IACR Transactions on Cryptographic Hardware and Embedded Systems
ISSN 2569-2925, Vol. 2020, No. 4, pp. 86–128. DOI:10.13154/tches.v2020.i4.86-128

Rejection Sampling Schemes for Extracting Uniform Distribution from Biased PUFs

Rei Ueno, Kohei Kazumori and Naofumi Homma

Research Institute of Electrical Communication, Tohoku University
2–1–1 Katahira, Aoba-ku, Sendai-shi, 980-8577, Japan
E-mail: ueno@riec.tohoku.ac.jp, homma@riec.tohoku.ac.jp

Abstract. This paper presents an efficient fuzzy extractor (FE) construction for secure cryptographic key generation from physically unclonable functions (PUFs). The proposed FE, named acceptance-or-rejection (AR)-based FE, utilizes a new debiasing scheme to extract a uniform distribution from a biased PUF response. The proposed debiasing scheme employs the principle of rejection sampling, and can extract a longer debiased bit string compared to those of conventional debiasing schemes. In addition, the proposed AR-based FE is extended to ternary PUF responses (*i.e.*, ternary encoding of a PUF response). These responses can be derived according to cell-wise reliability of the PUF and are promising for extraction of stable and high-entropy responses from common PUFs. The performance of the AR-based FEs is evaluated through an experimental simulation of PUF-based key generation and compared with conventional FEs. We confirm that the proposed AR-based FE can achieve the highest efficiency in terms of PUF and nonvolatile memory (NVM) sizes for various PUF conditions among the conventional counterparts. More precisely, the AR-based FE can realize a 128-bit key generation with up-to 55% smaller PUF size or up-to 72% smaller NVM size than other conventional FEs. In addition, the ternary AR-based FE is up to 55% more efficient than the binary version, and can also achieve up-to 63% higher efficiency than conventional counterparts. Furthermore, we show that the AR-based FE can be applied to PUFs with local biases (*e.g.*, biases depending on cell location in SRAM PUFs), unlike all the conventional schemes, for which only global (or identical) biases are assumed.

Keywords: Physically unclonable function (PUF) · Secure key generation · Fuzzy extractor · Debiasing

1 Introduction

Silicon physically unclonable functions (PUFs) are essential for construction of secure and trustable information systems. PUFs exploit uncontrollable process variations (*e.g.*, the electrical lengths of wires) to generate hardware-intrinsic random numbers. With these physically unclonable and tamper-evident features, PUFs are expected to provide a hardware root-of-trust for secure cryptographic key generation and entity authentication, among other applications [Mae03]. PUFs are classified into two types according to their input space size: weak and strong PUFs. A weak PUF typically accepts only one challenge or the number of challenges linear to the number of PUF cells. A strong PUF has a larger challenge space, the size of which increases exponentially with the number of cells. A typical application of weak PUFs is cryptographic secret key generation [BGS+08,MTV09,MHV12], whereas strong PUFs are primarily used for entity authentication involving a challenge-and-response protocol [SD07].

Received: 2020-04-15 Accepted: 2020-06-15 Published: 2020-08-26

Table 1: PUF sizes required for reliable 128-bit key generation with buffering entropy (Values are derived from [DGV$^+$16])

Bias p_1	0.50	0.52	0.54	0.56	0.58	0.60	0.62	0.64	0.66
BER $\mathbb{E}[e]$	0.100	0.100	0.100	0.099	0.098	0.097	0.096	0.094	0.092
PUF size	1,270	1,275	1,530	2,295	2,550	3,825	5,100	7,650	13,005

This paper focuses on cryptographic key generation from weak PUFs. Memory-based PUFs such as SRAM PUF, latch PUF, and butterfly PUF [HBF08, GKST07, SHO07] are typical examples of weak PUFs. Many weak PUFs exploit differences in drive capability of NAND gates and/or the electrical lengths of two symmetric wires as process variations. For example, the responses of SRAM PUF and latch PUF are determined by the initial state of the powered-on SRAM and the metastability of latches, respectively, differences in which are generated via the aforementioned process variations. The important requirements for PUF-based key generation are that the PUF response should be reliable and unpredictable. PUF responses are deemed reliable (or stable) if the repeatedly generated responses are always or quite similar in value. An unstable PUF response degrades the reliability of PUF-based authentication and may contaminate the security of the system relying on the generated cryptographic key. Furthermore, the PUF response is deemed unpredictable if the PUF response bits are randomly determined during instantiation. A major condition for unpredictability is that each PUF response should be uniform, *i.e.*, it should be unbiased. If a PUF response has a nonnegligible bias, it should not be used directly for cryptographic key generation because key generation based on such a PUF would be insecure owing to the entropy loss of the PUF response [KLRW14, MLSW15].

In practice, many silicon PUFs frequently contain noise and biases that are nonnegligible for cryptographic key generation [BGS$^+$08, MLSW15]. These characteristics arise from the difficulties associated with the manufacture of PUFs having completely reliable and uniform responses. When unreliable PUFs are employed for cryptographic key generation, a fuzzy extractor (FE) is commonly employed in order to remove the PUF noise and to reconstruct the enrolled values from the noisy PUFs [Boy04]. An FE generates helper data from a PUF response and a random seed encoded by error-correcting code(s) (ECC(s)) in the enrollment phase. Then, the FE reproduces the random seed from the noisy PUF response and helper data using the corresponding ECC decoding in the subsequent reconstruction phase. Notably, the helper data are stored in a common nonvolatile memory (NVM) that can be observed by attackers. Here, if the PUF response is biased, an entropy loss (or entropy leakage) occurs from the helper data [KLRW14, MLSW15, DGV$^+$16]. When typical ECCs are used in the FEs, secure 128-bit key generation with a practical PUF size would be difficult unless $0.42 < p_1 < 0.58$, where p_1 is the occurrence probability of the value "1" in the PUF response (*i.e.*, the bias). Table 1 shows the required PUF size[1] for a reliable 128-bit key generation using a repetition-BCH concatenated code with buffering entropy[2] for various PUF biases, where $\mathbb{E}[e]$ denotes the averaged bit-error rate (BER) of PUF. The values in Tab. 1 are derived from [DGV$^+$16]. We confirm that the required PUF size rapidly increases by the increase of the bias when $p_1 \geq 0.58$.

To address this entropy loss problem, debiasing schemes that realize secure cryptographic key generation even from biased PUFs have been developed in previous works [YD10, HMSS12, MLSW15, AWSO17, HO17, Š17, SUHA18, USH19]. The main motivation behind

[1] In this paper, "PUF size" indicates the bit length of the PUF response. The hardware cost for implementing PUF is basically proportional to the PUF size, as one cell of (weak) PUF typically generates one bit response.

[2] Buffering entropy indicates the usage of random seed longer than the original security parameter (*e.g.*, 128 bits) such that the residual entropy should be greater than the security parameter. Note again that buffering entropy cannot realize a secure key generation with a practical size of PUF unless $0.42 < p_1 < 0.58$.

the development of such debiasing schemes is to design more efficient and practical FEs than those employing an entropy buffer (*i.e.*, FEs without debiasing scheme). In the previous studies, the performances of FEs with debiasing schemes have been evaluated based on the PUF size (*i.e.*, bit length of the response) required for a 128-bit key generation with a reconstruction failure rates of less than 10^{-6} as in [BGS+08]. This is because the debiased bit strings should be shorter than the original PUF responses, and the debiasing schemes induce an overhead in terms of PUF size. For example, the pioneering debiasing method developed by Meas *et al.* [MLSW15], which is based on von Neumann's randomness extractor [vN51] (VNC: von Neumann corrector), requires that some bits of the PUF response are discarded to obtain a debiased bit string. This is despite the fact that the total PUF size is smaller than that of the FE without any debiasing method when $p_1 < 0.42$ or $0.58 < p_1$. As another example, Hiller and Önalan proposed a debiasing method based on coset coding (CC) which performs an ECC encoding as a one-to-many mapping in order to mitigate an entropy leakage from helper data [HO17]. In other words, the CC-based debiasing requires that some bits of the decoded random seed are discarded; this also yields increased PUF size. However, Hiller and Önalan demonstrated that the CC-based debiasing could be effectively applied to a practical SRAM PUF with $p_1 = 0.54$ [HO17]. Recently, Ueno *et al.* proposed a debiasing method named biased masking (BM), which applied a biased mask (*i.e.*, artificially generated noise) to the PUF response to render it uniform [USH19]. The BM was efficiently applied to PUFs for which $0.36 \leq p_1 \leq 0.64$. However, the PUF size required for BM rapidly increases when $p_1 \geq 0.64$ (or ≤ 0.36), and the overheads can be critical if PUFs have larger biases. To enlarge the application range of PUF-based key generation, a debiasing method that can be efficiently applied to various biased PUFs is highly desirable.

