FROPUF: How to Extract More Entropy from Two Ring Oscillators in FPGA-Based PUFs

Qinglong Zhang[1,2,3], Zongbin Liu[1,2], Cunqing Ma[1,2], Changting Li[1,2,3], and Lingchen Zhang[1,2(✉)]

[1] Data Assurance and Communication Security Research Center, Beijing, China
[2] State Key Laboratory of Information Security, Institute of Information Engineering, CAS, Beijing, China
{qlzhang,zbliu,limiao12,jixiang,jing}@lois.cn
[3] University of Chinese Academy of Sciences, Beijing, China

Abstract. Ring oscillator (RO) based physically unclonable function (PUF) on FPGAs is popular for its nice properties and easy implementation. The conventional compensated measurement though proved to be particularly effective in extracting entropy of manufacturing features, only one bit entropy can be extracted from two ROs, which implies enormous consumption of hardware resources. Motivated by this, we propose an elegant and efficient method to extract at least 31 bits entropy from two ROs by utilizing the fine control of programmable delay lines of look up table (LUT), and denominate this new construction as Further ROPUF. We will elaborate how to take advantage of the underlying manufacturing variations of LUTs and display how deeper variations are extracted by the second order difference calculation method. Additionally, we reveal the consistency between the evaluation results on Xilinx FPGAs and by simulations, and the responds' low bit-error-rate of 1.85% manifests the proposed FROPUF maintains considerable reliability.

Keywords: PUFs · Ring Oscillator · Entropy · FPGA

1 Introduction

With flourishing development of embedded devices in modern age, cryptographic algorithms are easily implemented on FPGA for FPGA's reconfigurable nature. An indispensable premise for the security of cryptographic primitives is the competence to securely generate, store and retrieve secret keys. In general, it rests upon a protected memory which stores the private information reliably and shields it completely from unauthorized parties. Whereas this requirement is nontrivial to achieve in practice [1]. Recently, physically unclonable function (PUF)

L. Zhang—The work is supported by a grant from the National Natural Science Foundation of China (No.61402470).

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2017
R. Deng et al. (Eds.): SecureComm 2016, LNICST 198, pp. 675–693, 2017.
DOI: 10.1007/978-3-319-59608-2_37

has attracted wider attention as a technique to provide physical roots of trust in embedded systems [2–4]. Due to the submicron random variation during manufacturing, nominally identical logic circuit turns out to have individual physical features. The main idea of PUF is to utilize these intrinsic random manufacturing features to extract a unique electronic fingerprint, therefore providing an approach to issues such as cryptographic key generation [5], intellectual property (IP) protection [6,7], device authentication [8–10] and trusted computing etc.

Up to now, a variety of electronic PUFs have been proposed, such as SRAM PUF [11], Butterfly PUF [12], Glitch PUF [13], Flip-Flop PUF [14], Ring Oscillator PUF [15] and so on. However, some of them are not suitable for FPGAs. In the state of the art commercial FPGAs of Xilinx and Altera, the start-up values of SRAM are reset to a certain value according to the manufacturer's design, which leads to the failure of deploying SRAM PUF on these FPGAs. Moreover, many other PUF designs like Butterfly PUF and Arbiter PUF demand a careful routing symmetry, which is also difficult to implement on FPGAs. Especially for Butterfly PUF, even the fundamental element, a latch with a preset signal and a clear signal, is not provided on Xilinx's latest 6-series and 7-series FPGAs. RO PUF which is first proposed by Suh and Devadas [15] has been widely used due to its sensitivity to manufacturing variations, and particularly the hard-macro design technique simplifies the implementation of identical ROs on FPGA. However, besides these advantages, Maiti [16] pointed out that some factors like the systematic or correlated manufacturing variations and the regional environmental noise would degrade the uniqueness and reliability of RO PUF.

A lot of researches [15–21] have been done in order to strengthen the properties of RO PUF. In DAC 2007, Suh and Devadas [15] applied a post-processing technique called 1-*out-of-k* masking and greatly enhanced the reliability of the PUF's response, but resulted in a relatively large resource overhead. In J.Cryptol.2011, Maiti et al. [16] proposed a configurable RO technique to produce nearly 100% error-free PUF outputs over varying environmental condition without post-processing. This technique is quite effective to resolve PUF reliability issues on FPGAs. However, two configurable ROs generate only one bit response in making a tradeoff between reliability and the length of response sequence, and the calculation cost is relatively high.

To serve as a physical root of trust, it is vital for a PUF design to provide sufficient entropy in its response. Nevertheless, the most commonly used fuzzy extraction technics in PUFs always cause entropy loss [22,23]. As a result, the amount of extractable entropy of a PUF becomes another essential evaluation index. Given this, Habib et al. [24] proposed an FPGA PUF base on programmable LUT delays to acquire more entropy from two ROs. According to his research, when the logically unrelated inputs of LUT vary from '000' to '111', the RO's frequency changes irregularly. While in CHES 2011 [25,26], it stated that the loop delays of input '111' were on average about 10 picoseconds larger than the delay values of input '000'. Habib et al. pointed out this disagreement is caused by employing different devices (in [24] is Spartan-3E, while in [25,26] are Virtex-5 series). However, if the frequency varies regularly as the LUT inputs change, the method used in [24] would be invalid.

In this paper, by utilizing the fine adjustment of LUT's propagation path on FPGAs [25], we propose a comprehensive scheme to extract more available manufacturing features and through a second order difference calculation way, we are able to achieve at least 31-bit entropy from two ROs. Furthermore, this difference calculation method can efficiently reduce the effect of the systematic manufacturing variation and the regional environmental noise. To make it more persuasive, the evaluation results obtained from our experiments and simulations demonstrate that the proposed PUF construction possesses excellent reliability and uniqueness under varied temperatures.

Although RO PUF is threatened by modeling attacks, a secure one-way hash over the PUF's outputs, so called a Controlled PUF [27], is an efficient solution. The staple of this paper is how to extract more entropy from ROs, rather than a secure access to the response of PUFs.

