HAL open science

Reliability of ring oscillator PUFs with reduced helper data

Julien Béguinot, Wei Cheng, Jean-Luc Danger, Sylvain Guilley, Olivier Rioul, Ville Yli-Mayry

▶ **To cite this version:**

Julien Béguinot, Wei Cheng, Jean-Luc Danger, Sylvain Guilley, Olivier Rioul, et al.. Reliability of ring oscillator PUFs with reduced helper data. 18th International Workshop on Security (IWSEC 2023), Aug 2023, Yokohama, Japan. hal-04136988

HAL Id: hal-04136988

https://telecom-paris.hal.science/hal-04136988v1

Submitted on 6 Jul 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Reliability of Ring Oscillator PUFs with Reduced Helper Data

Julien Béguinot[1][0009−0000−0729−6965], Wei Cheng[1,2][0000−0001−9433−7576], Jean-Luc Danger[1][0000−0001−5063−7964], Sylvain Guilley[1,2][0000−0002−5044−3534], Olivier Rioul[1][0000−0002−8681−8916], and Ville Yli-Mäyry[3][0000−0002−2435−0350]

[1] LTCI, Télécom Paris, Institut Polytechnique de Paris, 19 place Marguerite Perey, Palaiseau, 91120, France. firstname.lastname@telecom-paris.fr
[2] Secure-IC S.A.S., 104 Bd du Montparnasse, Paris, 75014, France. firstname.lastname@secure-ic.com
[3] Secure-IC K.K., 2-15-1 Konan, Minato-ku, Level 28 Shinagawa Intercity Tower A, Japan. ville-oskari.ylimayry@secure-ic.com

Abstract. Enhancing the reliability of natively unstable Physically Unclonable Functions (PUFs) is a major requirement when the PUF is to generate secret identifiers like cryptographic keys. One traditional method is to rely on an addition of a public word: the Helper Data. However, it involves extra complexity and constitutes a vulnerability against attacks manipulating it. In this work, we show that for PUFs based on oscillations, such as Loop-PUFs (LPUF) can simultaneously increase the stability of the PUFs responses and reduce the required amount of helper data to decrease the complexity and increase the security. We proceed in two steps: First, we improve the reliability of the LPUF using dynamically determined repeated measurements and decision process. The number of repetitions per challenge is automatically tuned according to its reliability level and measurement window. Second, we investigate lightweight helper data (less than one byte). Experimental validation of our approach is carried out on 640 LPUFs to characterize the PUF reliability under different temperatures. This provides the assessment of the probability that a given Key Error Rate (KER) is achieved. This, in turn, yields the probability that there is an oscillator with arbitrarily low KER among any given number of oscillators. Performances remain notably stable when subject to increasing temperature.

Keywords: Ring Oscillator PUF · Reliability · Adaptively Controlled PUF · Sequential Probability Ratio Test · Lightweight Helper Data.

1 Introduction

When using cryptographic primitives, cryptographic keys are at the basis of encryption, digital signatures, etc., and their security is of utmost importance. Traditionally, physically adding a key to a device and providing key storage is delegated to the fabrication plants, assembly factories, and other third parties. Malicious subcontractors may potentially change, record, or alter the keys provided

to the device. Furthermore, keys embedded in silicon, or programmed into one-time-programmable memory can be read-back by reverse-engineering after production.

Physically Unclonable Functions (PUFs) have been introduced to avoid these problems. Taking advantage of minute manufacturing variations, the PUF gives a "close to random" *response* output from a *challenge* input. Often, however, the response data of various PUF constructions show some instability: Given a challenge, the PUF output exhibits some dynamic noise, so that the output for a given challenge input is not consistently the same. Such noise is essentially due to transistor-level noise and environmental sources (uncontrollable fluctuations in supply voltage, temperature variations, etc.).

The idea of using additional *helper data* to aid in PUF robustness was first introduced in works by Linnartz et al. [28] and Dodis et al. [12]. However, the use of helper data has various drawbacks: additional cost, PUF output bias conditioned on the helper data, manipulation attacks, etc. These act as motivation for designing PUFs with less helper data.

In this article, we present a method applicable to Ring Oscillator PUFs like Loop-PUFs (LPUFs) that allows to trade an increase in latency during key reconstruction to mitigate the problems induced by helper data. For that we proceed in two steps. First we improve the reliability of the LPUF [6] by adaptively controlling the number of required oscillations (as hinted in [21]). Second, we investigate the performances of lightweight helper data (less than one byte).

1.1 Related Works

Four previous works try to enhance PUF reliability without using helper data:

- Che et al. [5] present a physically unclonable function without helper data based on non-volatile memory. It is nonetheless questionable whether this construct is a PUF, as it keeps its value even when not powered.
- Wang et al. [43] leverage locally enhanced defectivity of *direct self assembly* to generate stable PUFs without helper data. Their construction differs from classical constructions in that it is not parametric but relies on random but permanent connection in the hardware layout.
- Herkle et al. [21] present an eye-opening oscillator for arbiter PUFs, which exploit the dead-zone of the arbiter PUF to decide whether a bit is reliable or not, and automatically request new oscillations accordingly. Monte-Carlo simulation with transient noise and 50 repetitions yields an expected Bit Error Rate (BER) of $9.2 \cdot 10^{-5} \pm 7.7 \cdot 10^{-4}$. However, this work is not validated experimentally and does not propose a mathematical model (only heuristics).
- Temporal majority voting improves the reliability of a decision without helper data [10]. The PUF is repeated a given (odd) number of times and the key is obtained by majority vote at the bit level. This amounts to using a repetition code over time.

Fueller et al. [14] proposed a construction for computational fuzzy extractor based on the learning with error problem. Based on Fueller construction, Herder et

al. [20] construct a computational fuzzy extractor based on the Learning Parity with Noise (LPN) computational hardness. As in our work, these constructions use side information retrieved on the fly as a trapdoor.

1.2 Contributions

This paper has four main contributions.

1. Using a stochastic model, we determine the survival function of the BER of the ring-oscillator based LPUF consolidating results from Schaub et al. [35].
2. By controlling adaptively the number of repetitions of a given oscillator, we then improve the reliability of the LPUF at the cost of higher latency. The reliability of the proposed system can be set by the user. A feedback-based mode of operation monitors each bits independently and decides when it is reliable enough in order to minimize the global Key Error Rate (KER). This formalizes the eye-opening concept used in Herkle et al. [21].
3. We validate our analysis on a hardware implementation on FPGA. We analyze through experiments on 640 different oscillators and different controlled temperatures (using a climate chamber) the results of the design.
4. We evaluate performances of a set of different lightweight helper data on our design, and also present a case where design is used without any helper data.

1.3 Outline

The paper is organized as follows. Sec. 2 recalls the vanilla LPUF design [6] and introduces our novel adaptive design. Sec. 3 addresses reliability in the context of LPUF and provides the distribution of the BER. Sec. 4 specifies the adaptive design threshold function and grounds it theoretically with its relation to sequential analysis. Sec. 5 integrates our design within a full fledged PUF and real measurements show that it achieves satisfactory reliability with lightweight helper data. Namely, we show how to reach arbitrary high reliability.

1.4 Notations

We use the following notations. Random variables are denoted in upper case and their realizations in lower case. The probability density function (p.d.f.) of a random variable X is denoted by f_X or simply f if not ambiguous. The survival function is denoted Φ_X or Φ. The p.d.f. of the standard normal distribution is denoted by ϕ and its survival function by Q. Vector values are in typeset bold and sets are denoted with calligraphic symbols. The XOR operation is denoted $\oplus$ and operates bit-wisely on vector values.

