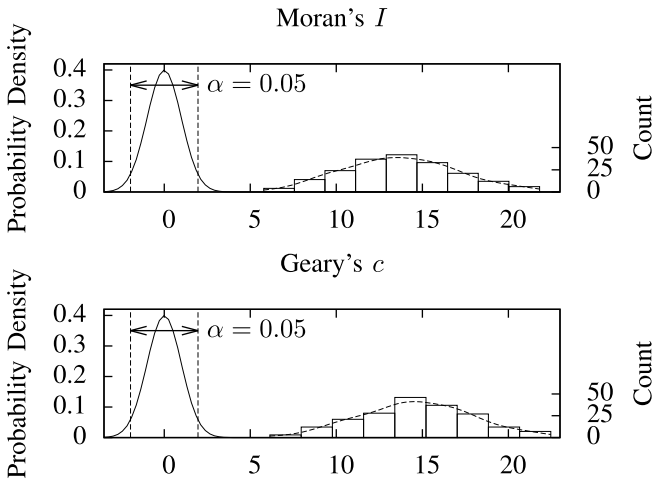

Fig. 2. Histograms and KDEs of z-scores under randomization assumption of Moran's I and Geary's c for RO-Maiti dataset with neighborhood radius $d = 1$.

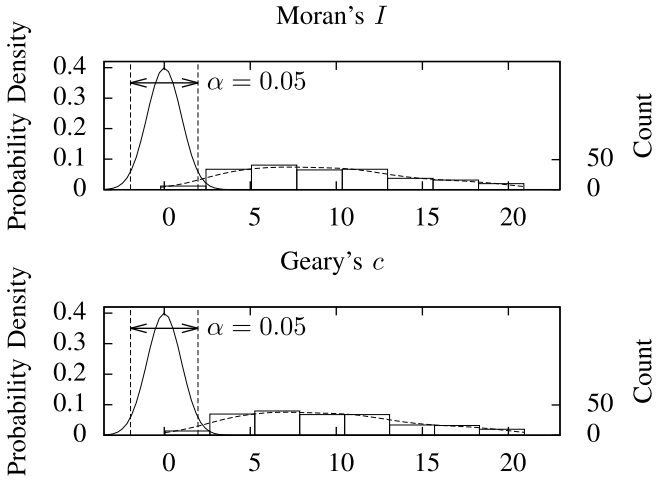

Fig. 3. Histograms and KDEs of z-scores under randomization assumption of Moran's I and Geary's c for RO-Maiti dataset after mean frequency alignment with neighborhood radius $d = 1$.

More precisely, of the 10 devices with highest Moran's I z-score, 8 where among the 10 devices with strongest speed gradient and the remaining 2 within the following 10.

For high security applications, the design should be modified to provide equal mean frequencies and the production process should be improved to remove the speed gradient. However, the degree to which the binary response is affected by such flaws can be attenuated by the bit extraction scheme, as outlined in the following.

2) Response Bits: To analyze the response bits extracted from the RO frequencies, the Join Count statistic fits better than Moran's I or Geary's c, because it is made for nominal data (like a logical one or zero) instead of interval data (like a frequency). Fig. 4 shows the Join Count statistic on bits extracted by mutually exclusive pairwise comparison (PWC) between physically adjacent ROs as in [33]. The spatial autocorrelation nearly disappears – even if the neighborhood radius is increased to 3 for higher sensitivity – because the PWC acts as a spatial high-pass filter and effectively reduces the correlation among the response bits. On the one hand,

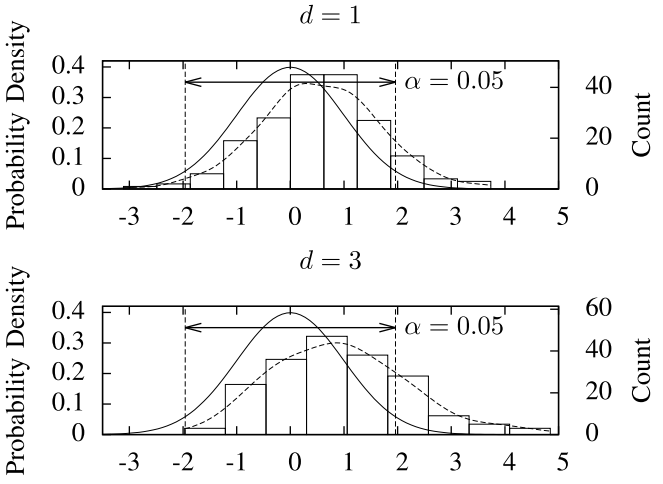

Fig. 4. Join Count statistic on response bits from mutually exclusive PWC on RO-Maiti dataset with neighborhood radius $d \in \{1, 3\}$.