This paper presents a new debiasing scheme and shows its application in the construction of an efficient FE, named the acceptance-or-rejection (AR)-based FE, which can accommodate a relatively wide range of PUF biases. The proposed debiasing scheme employs the principle of rejection sampling to obtain a uniform distribution from biased PUF responses. The basic concept of the proposed method is rejection (*i.e.*, discard) of some bits of the majority value (*i.e.*, "0" or "1") with an appropriate probability such that the accepted bit string can be uniformly distributed. In the enrollment phase, the AR-based FE determines the locations of accepted/rejected bits and generates an additional helper data named accepted cell location (ACL) data. In the reconstruction phase, the ACL data are used to reproduce PUF responses sampled during the enrollment. The ACL data and helper data can be stored in a common NVM observable by attackers, because there is no entropy leakage from these data under the assumed bias model as proven in this paper.

The major feature of the AR-based FE is that the rejection sampling scheme can be easily and efficiently implemented in hardware/software. In addition, the AR-based FE can extract a uniform distribution from PUFs with both local and cell-wise biases (*e.g.*, biases depending on cell locations in SRAM PUFs), whereas all the conventional schemes assume only global (or identical) biases. Moreover, the AR-based FE can be further improved by incorporating ternary encoding of PUF responses [YSI+11, YSI+13]; this improvement is confirmed on the basis of the empirical noise (or cell-wise reliability) model of PUF [Mae13].

In this paper, the efficiency of the AR-based FE is comprehensively evaluated using various ECC constructions. The proposed and conventional debiasing-based FEs are evaluated in the same strategies as those of previous studies, where the efficiency of FEs is evaluated based on the PUF and NVM sizes required for a 128-bit key generation with a reconstruction failure probability of less than 10^{-6}. The results show that the AR-based FE can achieve the highest efficiency (*i.e.*, the smallest PUF and/or NVM sizes) for a wider range of PUF biases and averaged BERs than the conventional FEs considered herein, with a comparable computational cost. For example, the AR-based FE can achieve at least 55% and 51% PUF size reduction and at least 79% and 46% NVM size reduction than the conventional FEs when the biases are 0.66 and 0.90, respectively. In addition, the ternary AR-based FE has up-to 55% higher efficiency than the corresponding binary FE, and can also achieve 57–63% higher efficiency in various conditions than its conventional counterparts.

The remainder of this paper is organized as follows. In Section 2, relevant preliminary information is provided and related works are summarized. The proposed method is presented and described in detail in Section 3. The performance evaluation is reported and the considerations are discussed in Sections 4 and 5, respectively. The paper is concluded in Section 6.

2 Preliminaries and Related Work

2.1 Basic notation

The following notation conventions are adopted in this paper. Bold lower-case characters represent binary (or ternary) vectors such as $\boldsymbol{x} = (x_1, x_2, \dots, x_i, \dots, x_m)$, where m $(= |\boldsymbol{x}|)$ denotes the length of $\boldsymbol{x}$ and x_i is the i-th bit (or digit) of $\boldsymbol{x}$. A random variable representing $\boldsymbol{x}$ is denoted by an upper-case italic character, *i.e.*, X, while $\mathbb{E}[X]$ denotes the expected value of X. The probability of an event A is denoted by Pr(A), and the simultaneous probability of events A and B is denoted by Pr(A, B). Shannon entropy of A is denoted by $\mathbb{H}(A)$ and the conditional entropy of A given B is denoted by $\mathbb{H}(A|B)$.

2.2 Bias model

In this paper, we consider two bias models: global bias and local (or cell-wise) bias. Figure 1 illustrates both bias models. We assume that a key generation system consists of v PUFs with m-bit response. In the example shown in Fig. 1, $v = 5$ and $m = 10$. In the global bias model, every cell of any PUF has an identical bias of p_1 (with a corresponding p_0), where p_1 and p_0 are the occurrence probabilities of being "1" and "0", respectively. The PUF response is assumed to be independent and identically distributed (i.i.d). In contrast, in the cell-wise bias model, the i-th cell of every PUF has a unique bias $p_{1,i}$ (with a corresponding $p_{0,i}$) depending on its address, where $p_{1,i}$ and $p_{0,i}$ are the occurrence probabilities of being "1" and "0" at the i-th cell, respectively. For example, in Fig. 1, the zeroth, first, and second cells have biases of $p_{1,0} = 0.30$, $p_{1,1} = 0.20$, and $p_{1,2} = 0.70$, respectively. In the cell-wise bias model, the PUF response bits are assumed to be independent of each other. The expected value of local biases $\mathbb{E}_i[p_{1,i}]$ is considered equal to the global bias p_1.

Even if a PUF has no global bias (*i.e.*, $p_1 = 0.5$), the min-entropy of the PUF under the cell-wise bias model can be significantly reduced if the PUF has nonnegligible local biases. This is because, in the cell-wise bias model, the min-entropy of PUF is given by $-\log_2 \Pi_i \max(p_{0,i}, p_{1,i})$, which should be equal to or less than that of the global bias model, given by $-\log_2 \max(p_0, p_1)^m$. Global bias models have been widely used in the study of PUF, FE, and debiasing to date; however, some PUFs have been shown to have cell-wise biases [MLSW15]. Figure 2 shows a typical sample of PUF without global bias but having cell-wise bias. The left and right half blocks of the PUF in Fig. 2 tend to output many 1s and 0s, respectively. More precisely, $p_{1,i} = 0.80$ for $0 \le i \le 4$ and $p_{1,i} = 0.20$ otherwise. Several PUFs are known to have such area-wise biases [MLSW15]. In addition, cell-wise biases which depend on word or byte patterns frequently appear in some SRAM PUFs because of the layout of SRAM cells. Moreover, a correlation between PUF responses is sometimes found in a set of PUFs with small inter-Hamming distance, and can also be interpreted as one of such cell-wise biases. Thus, the cell-wise bias model can be considered as a generalized bias model covering many known biases, such as those discussed above.

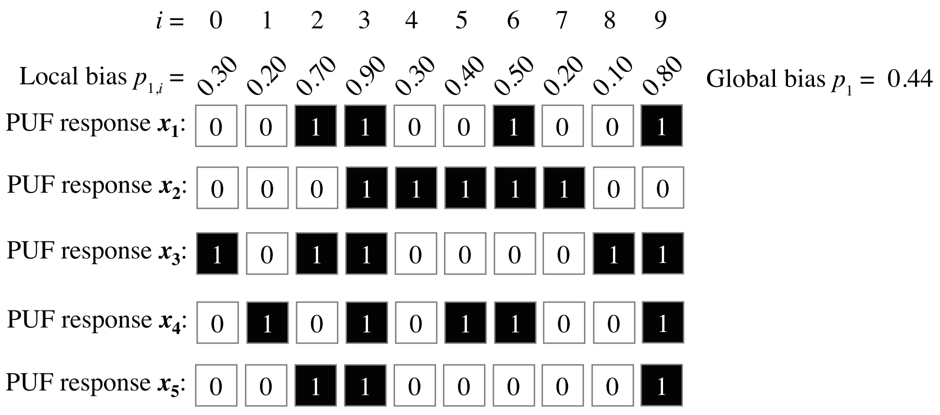

i = 0 1 2 3 4 5 6 7 8 9
Local bias $p_{1,i}$ = 0.30 0.20 0.70 0.90 0.30 0.40 0.50 0.20 0.10 0.80
Global bias p_1 = 0.44
PUF response x_1: 0 0 1 1 0 0 1 0 0 1
PUF response x_2: 0 0 0 1 1 1 1 1 0 0
PUF response x_3: 1 0 1 1 0 0 0 0 1 1
PUF response x_4: 0 1 0 1 0 1 1 0 0 1
PUF response x_5: 0 0 1 1 0 0 0 0 0 1

Figure 1: Sample global and cell-wise bias models.