In summary, our contributions in this paper are as follows:

- We propose an elegant method to extract more subtle manufacturing variations by second order difference calculation, which can efficiently reduce the impact of systematic variation and regional environmental noise to guarantee the PUF's reliability.
- We design a new construction named Further RO PUF (FROPUF), which can extract at least 31-bit entropy from only two ROs on FPGAs.
- We conduct both simulation and practical experiments to demonstrate that our new proposed PUF has a bit-error-rate of 1.85% at 27 °C and an average inter-distance of 49.32%.

The rest of the paper is organized as follows. Section 2 presents preliminaries for our paper. Section 3 describes our model for RO PUFs with fine control of LUT's inputs and proposes our new construction with second order difference calculation. Section 4 evaluates the performance of our PUF from simulations and practical experiments. Finally, we conclude this paper in Sect. 5.

2 Preliminaries

A typical example of RO based PUF is shown in Fig. 1. It consists of n identically laid-out ROs, RO_1 to RO_n, with frequencies f_1 to f_n respectively. In general, the challenge (i, j) is applied to the multiplexers to select a pair of ROs, RO_i and RO_j $(i \neq j)$. Due to intrinsic manufacturing variations, f_i and f_j actually differs from each other. Based on the compensated measurement proposed by Gassend et al. [2], a response bit r_{ij} can be generated by the comparison expression as follows:

$$r_{ij} = \begin{cases} 1 & if \quad f_i > f_j, \\ 0 & otherwise. \end{cases} \tag{1}$$

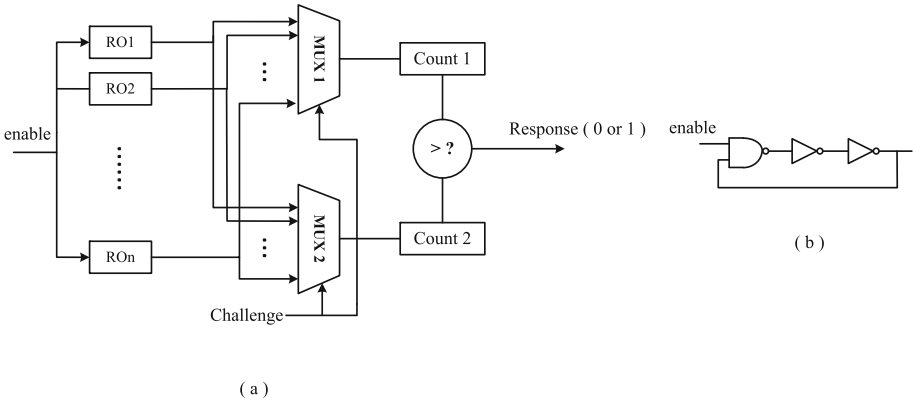

Fig. 1. (**a**) A typical example of RO PUF. (**b**) A three-stage ring oscillator

2.1 Evaluation Scheme of RO PUF

The evaluation scheme of a PUF is usually divided into three basic aspects, uniqueness, reliability and security [16].

- Uniqueness estimates how uniquely a PUF entity can be distinguished from others according to its responses.
- Reliability evaluates how stable the responses are with environmental factors (such as temperature, supply voltage) vary.
- Security is a PUF's ability in preventing adversaries from predicting the PUFs response.

Uniqueness can be measured by inter-distance. As defined in [28], when apply a particular challenge to the PUF's two different instances, the inter-distance is the hamming distance (HD) between their current responses. We estimate the uniqueness of a PUF base on the average inter-distance over a group of chips. With two PUF instantiations, denoted as i and j ($i \neq j$), both having a n-bit response, R_i and R_j respectively, the average inter-distance μ_{inter} among k chips is calculated as

$$\mu_{inter} = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{HD(R_i, R_j)}{n} \times 100\% \tag{2}$$

Reliability can be evaluated through intra-distance. It is the hamming distance between different evaluation values of the same response on the same PUF instance. Due to environmental factors, such as temperature variation, supply voltage fluctuation and circuit noise, PUF's responses are not perfectly reproducible. To evaluate the reliability of a PUF's response, we achieve n-bit response $m+1$ times from the PUF instance i at some environmental condition and select the first n-bit response as the reference response R_i and the other responses as $R_{i,j}$ ($1 \leq j \leq m$). The average intra-distance μ_{intra} can be calculated as follows:

$$\mu_{intra} = \frac{1}{m}\sum_{j=1}^{m}\frac{HD(R_i, R_{i,j})}{n} \times 100\% \tag{3}$$

2.2 Systematic Variation

In J.Cryptol.2011, Maiti et al. [16] pointed out that the total delay in a RO loop can be modeled as follows:

$$d_{LOOP} = d_{AVG} + d_{RAND} + d_{SYST} \tag{4}$$

where d_{AVG} = the nominal delay which is the same for all the identical ROs; d_{RAND} = delay variation due to manufacturing variation; d_{SYST} = delay variation due to the systematic variation. Then the difference between two ROs, a and b, can be calculated as follows:

$$\begin{aligned}\Delta d_{LOOP} &= (d_{AVG} + d_{RAND_a} + d_{SYST_a}) - (d_{AVG} + d_{RAND_b} + d_{SYST_b}) \\ &= \Delta d_{RAND} + \Delta d_{SYST}\end{aligned} \tag{5}$$

According to formula (5), a single response bit r_{ab} of these two ROs is not only decided by the random manufacturing variation, but also by the systematic variation. Maiti et al. noted that the systematic manufacturing variation could lead to a gradual change in the RO's loop delay as a function of the physical location, and its existence hazards RO PUF's uniqueness. In [16] a solution is given. Because two closely located ROs will have similar d_{SYST} in (4), their d_{SYST} can be counteracted by difference calculation.