2 LPUF Model

2.1 Model for a Single LPUF (Vanilla LPUF)

Ring oscillator PUF designs were first suggested by Gassend et al. [15, 16]. A number of works followed this design such as [29, 30, 39, 44]. A LPUF [6] is a

reconfigurable ring oscillator made up of n delay elements. That is, for a fixed challenge word $c \in \{0,1\}^n$ the ring oscillator outputs a noisy observation of the sum of the configured delays $\sum_{i=1}^{n} d_i^{c_i}$, where $d_i^{c_i}$ denotes the corresponding delay of the i-th element when challenged by a bit $c_i \in \{0,1\}$. Typically, each c_i selects one of two possible paths, and the delay $d_i^{c_i}$ is modeled as Gaussian with unknown mean and fixed variance.

The LPUF operates differentially to eliminate part of the noise due to environmental conditions (temperature, voltage, etc.) and to center the delay measurements. It measures the delay for both challenge codeword c and its complementary $\bar{c}$, and outputs the difference of the two delays. The resulting LPUF output on challenge word c is $\Delta_c = \sum_{i=1}^{n}(d_i^{c_i} - d_i^{\bar{c}_i}) = \sum_{i=1}^{n}(-1)^{c_i}\Delta_i$, where the $\Delta_i = d_i^0 - d_i^1$ are normally distributed with zero mean and fixed variance $\mathcal{N}(\delta = 0,\, \tau^2)$. Typically, the random bit generated by the PUF is $B_c = \mathrm{sign}(\Delta_c)$.

The dynamic noise can be described as flicker noise (a.k.a. $1/f$-noise or pink noise) which is a low frequency noise arising from the transistor commutation in the ring oscillator. For simplicity it is considered here to be zero-mean additive white Gaussian noise (AWGN) $\mathcal{N}(0, \sigma^2)$.

2.2 System Modeling for LPUF

We consider the following system modeling for the LPUF, that subjects to

1. **Randomness** measured by entropy: $H(\mathbf{S}|\mathbf{W}) \geq E$ (or $H_\infty(\mathbf{S}|\mathbf{W}) \geq E$)
2. **Reliability** subject to two parameters α, β: $\mathbb{P}_{PUF}(\mathbb{P}(\mathbf{S} \neq \hat{\mathbf{S}}|PUF) \geq \alpha) \leq \beta$
3. **Lightweight**: low memory and efficiently computation.

There are four degrees of freedom to optimize these three properties:

Challenge Code. The LPUF is restricted to a given set $\mathcal{C} = \{c_1, \ldots, c_M\}$ of M challenges which forms a binary code of length n and size M. In this work we focus on weak PUF or physically obfuscated key with a small number of challenges. Rioul et al. [34] show how to select optimally the first $M = n$ codewords. At most $M = n$ challenges with independent responses can be selected, and up to equivalence, these challenges are given by the Hadamard code of length n. The entropy can be increased by selecting more challenges, yet the orthogonality cannot be preserved anymore. Solé et al. [36] suggested that selecting the vector that minimizes the deviation from the family of vector is optimal with respect to the entropy. In the sequel we restrict the PUF input to $2^6 = 64$ Hadamard challenges for two reasons: (a) it is easy to construct Hadamard matrices of size 2^i by iterative tensor product, which facilitates hardware implementation; (b) LPUF are prone to modeling attacks (with machine learning techniques) when the challenges are not orthogonal.

Helper Data Algorithms [28]. These include error correcting codes [10], bit selection [10,35] or even zero-leakage helper data [17,18,38]. From a theoretical point of view with the framework of secure sketch, fuzzy extractor [12], robust fuzzy extractor [3, 11] and computational fuzzy extractor [14, 20] are used.

Quantization Procedures are used for zero-leakage helper data [18, 38], Two-Metric helper data [9], modelling resistance [37] or to increase entropy [25]. In this work, for simplicity, we only keep the most significant bit (i.e. the sign) of the output.

Noise Channel. Assuming a binary symmetric channel (BSC), information theoretic limits can be derived for the code-offset construction [19,24]. Maringer et al. [31] suggest an improved model with varying BSC and shows that channel state information increases the capacity of the PUF by more than 25% compared to the case where no such information is available.

The main objective of this paper is to design an adaptive procedure that modifies the noise channel to improve the reliability of the LPUF. For this aim, we add a feedback link to the channel and request retransmissions to achieve a given reliability constraint. As a byproduct we use less helper data. The system modeling of LPUF is depicted in Fig. 1 and the generic procedure is described in Alg. 1.

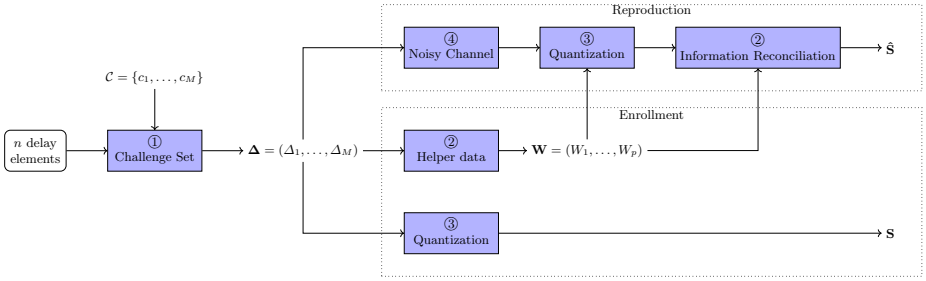


Fig. 1: System modeling of the LPUF

Algorithm 1: Adaptive LPUF

Data: A threshold function $A : \mathbb{N}^* \mapsto \mathbb{R}^+$, a maximal number of repetitions T_{max} and PUF which can be called by a codeword c of a fixed code $\mathcal{C}$.
Result: LPUF key whose reliability is ensured by A.

```
for c ∈ C do
    Δ_acc ← 0                                  /* Accumulated Measurements */
    t ← 0                                      /* Number of measurements */
    ok ← 0                                     /* Feedback */
    while not ok do
        Δ_acc ← Δ_acc + PUF(c)                 /* New measure */
        t ← t + 1
        ok ← (|Δ_acc| ≥ A(t) or t ≥ T_max)     /* Feedback exit condition */
    B_c ← (Δ_acc ≥ 0)                          /* Final quantization */
return (B_c)_{c∈C}
```

2.3 Hardware Description

The proposed vanilla LPUF design FPGA architecture is depicted in Fig. 2. It is implemented in a Basys3 board relying on Artix-7 FPGA. We repeated the

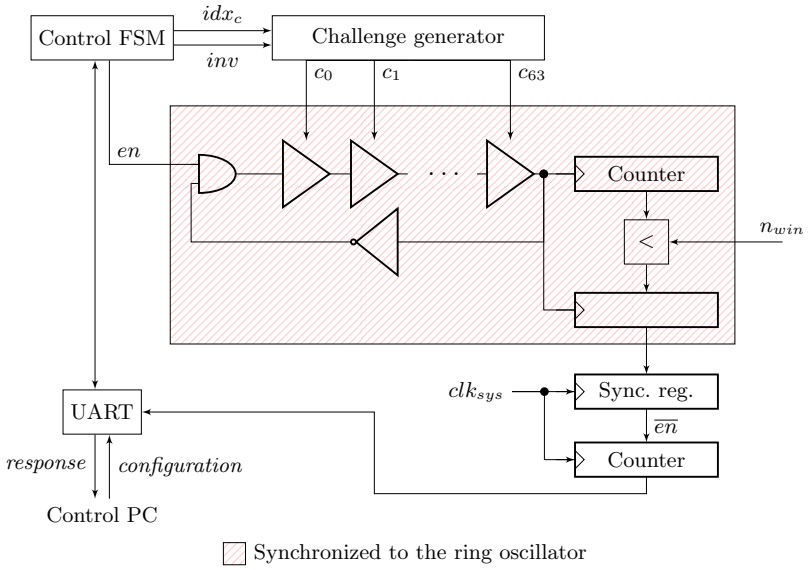


Fig. 2: Overview of the experimental PUF hardware implementation. Elements depicted inside the red area are timed with the ring oscillator as the clock signal.