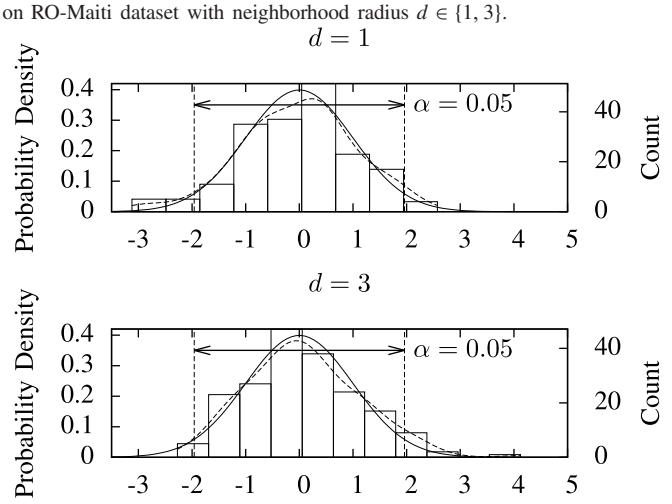

Fig. 5. Join Count statistic on response bits from mutually exclusive PWC on RO-Maiti dataset after mean frequency alignment with neighborhood radius $d \in \{1, 3\}$.

this means that underlying layout issues cannot be detected anymore. On the other hand, this shows that the consequences of suboptimal layouts can be reduced by good bit extraction schemes. With additional mean frequency alignment before the PWC, the spatial autocorrelation becomes almost completely invisible, as shown in Fig. 5.

But for PWC to act as spatial high-pass filter, it needs to be done in a mutually exclusive way. If, for example, PWC is done in an overlapping way to extract $P - 1$ instead of $\frac{P}{2}$ bits as in [7], strong negative spatial autocorrelation arises. This is because every RO that is not at the left or right edge of the design is used twice: It is taken once as the right RO and once as the left RO for PWC and thus influences both bits in opposite directions. The resulting negative spatial autocorrelation even after mean frequency alignment is visible in Fig. 6.

B. SRAM-XMC

The SRAM-XMC dataset is the third publicly available PUF dataset on the internet after the AIST dataset [34] and the RO-Maiti dataset. Since SRAM PUFs directly provide nominal data, analysis starts with the Join Count statistic like for the response bits of RO-Maiti and analysis of interval

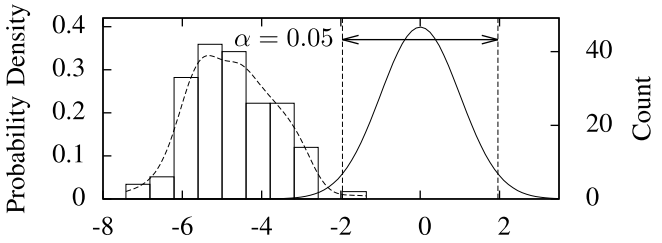

Fig. 6. Join Count statistic on response bits from overlapping PWC on RO-Maiti dataset after mean frequency alignment with neighborhood radius $d = 1$. Note that for better locatability of the bits, else than in [7], comparisons across linebreaks are not performed.

data is skipped. The motivation for analysis in this case is rather how much and which postprocessing is necessary for acceptable PUF behavior than how to improve the design, because the SRAM of commercial off-the-shelf (COTS) microcontrollers (MCUs) is not dedicated to serve as PUF and will in most cases not be changed to fit this purpose.