2.3 Programmable Delay Lines

LUT is the main programmable delay logic unit of FPGA, and the construction of a 3-input LUT is shown in Fig. 2. The LUT is composed of a set of SRAM cells and a tree-like structure of multiplexers (MUXs). The former stores the intended functionality and the latter enables selection of each individual SRAM cell content. A LUT can be configured as an inverter, whose output (O) is always an inversion of its first input (A_1), and the inputs (A_2 and A_3) are logically irrelevant with A_1 and O. In CHES 2011, Majzoobi et al. [25] proposed a novel technique to adjust the propagation path in minute increments /decrements by using only a single LUT on reconfigurable FPGA platform. The mechanism changes the propagation path inside the LUT by altering the logically irrelevant inputs. Although the inputs A_2 and A_3 have no influence on the inverters logic, their values affect the signal propagation path from input A_1 to output O. Majzoobi et al. pointed that when $A_2A_3 = 00$ and $A_2A_3 = 11$, the propagation path from A_1 to O is the shortest and the longest respectively as shown in Fig. 2. The latest Xilinx series products, Virtex-5,6,7 and Spartan 6, adopt 6-input LUTs. Therefore, as the method proposed by Majzoobi, a programmable delay inverter can be implemented with at most $2^5 = 32$ discrete levels for controlling the propagation delay. For example, it is an example of this

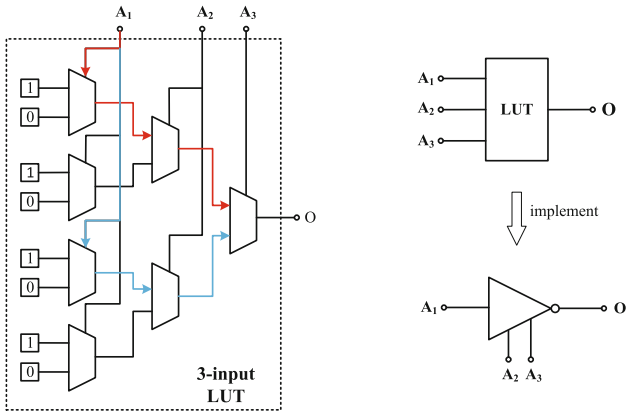

Fig. 2. Programmable delay lines using an LUT

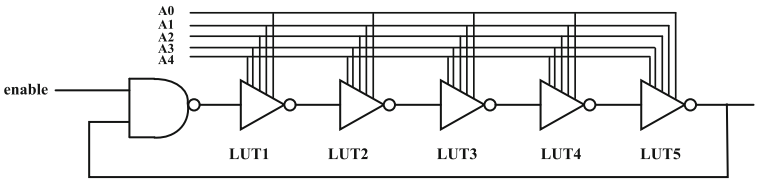

Fig. 3. An example of this fine control for 5-stage ROs

fine control for 5-stage ROs and the LUTs are 6-input in Fig. 3. Five of these inputs are configured as delay control.

Based on the multiple control of the LUT's propagation delay, Habib et al. [24] managed to extract more entropy from a pair of RO. According to their experiment results, the frequency varies significantly with the LUT's input sequence and the frequency's changing pattern is irregular. Based on this, Habib et al. were able to extract more entropy by comparing two RO's frequencies under all configurations from '000' to '111' correspondingly. However, the experiment results of Majzoobi et al. [25], demonstrates that the propagation delays of input '11111' are on average about 10 picoseconds larger than the corresponding delays of input '00000'. Habib et al. explained that the reason might be the Spartan 3E devices they used were based on 90 nm technology, while Majzoobi et al. employed Virtex-5 devices which were 65 nm technology. Therefore, if the frequency varies in a rough order depending on the LUT's input sequence, the result of the proposed method in [24] will lose efficiency on Virtex-5 devices.

3 Our Proposed Further ROPUF

Based on the propagation delay model proposed by Majzoobi et al. [25], we present a model which is involved with the subtle manufacturing variation of

different LUT's inputs. Consider a RO l consisting of 6-input LUTs, its loop delay can be modeled as follows:

$$d_{LOOP(l,j)} = d_{AVG} + d_{RAND(l,j)} + d_{SYST(l,j)} \quad (1 \leq j \leq 32) \tag{6}$$

where d_{AVG} is the nominal delay which is the same for all the identical ROs; $d_{RAND(l,j)}$ represents the delay variation due to the random manufacturing variation when LUTs are driven by the j^{th} input; $d_{SYST(l,j)}$ denotes the delay variation due to the systematic variation. The variables $d_{RAND(l,j)}$ and $d_{SYST(l,j)}$ could be positive and negative. For a RO with different LUT inputs, in_{j_1} and in_{j_2}, these two $d_{SYST(l,j_1)}$ and $d_{SYST(l,j_2)}$ are extremely close as shown in [16]. Therefore, in formula (6), the subscript of $d_{SYST(l,j_1)}$ and $d_{SYST(l,j_2)}$ can be modified to $d_{SYST(l)}$, where l is only related to the RO's location. And formula (6) will change into formula (7) as follows:

$$d_{LOOP(l,j)} = d_{AVG} + d_{RAND(l,j)} + d_{SYST(l)} \quad (1 \leq j \leq 32) \tag{7}$$

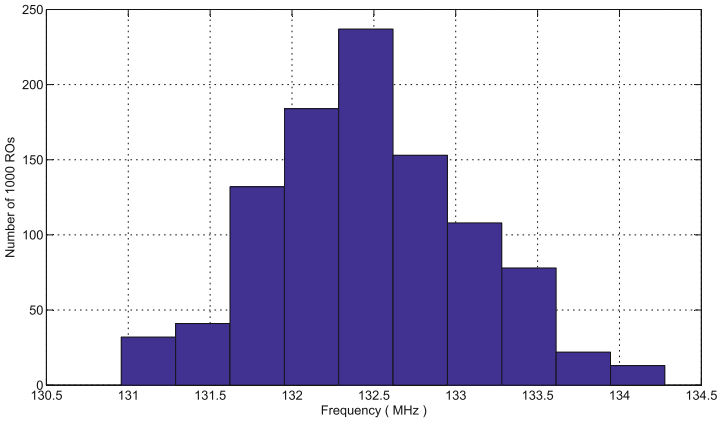

Fig. 4. The histogram distribution of 1000 ROs' frequencies

Moreover, when apply the same LUT input in_j on a group of L ROs, L variable values $d_{RAND(1,j)}$, $d_{RAND(2,j)}$, $\cdots$,$d_{RAND(L,j)}$ are acquired. According to ReConFig 2008 [29] and HOST 2011 [30], these values are almost complying with Gaussian distribution. Figure 4 shows the distribution of our experimental data, and it also seems like a normal distribution. Therefore, we assume that these L values are normally distributed. Apply this assumption to other LUT input configurations and we can achieve 32 normal distributions as follows:

$$(d_{RAND(1,j)}, d_{RAND(2,j)}, \cdots, d_{RAND(L,j)}) \sim N(\mu_j, \sigma_j^2) \quad (1 \leq j \leq 32) \tag{8}$$

Which indicates the random variable $d_{RAND(j)}$ is a normal distribution with mean μ_j and standard deviation σ_j.

