Design and Implementation of a Group-based RO PUF

Chi-En Yin and **Gang Qu**

Electrical and Computer Engineering Department
and Institute for Systems Research
University of Maryland, College Park, MD USA

Qiang Zhou

Department of Computer Science and Technology
Tsinghua University, Beijing, P. R. China

ABSTRACT

The silicon physical unclonable functions (PUF) utilize the uncontrollable variations during integrated circuit (IC) fabrication process to facilitate security related applications such as IC authentication. In this paper, we describe a new framework to generate secure PUF secret from ring oscillator (RO) PUF with improved hardware efficiency. Our work is based on the recently proposed group-based RO PUF with the following novel concepts: an entropy distiller to filter the systematic variation; a simplified grouping algorithm to partition the ROs into groups; a new syndrome coding scheme to facilitate error correction; and an entropy packing method to enhance coding efficiency and security. Using RO PUF dataset available in the public domain, we demonstrate these concepts can create PUF secret that can pass the NIST randomness and stability tests. Compared to other state-of-the-art RO PUF design, our approach can generate an average of 72% more PUF secret with the same amount of hardware.

I. INTRODUCTION

A Motivation

With the increasing desire for security, privacy protection, and trustworthy computing, it becomes vital to protect cryptographic keys stored on computing devices. In the design of a security system, it is normally assumed that cryptographic keys are stored in tamper-proof devices such that the price to steal or compromise the keys will be forbiddingly high. Recently, the concept of physical unclonable function (PUF) has been applied to key generation and storage [1-3].

By means of storing secrets in its unique intrinsic physical features that are randomly determined by fabrication variations such as the subtle difference in the delays of two wires with equal length at the design phase, PUF achieves a higher level of protection without relying on persistent power. The validity of the claim rests on the insight that attempts in conducting invasive attacks will alter the unique intrinsic features and therefore destroy the secret hidden in the victim devices with a higher probability. Owing to its strong unforgeability, PUF has found applications in many security-related areas [4-9] such as circuit identification and authentication, hardware intellectual property protection, device remote activation, and trustworthy computing. Most of the existing works propose various PUFs and their potential applications with little consideration on the following two metrics:

- **hardware efficiency**: how to generate the secret or key with the minimum hardware overhead.
- **security of the key/secret**: how to ensure that the secret generated by a PUF is random and hard to break.

978-3-9815370-0-0/DATE13/©2013 EDAA

It is hard to adopt a silicon PUF in many applications such as embedded trustworthy computing systems when the PUF is expensive, in terms of hardware, to implement. Meanwhile, the "uncontrollable" physical features behind PUFs do not guarantee the randomness of the PUF secret and "physical unclonable" is insufficient to address the security concerns.

The longest increasing subsequence-based grouping algorithm (LISA) was proposed in [10] to enhance the hardware utilization of ring oscillator (RO) PUF. LISA is conceptually different from the conventional RO PUF that uses RO pairs to generate PUF secret because it generated multiple bits simultaneously based on a group of ROs. However, the authors in [10] failed to address the following problems:

- fabrication variation has a significant impact on the PUF secret generated by the group-based RO PUF. This problem is not identified in [10].
- the group-based RO PUF has several unique features. It is not appropriate and will not be efficient to use the design flow for pair-based RO PUF, as [10] did, without leveraging these features.
- LISA requires the measurement of each RO's delay (or speed) at two extreme temperature (0^oC and 100^oC), which may not be realistic.

In this paper, we study the above challenges and propose several solutions to design and implement the group-based RO PUF.

B Ring Oscillator PUF

The basic RO PUF consists of two identical ring oscillators, which due to fabrication variation will have tiny difference in delay (or speed). One bit can be defined by comparing the speed and this bit will be equally likely to be a 0 or a 1 as long as the fabrication variation is random.

However, the bit we extract (or generate) from a pair of ROs this way may not be reliable and unclonable. For example, operating environment such as temperature and voltage has significant impact on delay. When this impact is sufficiently large, it becomes possible that one RO is faster than the other at one temperature, but becomes slower at another temperature, causing the bit generated from this pair of ROs to flip when temperature changes [2].

One way to solve this is to select only those pairs with frequency difference large enough to sustain temperature variation. For instance, Suh and Devadas [2] proposed to generate one reliable unclonable bit from 8 ROs that are hard wired as a unit as showed in Figure 1, where N = 8. The fastest and slowest ROs in each unit are picked to form a pair to generate a bit. This scheme uses two ROs with large speed difference and thus will be more reliable subject to operating environment changes. The use of N ROs,

multiplexers, counters, and comparators makes this approach very hardware expensive.