The cheaper price for COTS devices is frequently gained by less insight into the device, for example missing layout information. This makes spatial analysis more difficult, since the weights have to be guessed from an assumed SRAM layout. But except for SRAM, most other types of PUF are less likely to suffer from this problem anyway: For FPGA-based PUFs, coarse layout information is provided by the development tools after implementation on a particular type of FPGA. While this is, again, insufficient when using one of the block SRAMs as PUF, it can be sufficient for PUFs built in fabric, as shown in the previous subsection. For PUFs implemented on an ASIC, precise layout information in standard-cell granularity is naturally available to the designer, except where external IP blocks, e.g. third-party SRAM, are used. Furthermore, if the PUF design is to be certified, extensive documentation – regularly including layout information – has to be provided to the certifier and is thus also available to be used in SPACA.

Fig. 7 shows the Join Count statistics for two assumed line widths of the SRAM, 32 bit and 64 bit. The MCU has a native word width of 32 bit, so the former was an initial choice, but state-of-the-art space-saving layout techniques for SRAM suggest the latter, so both assumptions were tested. Results for a neighborhood radius of 3 are added to show spatial correlations that do not affect direct neighbors but others in the proximity.

The most noticeable observations from Fig. 7 are the extremely large z-scores for $l_x = 64$. Therefore, the SND, which would be unrecognizable due to axes scaling, is omitted. The large z-scores result from a bisection of bits regarding Bit-Alias [7] values, where the first 32 bit in each line are distributed around 0.6 and the bits in the second half of the line are distributed around 0.4. Like the unequal mean frequencies in the RO-Maiti dataset, this is detected through the cross-sensitivity of the spatial autocorrelation tests. A Bit-Alias test, cf. Section II, can be used in the same way as an analysis of the mean frequency along devices to distinguish the reason for the conspicuous SPACA result.

Considering this cross-sensitivity, it is surprising that the plots for $l_x = 32$ in Fig. 7 tremendously underestimate the issue of unequal Bit-Alias values. This is a special case where

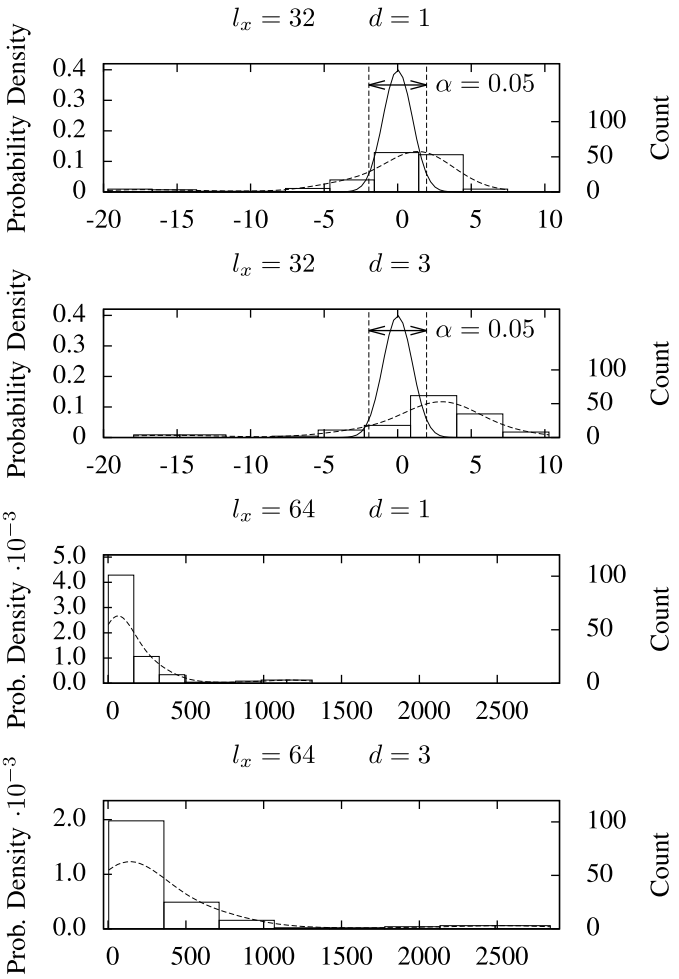

Fig. 7. Join Count statistic on SRAM-XMC dataset for assumed line width $l_x \in \{32, 64\}$ and with neighborhood radius $d \in \{1, 3\}$. Note that the plots for $l_x = 64$ do not have the standard normal distribution overlaid as it would be illegible due to extreme axes scaling.