3.1 Second Order Difference Calculation

On the basis of the above description, for a group of L ROs, by varying the LUT's input from '00000' to '11111', we can obtain $32 * L$ different $d_{LOOP(l,j)}$ which are in the same form of formula (9). Then we generate responses by second order difference calculation.

According to the above description, for a group of L ring oscillators, by varying the LUT's input from '00000' to '11111', we can get $32 * L$ different $d_{LOOP(l,j)}$ which has the similar form in formula (9). We propose an elegant method to generate responses based on second order difference calculation.

$$d_{LOOP(l,j)} = d_{AVG} + d_{RAND(l,j)} + d_{SYST(l)} \quad (1 \le j \le 32, 1 \le l \le L) \tag{9}$$

Our proposed method can be divided into two steps and here we present a neat example to illustrate our method.

1. For a RO l, select $d_{LOOP(l,j)}$ and $d_{LOOP(l,j+1)}$, then calculate the difference value $\Delta d_{LOOP(l,j)}$, $1 \le j \le 31$.
2. For two ROs l_1 and l_2, generate one bit $r_{l_1,l_2,j}$ $(1 \le l_1 \ne l_2 \le L, 1 \le j \le 31)$ as follows.

$$r_{l_1,l_2,j} = \begin{cases} 1 & if \quad \Delta d_{LOOP(l_1,j)} > \Delta d_{LOOP(l_2,j)}, \\ 0 & otherwise. \end{cases} \tag{10}$$

Through these two steps, we will get a 31-bit response from two of these L ROs, thus 31*(L-1) bits can be extracted. Based on formula (9), $\Delta d_{LOOP(l,j)}$ can be calculated as follows:

$$\begin{aligned} \Delta d_{LOOP(l,j)} &= d_{LOOP(l,j)} - d_{LOOP(l,j+1)} \\ &= (d_{AVG} + d_{RAND(l,j)} + d_{SYST(l)}) - (d_{AVG} + d_{RAND(l,j+1)} + d_{SYST(l)}) \\ &= d_{RAND(l,j)} - d_{RAND(l,j+1)} \end{aligned} \tag{11}$$

Observing formula (11), we note that the systematic variation is neatly removed by this first order difference calculation. According to the assumption condition (8) that both $d_{RAND(j)}$ and $d_{RAND(j+1)}$ are normally distributed, we can get the distribution of the random variable $\Delta d_{LOOP(j)}$ as follows:

$$\Delta d_{LOOP(j)} \sim N(\mu_j - \mu_{j+1}, \sigma_j^2 + \sigma_{j+1}^2 - 2 * r_j * \sigma_j^2 * \sigma_{j+1}^2) \quad (1 \le j \le 31) \tag{12}$$

where r_j is the correlation coefficient between these two random variables. Let $\mu_{LOOP(j)}$ and $\sigma_{LOOP(j)}$ represent $\mu_j - \mu_{j+1}$ and $\sigma_j^2 + \sigma_{j+1}^2 - 2 * r_j * \sigma_j^2 * \sigma_{j+1}^2$ respectively. Through the second step of second order difference calculation, we can calculate the distribution of the random variable $R_{l_1,l_2,j}$ as follows:

$$R_{l_1,l_2,j} \sim N(0, 2 * \sigma_{LOOP(j)}^2) \quad (1 \le j \le 31) \tag{13}$$

For the mean of $R_{l_1,l_2,j}$'s distribution is zero, as the result of our second order difference calculation method, probabilities for $r_{l_1,l_2,j}$ equals '0' and '1' are the same. Theoretically, every response bit has 50% probability to be '0' or '1' and if these 31 bits have no correlation, it can be stated that 31-bit entropy is extracted from these two ROs.

In order to evaluate the randomness and entropy of responses, we will carry out NIST test suits on the responses generated by FROPUF in Sect. 4.4.

3.2 Analysis of the Second Order Difference Calculation

The key point to extract more entropy from two ROs is to extract more manufacturing features whose magnitude may be close to that of noise. Therefore we should suppress or eliminate the noise to the greatest extent.

In the conventional architecture of RO PUF, every RO has unique signal propagation path. While in programmable delay line model, the fine control of LUT's logically irrelevant inputs leads to different propagation paths. Habib et al. [24] have tried to extract more responses by utilizing this character. However, these propagation paths have a rough order on Xilinx Virtex-5, 6, 7 series devices. In that case, although the direct comparison between the corresponding delays of two ROs can acquire a 32-bit response sequence, the correlation between these bits may give rise to a large amount of entropy loss.

In our proposed scheme, we take advantage of the similarity of adjacent ROs' systematic variations and effectively eliminate the influence of systematic factors by the first order difference calculation between loop delays of the same RO. Follow the second step, we obtain the second order difference as formula (14) shows. This result is affected by the combination of two ROs' manufacturing features. As described in Sect. 3.1, the value of each response bit is decided by the sign of formula (14).

$$(d_{LOOP(l,j)} - d_{LOOP(l,j+1)}) - (d_{LOOP(l+1,j)} - d_{LOOP(l+1,j+1)}) \tag{14}$$

Rewrite formula (14) we get:

$$(d_{LOOP(l,j)} - d_{LOOP(l+1,j)}) - (d_{LOOP(l,j+1)} - d_{LOOP(l+1,j+1)}) \tag{15}$$

Observing formula (15), you will see the most important difference between Habib et al. and us is that we compare the relative magnitude of two ROs' corresponding loop delays. And also because of this, our scheme is able to extract more subtle manufacturing features.

Furthermore, the second order difference calculation method maintains the primary idea presented by Gassend et al. [2] which alleviates the impact of environmental fluctuations by comparing two ROs' frequencies to generate one bit response. To sum up, the second order difference calculation involves two difference functions which reduce both influence of the systematic variation and of the environmental fluctuations.