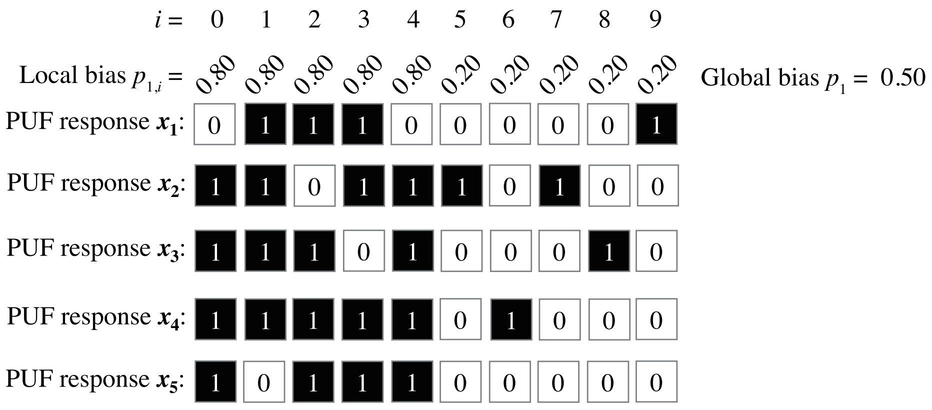

i = 0 1 2 3 4 5 6 7 8 9
Local bias $p_{1,i}$ = 0.80 0.80 0.80 0.80 0.80 0.20 0.20 0.20 0.20 0.20
Global bias p_1 = 0.50
PUF response x_1: 0 1 1 1 0 0 0 0 0 1
PUF response x_2: 1 1 0 1 1 1 0 1 0 0
PUF response x_3: 1 1 1 0 1 0 0 0 1 0
PUF response x_4: 1 1 1 1 1 0 1 0 0 0
PUF response x_5: 1 0 1 1 1 0 0 0 0 0

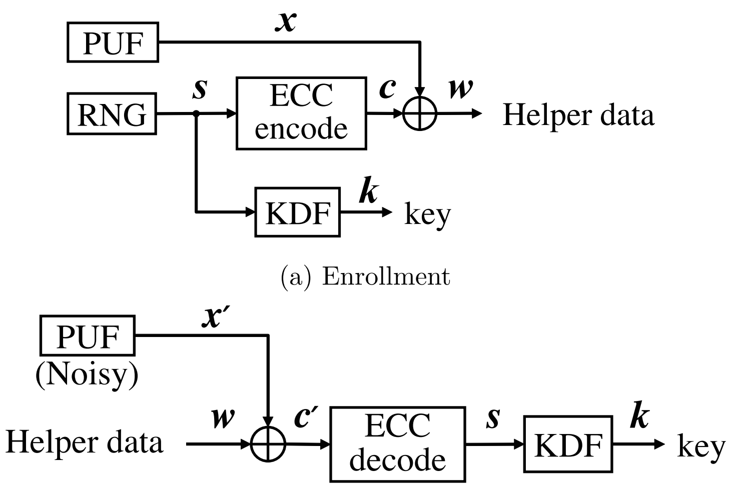
Figure 2: Example of PUF response with typical cell-wise bias, where there are two areas with different biases.

Note here that the cell-wise bias model clearly differs from a reliability model where each cell has its own BER, which is called an empirical noise model or cell-wise reliability model [Mae13, YSI+11, YSI+13]. In detail, for a cell-wise bias model, the i-th cell has a bias depending on its address if we have many PUFs; for an empirical noise model, each cell in a PUF has its own BER and each PUF may have its own distribution of BERs independent of the cell-wise bias. It has been also shown that many PUFs follow the empirical noise model.

2.3 Fuzzy extractor (FE)

An FE is used for reliable reconstruction of the enrolled value from a noisy PUF response. In this paper, we focus on FEs with a code-offset that employ ECC for error correction. Such FEs are widely deployed in many modern PUF-based key generation systems [BGS+08, DGSV15, DGV+16].

Figure 3 shows the block diagrams of an FE with code-offset at (a) enrollment and (b) reconstruction phases, where $\boldsymbol{x}$ denotes a PUF response; $\boldsymbol{s}$ denotes a random (secret) seed; $\boldsymbol{k}$ denotes a cryptographic key; $\boldsymbol{c}$ denotes a codeword given by ECC-encoding $\boldsymbol{s}$; $\boldsymbol{w}$ denotes a helper data; and $\boldsymbol{x}'$ and $\boldsymbol{c}'$ is a PUF response and codeword affected by noise, respectively. Here, $\oplus$ indicates the bit-parallel-XOR operation. In the enrollment phase,



(a) Enrollment

(b) Reconstruction

Figure 3: Block diagram of FE.

we first generate a random seed $\boldsymbol{s}$ using a random number generator (RNG) and then enroll a secret key $\boldsymbol{k}$ using a key derivation function (KDF), which is usually realized via a cryptographic hash function such as Keccak and spongent [MHV12] or a universal hash such as a Toeplitz hash [BGS+08, MTV09]. In the enrollment phase, a helper data $\boldsymbol{w}$ is also computed by the bit-parallel-XOR of the PUF response $\boldsymbol{x}$ and ECC codeword $\boldsymbol{c}$, which is the ECC-encoded result of $\boldsymbol{s}$. Here, $\boldsymbol{w}$ should be stored in an NVM on the client device with the PUF to enable appropriate reconstruction. In the reconstruction, we first compute the received codeword $\boldsymbol{c}'$ by the bit-parallel-XOR of the helper data $\boldsymbol{w}$ and the noisy PUF response $\boldsymbol{x}'$. As $\boldsymbol{c}'$ is considered as the codeword affected by noise included in the PUF response, we reproduce the random seed $\boldsymbol{s}$ (and $\boldsymbol{k}$) through ECC decoding. Whereas the enrollment is not usually performed through real-time processing, the reconstruction should be performed on a client device with a PUF. The computational resources for the reconstruction are frequently much severe, although the computation costs at the enrollment including measurement of PUF response, random seed generation, ECC encoding, and key derivation are also sometimes non-trivial.

The helper data are regarded as a public data available to the attacker because they are stored in a common NVM space. Therefore, we should consider the conditional entropy $\mathbb{H}(X|W)$ (or $\mathbb{H}(S|W)$) of X (or S) for a given W for a cryptographic key generation.[3] For example, a 128-bit PUF-based key generation should satisfy $\mathbb{H}(X|W) \geq 128$. It has been proven that $\mathbb{H}(X|W) = |\boldsymbol{s}|$ if X is uniformly distributed [MLSW15]. In this case, the 128-bit key generation can be easily realized using a 128-bit random seed. However, $\mathbb{H}(X|W)$ ($\mathbb{H}(S|W)$) significantly decreases owing to the bias of PUF [KLRW14, MLSW15]. This is called the entropy loss (entropy leakage). Intuitively, as shown by Hiller and Önalan, the channel from C to W can be considered as a binary symmetric channel with an error probability of p_1 [HO17], where X is regarded as an additive noise. The channel model indicates that the attacker can estimate $\boldsymbol{c}$ (and $\boldsymbol{s}$) by the ECC decoding of $\boldsymbol{w}$ if the noise level is insufficiently high (*i.e.*, the PUF response is highly biased). In [MLSW15], Maes *et al.* showed that it is difficult to realize a 128-bit key generation with a practical size of the PUF if $p_1 > 0.58$ (or $p_1 < 0.42$). For such biased PUFs, the usage of a debiasing scheme is a promising solution to realize an efficient key generation. Interested readers are

[3]Maes *et al.* proved that $\mathbb{H}(X|W) = \mathbb{H}(S|W)$; and therefore, they are essentially equivalent [MLSW15].

Table 2: Debiased bit extraction by VNC-based debiasing

Enrollment			Reconstruction		
Input	Output		Input		Output
$x_{2i}x_{2i+1}$	u_i	d_i	$x'_{2i}x'_{2i+1}$	d_i	u'_i
00	Discarded	0	0-	1	0
01	0	1	1-	1	1
10	1	1	--	0	Discarded
11	Discarded	0			

referred to [KLRW14, MLSW15, DGV+16, HO17] for more further details on the entropy loss/leakage.

2.4 Debiasing

A debiasing scheme is employed to derive a uniform distribution from a PUF-induced biased distribution. Use of a debiasing scheme enables the design of secure key generation systems with relatively small-sized PUFs even if PUFs have nonnegligible bias. Such debiasing methods have various underlying concepts, *e.g.*, use of a randomness extractor and addition of entropy by inducing randomness at the enrollment. In the following, we briefly describe previous works in which debiasing schemes for PUFs were proposed. We here describe six FEs: VNC-, CC-, maskless debiasing (MD)-, BM-, and trivial debiasing (TD)-based FEs, and FE with complementary index-based syndrome (C-IBS) [YD10, HMSS12, MLSW15, AWSO17, HO17, Š17, SUHA17, SUHA18, USH19]. Note that the global bias model is assumed in all the conventional schemes.