positive spatial autocorrelation in one direction cancels with negative spatial autocorrelation in another direction. A line wrap after every 32 bit leads to an alternating pattern across lines, where a line with all bits biased to zero is followed by a line with all bits biased to one. This corresponds to negative spatial autocorrelation. At the same time, though, the bits within a line are all biased to the same value, which corresponds to positive spatial autocorrelation. Since the tests only use weights which we chose based on the taxicab norm, they are not sensitive to directional information. This explains why the tests simply add up spatial autocorrelation in any direction and allow for negative and positive ones to cancel each other out. Although this is a pitfall, we consider it a special case. To overcome this problem, the test can be repeated with weights that credit distance in different axes unequally. The price for this second test is just one multiply-add if it is implemented in combination with the first test. Even if few degenerated distributions might still be constructed where negative and positive spatial autocorrelation cancel out despite of using the suggested method, it is very unlikely for such a case to appear in practice.

Fig. 9 exemplifies the result of using unequal weights for different directions. It shows results of the Join Count

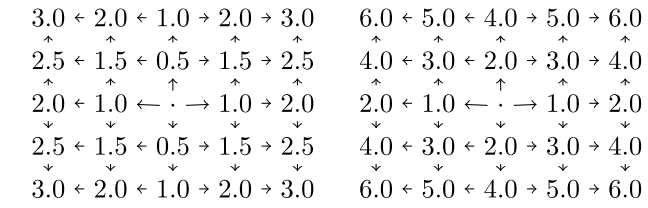

Fig. 8. Examples of the irregular taxicab norm. Compared to a regular taxicab norm, blocks are no longer quadratic but just rectangular, so steps in different directions add a different amount to the overall distance.

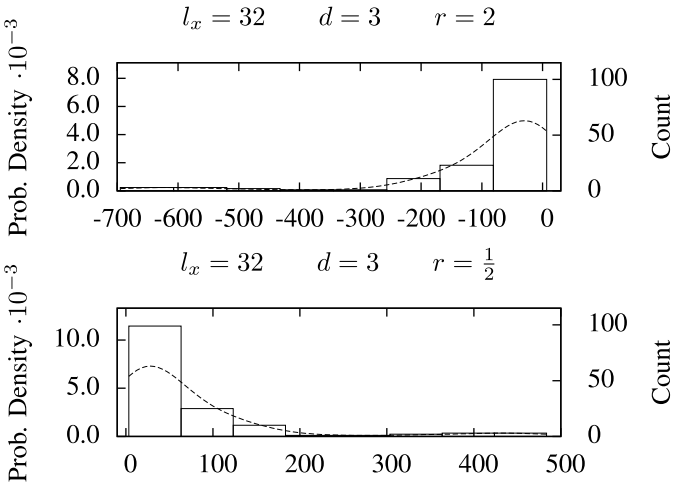

Fig. 9. Join Count statistic on SRAM-XMC dataset for assumed line width $l_x = 32$ using an *irregular taxicab norm* with ratio $r \in \{0.5, 2\}$ and neighborhood radius $d = 3$. Note that the plots do not have the standard normal distribution overlaid as it would be illegible due to extreme axes scaling.

statistic with weights derived from what we call an *irregular taxicab norm*. In this norm, a step in vertical direction is either shorter or longer than a step in horizontal direction, cf. Fig. 8. If steps in vertical direction, i.e. within a line of the SRAM-XMC dataset, cost twice as much as a horizontal step, the negative spatial autocorrelation between lines dominates as shown in the upper plot of Fig. 9. Likewise, if steps in vertical direction cost only half, the positive spatial autocorrelation within lines dominates as shown in the lower plot of Fig. 9.