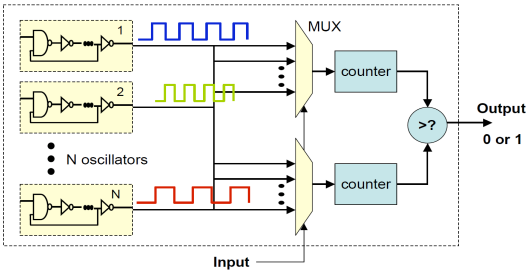

Fig. 1. 1-out-of-N ring oscillator PUF architecture [2].

Another known weakness of RO PUF is that it is subject to the negative effect of correlated or systematic variation, which weakens the security of PUF secret. Maiti and Schaumont [12] proposed to avoid this by considering only ROs that are physically close. However, their PUF secret is not uniformly distributed and not statistically random.

C Group-Based RO PUF and LISA

If we restrict the use of ROs no more than once, schemes by pairing N ROs can extract no more than $\lfloor N/2 \rfloor$ bits. Yin and Qu [10] proposed a group-based RO PUF to beat this $\lfloor N/2 \rfloor$ upper bound by comparing more than two ROs at the same time.

Suppose we order 4 ROs, {A,B,C,D}, by their speed from the fastest to the slowest. There will be 4! = 24 different orderings (or permutations), {ABCD}, {ABDC}, {ACBD}, {ACDB} … {DCBA}. Each ordering is unique and equally likely if the ROs' speeds are distinct and random, which what it is believed due to fabrication variation. We can encode these permutations with five bits 00000, 00001, 00010, 00011 … 10111, respectively as shown in Figure 2 and call this Compact Syndrome Coding (CSC). This bit string can be considered as the random physical unclonable secret extracted from RO PUF in binary. For example, if the speed goes from high to low in the order of ADCB, then we say the secret information is 00101. In this way, we can have 24 pieces of different secret information, much more than what $\lfloor 4/2 \rfloor$ = 2 bits can represent according to the pair-based approach.

00000_2	$\{ABCD\}$	01000_2	$\{BCAD\}$	10000_2	$\{CDAB\}$
00001_2	$\{ABDC\}$	01001_2	$\{BCDA\}$	10001_2	$\{CDBA\}$
00010_2	$\{ACBD\}$	01010_2	$\{BDAC\}$	10010_2	$\{DABC\}$
00011_2	$\{ACDB\}$	01011_2	$\{BDCA\}$	10011_2	$\{DACB\}$
00100_2	$\{ADBC\}$	01100_2	$\{CABD\}$	10100_2	$\{DBAC\}$
00101_2	$\{ADCB\}$	01101_2	$\{CADB\}$	10101_2	$\{DBCA\}$
00110_2	$\{BACD\}$	01110_2	$\{CBAD\}$	10110_2	$\{DCAB\}$
00111_2	$\{BADC\}$	01111_2	$\{CBDA\}$	10111_2	$\{DCBA\}$

Fig. 2. Motivation for the group-based RO PUF[10].

A longest increasing subsequence algorithm (LISA) is proposed in [10] to partition the ROs into groups, where every pair of ROs in each group will have a fixed relative speed under different operating temperature (that is, one RO is always faster than the other). Then multiple bits can be generated based on the speed orderings of the ROs in each group as shown in Figure 2. This guarantees that the generated PUF bits are reliable, but it is required to measure each RO's speeds at the lowest and highest operating temperatures.

D Standard PUF Design Flow

PUF design includes secret enrollment and secret regeneration. The typical workflow of a RO PUF enrollment process has the following four steps:

- **Fabrication Variation Extraction**: The first step in PUF design is to measure the unique characteristics endowed from the uncontrollable fabrication process. In the case of RO PUF, this corresponds to a full speed (or frequency) characterization of all ROs [14].
- **Secret Selection:** This step selects secure and reliable secrecy out of the variation profile measured in the previous step. Existing approaches include 1-out-of-8 coding [2], the index-based syndrome (IBS) coding [11] and the chain-like neighbor coding [12,14,15].
- **Error Correction:** To further enhance reliability, error correcting code (ECC) is applied. Codes have been used for RO PUFs include Hamming and BCH codes [2].
- **Tests for Security and Reliability:** Randomness test ensures the security of the PUF secret. Reliability test verifies the PUF secret can be regenerated under environmental fluctuations such as in temperature and supply voltage.

As we will see in Figure 3. certain public helper data will also be computed and stored in the chip to facilitate the PUF secret regeneration.

II. LIMITATIONS OF THE CURRENT GROUP-BASED RO PUF AND LISA

Although the group-based RO PUF makes it possible to generate more than $\lfloor N/2 \rfloor$ bits from N ROs, it has three major deficiencies:

First, LISA builds group-based RO PUF from the frequency measurements at the two operating temperature boundaries (0°C and 100°C). This conservative algorithm tends to partition the ROs into small groups, which is inefficient in generating PUF secret. Furthermore, this becomes impractical when we consider other extreme operating environments such as voltage variation.