test on 5 instances of the same board (see Fig. 8). We expect the results to be consistent with different placement within the same FPGA and with different FPGAs. Namely even if the signal to noise ratio may differ from one instance to another the mechanism should work alike. The PUF interface uses the UART communication with a control PC. The PUF core consists of a single ring oscillator implemented with 64 5-input LUTs as delay elements. As the delay element requires only two inputs (the signal to be delayed and the challenge bit), this leaves three bits, which we call "pins" subsequently, to configure the 8 delay paths inside each look-up table. Hence, a single ring oscillator can have 8 different delay chains by the pins configuration. In Fig. 2, the area shaded in red denotes the parts of the implementation that are synchronized to the ring oscillator. Registers and logic outside of the shaded area are synchronized to the system clock of the FPGA design. The design can be configured with window size n_{win}, which defines the number of LPUF periods the design uses to time the counter (synchronized to the system clock) interval.

During the ring oscillator measurement, first, a challenge is chosen from the possible challenges. Its bits are used to configure the delay elements. Then, the delay loop is activated and the amount of pulses during a given time frame (n_{win}) is counted. Next, the *inv* bit in Fig. 2 is set, and the inverse of the same challenge is used to evaluate the ring oscillator loop count. Finally, the difference of the counts is considered the result of the measurement for the given challenge.

For each Basys3 boards with 16 oscillators and 8 pins configurations and the control logic, the proposed design required 13585 LUTs slices, 7844 F7 Muxes slices, 242 F8 Muxes slices and 16 block RAM. The synthesizer tool indicates that each oscillator requires a power of about 11 mW.

3 Reliability Analysis

3.1 Reliability of the LPUF

For a fixed challenge c, the bit error rate BER_c of this challenge is the probability that the bit output flips, and the global key error rate KER is the probability that *at least* one bit of the key flips. Since the challenges are assumed independent one has

$$KER = 1 - \prod_{c \in \mathcal{C}} (1 - BER_c), \qquad BER = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} BER_c. \tag{1}$$

This shows that at low expected bit error rate the expected key error rate scales linearly with n. Lemma 1 shows that for a fixed average BER, the KER is minimized when all bit error rates are equal:

Lemma 1 (Equal BERs). *One has $KER \geq 1 - (1 - BER)^n$ with equality if and only if all bit error rates BER_c are equal.*

Proof. Arithmetic-geometric mean inequality applied to the $1 - BER_c$. □

Our first take-away is that for a fixed average BER we should target uniform BERs for all the challenges.

The delay difference Δ_c is drawn from $\mathcal{N}(\delta, \tau^2)$. The design sequentially requests new measurements $X_1, X_2, \ldots$ which are i.i.d. $\mathcal{N}(\Delta_c, \sigma^2)$. The goal is to decide optimally the sign of Δ_c. This can be reformulated as a Bayesian statistical hypothesis testing problem: $H_0 : \Delta_c \geq 0$ against $H_1 : \Delta_c < 0$. At step t, $X^t = \sum_{i=1}^t X_i$ is stored and the other measurements $\mathbf{X} = (X_1, \ldots, X_t)$ are discarded without loss of optimality, since X^t is an *sufficient statistic*:

Lemma 2. *$X^t = \sum_{i=1}^t X_i = S(\mathbf{X})$ is an sufficient statistic of the parameter $\theta = sgn(\Delta_c)$.*

Proof. Apply the Fisher-Neyman factorization theorem:

$$f_\theta(\mathbf{X}) = \prod_{i=1}^t e^{-\frac{(\theta|\Delta_c| - X_i)^2}{2\sigma^2}} = \exp\left(-\frac{1}{2\sigma^2} \sum (\Delta_c^2 + X_i^2) - \frac{\theta|\Delta_c|}{\sigma^2} X^t\right) = h(\mathbf{X}) g_\theta(S(\mathbf{X})).$$

The second take-away is that for white noise the optimal decision is based only on the sum of the past delays. In particular, for an optimal design it is enough to store this sum. The Bayes decision for the problem is to take the sign of this statistic. We recall the expression for the expected BER and KER when this optimal procedure is used assuming that the delays are not biased ($\delta = 0$):

Lemma 3 (Expected BER [35]). *The BER and expected BER with t repetitions obtained with the same derivations as Schaub et al. are*

$$BER_c(t) = Q\left(\frac{|\delta_c|}{\sqrt{t}\sigma}\right) \qquad \mathbb{E}[BER](t) = \frac{1}{\pi} \arctan\left(\frac{1}{\sqrt{t\gamma}}\right). \tag{2}$$

where $\gamma = \tau^2 \sigma^{-2}$ denotes the signal-to-noise ratio (SNR).

From these expressions we can go further in the analysis by deriving the probability density function of the bit error rate and its survival function:

Theorem 1 (BER Distribution). *The BER p.d.f. is*

$$f_{BER}(u) = 2(t\gamma)^{-\frac{1}{2}} \exp\left(-\frac{Q^{-1}(u)^2}{2}((t\gamma)^{-1} - 1)\right). \tag{3}$$

If the SNR $\gamma = 1$ the bit error rate is uniformly distributed in $[0, \frac{1}{2}]$, else $f_{BER}(0) = +\infty$ and $f_{BER}(\frac{1}{2}) = 2(t\gamma)^{-\frac{1}{2}}$. The bit error rate survival function is

$$\Phi_{BER}(u) = 1 - 2Q((t\gamma)^{-\frac{1}{2}} Q^{-1}(u)) \overset{t\gamma \to \infty}{\approx} \sqrt{\frac{2}{\gamma \pi t}} Q^{-1}(u). \tag{4}$$

Proof. The survival function is derived from Lemma 3 and the distribution of Δ_c. The p.d.f. is obtained from the survival function by computing its derivative.

Corollary 1. $\mathbb{P}(BER < \alpha) = \beta$ *for* $\gamma t = Q^{-1}(\alpha)^2 Q^{-1}(\frac{1-\beta}{2})^{-2} \overset{\beta \to 0}{\approx} 8\pi Q^{-1}(\alpha)^2 \beta^{-2}$.
Proof. Apply Thm. 1 and Taylor expansion of Q^{-1} about $\frac{1}{2}$.

The expression of Thm. 1 lead to our third take away:

- First, it shows that the bit error rate distribution does not vanish in $\frac{1}{2}$ and is bounded away from 0 by a factor $2(t\gamma)^{-\frac{1}{2}}$. This "large tail" is detrimental to the the PUF. Even with large SNR we cannot get rid of very bad bit error rate. Increasing the SNR and number of repetitions t is not enough to get rid of all errors. That is increasing n_{win} is not enough to remove all errors.
- Second, the expression of the survival function enables us to compute the probability β that the bit error rate of a challenge exceed a targeted bit error rate α as shown in Thm. 1. This scales to a n bit key since the probability β_n that the worst bit error rate of the key exceed α is $\beta_n = 1 - (1-\beta)^n \approx n\beta$.