3.3 Simulation of Second Order Difference Calculation

To prove the effectiveness and correctness of our proposed model through simulation, some necessary parameters needs to be collected from experimental data on FPGA. In order to reflect individual differences, manufacturing variation and environmental change should be considered during simulation. Manufacturing variation is generally divided into systematic variation and random variation. Systematic variation is mainly affected by the ROs location on the wafer or chip. There is a research pointing out that systematic variation dominates the frequencies of ROs in one region are average larger than those in another region [16]. On the contrary, random variation has no relationship with components' spatial location. Therefore, we assume the parameters are as follows:

- Systematic delay d_{SYST} affected by spatial location: $\sim \mathrm{N}(0,\sigma^2_{syst})$.
- Component delay d_{RAND} affected by random variation: $\sim \mathrm{N}(\mu_j,\sigma^2_j)$, where j represents the j^{th} input for LUTs.

According to the parameters defined above, the delay value for different LUT inputs of different ROs can be generated by simulation, and we can also calculate every response bit following Algorithm 1, where sampling y from a distribution $\mathrm{N}(\mu,\sigma^2)$ is denoted as $y \longleftarrow \mathrm{N}(\mu,\sigma^2)$.

Algorithm 1. Simulation Algorithm of Second Order Difference Calculation

Settings: $\cdot MAX_{NumRO}$ is the number of ring oscillators.
$\cdot MAX_{NumIn}$ is the number of different LUT's inputs.
Output: $r_{l,j}$, $0 \le l \le MAX_{NumRO} - 1$, $0 \le j \le MAX_{NumIn} - 1$

for l = 1 to MAX_{NumRO} **do**
  $d_{SYST(l)} \leftarrow \mathrm{N}(0,\sigma^2_{syst})$
  for j = 1 to MAX_{NumIn} **do**
    $d_{RAND(l,j)} \leftarrow \mathrm{N}(\mu_l,\sigma^2_l)$
  end for
end for
for l = 1 to $MAX_{NumRO} - 1$ **do**
  for j = 1 to $MAX_{NumIn} - 1$ **do**
    $\Delta_{LOOP(l,j)} \leftarrow (d_{AVG} + d_{RAND(l,j)} + d_{SYST(l)}) - (d_{AVG} + d_{RAND(l,j+1)} + d_{SYST(l)})$
    $\Delta_{LOOP(l+1,j)} \leftarrow (d_{AVG} + d_{RAND(l+1,j)} + d_{SYST(l+1)}) - (d_{AVG} + d_{RAND(l+1,j+1)} + d_{SYST(l+1)})$
    if $\Delta_{LOOP(l,j)} \ge \Delta_{LOOP(l+1,j)}$ **then**
      $r_{l,j} \leftarrow 1$
    else
      $r_{l,j} \leftarrow 0$
    end if
  end for
end for

4 Evaluation

In this section, we present the evaluation result of our FROPUF on 15 Virtex-5 XC5VLX110T-1FF1136 FPGAs, 10 Virtex-6 XC6VLX240T-1FF1156 FPGAs and 5 Kintex-7 XC7K325T-2FFG900 FPGAs. On the basis of practical measurements, we firstly acquire parameters for simulation. Then by comparing experimental and simulative results, we reveal the consistency of our simulation model and the practical architecture.

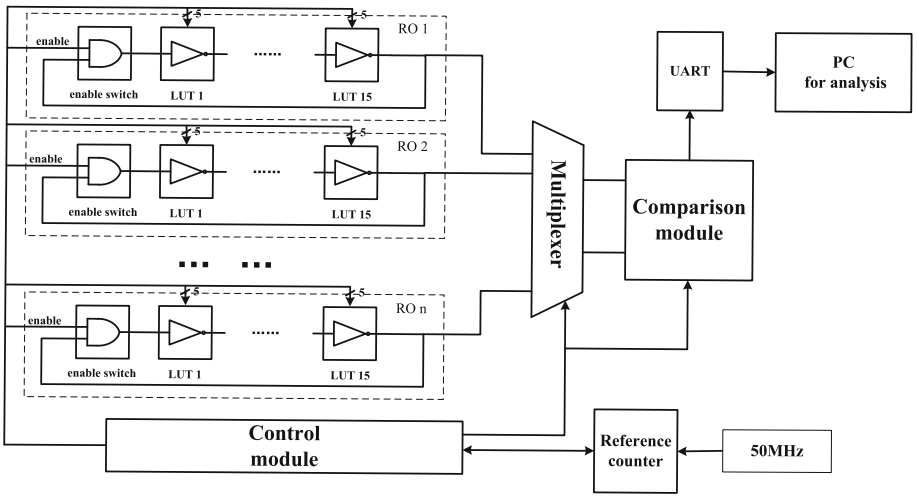

Fig. 5. The experimental evaluation system

Figure 5 shows our experimental evaluation system on Virtex-5 FPGA. A 50-MHz clock signal generated by an on-board oscillator and is applied to the reference counter. In Fig. 5, we place 200 ring oscillators and each of them is composed of 16 LUTs. 15 LUTs are instantiated as inverters with 5 configuration inputs and the last one serves as an enable switch. All the 16 LUTs are deployed in four adjacent slices, which means every RO occupies 4 slices of FPGA. Hard Macro technique is adopted to guarantee identical layout of these 200 ROs. The whole system is mainly controlled by the control module, which is responsible for the inverters' configuration inputs and control signals of multiplexer and reference counter. In order to evaluate the responses generated by our FROPUF, we utilize UART interface to transmit these responses to PC for analysis.

As high frequency makes RO instable, we select 15-stage RO whose frequency is about 132 MHz. To demonstrate the validity of our design, the configuration inputs for all LUTs are the same, i.e. the configuration input space is 2^5. In our evaluation system, there are 200 ROs which consume 800 slices. The other control module and the UART module have 213 and 126 slices separately.

In normal environmental condition, we perform a basic experiment on delay characteristics described in Sect. 3 and extract the parameter required by simulation. The obtained parameters are shown in Table 1. Based on these parameters and our simulation model, we can calculate the *intra-distance* and *inter-distance* to evaluate the reliability and uniqueness of FROPUF.