Second, it is well-known that the semiconductor fabrication process has a strong spatial correlation, referred to as systematic variation. This has to be considered in the design of RO PUF because otherwise the bits generated will have strong correlation with the ROs' position on the chip and thus will be easy to break [13-15]. Group-based RO PUF creates multiple bits from a group of ROs and will have stronger spatial correlation. The authors in [10] fail to address this very important security concern.

Last but not the least, the CSC scheme in group-based RO PUF (see Figure 2) does not work well with error correcting codes (ECC). For instance, the ordering of BACD is encoded as 00110. When RO C becomes faster than A due to certain environment change or circuit aging, the order becomes BCAD, which has code 01000 and the ECC has to be able to correct three bits in this case. On the other hand, for the very unusual event that BACD changes to DCAB, the ECC only needs to correct the first bit. This indicates that the CSC scheme is not suitable for ECC because the probability that an error occurs does not correlate well with the number of bits that needs to be corrected when that error occurs.

To summarize, if we implement the concept of group-based RO PUF proposed in [10] using the standard PUF design flow, we find that the fabrication variation extraction phase is expensive and may not be practical; the secret selection algorithm LISA is pessimistic and may not be efficient in generating PUF secret; the CSC scheme is not suitable for error correction; and the nature of group-base PUF secret generation may have serious security flaw due to the spatial correlation of fabrication variation, but there is no report on security and stability tests.

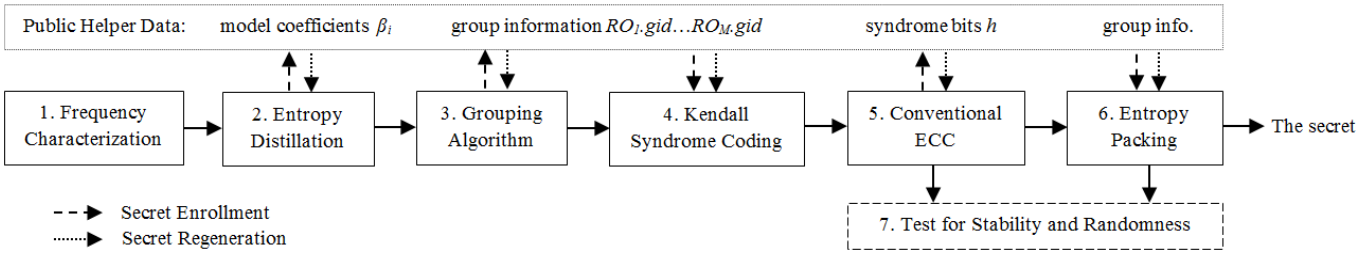

Fig. 3. Overview of the proposed group-based RO PUF design and implementation

III. DESIGN AND IMPLEMENATION OF THE GROUP-BASED RO PUF

A RO PUF typically consists of a group of ROs as well as countes and multiplexers to help collect the frequency readings of the RO array. Figure 3 depicts the overview of our proposed architecture to design and implement the group-based RO PUF.

1. In the frequency characterization phase, we will adopt exact the same method as used in pair-based RO PUF. Indeed, we will use the public data set for pair-based RO PUF [14] to build the group-based RO PUF.
2. We propose an entropy distillation phase where we will filter out the semiconductor's spatial trend in fabrication variation such that our PUF secret selection will be based only on the random variation.
3. We develop a low complexity algorithm to replace LISA for the partitioning of ROs to groups based on the frequency characterization in phase 1.
4. We propose to use Kendall Syndrome Coding (KSC) instead of CSC to facilitate the error correction process.
5. The error correction phase is the same and any ECC can be used.
6. An entropy packing phase is introduced to enhance the security and efficiency of KSC.
7. We use the public RO PUF data set [14] to test the stability of the generated PUF secret and use standard NIST test suites to test its randomness.

In the rest of this section, we will elaborate the technical details of phases 2, 3, 4, and 6. The test results of phase 7 will be reported in Section 4.

A Entropy Distiller

Fabrication variation consists of a systematic component and a random component. The main causes of the systematic variation are attributed to equipment and process non-uniformity such as focus shift of photolithography, gradient of thermal annealing, dissimilar interactions between circuit layout and the chemical mechanical polishing process. The goal of the distiller is to model the systematic variation such that we can filter out most of it and build PUF secret from the remaining true random variation (Hence, it is not necessary to model the systematic variation accurately, which is still an open problem). For this purpose, we use the polynomial regression model because of its simplicity.