3.2 Comparison with Temporal Majority Voting

Temporal majority voting is a technique introduced to make a decision more reliably [10]. It repeats measurements t times and decide the bit by majority voting. If there are t repetitions and a bit error rate of BER, then the new corrected error is given by the survival function of the binomial distribution of parameter n and parameter BER evaluated in $\frac{n-1}{2}$ [10]. This technique is relevant when no information about the reliability is available. It is yet sub-optimal for the LPUF for which a side information on the reliability of the bit is available. Indeed, for temporal majority voting we have $BER(t) = \sum_{t=\frac{n-1}{2}}^{n} \binom{n}{t} BER(1)^t (1 - BER(1))^{n-t}$, and with our method (Bayes decision), using the equation of Lem. 3, we have the exponentially better evaluation:$BER(t) = Q(\sqrt{t} Q^{-1}(BER(1)))$.

Our fourth take away is that while temporal majority voting makes sense for PUF constructions without a "confidence information" about the PUF outputs for oscillation based PUF it is preferable to use the optimal strategy based on the sign of the sum of the accumulated difference.

4 Sequential Probability Ratio Test (SPRT)

We now refine the previous approach where the PUF estimates the reliability of the bits on the fly. This requests repetitions for each bit until a certain reliability is achieved. Instead of fixing the number of repetitions in advance, the number of repetitions is chosen adaptively to minimize the required number of sample for a fixed targeted probability of error. The number of repetitions is *not* constant, but found as a (S_t)-*stopping time* denoted T which depends only on (S_t). The requested quality of service imposes some constraints on T. In particular, the number of repetitions should be bounded and not too long on average: $\mathbb{P}(T \leq T_{max}) = 1$ and $\mathbb{E}[T] \leq T_{avg}$.

4.1 Sequential Analysis

We recall that the posterior distribution of Δ_c given X^t is normally distributed:

Lemma 4 (Ex 3.7 [13]). *The delay Δ_c given the observation $X^t = x$ is normally distributed with mean $\delta_{t,x}$ and variance τ^2_{tx} given by $\delta_{t,x} = \frac{x\tau^2+\sigma^2\delta}{\tau^2 t+\sigma^2}$ and $\tau^2_{t,x} = \frac{\sigma^2\tau^2}{t\tau^2+\sigma^2}$. Interestingly, the posterior variance does not depend on the observations but only on the prior distribution and the number t of repetitions.*

Proof. By factorization under canonical form: Using Bayes' formula,

$$\log f_{\Delta|X^t=x}(u) \propto \frac{(u - \delta)^2}{\tau^2} + \frac{(ut - x)^2}{t\sigma^2} = u^2 \underbrace{(\tau^{-2} + t\sigma^{-2})}_{\tau^{-2}_{t,x}} - 2u \underbrace{(\delta\tau^{-2} - x\sigma^{-2})}_{\tau^{-2}_{t,x}\delta_{t,x}} + cst.$$

This expression can be simplified by making a change of variable as suggested in [2]. Let $t_0 = \gamma^{-1}$ and $T = t + t_0$. The normalised process $S_t = \frac{X^t+t_0\delta}{\sigma\sqrt{T}}$ is also a sufficient statistic; the posterior distribution of Δ_c given $S_t = s$ is now given by $\delta_{t,s} = \frac{\sigma s}{\sqrt{T}}$ and $\tau_{t,s} = \frac{\sigma}{\sqrt{T}}$. From this expression we can compute the posterior probability of H_0 and H_1 as given in Th. 2.

Theorem 2. *One has $\mathbb{P}(H_0|S_t = s) = 1 - \mathbb{P}(H_1|S_t = s) = Q(-s) = 1 - Q(s)$, and In particular the Bayes decision at step t is to decide the sign of Δ_c to be the sign of S_t and the posterior probability of error is $\mathbb{P}_e(s) = Q(|s|)$, which corresponds to a Neyman-Pearson test of H_0 against H_1.*

Proof. Immediate from Lemma 4.

We recall that Neyman-Pearson test is the most powerful test[4] at level α to test simple hypothesis. By Karlin-Rubin theorem [27], it is also universally most powerful test for composite hypothesis whose likelihood is non decreasing.

The expression of Thm. 2 can help to correct error more efficiently using this information for soft decoding [10]. It was already observed that for ring oscillator a confidence information could be re-derived [20]. This is one advantage of the LPUF over SRAM-PUF for which deriving a probability of error seems harder as it would require multiple power up.

[4] The power of a statistical test is the probability that it correctly rejects H_0.

4.2 Wald's Sequential Probability Ratio Test & Constant Boundary

A first approach is to force the probability of error to reach a given level α before taking a decision as suggested in Sec. 3 with Lemma 1. Using Thm. 2 this is equivalent to set $\alpha \geq Q(|S_T|)$ i.e. $|X^T| \geq \sigma Q^{-1}(\alpha)\sqrt{T}$.

Proposition 1 (Square Root Boundary). *If $A(t) = \sigma Q^{-1}(\alpha)\sqrt{\gamma^{-1} + t}$ in Alg.1 then the BER of the LPUF is at most α.*

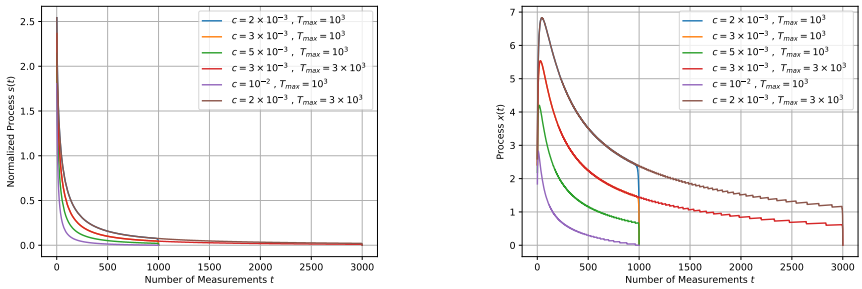
This is equivalent to set the likelihood ratio of the two hypothesis to a given level. We recognize Wald's sequential probability ratio test (SPRT) [41]. Wald's SPRT has been proven to be optimal for testing simple hypothesis [42]. If a sequential test achieves the same probability of detection and false alarm it requires at least as many measurements as Wald's test. Though, contrary to Neyman-Pearson test, the optimality result does not extend to composite hypothesis testing. This indicates that the procedure can be improved. For the LPUF, the SPRT is truncated since $T \leq T_{max}$ is imposed. As a consequence, the bit error rate is higher since a proportion β of the challenges is not stabilized with the SPRT as they require more than T_{max} measurement repetitions on average. This proportion is $\beta = 1 - 2Q(\frac{\sigma Q^{-1}(\alpha)}{\sqrt{T_{max}}}) \approx \sqrt{\frac{2\sigma^2 Q^{-1}(\alpha)^2}{\pi T_{max}^2}}$. It is useless to decrease further α if a high proportion of the challenges cannot be stabilized. If we target $\beta \approx \alpha$, we obtain that we should have $T_{max} \approx \sqrt{\frac{2}{\pi}}\frac{\sigma Q^{-1}(\alpha)}{\alpha}$.

The simplest sequential test to implement is the constant boundary. The probability β that the expected measurement time for a challenge exceed T_{max} is $\beta = 1 - 2Q(\frac{a}{\sqrt{\gamma T_{max}}}) \approx \sqrt{\frac{2a^2}{\pi\gamma T_{max}^2}}$.

4.3 Improving the Boundary

Both Wald SPRT test and the constant boundary are sub-optimal solutions. Some works to test the sign of the mean of a Gaussian process or to test the sign of the drift of a Wiener process already exists [2, 4, 7, 8]. These works explores diverse prior assumptions, and loss functions. The main approach is to consider this problem as a Markov decision problem where the terminal cost function is the probability of error with a constant sampling cost $c > 0$ and to approximate the process as a drifted Wiener process. With the recursive equation provided in [2] we computed an approximation of the optimal solution in the finite horizon case for different sampling cost as shown in Fig. 3. If Wald's test was optimal, then the solution in the s state would be a constant. Results show that this is not the case. If we look at the x state representation, we observe that the optimal boundary increases rapidly to a maximum and then decreases slowly to zero. Moreover, the horizon T_{max} does not affect the shape of the boundary.