Table 1. Parameters for simulation in normal environmental condition

Parameter	Value
Standard deviation of systematic delay $\sigma_{d_{SYST}}$ $(\%^2)$	3.5336
Standard deviation of component random variation $\sigma_{d_{RAND}}$ $(\%^2)$	4.7636

4.1 Reliability

Reliability is mainly evaluated by intra-distance and reflects how stably the PUF can reproduce its responses. Figure 6 plots the evaluation results from simulation and experimental results of our evaluation systems. The simulative average error rate is around 1.25%. Steps to calculate the average intra-distance of practical experiments are depicted as follows:

1. Let every two ROs generate 31-bit response 200 times and record as $\mathrm{RES}_{l,k,t}$. Where $1 \leq l \leq 15$ denotes the index of FPGA boards, $1 \leq k \leq 100$ denotes the index of RO pairs and $1 \leq t \leq 200$ denotes the the number of measurement.
2. For every RO pair, obey formula (3) to calculate the intra-distance of its 200 responses.
3. Average all the RO pairs intra-distance.

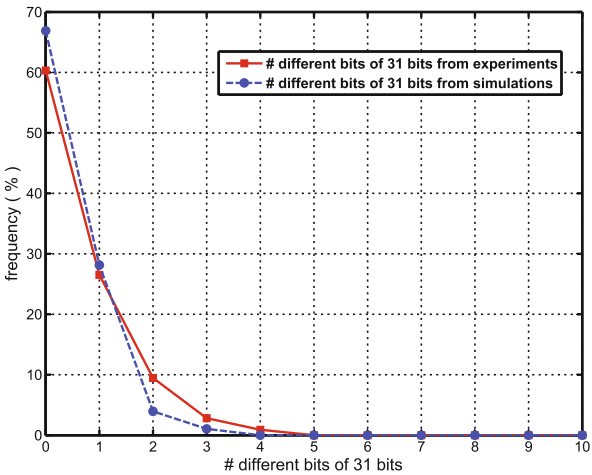

Fig. 6. The intra-distance evaluation from practical experiments and simulations

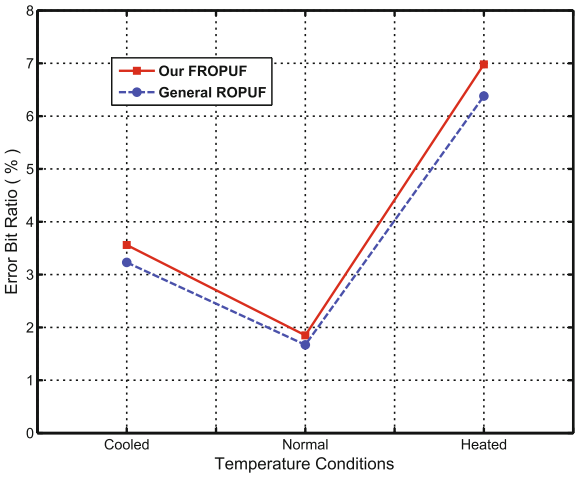

Fig. 7. The error bit ratio under different temperature conditions

The measurement is carried out at normal temperature (27 °C) on 15 Virtex-5 XC5VLX110T- 1FF1136 FPGAs. The experimental average error rate is around 1.85%, which is comparable to other RO PUF designs [5,18]. In addition, Fig. 6 indicates that the behavior of experimental error rate can be assessed by simulation with high accuracy.

The change of temperature is a major disturbance for RO based PUF, therefore we conduct experiments under different temperatures to investigate the FROPUF's property. Figure 7 shows that as the temperature rising (up to about 70 °C), the intra-distance continues to increase until 6.98%, which is about the half of 15% assumed in [31]. According to Figure 7, FROPUF achieves almost the same error bit rate as the general ROPUF.

4.2 Uniqueness

Uniqueness is mainly evaluated by inter-distance and Fig. 8(a) is a histogram of hamming distances between different PUF instances' responses in practice. Every instance consists of two ROs and produces a 31-bit response. We totally deploy 1500 instances on 15 Virtex-5 FPGAs. The result shows that the average inter-distance is about 49.32%, which indicates that FROPUF instances possess enough uniqueness to be identified from each other. Figure 8(b) shows the result of the same evaluation by simulation. Through Fig. 8(a) and (b), we conclude that the simulation is able to evaluate the uniqueness of responses generated by PUF instances.

4.3 The Randomness Evaluation

NIST test suites are carried out to evaluate the randomness of the responses generated by FROPUF. The length of the response generated by each instantiation

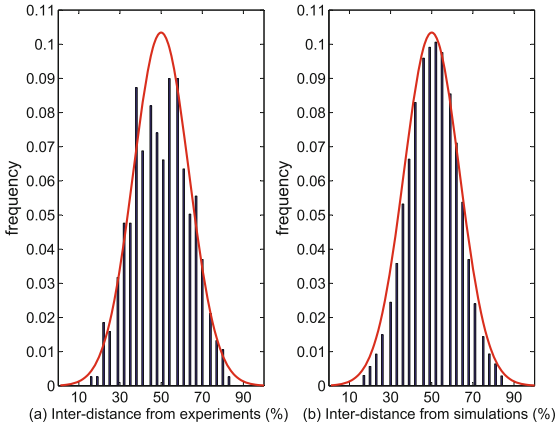

Fig. 8. The inter-distance evaluation from practical experiments and simulations

is 31 bits and we get totally 46500 bits responses. For the limitation of the length of response, we conduct 9 basic NIST tests and the result is shown in Table 2. The *Frequency* test manifests that the response bit has nearly 50% to be '1' and 50% to be '0' and this practical result is similar to the theoretical analysis in Sect. 3.1.

Table 2. The result of NIST for FROPUF's responses

Statistical test	P-value	Proportion
Frequency	0.350485	10/10
Blockfrequency	0.911413	10/10
Cumulativesums(forward)	0.739918	10/10
Cumulativesums(backward)	0.035174	10/10
Runs	0.534146	10/10
Longestruns	0.628713	10/10
Rank	0.122325	10/10
FFT	0.523478	10/10
Serial(∇^1)	0.712378	10/10
Serial(∇^2)	0.328793	10/10
Linearcomplexity	0.189283	10/10

In Table 3, we list some designs extracting responses from ROs and make comparisons of the variable *Bits per Ring*. The result shows that in our architecture, we can extract 16.5 bits entropy per ring, which is 7 times larger than that of Habib et al. [24], and moreover it is 31 times larger than that of the general RO PUF.