VNC-based FEs: Maes *et al.* [MLSW15] proposed debiasing methods based on the VNC [vN51], which was used to extract a uniform bit string from an bit string with a bias of p_1. FEs with VNC-based debiasing are called VNC-based FEs in this paper. In the VNC-based debiasing, the bit string is divided into pairs of two consecutive bits. If the bits in a pair differ from each other (*i.e.*, having values of “01” and “10”), the first bit is retained as a response bit; otherwise (*i.e.*, the values are “00” and “11”), the pair is discarded. Table 2 summarizes the extraction of debiased bit string with VNC-based debiasing at (a) enrollment and (b) reconstruction, and Fig. 4 shows an example of VNC-based debiasing. The mapping shown in Tab. 2 is applied to a PUF response, and the resulting debiased bit string is used for helper data generation and reconstruction instead of the original PUF response. In the enrollment of VNC-based FE, the VNC is applied to the (biased) PUF response to obtain a debiased bit string $\boldsymbol{u}$. Furthermore, an additional helper data named debiasing data $\boldsymbol{d}$, which indicates the locations of the retained/discarded pairs, is generated. In the reconstruction, $\boldsymbol{u}$ is reproduced from the noisy PUF response, helper data, and debiasing data. In [MLSW15], Maes *et al.* proposed three FE constructions based on VNC-based debiasing, which differed in terms of efficiency and reusability.[4] Subsequently, Delvaux *et al.* showed that a nonnegligible cost in PUF size is required to render a debiasing-based FE reusable [DGV+16]. Suzuki *et al.* [SUHA17, SUHA18] later improved the efficiency of VNC-based FEs by means of ternary encoding of the PUF response [YSI+11, YSI+13], exploiting the cell-wise reliability model [Mae13].

CC-based FE: Hiller and Önalan proposed a debiasing method that mitigated entropy loss on the basis of coset coding [HO17]. In this paper, we refer to the FEs based on coset coding as CC-based FEs. A CC-based FE primarily aims to induce entropy at the ECC

[4]An FE is said to be reusable if the FE can enroll an identical PUF using different seeds many times with negligible secrecy leakage compared to that of a single enrollment [Boy04, MLSW15].

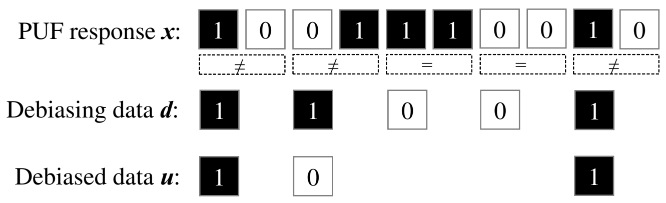


Figure 4: Example of VNC-based debiasing.

encoding with one-to-many mapping in a random manner. Furthermore, a CC-based FE is easily implemented as a coset coding, where some bits of a random seed are discarded after ECC decoding during the reconstruction. No additional resource-consuming operation is required for its enrollment. The CC-based FE mitigates the secrecy leakage from helper data; however, it does not necessarily reduce this leakage sufficiently if the bias is large [USH19]. Note that discarding some bits of a random seed also results in a larger PUF size. In the previous work, the efficiency of another CC-based FE with differential sequence coding (DSC) [HYS16] was demonstrated through its application to an SRAM PUF with $p_1 = 0.54$ and an averaged BER of 0.10. However, a method of efficiently applying CC-based FEs to a wider variety of PUFs has not yet been clarified.

MD-based FE: An FE based on maskless debiasing (MD) was proposed by Aysu *et al.* [AWSO17]. The MD-based FEs use many-to-one mapping to derive a debiased bit string from a biased PUF response, where the many-to-one mapping is designed depending on the PUF bias such that the output is unbiased. For example, if the PUF bias is given by $p_1 = 0.70$, a bit string with a bias of 0.49 can be obtained using a two-to-one mapping where the values of "00," "01," and "10" are mapped to "0" and "11" is mapped to "1." However, it is difficult to design such mappings with a sufficiently small input bit length except for specific biases. In addition, the averaged BER of the mapping output is greater than that of the original PUF response; this is also considered as an overhead yielding increased PUF size.

Trivial debiasing (TD): A trivial debiasing (TD) scheme was presented by Škorić [Š17], which follows the condense-then-fuzzy-extract scheme with an FE (or secure sketch) based on the syndrome construction. In condense-then-fuzzy-extract construction, a randomness condenser is first applied to the source (*i.e.*, the PUF response), which is followed by application of a secure sketch [CFP⁺16]. Let m and l be the length of PUF response and the actually required length for the FE, respectively. Let $\mathcal{M}_m$ be a set of indices of PUF response bit, that is, $\mathcal{M}_m = [0, 1, \ldots, i, \ldots, m-1]$. Let Ω be a parametric integer. The TD scheme adds an entropy of $\log_2 \Omega$.

In the TD scheme, a subset of $\mathcal{M}_m$ with a size of l, denoted by $\mathcal{U}_l$, is first randomly drawn such that all indices of PUF response bit having the rarest-occurring value ("0" or "1") are included in $\mathcal{U}_l$ (*e.g.*, $\mathcal{U}_l$ should contain all indices included in $\{i \mid x_i = 0\}$ if $p_1 \geq 0.5$). One of helper data $\boldsymbol{w}$ is generated as the syndrome of the PUF response bits with indices included in $\mathcal{U}_l$. Then, $\mathrm{DRSG}_l(\theta, \chi)$ is taken as a deterministic generator of a random subset of $\mathcal{M}_m$ with a size of l from a random seed θ and counter χ (hereafter, deterministic random subset generator is abbreviated as "DRSG"). A random permutation $\pi : \mathcal{M}_m \to \mathcal{M}_m$ is determined using a random value $\omega \in [0, \Omega - 1]$, such that $\mathrm{DRSG}_l(\theta, \omega) = \{\pi(i) \mid i \in \mathcal{U}_l\}$. Finally, Helper data is obtained as the triple of $\boldsymbol{w}$, θ, and π, and $\mathrm{DRSG}_l(\theta, \chi)$ is published.

In the reconstruction, set $\mathcal{R}$ consisting of all indices with the rarest-occurring value included in noisy PUF response $\boldsymbol{x}'$ (*i.e.*, indices included in $\{i \mid x_i = 0\}$ if $p_1 \geq 0.5$) is first extracted. Then, subset $\mathcal{L}^\chi = \mathrm{DRSG}_l(\theta, \chi)$ is computed for all $\chi \in [0, \Omega - 1]$. Next, ω is

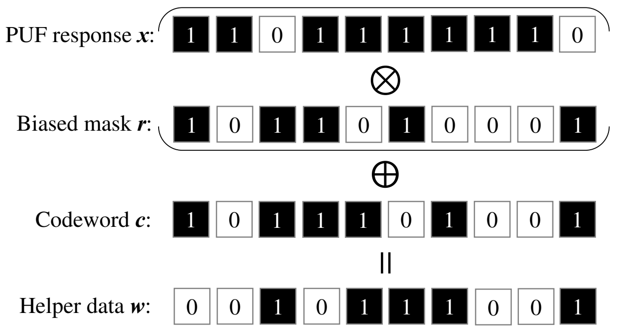

PUF response x: 1 1 0 1 1 1 1 1 1 0
⊗
Biased mask r: 1 0 1 1 0 1 0 0 0 1
⊕
Codeword c: 1 0 1 1 1 0 1 0 0 1
=
Helper data w: 0 0 1 0 1 1 1 0 0 1

Figure 5: Example of BM-based debiasing.

estimated as $\mathcal{L}^\omega$ is the closest to $\mathcal{R}$, and $\mathcal{U}_l$ is reconstructed using $\mathcal{L}^\omega$ and π^{-1}. Finally, $\boldsymbol{x}$ is reconstructed using $\boldsymbol{w}$, $\boldsymbol{x}'$, and $\mathcal{U}_l$, with the syndrome decoding.

Randomness condensers are known to be more efficient than randomness extractors in terms of entropy loss. Škorić gave the retained entropy of the TD from the m-bit p_1-biased PUF response ($p_1 > 0.5$) as $2m(1-p_1)-2$, which is larger than that of the VNC ($mp_1(1-p_1)$) [Š17]. Moreover, the TD scheme can induce an additional entropy of $\log_2 \Omega$ using fake permutations (*i.e.*, $\mathrm{DRSG}_l(\theta, \chi)$). However, there is no quantitative evaluation on the resulting PUF size and BER of bit string extracted through TD. In addition, no concrete construction for building blocks nor the process of determining parameters to design and evaluate actual key generation systems have been shown. Note that, Škorić applied TD to the secure sketch based on syndrome construction [Š17] (initially presented as a quantum oblivious transfer protocol in [BBCS91]); however, its application to a secure sketch/FE based on a code-offset was not shown.