Bather [2] shows that an asymptotic for the optimal boundary is $(8\pi ct)^{-\frac{1}{2}}$ for large values of t. Chernoff [4, 7, 8] obtains a similar first order term with a terminal cost function proportional to the amplitude of the drift/delay.



(a) Results in the s state space representation.

(b) Results mapped back into the X^t state space representation.

Fig. 3: Results with dynamic programming for different sampling costs and horizon.

5 Combining with Lightweight Helper Data

We repeat the vanilla LPUF measurement 10^3 to verify how the output changes over the time and show the results in Fig. 4. We see that the responses are stable in time and perturbed by noise. We can visualize when errors happen i.e. when the trajectory for a given challenge crosses the line $y = 0$. We consider two different window-sizes $n_{win} = 2^{12}, 2^{19}$ and observe that even by multiplying the window-size by 2^7 the LPUF output is not stabilized even if the results improve. These results have to be compared with the result of the adaptive design for both boundaries. Compared to the classical design where increasing n_{win} "scales" the LPUF outputs the new design "push away" the delay difference close to zero. On two sub-figures on the right we see that their remain some bits that could not be stabilized by the design. In the first case (square root boundary) there was not enough measurements for the delays. As some errors persists we investigate a set of small helper data.

5.1 Bit Flipping

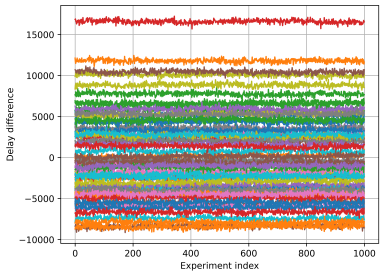
The posterior probabilities of error can be used as a side information to construct the list of the k most likely key from the PUF outputs. We validate this approach by computing the expected number of guesses necessary to find the correct key. This quantity is known as *guessing entropy* and introduced by Massey for cryptographic purposes [32]. Using the lower bound of Arikan [1] improved by Rioul [33] we obtain when $\gamma \neq 1$, $\frac{1}{\ln(2^{n+1}+1)}\left(1 + 4\int_0^\infty \phi(\delta)\sqrt{Q(\delta\sqrt{\gamma})Q(-\delta\sqrt{\gamma})}\,\mathrm{d}\delta\right)^n \leq GE \leq \frac{1}{2}\left(1 + 4\int_0^\infty \phi(\delta)\sqrt{Q(\delta\sqrt{\gamma})Q(-\delta\sqrt{\gamma})}\,\mathrm{d}\delta\right)^n + \frac{1}{2}$. For $\gamma = 3 \cdot 10^2$, $5 \cdot 10^2$ and 10^3 it follows that 47, 18 and 7 trials are respectively required on average.

5.2 Correction With Few Bits of Parity

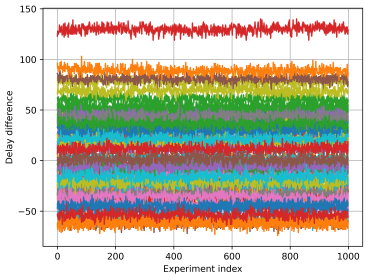
We suggest to use few bits of parity to improve the reliability of the LPUF design. The syndrome of the PUF output is stored as a helper data [10, 12]. A code offset construction would work as well but it would require to store a whole code-word as

helper data. The experiments performed in this section are computed with the 64 challenges of 5 Basys 3 FPGA boards with 16 independent LPUF implementations in parallel using 8 different internal routing configurations. As a reference Van Herrewege et al. [22] use a $[255, 21, t = 55]_2$ BCH code with 234 bits of helper data correcting up to 55 errors for a key of length 255.

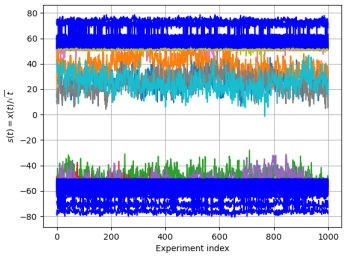
Parity Check Code The parity check code requires a single bit of redundancy. Since its minimal distance is 2, it can detect one error but cannot correct any. However, the parity check code can correct one erasure. We erase the least reliable bit using the reliability information of the LPUF and correct this erasure. This is related to the idea of soft decoding for PUF as suggested by Delvaux et al. [10].



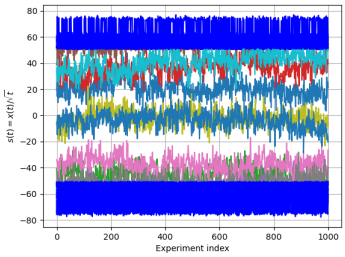
(a) Vanilla LPUF, window-size 2^{19} clock-cycles.



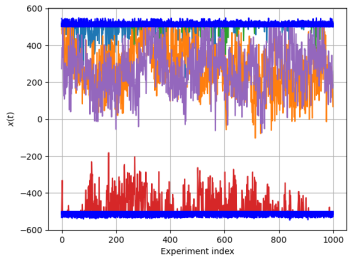
(b) Vanilla LPUF, window-size of 2^{12} clock-cycles.



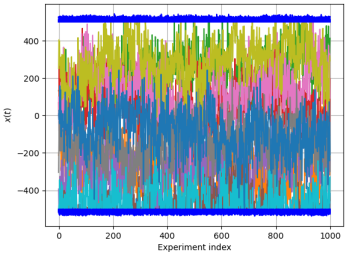
(c) Oscillator in board 1, PUF 14, pins 2, stabilization with a square root threshold 50.



(d) Oscillator in board 3, PUF 14, pins 0, no stabilization with a square root threshold 50.



(e) Oscillator in board 1, PUF 14, pins 2, stabilization with the constant threshold 500.



(f) Oscillator in board 3, PUF 14, pins 0, no stabilization with the constant threshold 500.

Fig. 4: Example of results of the LPUF design with two types of boundary thresholds.

Parity Check Codes in Parallel (PCP) The n bits output of the LPUF is split into N (where N divides n) blocks of size N/n, and a parity check code in each block is applied . The idea of codes in parallel was first suggested by Delvaux et al. [10]. This scheme requires N bit of parity. We apply the decoding procedure for a parity check code in each of the N blocks. We tried this approach with $N \in \{1, 2, 4, 8, 16\}$ blocks.

Single Error Correction Double Error Detection (SECDED) code. We can consider a SECDED code [23] which requires 8 bits of helper data. We consider a SECDED code of length 64 punctured from the $[127, 120, 3]_2$ Hamming code and extended with a parity check equation. In particular, the minimum distance of this code is at least 4. For this approach we investigate two different decoders.

The first decoding procedure (SECDED-ALG) is as follows. We compute the syndrome of the reproduced keys and xor it with the helper data. If there is no error we return the key as is. Else there are two cases:

- If the last parity-check bit is a one, there is an odd number of error. If the syndrome correspond to a one error pattern, the bit indicated by the syndrome is flipped. Else, there is a least three errors. A pair of unreliable bits is flipped until a one error pattern is found.
- If the last parity-check bit is even, there is an even number of errors. One unreliable bit is flipped until the syndrome corresponds to a one error pattern.

The second decoding procedure (SECDED-ML) does not exploit the algebraic structure of the code. It enumerates key candidates "by decreasing order of likelihood" until a syndrome without error is found as already suggested in [10]. This second option provides better results but requires a bit more computations. It differs from [10] exhaustive enumeration in that we try key candidates by decreasing likelihood. As the guessing entropy computed in the previous section is small, the computation overhead is limited on average contrarily to a naive exhaustive enumeration.