Table 3. Comparison of the entropy extracted from two ROs

	Our work	General RO PUF	Habib et al. [24]	Maiti et al. [16]
Number of ring oscillators	2	2	130	512
Average independent response bits	31	1	318	511
Bits per ring	16.5	0.5	2.44	≈ 1

4.4 The Evaluation Result on Other FPGAs

The above reliability and uniqueness are tested on Xilinx Virtex-5 FPGAs. We also conduct experiments on Xilinx Virtex-6 and Kintex-7 FPGAs. The results show that the intra-distance and inter-distance is 1.68% and 49.12% on Virtex-6 FPGAs, and is 1.62% and 48.95% respectively on Kintex-7 FPGAs. Therefore, our new proposed FROPUF is also available on these new fashioned FPGA products.

5 Further Discussion

In Sect. 3, our proposed second order difference calculation just generates a 31-bit response. However, follow Algorithm 2, we can get a 496-bit response from two ROs. Obviously, the Shanon entropy of this 496-bit response is less than 496 bits. On the observation of Sect. 3, a lower bound entropy of this 496-bit response is 31 bits. Based on the model proposed in Sect. 3, we can calculate the Shanon entropy of this 496-bit response as follows.

Based on the model proposed in Sect. 3, we have the formula (16) as follows because we assume no prior information about the response when only the manufacturing variation is present.

$$Prob(r_i = 1) = Prob(r_i = 0) = 0.5 \quad (1 \leq i \leq 496) \tag{16}$$

However, because of the existence of correlations, if we know some bits' value, some other bits' information leaks out. For example, if r_1 and r_2 are known, the value of r_{32} distributes as follows.

$$Prob(r_{32} = 0) = \begin{cases} 1 & if \quad r_1 = 1 \quad and \quad r_2 = 0, \\ 0.5 & if \quad r_1 = 1 \quad and \quad r_2 = 1, \\ 0.5 & if \quad r_1 = 0 \quad and \quad r_2 = 0, \\ 0 & if \quad r_1 = 0 \quad and \quad r_2 = 1. \end{cases} \tag{17}$$

Figure 9 shows the correlation between these 496 bits response. Based on formula (16) and Fig. 9, we can calculate the Shanon entropy of the 496-bit response as follows. The 31 bits in the first row of Fig. 9 have 31 bits Shanon entropy, and the 30 bits in second row have 15 bits Shanon entropy and so on. We can acquire that the responses in the i^{th} row have $\frac{1}{2^{i-1}}(32 - i)$ bits Shanon entropy. As a result, the Shanon Entropy of this 496-bit response is $\sum_{i=1}^{31} \frac{1}{2^{i-1}}(32 - i) = 60$ bits.

Algorithm 2. Simulation Algorithm of Second Order Difference Calculation

Settings: · There are two ROs, A and B.
· Both A and B have 5-bit configuration inputs.
· According to 32 different inputs, there will be $Counter_{A_j}$ and $Counter_{B_j}$, $1 \leq j \leq 32$

Output: · Response r_i, $1 \leq i \leq 496$

i ⟵ 0
for m = 1 to 32 **do**
  for n = m+1 to 32 **do**
    i ⟵ i + 1
    $\Delta Counter_{A(m,n)} \longleftarrow Counter_{A_m}$ - $Counter_{A_n}$
    $\Delta Counter_{B(m,n)} \longleftarrow Counter_{B_m}$ - $Counter_{B_n}$
    if $\Delta Counter_{A(m,n)} \geq \Delta Counter_{B(m,n)}$ **then**
      $r_i \leftarrow 1$
    else
      $r_i \leftarrow 0$
    end if
  end for
end for

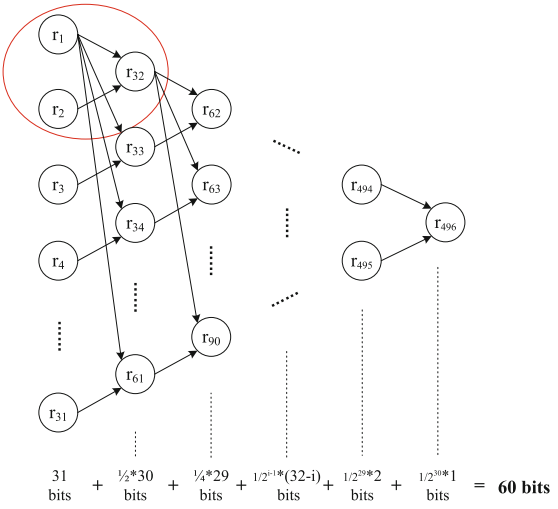

Fig. 9. The Shanon Entropy of the 496-bit response

6 Conclusion

In this paper, we propose a new architecture called Further RO PUF, which takes advantage of LUT's fine control to achieve more random manufacturing variations. Through the second order difference calculation, we on one hand extract more subtle features from RO pairs, and on the other hand, neatly reduce the

influence of systematic variation and environmental fluctuation to strengthen the reliability. The key point of FROPUF is that we can extract at least 31 bits entropy from only two ROs and according to our analysis, the Shanon Entropy of the response reaches 60 bits. By conducting simulation and practical experiments, we found the intra-distance of FROPUF is only 1.85% at 27 °C and never exceed 10% with drastic temperature changes, and the inter-distance is about 49.32%, which guarantees the uniqueness of different PUF instances.