A k^{th}-order polynomial regression is a form of linear regression in which the relationship between independent variables and a dependent variable is modeled by a polynomial of order k. For a RO PUF with its m ROs arranged in r rows by c columns, the Cartesian coordinates (x, y) (or its physical coordinates (v_x, h_y) on the chip) of ROs are regarded as two independent variables and the oscillating frequency $z_{x,y}$ is a variable dependent on v_x and h_y. In such a two dimensional setting, a polynomial regression model of order k takes the following general form

$$z_{x,y} = \sum_{i=0}^{k} \sum_{j=0}^{i} \beta_{k,i,j} v_x{}^{i-j} h_y{}^{j} + \epsilon_{k,x,y} \qquad (1)$$

where $1 \le x \le c$; $1 \le y \le r$; $z_{x,y}, \beta_{k,i,j}, \varepsilon_{k,x,y} \in R$. On the right hand side of the equation, the summation term models the systematic variation and the residual term $\varepsilon_{k,x,y}$ models the random variation. In the k^{th}-order polynomial model, there will be $m = c \times r$ equations in the form of Eqn. (1) with $1 \le x \le c$ and $1 \le y \le r$. Because $0 \le j \le i \le k$, the number of unknowns $\beta_{k,i,j}$'s will be $n = (k+1)(k+2)/2$, which is much less the number of ROs on the chip (normally in the order of hundreds to generate any meaningful PUF secret). This results in an over-determined system (i.e. m > n), which can be solved by standard approaches such as the least squares method.

Like the frequency characterization phase, this phase can be done during testing for each chip. The obtained coefficients $\beta_{k,i,j}$'s will be stored as the public helper data, while the random variation $\varepsilon_{k,x,y}$ will be used to generate PUF data. When we need to regenerate the same PUF data, the frequency of each RO will be measured again, then the systematic variation will be filtered out from the measurement $z'_{x,y}$ by using the stored help data $\beta_{k,i,j}$'s to reveal the random variation $\varepsilon'_{k,x,y}$, which will be used in later phases to retrieve the original PUF secret, as shown in Figure 3.

B Grouping Algorithm

The goal of the step is to partition the m ROs G into subgroups $G_1, G_2, \ldots$ based on their frequency measurements RO_i's at a single operating environment in order to ***maximize*** $\sum \log_2(|G_i|!)$ under the constraints of

a) $G_i \cap G_j = \Phi$ for any $i \neq j$
b) $G_1 \cup G_2 \cup \ldots = G$, or $|G_1| + |G_2| + \ldots = m$
c) the RO frequency difference $|RO_i - RO_j| \ge f_{th}$ for any two ROs in the same group.

The optimization objective is the amount of PUF secret that can be generated from the given ROs using the group-base approach. Constraint a) ensures that no RO will be used more than once in order to avoid potential security concerns. Constraint b) indicates that all the ROs will be used once, which is necessary to maximize our objective function. The original LISA approach [10] measures each RO's frequency at two extreme operating conditions to provide the robustness of the PUF secret. We use only one measurement and require the frequency discrepancy between any two ROs in the same group to be higher than a given threshold f_{th}, in constraint c). This constraint combined with error correcting code will enable us to regenerate the PUF secret robustly.

Figure 4 shows a constructive grouping algorithm that determines the group index for each RO one by one (the `for` loop in line 3).

`last[]` in line 2 records the slowest RO in each group. The `while` loop in lines 5-10 add RO_i to the first group whose slowest RO is at least f_{th} faster than RO_i to meet constraint c). This grouping information, RO_i`.gid`, is also kept as the public helper data and will be used when PUF secret needs to be regenerated (see Figure 3).

```
Input:
1)  m ROs sorted by the frequency measurements
    RO1>RO2>...>ROm.
2)  frequency discrepancy threshold fth.
Output: the group index for each RO, ROi.gid
Algorithm:
1.  RO0 = ∞;
2.  for (i=1; i≤m; i++)  last[i] = 0;
3.  for (i=1; i≤m; i++)
4.    done = 0; j = 1;
5.    while (!done)
6.      if (ROlast[j] − ROi > fth)
7.        ROi.gid = j;
8.        last[j] = i;
9.        done = 1;
10.     j++;
```

Fig. 4. Pseudo-code of the proposed grouping algorithm.

C Kendall Syndrome Coding

When we have n ROs in a group and define PUF secret based on the order of their frequency RO_i (i=1,2,…,n), an error during PUF secret regeneration occurs when the measured frequencies do not follow the same order. For example, when four ROs {A,B,C,D} are enrolled in the order of BACD, from the fastest to the slowest, a measurement in the order of BCAD would be an error.

As we have demonstrated in Section 2, when we use traditional compact syndrome coding (CSC) for group-based PUF, the cost for error correcting code (ECC) will be high. In the above example, BACD has code 00110 and BCAD's code is 01000, which means that the ECC needs to be able to correct three bit errors in this case.