C. TS-IFX

After two datasets with significant spatial autocorrelation, our third and last dataset can serve as a positive example. The TS-IFX dataset is obtained from three test wafers of a proprietary PUF primitive called Two-Stage ID [29]. It works inherently differential like SRAM and thus provides only binary data. But other than general purpose SRAM the ASIC layout is explicitly designed for symmetry, so it does not suffer from large-scale Bit-Alias like most other SRAM PUFs. Due to separated amplification and decision phases, it is also much more reliable than canonical SRAM PUFs. The test wafers were thoroughly characterized, which makes it the only dataset in our work where data at varying environmental conditions – such as temperature, supply voltage, etc. – is available.

Fig. 10 shows that there is no spatial autocorrelation even at the corners of the operating conditions. To verify that spatial

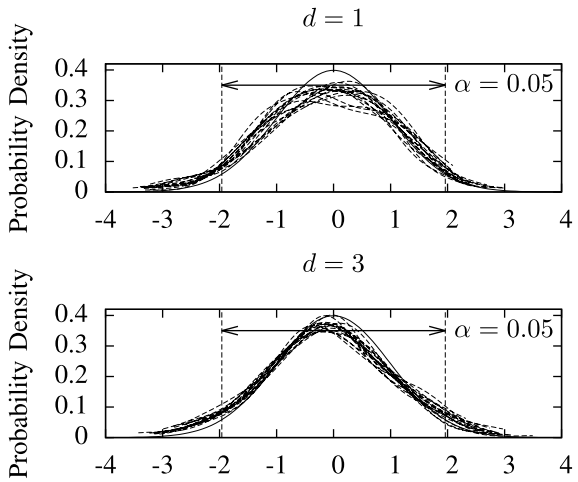

Fig. 10. KDEs of Join Count statistic on TS-IFX dataset for 15 different environmental conditions, including temperature, supply voltage, and amplification current sweeps. Histogram bars are omitted for legibility.

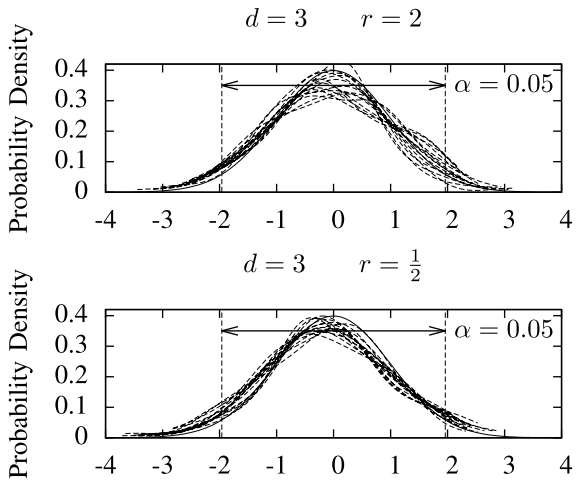

Fig. 11. KDEs of Join Count statistic on TS-IFX dataset as in Fig. 10 but using an *irregular taxicab norm* with ratio $r \in \{0.5, 2\}$ and neighborhood radius $d = 3$. Histogram bars are omitted for legibility.

autocorrelation does not cancel out like for the SRAM-XMC dataset, the test was repeated using an irregular taxicab norm (see Fig. 11).

VII. Quantitative Evaluation

In the previous section we visually compared the distribution of the z-scores with an SND, which is the expectation for H_0. While this is intuitive and works well if the data is far from randomly located and thus the distributions differ strongly, it is informal and cannot be automated. But there are mathematical tools to perform this comparison and fulfill the claim of fully automated testing:

During qualification of a new product, a large sample of devices will be thoroughly tested and entire PUF responses from all test devices are available for analysis, like in the previous section. In this situation, the challenge is to make a proper quantitative statement on how probable it is that the design itself does not contain errors that lead to spatial correlations. Such quantitative statements are also necessary to define common criteria for certification,[2] Although other analysis such as a PCA can be performed in this scenario, it is difficult to derive a quantitative statement on the probability of correlations from its result. For the spatial autocorrelation tests, the expected asymptotic behavior is known and normality tests such as the Chi-Square, the Shapiro-Wilk, or the Anderson-Darling test can be used to replace the visual inspection of histograms. The obtained p-value gives an estimate on how probable it is, that the spatial autocorrelation statistics is indeed an SND.