References

1. Ruhrmair, U., Holcomb, D.E.: PUFs at a glance. In: Design, Automation and Test in Europe Conference and Exhibition (DATE), pp. 1–6 (2014)
2. 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, pp. 148–160 (2002)
3. Katzenbeisser, S., Kocabaş, Ü., Rožić, V., Sadeghi, A.-R., Verbauwhede, I., Wachsmann, C.: PUFs: myth, fact or busted? a security evaluation of physically unclonable functions (PUFs) cast in silicon. In: Prouff, E., Schaumont, P. (eds.) CHES 2012. LNCS, vol. 7428, pp. 283–301. Springer, Heidelberg (2012). doi:10.1007/978-3-642-33027-8_17
4. Ravikanth, P., Recht, B., Taylor, J., Gershenfeld, N.: Physical One-Way Functions, vol. 297. American Association for the Advancement of Science, Washington, DC (2002). pp. 2026–2030
5. Maes, R., Van Herrewege, A., Verbauwhede, I.: PUFKY: a fully functional PUF-based cryptographic key generator. In: Prouff, E., Schaumont, P. (eds.) CHES 2012. LNCS, vol. 7428, pp. 302–319. Springer, Heidelberg (2012). doi:10.1007/978-3-642-33027-8_18
6. Guajardo, J., Kumar, S.S., Schrijen, G.-J., Tuyls, P.: FPGA intrinsic PUFs and their use for IP protection. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 63–80. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74735-2_5
7. Roy, J.A., Koushanfar, F., Markov, I.L.: EPIC: Ending piracy of integrated circuits. In: Proceedings of the Conference on Design, Automation and Test in Europe, pp. 1069–1074 (2008)
8. Koeberl, P., Li, J., Maes, R., Rajan, A., Vishik, C., Wójcik, M.: Evaluation of a PUF device authentication scheme on a discrete 0.13 um SRAM. In: Chen, L., Yung, M., Zhu, L. (eds.) INTRUST 2011. LNCS, vol. 7222, pp. 271–288. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32298-3_18
9. Devadas, S., Suh, E., Paral, S., Sowell, R., Ziola, T., Khandelwal, V.: Design and implementation of PUF-based unclonable RFID ICs for anti-counterfeiting and security applications. In: 2008 IEEE International Conference on RFID, pp. 58–64 (2008)
10. Tuyls, P., Škorić, B.: Strong authentication with physical unclonable functions. In: Petković, M., Jonker, W. (eds.) Security, Privacy, and Trust in Modern Data Management. Data-Centric Systems and Applications, pp. 133–148. Springer, Heidelberg (2007)
11. Holcomb, D.E., Fu, K.: Bitline PUF: building native challenge-response PUF capability into any SRAM. In: Batina, L., Robshaw, M. (eds.) CHES 2014. LNCS, vol. 8731, pp. 510–526. Springer, Heidelberg (2014). doi:10.1007/978-3-662-44709-3_28

12. Kumar, S.S., Guajardo, J., Maes, R., Schrijen, G-J., Tuyls, P.: Extended abstract: the butterfly PUF protecting IP on every FPGA. In: IEEE International Workshop on Hardware-Oriented Security and Trust, HOST 2008, pp. 67–70 (2008)
13. Suzuki, D., Shimizu, K.: The glitch PUF: a new delay-PUF architecture exploiting glitch shapes. In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 366–382. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15031-9_25
14. Maes, R., Tuyls, P., Verbauwhede, I.: Intrinsic PUFs from flip-flops on reconfigurable devices. In: 3rd Benelux Workshop on Information and System Security (WISSec 2008) (2008)
15. Suh, G.E., Devadas, S.: Physical unclonable functions for device authentication and secret key generation. In: Proceedings of the 44th Annual Design Automation Conference, pp. 9–14 (2007)
16. Maiti, A., Schaumont, P.: Improved ring oscillator PUF: an FPGA-friendly secure primitive. J. Cryptology **2**, 375–397 (2011)
17. Gao, M., Lai, K., Qu, G.: A highly flexible ring oscillator PUF. In: Proceedings of the 51st Annual Design Automation Conference on Design Automation Conference. ACM (2014)
18. Rahman, T., Forte, D., Fahrny, J., Tehranipoor, M.: ARO-PUF: an aging-resistant ring oscillator PUF design. In: Proceedings of the Conference on Design, Automation & Test in Europe (2014)
19. Cherkaoui, A., Fischer, V., Aubert, A., Fesquet, L.: Comparison of self-timed ring and inverter ring oscillators as entropy sources in FPGAs. In: Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1325–1330. IEEE (2012)
20. Yin, C-E., Qu, G., Zhou, Q.: Design and implementation of a group-based RO PUF. In: Proceedings of the Conference on Design, Automation and Test in Europe, pp. 416–421. EDA Consortium (2013)
21. Dominik, M., Heyszl, J., Heinz, B., Schuster, D., Stumpf, F., Sigl, G.: Localized electromagnetic analysis of RO PUFs. In: 2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 19–24. IEEE (2013)
22. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 523–540. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24676-3_31
23. Bösch, C., Guajardo, J., Sadeghi, A.-R., Shokrollahi, J., Tuyls, P.: Efficient helper data key extractor on FPGAs. In: Oswald, E., Rohatgi, P. (eds.) CHES 2008. LNCS, vol. 5154, pp. 181–197. Springer, Heidelberg (2008). doi:10.1007/978-3-540-85053-3_12
24. Habib, B., Kris, G., Kaps, J-P.: FPGA PUF based on programmable LUT delays. In: 2013 Euromicro Conference on Digital System Design (DSD). IEEE (2013)
25. Majzoobi, M., Koushanfar, F., Devadas, S.: FPGA-based true random number generation using circuit metastability with adaptive feedback control. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 17–32. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23951-9_2
26. Majzoobi, M., Koushanfar, F., Devadas, S.: FPGA PUF using programmable delay lines. In: 2010 IEEE International Workshop on Information Forensics and Security (WIFS) (2010)
27. Rührmair, U., Sehnke, F., Sölter, J., Dror, G., Devadas, S., Schmidhuber, J.: Modeling attacks on physical unclonable functions, pp. 237–249 (2010)

28. Maes, R., Verbauwhede, I.: Physically unclonable functions: a study on the state of the art and future research directions. In: Sadeghi, A.-R., Naccache, D. (eds.) Towards Hardware-Intrinsic Security. Information Security and Cryptography, pp. 3–37. Springer, Heidelberg (2010)
29. Wold, K., Tan, C.H.: Analysis and enhancement of random number generator in FPGA based on oscillator rings. In: Proceedings of the 2008 International Conference on Reconfigurable Computing and FPGAs, pp. 385–390. IEEE Computer Society (2008)
30. Maiti, A., Casarona, J., McHale, L.: A large scale characterization of RO-PUF. In: IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 94–99. IEEE (2010)
31. Maes, R., Tuyls, P., Verbauwhede, I.: Low-overhead implementation of a soft decision helper data algorithm for SRAM PUFs. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 332–347. Springer, Heidelberg (2009). doi:10.1007/978-3-642-04138-9_24