Hash of the key ? For the SECDED code we suggested an exhaustive search decoder which does not use the algebraic property of the code. The code structure is not compulsory to correct the PUF output. As a proof of concept, we verify that the key can be found if the a hash of the key (SHA256) is stored as helper data. This is not information theoretic secure but computationally secure if the hash function is one way. The underlying idea is to consider cases where successive keys in a list can be tried. For instance, if the key is used to decrypt and AES-GCM, different keys can be tried until success. The main issue here is that this implies a larger computational overhead. In a sense, this is a computational fuzzy extractor scheme. Eventually, the construction based on the LPN problem would be more relevant here. Sieving a list of key candidates with a "tag" was first suggested by Dodis [12], our suggestion differs in that we don't sieve a list obtained by the list decoding of a code but rather sort the key candidates by decreasing likelihood.

Configuring the LPUF in the Most Favorable Pin Configuration The LPUF design is a re-configurable ring oscillator whose delay elements are implemented by LUT-5. 2 wires are used for the challenge and output which leaves 3 wires for configuring the LPUF (8 configurations). One way to improve the LPUF results is to use the most favorable pin configuration out of m configurations. For an ASIC implementation, we could consider the best oscillator among a list of different oscillators. This selection has two beneficial impacts on the design. It reduces the key error rate, and also reduces the latency of the design.

5.3 Key Error Rate of the Adaptive Design

The performance is estimated with 1000 iterations for the 640 available oscillators with $n_{win} = 2^{10}$ and maximum 300 repetitions. Experimental results are shown in Fig. 5 and 6.

The average KER obtained at 30 degrees is about 0.37 without any helper data, 3.1×10^{-2} with SECDED-ML and 6.4×10^{-3} with a tag (SHA256). If we choose the best pin configuration among the 8 pins configuration the KER reduces to 1.5×10^{-3} with SECDED-ML. The figures can be read as follows: by looking for a given targeted KER on the y-axis we obtain on the x-axis the probability that an oscillator achieves a KER lower than the targeted KER. For instance, for SECDED-ML the probability that the KER is lower than 10^{-3} is little less than 0.8. The ideal case corresponds to a vertical line in $w = 1$ and the worst case corresponds to the horizontal line $y = 1$. It happens that some vertical curves super-impose in the final vertical line (in Fig. 6).

We do not observe significant changes in performances between the square-root boundary and the constant boundary. Hence we advocate to use the simpler constant boundary as introduced by Herkle et al. [21]. It is interesting to observe that despite higher temperature should increase the KER because of larger noise variance the performances are barely affected by the temperature. This shows that the proposed design compensate well larger temperature by dynamically adapting the number of required repetitions to achieve the targeted reliability. Selecting the most reliable oscillators among a list of m oscillators enables to reduce arbitrarily the KER as shown in Fig. 6. Equivalently, discarding bad oscillators greatly improves the performances. At 30 degrees with SECDED-ML the average KER of the 79% best oscillators reduces to 10^3.

Admittedly, removing all helper data while maintaining a very low key error rate seems impossible in general. The proposed design has some limits:

- The impact of repetitions on low frequency noise is limited. Especially its impact on the Pink noise arising from transistor commutations is limited.
- We do not claim that the design prevent errors due to unstable power supply.

The number of measurements at 30 degrees is shown in Fig. 7. The average number of repetitions is approximately 81 (and increases to 83 at 60 degrees). Most of the challenges are stabilized in a first bump. Then we observe a final bump at $T_{max} = 300$ which corresponds to challenges that could not be stabilized

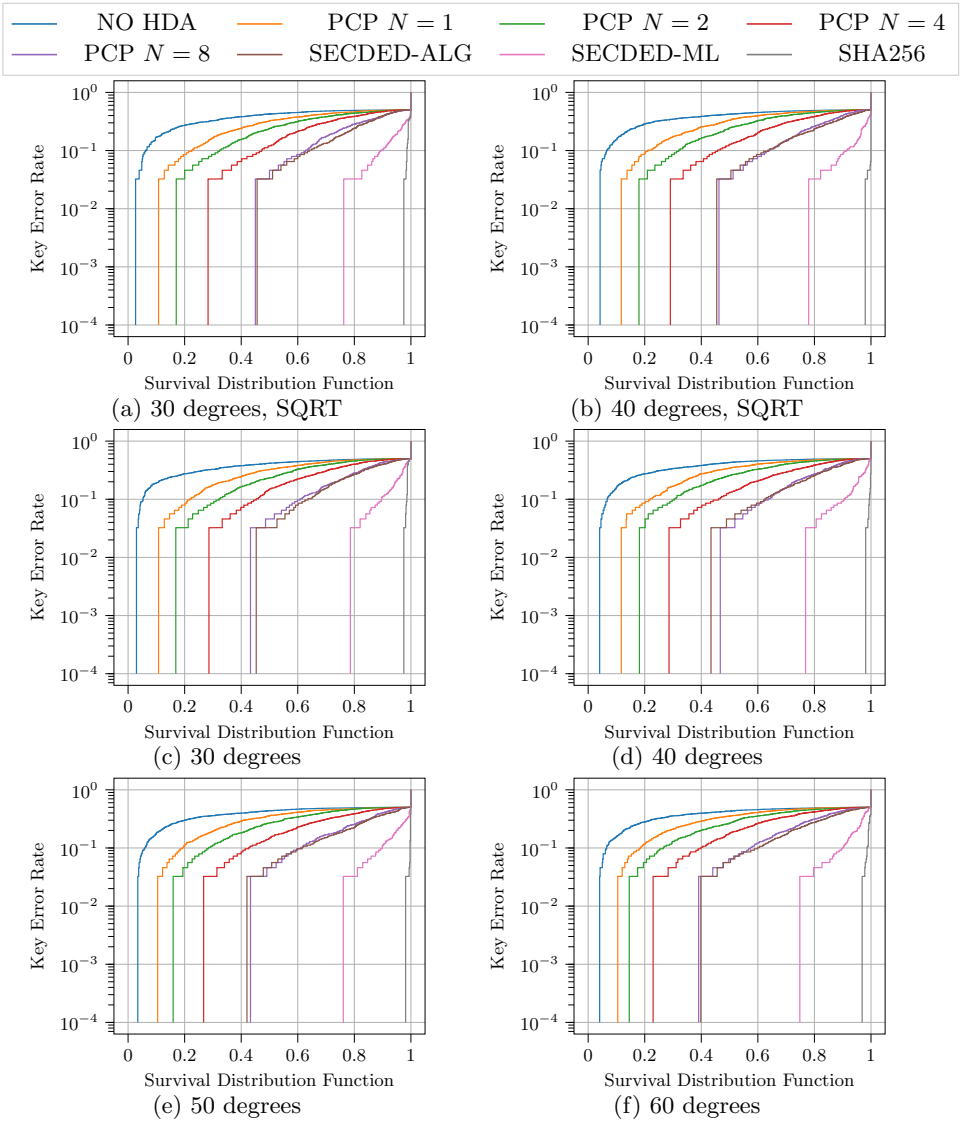
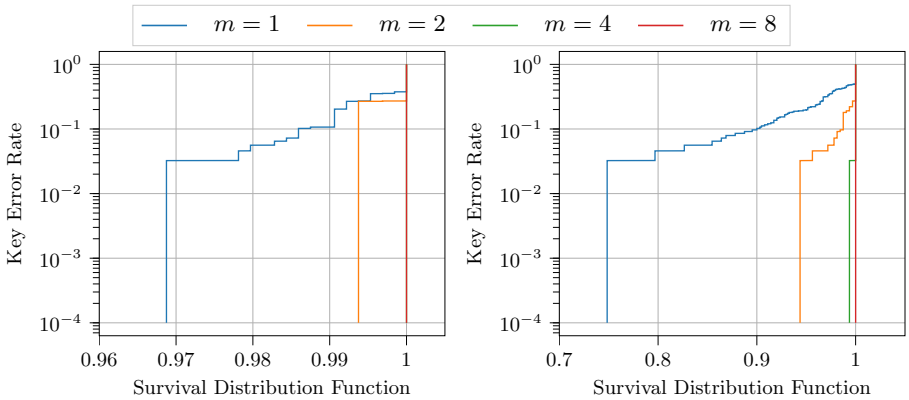


Fig. 5: KER of the design under different temperatures with light HDAs.

and reached the maximum number of repetitions. If we increase T_{max}, the height of this bump (and the KER) would decrease at the cost of a larger latency.