BM-based FE: Ueno *et al.* [USH19] proposed an FE based on the debiasing method called biased masking (BM). Figure 5 shows an example of BM-based debiasing with $p_1 \geq 0.5$. The basic concept of BM is addition of an artificial noise to a biased PUF response such that the resulting bit string is uniform. Here, the addition of artificial noise is realized as bit-parallel-AND or -OR with a PUF response. The artificial noise (*i.e.*, a mask value) is randomly generated with a proper bias. The added noise makes it difficult for attackers to obtain secret information, but a valid PUF user can remove the noise by performing ECC decoding. The BM overhead is given as the increase in the averaged BER at the ECC decoding during the reconstruction. Ueno *et al.* showed that the overhead could be tolerated by simply increasing the length of repetition code [USH19]. Their evaluation revealed that the BM-based FE can efficiently realize a 128-bit key generation from PUFs with a bias of $0.36 \leq p_1 \leq 0.64$, whereas the overhead of BM increases significantly for $p_1 < 0.36$ and $0.64 < p_1$. Although enrollment of a BM-based FE requires an RNG and a bit-parallel-AND or -OR operation, the corresponding reconstruction does not require additional computation beyond ECC decoding.

FE with C-IBS: In [YD10], Yu and Devadas proposed an FE with index-based syndrome (IBS), which differs from FEs with code-offsets in that the helper data is generated using a unique encoding method and not a bit-parallel-XOR operation. Hiller *et al.* generalized the IBS as a complementary IBS (C-IBS) for improved efficiency [HMSS12]. FEs with IBS and C-IBS exploit the bit-wise reliability of the PUF response given from the empirical noise model [Mae13] so that only several reliable cells are used. The basic concept of IBS and C-IBS involves the generation of helper data as the location of reliable cells, rather than use of a bit-parallel-XOR operation involving the codeword $\boldsymbol{c}$ and PUF response $\boldsymbol{x}$.

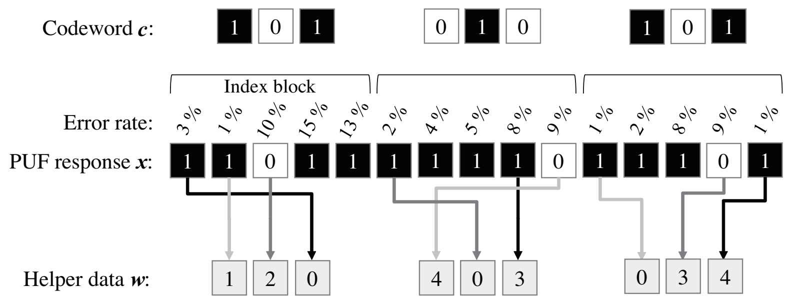


Figure 6: Example of enrollment with C-IBS.

Figure 6 is an overview of the enrollment with C-IBS. C-IBS utilizes an ECC consisting of two codewords of "10101...1" and "01010...0," which is called C-IBS code, instead of repetition code. In Fig. 6, the length of C-IBS code (denoted by ν_1) is 3. For the C-IBS, the BERs of each cell are first derived. Then, $\boldsymbol{x}$ is first divided into blocks consisting of λ cells named C-IBS (or index) block. Here, λ should be determined such that $\lambda > \nu_1$. Figure 6 shows the case of $\lambda = 5$ and $\nu_1 = 3$. If the first bit of the z-th block of $\boldsymbol{c}$ is "1" and "0," the helper data corresponding to the bit are generated as the address of cells most reliably outputting "1" and "0" in the z-th block, respectively. The subsequent bits are enrolled in the same manner, although the cells that have been already assigned are never used again. In the case of Fig. 6, the zeroth bit in the leftmost block of $\boldsymbol{w}$ is given as 1 because the zeroth bit in the leftmost block of $\boldsymbol{c}$ is "1" and the first cell most reliably outputs "1" in the first block of $\boldsymbol{x}$. Then, the first bit of $\boldsymbol{w}$ is given as 2 because the first bit of $\boldsymbol{c}$ is "0" and the second cell most reliably outputs "0." Moreover, the second bit of $\boldsymbol{w}$ is 0 because the second bit of $\boldsymbol{c}$ is "1" and the zeroth cell outputs "1" with the second highest reliability after the first cell, which has already been assigned. At the reconstruction, C-IBS reads the cells as indicated by helper data in order to obtain $\boldsymbol{c}'$, and then performs the ECC decode.

Although the IBS was not originally presented as a debiasing scheme, IBS and C-IBS are now considered to be useful for key generation even from biased PUFs [DGV⁺16]. Although IBS and C-IBS were designed to utilize reliable cells, (C-)IBS requires index blocks to contain both cells reliably outputting both "0" and "1" for a reconstruction. Otherwise, even if a codeword contains "1" or 1s, there is a possibility that no or deficient reliable cell(s) will output "1" in the corresponding index block, and *vice versa*. This scenario causes inconsistencies between the enrollment and reconstruction and degrades the reliability of reconstruction. For example, in Fig. 6, the third bit of the second-block $\boldsymbol{c}$ is wrongly enrolled because there is no more cell with the value of "0." Therefore, to avoid such inconsistencies, if the PUF bias is large, the bit length of each index block should be longer; this yields an increased PUF size.

3 Proposed Method

3.1 Basic concept

This section presents the proposed AR-based FE, which is based on a new debiasing scheme that employs the principle of rejection sampling (*i.e.*, the acceptance-or-rejection method), named the AR-based FE. We first describe the AR-based FE with the global bias model in Sections 3.1–3.4, and then extend it to the cell-wise bias model in Section 3.5. Subsequently, in Section 3.6, we present another extension of the proposed FE on the basis of the empirical noise model for enhanced efficiency.

Rejection sampling is a method of deriving an arbitrary distribution from another easily available distribution. In the case of discrete distribution, we consider a probabilistic distribution as a probability mass function (pmf). Let $p_{\mathrm{tar}}(x)$ be the target distribution expressed as a pmf, and $p_{\mathrm{prop}}(x)$ be an attainable proposal distribution. In rejection sampling, we first determine a constant h satisfying

$$p_{\mathrm{tar}}(x) \leq h p_{\mathrm{prop}}(x), \tag{1}$$

for all x. We then obtain the distribution represented by $p_{\mathrm{tar}}(x)$ as follows: (i) sample a is derived from $p_{\mathrm{prop}}(x)$; (ii) a is accepted with a probability of $p_{\mathrm{tar}}(a)/hp_{\mathrm{prop}}(a)$ or else rejected. One of the simplest approaches to implement Step (ii) is using another sample b from the uniform distribution of $[0, hp_{\mathrm{prop}}(a)]$ and thresholding b with $p_{\mathrm{tar}}(a)$.

The proposed AR-based FE performs rejection sampling to derive a uniform distribution from the biased distribution given as the PUF response. However, the rejection sampling scheme is not a reproducible operation because an acceptance/rejection of sample is randomly determined and no information about acceptance/rejection is retained. To reproduce the debiased bit string in the reconstruction phase, the proposed AR-based FE utilizes additional helper data, named accepted cell location (ACL) data, which indicates the locations of accepted/rejected cells; this is similar to VNC-based FEs. Usage of ACL data enables the reproduction of a uniformly distributed bit string obtained through rejection sampling in the enrollment phase. In comparison with straight-forward FEs, no computational overhead is required during the reconstruction, whereas additional RNG and bit-parallel logic operations (*e.g.*, OR operations) are required in the enrollment similarly to the process of BM. As the enrollment essentially does not require any real-time processing and the client device with a PUF is sometimes constrained in terms of resources, the proposed FE is applicable to many PUF-based key generation systems.

3.2 FE construction

Figure 7 shows block diagrams of the AR-based FE during (a) enrollment and (b) reconstruction. The primary characteristics of the AR-based FE are that new operations named reproducible rejection sampling (RRS) and accepted cell extraction (ACE) operations are applied during the enrollment and reconstruction, respectively. The RRS operation performs rejection sampling on a PUF response and generates ACL data. The ACL data is stored into an NVM as well as helper data because there is no entropy leakage from the pair of ACL and helper data as proven in Section 3.3. The ACE operation regenerates the debiased PUF response according to ACL data. Usage of ACL data makes it possible to reproduce the debiased bit string by the rejection sampling, whereas a straight-forward rejection sampling is not reproducible.