Considering constraint c) in our proposed grouping algorithm, we conclude that most of the errors happen in the form of a flip between two ROs whose frequencies are adjacent when they are ordered. The change from BACD to BCAD is one example where RO C becomes faster than RO A. This is because when we construct the group, any two adjacent ROs must have their frequencies differ by at least f_{th}. Therefore, a flip between non-adjacent ROs will require a change on RO frequency to be $2f_{th}$ or higher, which is very unlikely. Base on this observation, we propose the following non-minimum length encoding scheme:

For n ROs with frequency RO_i (i=1,2,…,n), we define its code as a k-bit string

$$s_{gi} = \delta(1,2)\delta(1,3)\ldots\delta(1,n)\delta(2,3)\ldots\delta(2,n)\ldots\delta(n-1,n) \quad (2)$$

where $k=n(n-1)/2$, $\delta(i,j) = 0$ if $RO_i < RO_j$ and $\delta(i,j) = 1$ otherwise. The advantage of this coding scheme is that when a flip between two adjacent ROs occurs, there will be only one bit error in the coding. For example, BACD is coded as *100000* and BCAD's code is 110000, they differ only at the second bit.

We refer to this coding scheme as Kendall syndrome coding (KSC) because of its similarity to the notion of *Kendall tau distance* [16]: Given two permutations $\sigma=(\sigma(1), \sigma(2), \ldots, \sigma(n))$ and $\pi=(\pi(1), \pi(2), \ldots, \pi(n))$ on same set of n elements (such as integers 1, 2, …, n), the *Kendall tau distance* $d_\tau(\sigma,\pi)$ is defined by

$$d_\tau(\sigma,\pi) = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} s(i,j) \quad (3)$$

where $s(i,j) = 1$ if $\{\sigma(i)<\sigma(j)$ and $\pi(i)>\pi(j)\}$ or $\{\sigma(i)>\sigma(j)$ and $\pi(i)<\pi(j)\}$ and $s(i,j) = 0$ otherwise. For example, when $\sigma=(1,3,2)$ and $\pi=(1,2,3)$, we have $s(1,2)=0$, $s(1,3) =0$, $s(2,3)=1$ and thus $d_\tau(\sigma,\pi) =1$. The permutation σ can also be conveniently retrieved from its definition of Kendall tau distance $d_\tau(\sigma,\pi)$ to the identity permutation $\pi=(1,2,\ldots,n)$ [16]:

$$\sigma(i) = 1 + \sum_{j=1}^{i-1}(1 - s(i,j)) + \sum_{j=i+1}^{n} s(i,j) \quad (4)$$

The bit strings generated by equation (2) from each group of ROs will be concatenated, but it cannot be considered as the PUF secret because it is not robust, secure, and efficient. We now address these concerns.

D ECC and Entropy Packing

A linear block code (p, q, t) can be used to correct runtime errors, where p is the block size, q is the number of information bits, p-q is the number of parity bits that enables the correction of up to t errors within the block. We adopt the code-offset technique with syndrome bits because it can bound the min-entropy loss [2].

When we enroll (or determine) the PUF secret, the KSC coded output is divided into *p-bit* blocks. Each block is considered as a linear block code (p, q, t), where the first q bits are information based on which we can compute p-q parity bits following certain ECC scheme (such as BCH or Hamming code); these parity bits will be exclusive-or-ed (⊕) with the last p-q bits in the block to produce the syndrome bits h. The syndrome bits will be saved as public helper data to assist secrecy regeneration (see Figure 3).

To recover the enrolled secret from a new measurement of RO frequencies, we generate *p-bit* blocks and exclusive-or (⊕) the last p-q bits with the saved syndrome bits h to retrieve the p-q parity bits. These parity bits will replace the last p-q bits to form a new *p-bit* block with the original first q bits. As long as there are no more than t bits of error, ECC will enable us to regenerate the PUF secrecy (the first q bits in each block) correctly.

Compared to CSC, KSC facilitates ECC because it reduces the number of bit flips in its code words when the RO frequency measurements have an order different from the one at enrollment. That is, an ECC will need to correct fewer errors (smaller t) in KSC coded words than in CSC coded words. However, KSC leaves many code words unused. Consider a group of three ROs, codes 010 and 101 are invalid according to the definition in equation (2). (The two 0's in 010 indicate $RO_1<RO_2$ and $RO_2<RO_3$, so $RO_1<RO_3$ and the second bit cannot be 1). On one hand, this makes KSC inefficient. On the other hand, such logical dependency will make the code words non-uniform and weaken the security of the PUF data.