6 Conclusion

We present a methodology to make a robust ring oscillator PUF. This method reduces the BER at the cost of a longer latency but is founded theoretically on



(a) SHA256 (b) SECDED-ML

Fig. 6: Key Error Rate of the design at 60 degrees and constant boundary when the best oscillators among m oscillator is selected.

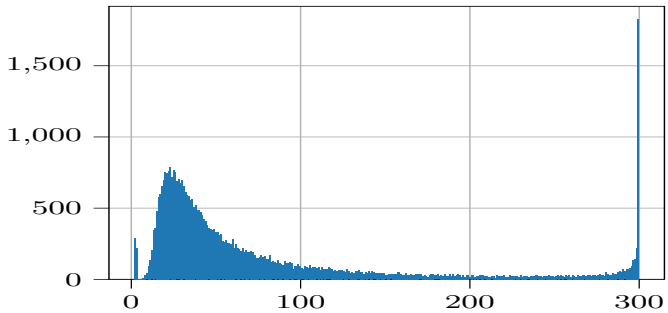


Fig. 7: Histogram of Number of Repetitions at 30 degrees

sequential analysis and validated experimentally with measurements on an FPGA board. We investigate a set of lightweight helper data and show that the key error rate can decrease further. In some practical use cases, the LPUF can even be used "without any helper data" provided that a signature or a valid hash with the enrolled secret is available.

There remain multiple aspects to investigate for future works. For example, the impact of correlated noise on the errors and the optimal boundary. The energy consumption could also be studied as in the one hand repeated measurements lead to more energy consumption, but on the other hand we limit post-processing. Depending on the device constraints there should exists an optimal trade-off to exhibit. Potential side-channel attacks are also left as perspectives. In particular, it should be confirmed that the fact that this design is not constant time is not a problem in this case (timing attack). Further, it would be interesting to know whether the repeated measurements facilitates power or electromagnetic side-channel attacks. As the response is not stored in a flip-flop until the decision is made the effect on side channel attack at the output should be benign. Though, one may identify if a challenge is stable through its corresponding measurement time. Still an efficient countermeasure against side-channel has already been proposed and discussed in [40].
The python code used for the analysis is available at [26].

A Experimental Setup

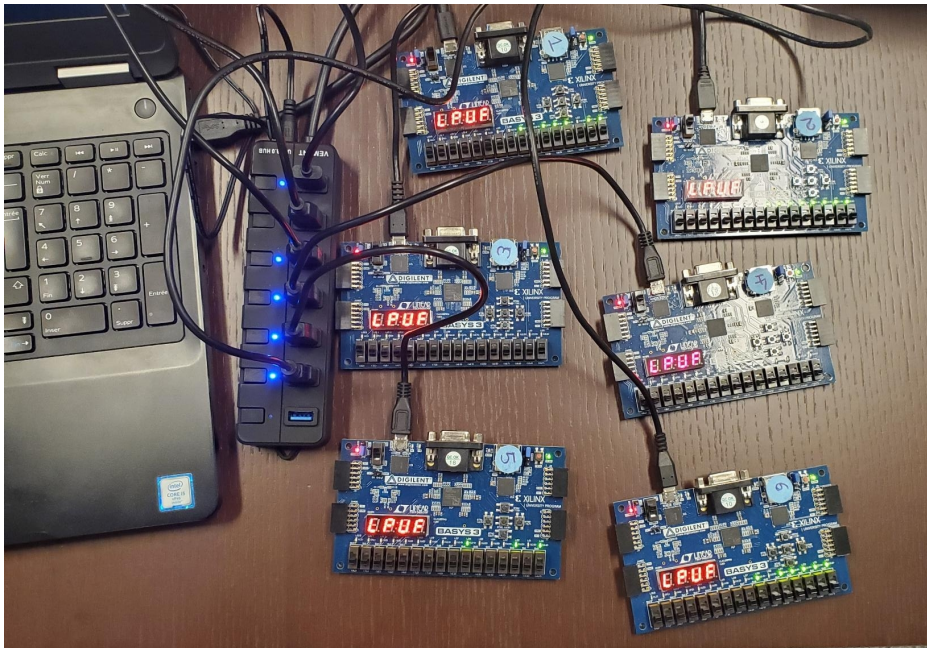


Fig. 8: Initial experimental setup with 6 boards at room temperature.

References

1. Arikan, E.: An inequality on guessing and its application to sequential decoding. IEEE Transactions on Information Theory **42**(1), 99–105 (1996)
2. Bather, J.: Bayes procedures for deciding the sign of a normal mean (1962)
3. Becker, G.T.: Robust fuzzy extractors and helper data manipulation attacks revisited: Theory versus practice. IEEE Transactions on Dependable and Secure Computing **16**, 783–795 (2019)
4. Breakwell, J., Chernoff, H.: Sequential tests for the mean of a normal distribution. ii. (large t) (1964)
5. Che, W., Plusquellic, J., Bhunia, S.: A non-volatile memory based physically unclonable function without helper data. In: 2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). pp. 148–153. IEEE (2014)
6. Cherif, Z., Danger, J.L., Guilley, S., Bossuet, L.: An easy-to-design PUF based on a single oscillator: The loop PUF. In: 15th Euromicro Conference on Digital System Design, DSD 2012, Çeşme, Izmir, Turkey, September 5-8, 2012. pp. 156–162. IEEE Computer Society (2012). https://doi.org/10.1109/DSD.2012.22, http://dx.doi.org/10.1109/DSD.2012.22
7. Chernoff, H.: Sequential tests for the mean of a normal distribution (1961)
8. Chernoff, H.: Sequential tests for the mean of a normal distribution. iii. (small t) (1965)
9. Danger, J.L., Guilley, S., Schaub, A.: Two-Metric Helper Data for Highly Robust and Secure Delay PUFs. In: IEEE 8th International Workshop on Advances in Sensors and Interfaces, IWASI 2019, Otranto, Italy, June 13-14, 2019. pp. 184–188. IEEE (2019). https://doi.org/10.1109/IWASI.2019.8791249, https://doi.org/10.1109/IWASI.2019.8791249
10. Delvaux, J., Gu, D., Schellekens, D., Verbauwhede, I.: Helper data algorithms for PUF-based key generation: Overview and analysis. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **34**(6), 889–902 (2015). https://doi.org/10.1109/TCAD.2014.2370531
11. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. SIAM journal on computing **38**(1), 97–139 (2008)
12. Dodis, Y., Reyzin, L., Smith, A.D.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In: Cachin, C., Camenisch, J. (eds.) Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings. Lecture Notes in Computer Science, vol. 3027, pp. 523–540. Springer (2004). https://doi.org/10.1007/978-3-540-24676-3“¨31, https://doi.org/10.1007/978-3-540-24676-3_31
13. Ekström, E., Vaicenavicius, J.: Bayesian sequential testing of the drift of a brownian motion (2015). https://doi.org/10.48550/ARXIV.1509.00675, https://arxiv.org/abs/1509.00675
14. Fuller, B., Meng, X., Reyzin, L.: Computational fuzzy extractors. Cryptology ePrint Archive, Paper 2013/416 (2013), https://eprint.iacr.org/2013/416, https://eprint.iacr.org/2013/416
15. Gassend, B., Clarke, D., van Dijk, M., Devadas, S.: Controlled physical random functions. In: 18th Annual Computer Security Applications Conference, 2002. Proceedings. pp. 149–160 (2002). https://doi.org/10.1109/CSAC.2002.1176287