We first show the process for extracting a uniform distribution from a biased PUF response through the RRS and ACE operations via rejection sampling. As we should derive a debiased bit string from a biased PUF response in the global bias model, the input and output of the rejection sampling (*i.e.*, the proposal and target distributions) are given as a Bernoulli distribution with a bias of p_1 and a uniform distribution, respectively. As we assume that the PUF response is independently distributed, we perform the rejection sampling in a bit-wise manner. Thus, Eq. (1) can be instantiated for the RRS operation as the following set of inequalities:

$$\begin{cases} 0.5 & \leq & hp_0, \\ 0.5 & \leq & hp_1. \end{cases} \tag{2}$$

Here, if we minimize the parameter h as $h = (2\min(p_0, p_1))^{-1}$, we can minimize the probability of rejection at sampling. In this optimal case, when $p_1 \geq p_0$, the rejection probabilities of "0" and "1" are given by $1 - 0.5/hp_0 = 0$ and $1 - 0.5/hp_1 = 1 - p_0/p_1$,

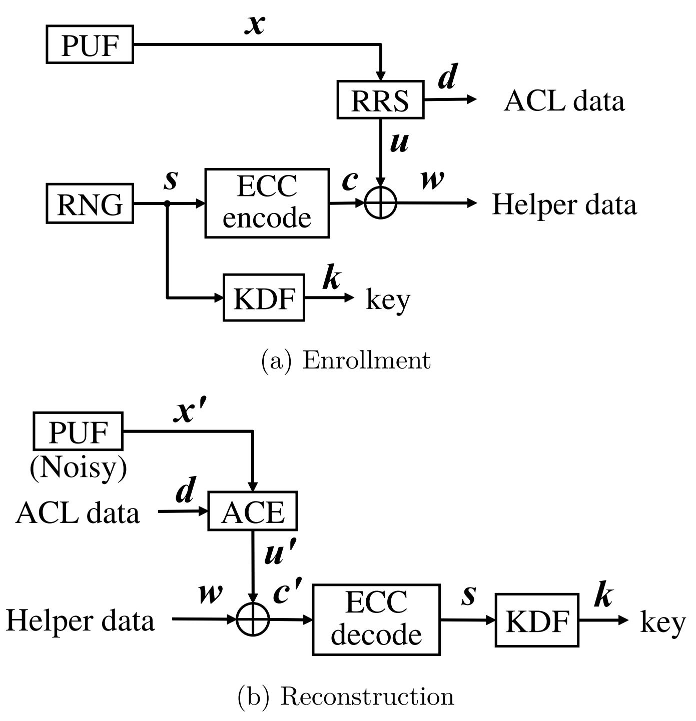


Figure 7: Block diagram of AR-based FE.

respectively. When $p_1 < p_0$, they are given by $1 - p_1/p_0$ and 0, respectively. Thus, we extract a debiased bit string with the minimized rejection probabilities.

We next describe the implementation of RRS and ACE operations, which requires no real-time processing in the reconstruction phase and induces little computational overhead compared to the conventional FEs. Figure 8 shows an example of RRS operation during enrollment. The RRS operation performs the above bit-wise rejection sampling to obtain a debiased bit string and generates the corresponding ACL data. Let m be the length of PUF response. To realize the bit-wise rejection sampling with fewer computational resources, the RRS operation consists of the following three steps: (a) generation of an m-bit random number $\boldsymbol{r}$ with a bias of q_1, which is given by

$$q_1 = \frac{\min(p_0, p_1)}{\max(p_0, p_1)}; \tag{3}$$

(b) computation of ACL data $\boldsymbol{d}$ by the bit-parallel-OR of $\bar{\boldsymbol{x}}$ and $\boldsymbol{r}$ when $p_1 \geq 0.5$ or the bit-parallel-OR of $\boldsymbol{x}$ and $\boldsymbol{r}$ when $p_0 > 0.5$, where $\bar{\boldsymbol{x}}$ $(= \boldsymbol{x} \oplus 1^m)$ denotes the bit string in which each bit of $\boldsymbol{x}$ is inverted; and (c) derivation of a debiased bit string $\boldsymbol{u}$ according to $\boldsymbol{d}$. In Step (c), we assign (*i.e.*, accept) x_i to u_i if $d_i = 1$, whereas we discard (*i.e.*, reject) x_i if $d_i = 0$, where x_i, u_i, and d_i denote the i-th bit of $\boldsymbol{x}$, $\boldsymbol{u}$, and $\boldsymbol{d}$, respectively.[5] In contrast, the ACE operation during reconstruction is a deterministic algorithm and can be realized using Step (c) only, in accordance with the $\boldsymbol{d}$ from the RRS operation, which reproduces the debiased bit string derived by that operation.

Algorithms 1 and 2 are the algorithmic descriptions of RRS and ACE operations, respectively. Here, m and l denote the PUF size and the required length of debiased bit

[5]Here, when $d_i = 0$, u_i is considered to have the value of "discarded" in order to simplify the notation. However, the actual length of $\boldsymbol{u}$ is l, which is shorter than m.

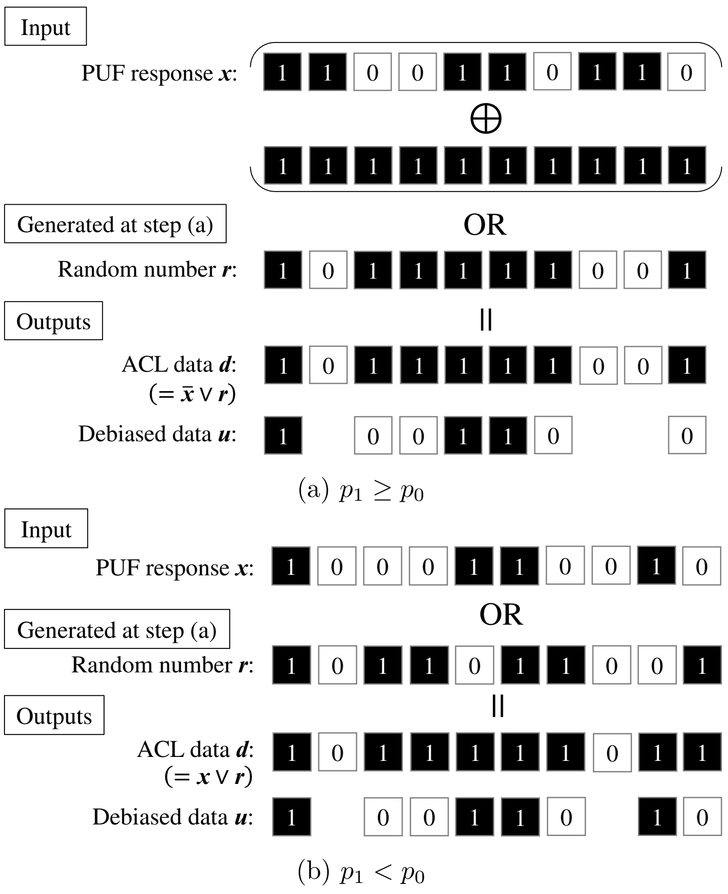


Figure 8: Example of RRS operation.

string, respectively. Note that the required length (*i.e.*, l), which equals to the helper data size, should be pre-determined in accordance with the coding rate of used ECC and required key length (The ECC is designed according to the averaged BER of PUF). In RRS operation, Lines 9–15 perform the bit-wise rejection sampling according to $\boldsymbol{d}$, which is generated from $\bar{\boldsymbol{x}}$ or $\boldsymbol{x}$ and $\boldsymbol{r}$ for the reproducibility. RRS operation returns $\perp$ as an enrollment failure at Line 20 if the RRS operation cannot extract sufficient debiased bits, because such a PUF may lead to the secrecy leakage. To perform an enrollment with a negligible failure probability, we should determine the PUF size m appropriately. We describe how to determine m for given l and p_1, p_0 such that the enrollment failure probability is smaller than a threshold in Section 3.4. On the other hand, ACE operation corresponds to the Lines 9–16 of RRS operation, which reproduces the debiased bit string $\boldsymbol{u}$ as $\boldsymbol{u}'$ from $\boldsymbol{x}'$ and $\boldsymbol{d}$.