A different scenario is the continuous end-of-line production test to maintain the quality of the product. In this case the PUF response should not be stored on a central server for statistical tests, because this creates a single point of compromise. Instead, it should not leave the device at any time. A good solution would thus be if the device itself could perform the test and output a value that does not reveal exploitable information on the PUF response. However, this means that the statistic needs to be computable from a single value per device and that this value needs to be computable on very resource constrained devices. A solution to this can be the Join Count statistic, because most of its operations are simple XORs and the calculation of $E[BW]$ and $E[BW^2]$ can be simplified to just a few multiplications and additions as long as the size and shape of the PUF response is fixed. The z-scores produced this way by each device can then be gathered without security concerns and continuously tested for deviation from a normal distribution.

The result of the test for spatial autocorrelation also aids the selection of an adequate post-processing method. While it is generally advisable to optimize the PUF design for highest quality of the raw response, a complete removal of spatial correlations might not always be viable or the most cost or area efficient way. If the security concept of the device allows it, one might thus decide to allow for a certain amount of spatial autocorrelation in the raw data and select a post-processing that can cope with it. Vice versa, if the post-processing chosen for other reasons comes with the ability to compensate for spatial correlation, it is not required to achieve a raw response entirely free of spatial autocorrelation. For both situations it would be desirable to change the null-hypothesis to an assumption like "the spatial autocorrelation is sufficiently low", but this seems difficult, as the asymptotic distribution is very hard to determine for such cases. However, as our experiments have shown, the absolute value of the z-scores grows continuously to the amount of spatial autocorrelation in the dataset. It might thus be fair enough to enlarge the acceptance intervals to account for the ability of the post-processing to handle a certain extent of spatial autocorrelation.

VIII. Conclusion

In this paper we introduced SPACA to the field of PUFs and propose it to become a standard test for the assessment

[2]Note that an actual certification guideline should assess the result of SPACA with consideration of the application and overall system design. Also, further research into appropriate distance metrics and neighbourhood bounds would be necessary, which is outside the scope of this work.

of the quality of PUFs. This is motivated by our observation that canonical tests are insensitive to intra-device correlations. Therefore SPACA provides valuable insights. An example is given, for how SPACA can be combined, e.g. with the Bit-Alias test – as one representative for canonical tests – to distinguish variations of the means from true spatial autocorrelation.

Further, this paper provides the mathematical background behind SPACA tests and shows how to apply them on PUF data. The capabilities of SPACA are exemplified on three real-world PUF datasets to discuss the findings and limitations of the metrics. For the three discussed statistics (*Moran's I*, *Geary's c*, and the *Join Count statistic*) we conclude that Join Count statistic might be most beneficial in many PUF scenarios, because it is tailored to nominal data such as response bits. It can also be easily implemented even on resource constrained devices. Finally, the usability of the tests in real-world production testing is shown in this work.

ACKNOWLEDGMENT

Permanent ID and revision date of this document: b611a927a755bd61ec63c93b39888f1a225e0754 2018-1-11.

REFERENCES

[1] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. 44th Annu. Design Autom. Conf.*, 2007, pp. 9–14.

[2] Q. Chen, G. Csaba, P. Lugli, U. Schlichtmann, and U. Rührmair, "The bistable ring PUF: A new architecture for strong physical unclonable functions," in *Proc. IEEE Int. Symp. Hardware-Oriented Secur. Trust*, Jun. 2011, pp. 134–141.

[3] A. Maiti and P. Schaumont, "Improving the quality of a physical unclonable function using configurable ring oscillators," in *Proc. Int. Conf. Field Programm. Logic Appl. (FPL)*, Aug./Sep. 2009, pp. 703–707.

[4] 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.*, vol. 7. 2007, pp. 1–12.

[5] U. Rührmair and D. E. Holcomb, "PUFs at a glance," in *Proc. Design, Autom. Test Eur. Conf. Exhibit. (DATE)*, Mar. 2014, pp. 1–6.