To solve this problem, we add an entropy packing phase in the PUF secret regeneration process (see Figure 3). After we retrieve the KSC coded string with the help of ECC, we encode the information of each group (that is, the frequency order of all the ROs in the same group) in the most compact form such as the lexicographic order in Figure 2. Assuming that there are n ROs in a group, this information will be a permutation of $\{1,2,\ldots,n\}$. Note that $\delta(i,j)$ defined in equation (2) equals to $s(i,j)$ in equation (3) when $\pi=(1,2,\ldots,n)$ is the identity permutation. From equation (4), we can determine the permutation corresponding to the RO

frequency order at PUF secret regeneration time. Then we can apply standard procedures to convert this permutation information in a compact form.

IV. RESULTS ON RANDOMNESS AND STABILITY TESTS

In this section, we report the results on randomness and stability of the PUF secret generated by the proposed group-based approach from a public RO PUF dataset [14].

A Dataset and Test for Randomness

We use the RO PUF dataset built by researchers in Virginia Tech which is publically available at [14]. This dataset comprises frequency characterization of 125 Xilinx Spartan-3 (90-nm) FPGAs at different operating environment. For each FPGA, 512 ROs are placed in 32 rows and 16 columns. Measurements are collected with 1.2V supply voltage with ±10% and ±20% fluctuation, and temperature variation from 25°C to 65°C with a 10°C increment. We use the measurement at 1.2V and 25°C as the RO frequency characterization output of phase 1 in Figure 3.

NIST's statistical test suite for random and pseudorandom number generators designed for cryptographic applications [17] is used to test whether the PUF secret generated above is random. 11 out of the 15 tests in the suite are applicable and they are shown in the last column of Table 1. The parameters are set as below following NIST recommendations: bitstring length is 400 (except 120 for FFT Test in order to meet with the minimum length requirement), 32 as the block length for Frequency Test, 2 for Approximate Entropy Test and 5 for Serial Test.

P-VALUE	PROPORTION	STATISTICAL TEST
0.609372	63/64	Frequency
0.465914	62/64	BlockFrequency
0.517608	63/64	CumulativeSums (m-2)
0.109242	63/64	CumulativeSums (m-3)
0.023812	63/64	Runs
0.366511	63/64	LongestRun
0.817009	64/64	Rank
0.028264	21/21	FFT
0.517608	63/64	ApproximateEntropy
0.933004	64/64	Serial (forward)
0.380722	64/64	Serial (backward)

Table 1. Parameters used in cost-performance calculation.

NIST test results are interpreted in two ways: (1) the proportion of total bitstrings that pass a test shall be above a minimum value; (2) the P-values of all bitstrings shall be uniformly distributed such that the P-value of the P-values is equal or greater than a minimum value; default settings were used in the test suite.

The test results from the first half of the dataset are used to guide us for the modeling of systematic variation in the distillation phase and for the selection of value f_{th}. In this case, the results suggest that we can select 1st-order polynomial to remove the systematic component of fabrication variation. When we use this model on the second half of the dataset, the PUF secret built based on the remaining true variation passes all the 11 NIST randomness tests for both proportion and P-value (see Table 1).

B Test for Stability

Stability measures PUF's ability to regenerate the enrolled PUF secret. It is affected by the selection of f_{th} and ECC. Large f_{th} will reduce errors and ECC can correct errors. We consider three classes of *BCH(n,k,t)* codes with n=31, 63, and 127, where n is the block size, k is the information bits per block, and t is the maximum number of correctable bit errors in a block. For f_{th}, we use values between 1 and 3 standard deviations of the 512 ROs' random variations (the variation after distillation in phase 2).

As we have discussed earlier, a group of $|g_i|$ ROs can generate roughly $log_2|g_i|!$ bits of information. However, for each *n-bit* block, we have a min-entroy loss of *n-k* bits due to the public disclosure of the syndrome bits for ECC in phase 5 (in Figure 3). Therefore, we define the effective min-entropy H_∞^{KSC} for KSC as

$$\sum_{i=1}^{|G|} \log_2 |g_i|! - \lceil \frac{\sum_{i=1}^{|G|} |s_{g_i}|}{n} \rceil (n-k) \qquad (5)$$

where |G| is the number of groups the grouping algorithm in Figure 4 returns, $|s_{gi}|$ is the length of the KSC code for group g_i as defined in equation (2). The min-entropy for other coding schemes such as CSC and IBS can be defined similarly, we denote them by H_∞^{CSC} and H_∞^{IBS}, respectively.

For a given f_{th}, if the ECC is unable to correct the errors in a block, we set the min-entropy value to be 0. Otherwise, the ECC will be able to provide stability and the min-entropy value shows the amount of robust bits that the PUF can generate. For the purpose of stability test, we enroll the data at 1.2V and 25°C from [14] as the PUF secret, and attempt to use data from the same source under the following six different environments to retrieve the PUF secret: (1.08V, 25°C), (1.32V, 25°C), (1.2V, 35°C), (1.2V, 45°C), (1.2V, 55°C), (1.2V, 65°C). Figure 5 reports the results when *BCH(31,k,t)* is used as the ECC, similar results hold for the cases of *BCH(63,k,t)* and *BCH(127,k,t)*.