3.3 Security of AR-based FE

In this subsection, we analyze the security of the proposed AR-based FE. We assume that an attacker can observe (but cannot modify) the helper data and ACL data stored in a common NVM, and that she exactly knows the biases of PUF (*i.e.*, p_1 in the global bias model and $p_{1,i}$ for all i in the cell-wise bias model). In other words, the attacker's goal is to estimate the PUF response and secret seed from the data stored in the NVM and the information given by the biases. This assumption corresponds to the scenario where the

Algorithm 1 RRS operation

```
Input: PUF response $\boldsymbol{x}$
Output: Pair of debiased bit string and ACL data $(\boldsymbol{u}, \boldsymbol{d})$ or $\perp$
 1: parameter $m, l, p_0, p_1$;
 2: function RRSOPERATION($\boldsymbol{x}$)
 3:     $q_1 \leftarrow \min(p_0, p_1)/\max(p_0, p_1)$;
 4:     $\boldsymbol{r} \leftarrow \mathrm{RNG}_{m,q_1}$;    ▷ Step(a): $\mathrm{RNG}_{m,q_1}$ generates $m$-bit $q_1$-biased random number
 5:     if $p_1 \geq p_0$ then
 6:         $\boldsymbol{x} \leftarrow \boldsymbol{x} \oplus 1^m$;    ▷ Compute $\bar{\boldsymbol{x}}$
 7:     end if
 8:     $\boldsymbol{d} \leftarrow \boldsymbol{x} \vee \boldsymbol{r}$;    ▷ Step (b): Generation of ACL date $\boldsymbol{d}$ from $\bar{\boldsymbol{x}}$ or $\boldsymbol{x}$ and $\boldsymbol{r}$
 9:     for $i$ from 0 to $m-1$ do    ▷ Step (c): Generation of debiased bit string by rejection sampling
10:         if $d_i = 1$ then
11:             $u_i = x_i$;
12:         else if $d_i = 0$ then
13:             reject $x_i$;
14:         end if
15:     end for
16:     $\boldsymbol{u} \leftarrow \mathrm{Align}(u_0, u_1, \ldots, u_{m-1})$;    ▷ Derive $\boldsymbol{u}$ as a complete bit string without discarded positions
17:     if $|\boldsymbol{u}| \geq l$ then
18:         return $(\boldsymbol{u}, \boldsymbol{d})$;
19:     else if $|\boldsymbol{u}| < l$ then
20:         return $\perp$;    ▷ Return $\perp$ if enrollment fails due to insufficient length of $\boldsymbol{u}$
21:     end if
22: end function
```

Algorithm 2 ACE operation

```
Input: Pair of noisy PUF response and ACL data $(\boldsymbol{x}', \boldsymbol{d})$
Output: Reproduced debiased bit string $\boldsymbol{u}'$
 1: parameter $m$;
 2: function ACEOPERATION($\boldsymbol{x}', \boldsymbol{d}$)
 3:     for $i$ from 0 to $m-1$ do
 4:         if $d_i = 1$ then
 5:             $u'_i = x'_i$;
 6:         else if $d_i = 0$ then
 7:             reject $x'_i$;
 8:         end if
 9:     end for
10:     $\boldsymbol{u}' \leftarrow \mathrm{Align}(u'_0, u'_1, \ldots, u'_{m-1})$;    ▷ Derive $\boldsymbol{u}'$ as a complete bit string without discarded positions
11:     return $\boldsymbol{u}'$;
12: end function
```

attacker performs a reverse-engineering on the NVM and has knowledge of some properties of the PUF. Under this assumption, the FEs are equivalent to a wiretap channel with the attacker acting as an eavesdropper [Wyn75, HO17]; therefore, the security can be validated and formally proven from the information-theoretic perspectives. The possibility of a data manipulation attack is discussed in Section 5.3.

We first derive the biases of the RRS operation outputs (*i.e.*, $\boldsymbol{u}$ and $\boldsymbol{d}$), to demonstrate that the RRS operation derives a bit string with no bias. When $p_1 \geq p_0$ (corresponding to Fig. 8(a)), the bias of $\boldsymbol{r}$ (*i.e.*, q_1) is equal to p_0/p_1. Let q'_1 and q'_0 be the biases of the ACL data $\boldsymbol{d}$ (*i.e.*, the probability of being "1" and "0" in $\boldsymbol{d}$, respectively). As $\boldsymbol{d}$ is derived by the bit-parallel-OR of $\bar{\boldsymbol{x}}$ and $\boldsymbol{r}$, q'_1 is given by

$$\begin{aligned} q'_1 &= p_0 + q_1 - p_0 q_1 \\ &= p_0 + p_1 q_1 \\ &= 2p_0. \end{aligned} \tag{4}$$

Then, let p'_1 and p'_0 be the biases of $\boldsymbol{u}$. As we accept x_i only if $d_i = 1$ only, p'_1 and p'_0 are given as the conditional probabilities of $x_i = 1$ and $x_i = 0$ under $d_i = 1$, respectively.

Thus, p'_1 and p'_0 are given by

$$\begin{aligned} p'_1 &= \Pr(x_i = 1 | d_i = 1) \\ &= \frac{\Pr(x_i = 1, d_i = 1)}{\Pr(d_i = 1)} \\ &= \frac{p_1 q_1}{q'_1} = \frac{1}{2}, \qquad (5) \\ p'_0 &= \Pr(x_i = 0 | d_i = 1) \\ &= \frac{\Pr(x_i = 0, d_i = 1)}{\Pr(d_i = 1)} \\ &= \frac{p_0}{q'_1} = \frac{1}{2}, \qquad (6) \end{aligned}$$

respectively. Here, $\Pr(x_i = 1, d_i = 1)$ (*i.e.*, the simultaneous probability of $x_i = 1$ and $d_i = 1$) equals to $p_1 q_1$, because it should occur when $x_i = 1$ and $r_i = 1$, with x_i and r_i being independent of each other. Note that it also follows the assumption that the PUF response is independently distributed. In addition, $\Pr(x_i = 0, d_i = 1) = \Pr(x_i = 0) = p_1$, because d_i is given by the OR of $\bar{x}_i = 1$ and r_i. For $p_1 < p_0$, we can obtain the biases of $\boldsymbol{d}$ and $\boldsymbol{u}$ in the same manner as the above. As q_1 equals to p_0/p_1 in this case, the bias of $\boldsymbol{d}$ (*i.e.*, q'_1) is given as

$$\begin{aligned} q'_1 &= p_1 + q_1 - p_1 q_1 \\ &= p_1 + p_0 q_1 \\ &= 2p_1. \qquad (7) \end{aligned}$$

The biases of $\boldsymbol{u}$ (*i.e.*, p'_1 and p'_0) are given as

$$p'_1 = \frac{p_1 q_1}{q'_1} = \frac{1}{2}, \qquad (8)$$

$$p'_0 = \frac{p_0}{q'_1} = \frac{1}{2}. \qquad (9)$$

Equations (5), (6), (8), and (9) imply that we can obtain a uniformly distributed bit string by RRS operation. Since ACE operation reproduces the debased bit string, the output of ACE operation is also unbiased.

The outputs of RRS and ACE operations should be a uniformly distributed bit string. The AR-based FE employs an ACL data that can be observed by the attacker, as is the case for the helper data. However, the attacker cannot obtain any information other than the locations of accepted PUF cells used in the key generation. Therefore, similarly to the VNC-based FEs [MLSW15], the security of the AR-based FE can be proven as Theorem 1.

Theorem 1. *Let $\mathbb{I}(S; (W, D))$ be the mutual information between random seed S and the pair of helper data W and ACL data D. The AR-based FE has no entropy leakage, that is, $\mathbb{I}(S; (W, D)) = 0$.*

Proof. $\mathbb{I}(S; (W, D))$ can be rewritten as

$$\begin{aligned} \mathbb{I}(S; (W, D)) &= \mathbb{I}(S; D) + \mathbb{I}(S; W | D) \\ &= \mathbb{I}(S; W | D) \\ &= \mathbb{H}(W | D) - \mathbb{H}(W | D, S), \qquad (10) \end{aligned}$$

because S and D are independent of each other (*i.e.*, $\mathbb{I}(S; D) = 0$). Let $\boldsymbol{c}$ and C be the codeword and the corresponding random variable, respectively. Let l be the length of $\boldsymbol{c}$.