[6] Y. Hori, T. Yoshida, T. Katashita, and A. Satoh, "Quantitative and statistical performance evaluation of arbiter physical unclonable functions on FPGAs," in *Proc. Int. Conf. Reconfigurable Comput. FPGAs*, Dec. 2010, pp. 298–303.

[7] A. Maiti, J. Casarona, L. McHale, and P. Schaumont, "A large scale characterization of RO-PUF," in *Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust (HOST)*, Jun. 2010, pp. 94–99.

[8] A. Maiti, V. Gunreddy, and P. Schaumont, "A systematic method to evaluate and compare the performance of physical unclonable functions," in *Embedded Systems Design with FPGAs*. Springer, 2013, pp. 245–267.

[9] T. Ignatenko, G.-J. Schrijen, B. Skoric, P. Tuyls, and F. Willems, "Estimating the secrecy-rate of physical unclonable functions with the context-tree weighting method," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2006, pp. 499–503.

[10] S. Katzenbeisser, Ü. Kocabaş, V. Rožić, A.-R. Sadeghi, I. Verbauwhede, and C. Wachsmann, "PUFs: Myth, fact or busted? A security evaluation of physically unclonable functions (PUFs) cast in silicon," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, 2012, pp. 283–301.

[11] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proc. 17th ACM Conf. Comput. Commun. Secur. (CCS)*, 2010, pp. 237–249. [Online]. Available: http://doi.acm.org/10.1145/1866307.1866335

[12] F. Ganji, S. Tajik, F. Fäßler, and J.-P. Seifert, *Strong Machine Learning Attack Against PUFs With no Mathematical Model*. Berlin, Germany: Springer, 2016, pp. 391–411. [Online]. Available: https://doi.org/10.1007/978-3-662-53140-2_19

[13] P. A. P. Moran, "Notes on continuous stochastic phenomena," *Biometrika*, vol. 37, nos. 1–2, pp. 17–23, 1950. [Online]. Available: http://www.jstor.org/stable/2332142

[14] R. C. Geary, "The contiguity ratio and statistical mapping," *Incorporated Statist.*, vol. 5, no. 3, pp. 115–127 and 129–145, 1954. [Online]. Available: http://www.jstor.org/stable/2986645

[15] P. A. P. Moran, "The interpretation of statistical maps," *J. Royal Stat. Soc. B, (Methodol.)*, vol. 10, pp. 243–251, 1948.

[16] F. Wilde, B. M. Gammel, and M. Pehl, "Spatial correlations in physical unclonable functions," in *Proc. 6th Conf. Trustworthy Manuf. Utilization Secure Devices (TRUDEVICE)*, Barcelona, Spain, 2016.

[17] B. Willsch, J. Hauser, S. Dreiner, A. Goehlich, and H. Vogt, "Statistical tests to determine spatial correlations in the response behavior of PUF," in *Proc. 12th Conf. Ph.D. Res. Microelectron. Electron. (PRIME)*, Jun. 2016, pp. 1–4.

[18] *PUF Quality Assessment Suite (PQAS), MATLAB Edition. Alternative Locations Can be Found by Searching for the Permanent ID of This Document*. [Online]. Available: https://gitlab.lrz.de/tueisec/PQAS

[19] M. Pehl, M. Hiller, and H. Graeb, "Efficient evaluation of physical unclonable functions using entropy measures," *J. Circuits, Syst. Comput.*, vol. 25, no. 01, p. 1640001, 2016.

[20] J. Delvaux, "Security analysis of PUF-based key generation and entity authentication," Ph.D. dissertation, Shanghai Jiao Tong Univ., Shanghai, China, 2017.

[21] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. 9th ACM Conf. Comput. Commun. Secur.*, 2002, pp. 148–160.

[22] A. D. Cliff and J. K. Ord, *Spatial Autocorrelation*. London, U.K.: Pion, 1973.

[23] A. D. Cliff and J. K. Ord, *Spatial Processes: Models & Applications*. London, U.K.: Pion, 1981.

[24] D. A. Griffith, *Spatial Autocorrelation: A Primer*. Washington, DC, USA: Association of American Geographers Resource, 1987.

[25] R. R. Sokal and N. L. Oden, "Spatial autocorrelation in biology: 1. Methodology," *Biol. J. Linnean Soc.*, vol. 10, no. 2, pp. 199–228, 1978.