16. Gassend, B., Clarke, D., van Dijk, M., Devadas, S.: Silicon physical random functions. In: Proceedings of the 9th ACM Conference on Computer and Communications Security. p. 148–160. CCS '02, Association for Computing Machinery, New York, NY, USA (2002). https://doi.org/10.1145/586110.586132, https://doi.org/10.1145/586110.586132
17. de Groot, J., Škorić, B., de Vreede, N., Linnartz, J.P.: Information leakage of continuous-source zero secrecy leakage helper data schemes. IACR Cryptology ePrint Archive **2012**, 566 (2012), http://eprint.iacr.org/2012/566
18. de Groot, J., Škorić, B., de Vreede, N., Linnartz, J.P.: Quantization in zero leakage helper data schemes. EURASIP J. Adv. Sig. Proc. **2016**, 54 (2016). https://doi.org/10.1186/s13634-016-0353-z, http://dx.doi.org/10.1186/s13634-016-0353-z
19. Günlü, O., Schaefer, R.F.: Low-complexity and reliable transforms for physical unclonable functions. In: ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 2807–2811 (2020). https://doi.org/10.1109/ICASSP40776.2020.9053107
20. Herder, C., Ren, L., van Dijk, M., Yu, M.D., Devadas, S.: Trapdoor computational fuzzy extractors and stateless cryptographically-secure physical unclonable functions. IEEE Transactions on Dependable and Secure Computing **14**(1), 65–82 (2017). https://doi.org/10.1109/TDSC.2016.2536609
21. Herkle, A., Becker, J., Ortmanns, M.: An Arbiter PUF employing eye-opening oscillation for improved noise suppression. In: 2018 IEEE International Symposium on Circuits and Systems (ISCAS). pp. 1–5 (2018). https://doi.org/10.1109/ISCAS.2018.8351361
22. Herrewege, A.V., Katzenbeisser, S., Maes, R., Peeters, R., Sadeghi, A.R., Verbauwhede, I., Wachsmann, C.: Reverse fuzzy extractors: Enabling lightweight mutual authentication for puf-enabled rfids. In: International Conference on Financial Cryptography and Data Security. pp. 374–389. Springer (2012)
23. Hsiao, M.Y.: A class of optimal minimum odd-weight-column SEC-DED codes. IBM Journal of Research and Development **14**(4), 395–401 (1970)
24. Ignatenko, T., Willems, F.M.J.: Information leakage in fuzzy commitment schemes. IEEE Transactions on Information Forensics and Security **5**, 337–348 (2010)
25. Immler, V., Hiller, M., Liu, Q., Lenz, A., Wachter-Zeh, A.: Variable-length bit mapping and error-correcting codes for higher-order alphabet PUFs. In: International Conference on Security, Privacy, and Applied Cryptography Engineering. pp. 190–209. Springer (2017)
26. Julien, B.: https://github.com/JulienBeg/Reliability-Ring-Oscillator-PUF
27. Karlin, S., Rubin, H.: Distributions possessing a monotone likelihood ratio. Journal of the American Statistical Association **51**(276), 637–643 (1956)
28. Linnartz, J.P., Tuyls, P.: New shielding functions to enhance privacy and prevent misuse of biometric templates. In: Kittler, J., Nixon, M.S. (eds.) Audio- and Video-Based Biometric Person Authentication. pp. 393–402. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
29. Maiti, A., Schaumont, P.: Improving the quality of a physical unclonable function using configurable ring oscillators. In: 2009 International Conference on Field Programmable Logic and Applications. pp. 703–707 (2009). https://doi.org/10.1109/FPL.2009.5272361
30. Maiti, A., Schaumont, P.: Improved Ring Oscillator PUF: An FPGA-friendly Secure Primitive. J. Cryptology **24**, 375–397 (2011). https://doi.org/10.1007/s00145-010-9088-4

31. Maringer, G., Xhemrishi, M., Puchinger, S., Garb, K., Liu, H., Jerkovits, T., Kürzinger, L., Hiller, M., Wachter-Zeh, A.: Analysis of communication channels related to physical unclonable functions. arXiv preprint arXiv:2112.02198 (2021)
32. Massey, J.L.: Guessing and entropy. In: Proceedings of 1994 IEEE International Symposium on Information Theory. p. 204. IEEE (1994)
33. Rioul, O.: Variations on a Theme by Massey. IEEE Transactions on Information Theory **68**(5), 2813–2828 (2022)
34. Rioul, O., Solé, P., Guilley, S., Danger, J.L.: On the entropy of physically unclonable functions. In IEEE International Symposium on Information Theory, ISIT (July 2016)
35. Schaub, A., Danger, J.L., Guilley, S., Rioul, O.: An Improved Analysis of Reliability and Entropy for Delay PUFs. 21st Euromicro Conference on Digital System Design (2018)
36. Solé, P., Cheng, W., Guilley, S., Rioul, O.: Bent Sequences over Hadamard Codes for Physically Unclonable Functions. In: 2021 IEEE International Symposium on Information Theory (ISIT). pp. 801–806 (2021). https://doi.org/10.1109/ISIT45174.2021.9517752
37. Stangherlin, K., Wu, Z., Patel, H., Sachdev, M.: Enhancing Strong PUF Security with Non-monotonic Response Quantization (2022). https://doi.org/10.48550/ARXIV.2206.03440, https://arxiv.org/abs/2206.03440
38. Stanko, T., Nur Andini, F., Skoric, B.: Optimized quantization in zero leakage helper data systems. Trans. Info. For. Sec. **12**(8), 1957–1966 (aug 2017). https://doi.org/10.1109/TIFS.2017.2697840, https://doi.org/10.1109/TIFS.2017.2697840
39. Suh, G.E., Devadas, S.: Physical unclonable functions for device authentication and secret key generation. In: 2007 44th ACM/IEEE Design Automation Conference. pp. 9–14 (2007)
40. Tebelmann, L., Danger, J.L., Pehl, M.: Self-Secured PUF: Protecting the Loop PUF by Masking. In: Constructive Side-Channel Analysis and Secure Design - 11th International Workshop, COSADE 2020, Lugano, Switzerland, October 5-7, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12244. Springer (October 5-7 2020)
41. Wald, A., Wald, A.: The sequential probability ratio test for testing a simple hypothesis H0 against a single alternative H1. Sequential Analysis **37**, 70 (1947)
42. Wald, A., Wolfowitz, J.: Optimum character of the sequential probability ratio test. The Annals of Mathematical Statistics pp. 326–339 (1948)
43. Wang, W.C., Yona, Y., Diggavi, S., Gupta, P.: LEDPUF: Stability-guaranteed physical unclonable functions through locally enhanced defectivity. In: 2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). pp. 25–30. IEEE (2016)
44. Yin, C.E.D., Qu, G.: LISA: Maximizing RO PUF's secret extraction. In: 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST). pp. 100–105 (2010). https://doi.org/10.1109/HST.2010.5513105