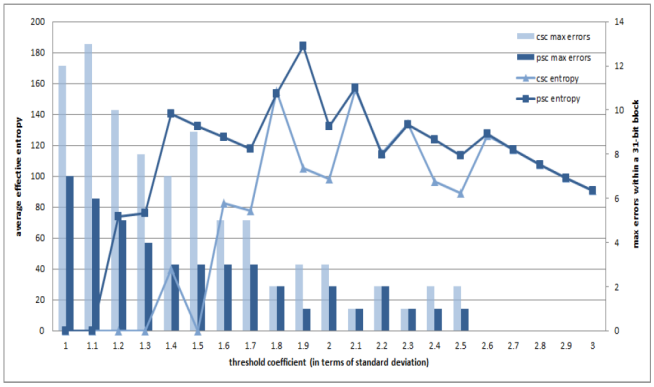

Fig. 5. The average min-entropy (line segments) and the maximum number of errors per block (vertical bars) for KSC and CSC with *BCH(31, k,t)* as the ECC. X-axis shows f_{th} goes from 1.0x to 3.0x, with a step of 0.1, standard deviation of the RO random variations.

First we see that as f_{th} goes from 1.0x to 3.0x standard deviation of the RO random variations, the maximum number of errors per block decreases. When f_{th} is 2.6x and higher, there is no error. This verifies that large f_{th} provides high stability. We also see that KSC (the dark bar) has fewer errors than CSC in general. In the 31-bit block, CSC suffers a maximum number of 13 error bits (at f_{th} = 1.1x), while KSC only has 7 error bits (at f_{th} = 1.0x).

Second, for the amount of PUF secret, we see that KSC reaches its maximum min-entropy of 184 bits at f_{th} = 1.9, while the maximum min-entropy for CSC is 160 bits. KSC is 15% better. The trend of min-entropy curve shows that (1) when f_{th} is small, the number of errors per block is too large for BCH to correct and the min-entropy remains 0, which means that we are unable to generate any robust PUF secret; (2) as f_{th} increases, the number of

errors in a block decreases and the min-entropy increases; (3) when f_{th} becomes very large (2.0x or higher), the min-entropy starts decreasing. This last observation is due to the fact that the size of each RO group will shrink as f_{th} increases, resulting a reduction of min-entropy from the first term in equation (5).

V. COMPARISON WITH PAIR-BASED RO PUF

We have described the design and implementation of the group-based RO PUF in section 3 and performed the randomness and stability tests on the PUF secret it generates in section 4. Now to demonstrate the performance of the proposed group-based RO PUF, we compare it with the best known pair-based approach, the index based scheme (IBS) reported in [11].

In the IBS-based RO PUF, ROs are partitioned into blocks, each block will generate one bit PUF secret by comparing the fastest RO and the slowest RO at enrollment time. The indices of the selected pair of ROs are kept as public helper data to regenerate the PUF secret.

We implement IBS-based RO PUF following the description in [11]. Because there is no distillation phase, we form IBS blocks by k ROs that are physically close to each other to reduce the systematic correlation [12]. Clearly, as k increases, the frequency discrepancy between the fastest and the slowest ROs in the same block will increase and the PUF bit becomes more stable. The same BCH schemes are used as ECC and we define IBS RO PUF's min-entropy similarly for comparison purpose.

Using the same public dataset [14], we observe that when k=6, there will not be any error as shown in Figure 6. However, similar to our group-based RO PUF, the maximum min-entropy is not achieved at the point when there is no error. In the IBS case, this is because smaller block size will result in more blocks (and thus higher min-entropy) when the total number of ROs is fixed.

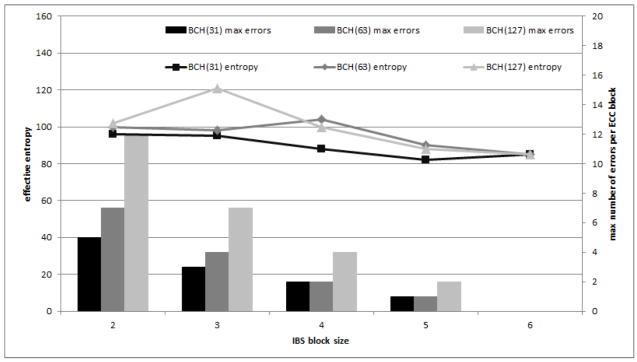

Fig. 6. The average min-entropy (line segments) and the maximum number of errors per block (vertical bars) for IBS with *BCH* as the ECC. X-axis shows the number of ROs in each block.