[26] R. R. Sokal and N. L. Oden, "Spatial autocorrelation in biology: 2. Some biological implications and four applications of evolutionary and ecological interest," *Biol. J. Linnean Soc.*, vol. 10, no. 2, pp. 229–249, 1978.

[27] M. J. de Smith, *STATSREF: Statistical Analysis Handbook—A Web-Based Statistics Resource*. Winchelsea, U.K.: Winchelsea Press, 2011, accessed: May 25, 2017. [Online]. Available: http://www.statsref.com/HTML/index.html

[28] F. Wilde, "Large scale characterization of SRAM on Infineon XMC microcontrollers as PUF," in *Proc.4th Workshop Cryptogr. Secur. Comput. Syst. (CS)*, Stockholm, Sweden, Jan. 2017, pp. 13–18.

[29] M. Bucci and R. Luzzi, "Identification circuit and method for generating an identification bit using physical unclonable functions," U.S. Patent 8 583 710, Nov. 12, 2013.

[30] A. Maiti. (2011). *PUF Download Data. Virginia Tech*. Accessed: Sep. 27, 2017. [Online]. Available: http://rijndael.ece.vt.edu/puf/download.html

[31] C.-E. Yin, G. Qu, and Q. Zhou, "Design and implementation of a group-based RO PUF," in *Proc. Design, Autom. Test Eur. Conf. Exhib.*, Mar. 2013, pp. 416–421.

[32] C.-E. Yin and G. Qu, "Improving PUF security with regression-based distiller," in *Proc. 50th Annu. Design Autom. Conf.*, 2013, Art. no. 184.

[33] F. Wilde, M. Hiller, and M. Pehl, "Statistic-based security analysis of ring oscillator PUFs," in *Proc. 14th Int. Symp. Integr. Circuits (ISIC)*, Dec. 2014, pp. 148–151.

[34] Y. Hori. (2010). *Yohei Hori's Web Site—Profile*. Accessed: Sep. 27, 2017. [Online]. Available: https://staff.aist.go.jp/hori.y/en/puf/

Florian Wilde was born in Munich, Germany, in 1987. He received the Dipl.-Ing. degree (with high distinction) in electrical and computer engineering with a focus on embedded systems for renewable energy applications from the Technical University of Munich in 2013.

He is currently pursuing the Dr.-Ing. degree with a focus on quality assessment of physical unclonable functions with the Department of Electrical and Computer Engineering, Technical University of Munich. During his years of study, he was with, among others, Infineon Technologies AG, Munich, and TUM CREATE Ltd., Singapore. Since 2014, he has been with the Chair for Security in Information Technology, Department of Electrical and Computer Engineering, Technical University of Munich. His broader research is targeted at securing embedded systems against all kinds of attacks, be it local or remote, physical, or via software.

Berndt M. Gammel was born in Munich, Germany in 1963. He received the Dipl.-Phys. degree in theoretical solid-state physics from the Technical University of Munich, Germany, and the Dr. rer. nat. degree with a focus on conductivity in the quantum Hall effect, in 1994. Since 1998, he has been with Infineon Technologies AG, Munich, Germany, where he was a responsible architect for high security controllers, shifting his focus to cryptography, coding theory, and physical attacks on hardware. Since 2007, he has been leading the Smart Card Security Innovation Group. His current research projects are concerned with analysis of and countermeasures against side-channel, probing, and fault attacks.

Michael Pehl (M'14) was born in Munich, Germany in 1978. He received the Dipl.-Ing. degree in electrical engineering and information technology and the Dr.-Ing. degree *(summa cum laude)* from the Technical University of Munich in 2006 and 2012, respectively.

Since 2012, he has been a Researcher with the Chair for Security in Information Technology, Department of Electrical and Computer Engineering, Technical University of Munich, where he is leading the Physical Unclonable Functions Group.

His research has been concerned with tools for design automation of analog circuits as well as with different aspects of physical unclonable functions, such as evaluation of PUFs, error correction for PUFs, side-channel attacks on PUFs, and applications of PUFs.