Table 2 below compares the PUF secret generation ability by the IBS-based approach and our proposed group-based approach with different ECCs. The "No ECC" row represents the cases when there is no error and ECC is not necessary. That is, block size $k=6$ in IBS and $f_{th} = 2.6x$ standard deviation in KSC (see Figure 5 and Figure 6). The last column "Gain" clearly shows that the proposed group-based approach outperforms the pair-based approach by a large margin, 72% on average. It means that based on the same dataset and hardware (ROs), the group-based approach can generate 72% more PUF secret. Or in another word, to generate the same amount of PUF secret, group-based approach will require roughly 42% less hardware. (1-1.0/1.72 ≈ 0.42).

	H_∞^{IBS}	H_∞^{KSC}	Gain
No ECC	85	128	50%
BCH(31)	96	184	92%
BCH(63)	104	189	82%
BCH(127)	121	197	63%
Average	107	190	72%

Table 2. The min-entropy achieved by pair-based RO PUF (H_∞^{IBS}) and the proposed group-based RO PUF (H_∞^{KSC}).

VI. CONCLUSIONS

In this work, we develop a method to design and implement the newly proposed group-based RO PUF. We introduced several new phases in the PUF secret enrollment process to make group-based RO PUF more effective. We use public RO PUF dataset to demonstrate the randomness and stability of the generated PUF secret. We also demonstrate that the group-based RO PUF significantly improves the hardware efficiency.

Acknowledgement: This work is supported in part by the National Science Foundation of China under grant 61228204.

VII. REFERENCES

[1] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical One-Way functions," Science, Sep 2002.

[2] G. E. Suh, S Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generateion," ACM DAC 2007.

[3] D. Lim, J-W. Lee, B. Gassend, M. van Dijk, E. Suh, and S. Devadas, "Extracting Secret Keys from Integrated Circuits," IEEE Transactions on VLSI Systems, Oct 2005.

[4] S. Devadas, V. Khandelwal, S. Paral, R. Sowell, E. Suh, T. Ziola, "Design and Implementation of Unclonable RFID ICs for Anti-Counterfeiting and Security Applications," RFID World 2008, Mar 2008.

[5] J. Guajardo, S. Kumar, G. Jan Schrijen, P. Tuyls, "FPGA Intrinsic PUFs and Their Use for IP Protection," Workshop on Cryptographic Hardware and Embedded Systems (CHES), Sep 10-13, 2007, Vienne, Austria.

[6] Jorge Guajardo, Sandeep S. Kumar, Geert-Jan Schrijen, Pim Tuyls, "Physical Unclonable Functions and Public-Key Crypto for FPGA IP Protection," International Conference on Field Programmable Logic and Applications (FPL), Aug 27-29, 2007, Amsterdam, Netherland.

[7] S. Kumar, J. Guajardo, R. Maes, G.J. Schrijen qnd P. Tuyls, "The Buttefly PUF: Protecting IP on every FPGA," In IEEE International Workshop on Hardware Oriented Security and Trust, Anaheim 2008.

[8] Jason H. Anderson, "A PUF design for secure FPGA-based embedded systems," ASP-DAC 2010

[9] U. Ruhrmair, S. Devadas, and F. Koushanfar. Security Based on Physical Unclonability and Disorder. In: M. Tehranipoor and C. Wang (eds.) Introduction to Hardware Security and Trust, pp. 65-102, Springer, 2011.

[10] C.-E. D. Yin and G. Qu, "Lisa: Maximizing ro puf's secret extraction," Proceedings of 3rd IEEE International Workshop on Hardware Oriented Security and Trust (HOST), June, 2010.

[11] M.-D. Yu and S. Devadas, "Secure and robust error correction for physical unclonable functions," IEEE Journal of Design & Test Computers, Vol. 27, Issue 1, Jan. 2010.

[12] A. Maiti, P. Schaumont, "Improving the quality of a Physical Unclonable Function using configurable Ring Oscillators," FPLA 2009.

[13] B. Škorić, P. Tuyls, "An efficient fuzzy extractor for limited noise," IACR ePrint 2009 http://eprint.iacr.org/2009/030

[14] A. Maiti and P. Schaumont, "A large scale characterization of ro-puf," Proceedings of 3rd IEEE International Workshop on Hardware Oriented Security and Trust (HOST), Jun. 2010.

[15] F. S. Dominik Merli and C. Eckert, "Improving the quality of ring oscillator pufs on fpgas," Proceedings of the 5th Workshop on Embedded Systems Security, Oct. 2010.

[16] H. Chadwick and I. Reed, "The equivalence of rank permutation codes to a new class of binary codes," IEEE Transactions on Information Theory, Vol. 16, No. 5, pp. 640–641, 1970.

[17] A. Rukhin, J. Soto, J. Nechvatal, et al, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," NIST Special Publication 800-22 Revision 1a, Apr. 2010.