Here, l equals to the length of the helper data (*i.e.*, $l = |\boldsymbol{c}| = |\boldsymbol{w}|$). Hence, $\mathbb{H}(W|D)$ should be bounded as

$$\mathbb{H}(W|D) \leq l. \tag{11}$$

Let $\boldsymbol{u}$ and U be the output of RRS operation and the corresponding random variable, respectively. We should consider only the case that the ACE operation successfully extracts a debiased bit sting of a certain length $|\boldsymbol{u}|$ which is equal to or longer than l (*i.e.*, $|\boldsymbol{u}| \geq l$). This is because the RRS operation returns $\perp$ (*i.e.*, enrollment failure) if $l < |\boldsymbol{u}|$ as described in Alg. 1, and therefore such a PUF is never used for the enrollment. Accordingly, let us consider helper data W as $W = C \oplus U_{0:l-1} = S\boldsymbol{G} \oplus U_{0:l-1}$, where $U_{0:l-1}$ represents a partial bit string from the zeroth to $(l-1)$-th bits of U and $\boldsymbol{G}$ is a generator matrix of ECC. Then, $\mathbb{H}(W|D, S)$ is given by

$$\begin{aligned}\mathbb{H}(W|D, S) &= \mathbb{H}(S\boldsymbol{G} \oplus U_{0:l-1}|D, S)\\ &= \mathbb{H}(U_{0:l-1}|D)\\ &= \mathbb{H}(U_{0:l-1})\\ &= l.\end{aligned} \tag{12}$$

This is because $U_{0:l-1}$ should be a uniform distribution as shown in Eqs. (5), (6), (8), and (9). In accordance with Eqs. (10)–(12), the entropy leakage is given by

$$\begin{aligned}\mathbb{I}(S; (W, D)) &= \mathbb{H}(W|D) - \mathbb{H}(W|D, S)\\ &\leq l - l\\ &= 0.\end{aligned} \tag{13}$$

Note that $\mathbb{I}(S; (W, D))$ should be non-negative; therefore, $\mathbb{I}(S; (W, D)) = 0$. □

Theorem 1 also indicates that $\mathbb{H}(S|W, D) = |\boldsymbol{s}|$ because $\mathbb{H}(S|W, D) = \mathbb{H}(S) - \mathbb{I}(S; (W, D))$. This means that we can realize a σ-bit key generation (*i.e.*, $\mathbb{H}(S|W, D) = \sigma$) using a σ-bit random seed.

3.4 How to determine PUF size for AR-based FE

This subsection describes how to determine PUF size m ($= |\boldsymbol{x}|$) in the AR-based FE with a given PUF bias, ECC, and actually required debiased bit length l ($= |\boldsymbol{c}|$) such that the enrollment failure probability is negligible. Note that $l = |\boldsymbol{c}|$ should be determined in advance according to the length of random seed $|\boldsymbol{s}|$ (equal to the security parameter σ here) and the coding rate of used ECC, which is designed according to the averaged BER of PUF [DGSV15].

As described in Section 3.3, the AR-based FE can realize a σ-bit key generation if a debiased bit string equal to or longer than l can be obtained by the RRS operation. As the AR-based FE employs randomness to determine acceptance/rejection of PUF cells, the resulting debiased bit string length (*i.e.*, $|\boldsymbol{u}|$) is given in a stochastic manner. Therefore, we should determine the PUF size m such that the probability $\Pr(|\boldsymbol{u}| < l)$ should be smaller than a threshold because we cannot enroll a PUF in the proposed FE when $|\boldsymbol{u}| < l$. Let $P^{(\mathrm{en})}_{\mathrm{thre}}$ be the threshold probability of failure in the enrollment of the AR-based FE. We should determine the PUF size (*i.e.*, m) that satisfies $P^{(\mathrm{en})}_{\mathrm{thre}} \geq \Pr(|\boldsymbol{u}| < l)$, given a PUF bias and a required debiased length.

As the rejection/acceptance of each cell should be determined independently according to p_1, the resulting distribution should be binomial. Therefore, the length of the accepted bit string can be represented by the cumulative mass function (cmf) of the binomial distribution with p_1 similarly to [MLSW15]. Let $\mathrm{cmf}_{\mathrm{bino}}(l; m, p)$ be the cmf of a binomial distribution having l-times successes with probability p from m-times challenges. If we need an l-bit debiased bit string for key generation, the expected probability of enrollment failure $\mathbb{E}[P_{\text{fail}}^{(\text{en})}]$ for the AR-based FE with an m-bit PUF having a bias of p_1 can be expressed as

$$\mathbb{E}[P_{\text{fail}}^{(\text{en})}] = \text{cmf}_{\text{bino}}\big(l; m, \mathbb{E}[p_{\text{accept}}]\big), \tag{14}$$

where $\mathbb{E}[p_{\text{accept}}]$ denotes the expected probability of acceptance. Thus, the PUF size should satisfy

$$P_{\text{thre}}^{(\text{en})} \geq \mathbb{E}[P_{\text{fail}}^{(\text{en})}] = \text{cmf}_{\text{bino}}\big(l; m, \mathbb{E}[p_{\text{accept}}]\big), \tag{15}$$

which is followed by

$$l \leq \text{cmf}_{\text{bino}}^{-1}\big(P_{\text{thre}}^{(\text{en})}; m, \mathbb{E}[p_{\text{accept}}]\big), \tag{16}$$

where $\text{cmf}_{\text{bino}}^{-1}$ denotes the inverse function of cmf_{bino} in terms of l. In the case of the AR-based FE with the global bias model, $\mathbb{E}[p_{\text{accept}}]$ is given by $\mathbb{E}[p_{\text{accept}}] = \mathbb{E}[\Pr(d_i = 1)] = \min(2p_0, 2p_1)$ as it should be equal to the expected number of 1's in $\boldsymbol{d}$ (*i.e.*, q_1' in Eq. (4) or (7)); and therefore, the PUF size m should be determined according to

$$l \leq \text{cmf}_{\text{bino}}^{-1}\big(P_{\text{thre}}^{(\text{en})}; m, \min(2p_0, 2p_1)\big). \tag{17}$$

Thus, a σ-bit PUF-based key generation system with the AR-based FE can be designed according to the following three steps: (a) identification of the averaged BER and the distribution of PUF response X (*i.e.*, PUF biases p_1 and p_0); (b) design of ECC(s) such that the expected reconstruction failure probability is less than a threshold $P_{\text{thre}}^{(\text{rec})}$, which determines the required length of debiased bit string (*i.e.*, $l = \sigma/\rho$, where ρ denotes the coding rate of the ECC); and (c) determination of the PUF size m according to Eq. (17).

3.5 Extension to cell-wise bias model

In addition to the global bias model discussed in Sections 3.1–3.4, we extend the AR-based FE to the cell-wise bias model. As shown in Fig. 2, Section 2.2, even if the PUF has no global bias, the min-entropy given from the cell-wise bias can be significantly worse. Some conventional debiasing methods may tolerate certain specific cell-wise biases. For example, VNC-based FEs may be applicable to the PUF with the area-wise bias shown in Fig. 2. However, these methods cannot be applied to all types of cell-wise biases. In fact, we can easily create examples for which the above debiasing scheme cannot extract a uniform distribution from a cell-wise-biased PUF response: a PUF response with $p_{1,i} = 0.80$ if i is odd and $p_{1,i} = 0.20$ otherwise is one of such examples. Thus, no debiasing method that is generally applicable to any type of cell-wise biases has been developed to date.

The basic concept of the extension proposed herein is adaptive determination of the rejection probability according to the bias of each cell (*i.e.*, $p_{1,i}$ for the i-th cell), as the AR-based FE employs bit-wise rejection sampling. For example, for the case shown in Fig. 1, the bit-wise rejection sampling is applied to the zeroth bit with the proposed distribution of $p_{1,0} = 0.30$, to the first bit with $p_{1,1} = 0.20$, to the second bit with $p_{1,2} = 0.70$, and so on. More simply and intuitively, we apply the RRS operation with $p_1 = 0.80$ and $p_1 = 0.20$ to the left and right halves of the PUF response in Fig. 2, respectively.

Formally, RRS operation for the extended AR-based FE is implemented by modifying the generation of $\boldsymbol{r}$ in Steps (a) and the computation of $\boldsymbol{d}$ in Step (b) (Section 3.2). In Step (a), we generate an m-bit number $\boldsymbol{r}$ such that the i-th bit of $\boldsymbol{r}$ (*i.e.*, r_i) has a bias of $q_{1,i}$ given by

$$q_{1,i} = \frac{\min(p_{0,i}, p_{1,i})}{\max(p_{0,i}, p_{1,i})}. \tag{18}$$

In Step (b), we compute the ACL data $\boldsymbol{d}$ as follows: A bit string $\boldsymbol{f}$ is generated, the i-th bit of which is "1" if $p_{1,i} \geq 0.5$; otherwise, it is 0. Then, the ACL data is computed using the relation $d_i = (x_i \oplus f_i) \vee r_i$. This indicates that, for the i-th cell, the RRS operation shown in Fig. 8(a) is applied by inverting x_i if $p_{1,i} \geq 0.5$; otherwise, that of Fig. 8(b) is applied. Step (c) of the RRS operation and ACE operation in the extended AR-based FE is implemented in the same manner as for the original case.

The bias of u_i (*i.e.*, the i-th bit of the debiased bit string $\boldsymbol{u}$) is derived similarly to that of the global bias model. Let $q'_{1,i}$ ($q'_{0,i}$) and $p'_{1,i}$ ($p'_{0,i}$) be the probabilities of "1" ("0" for d_i and u_i (*i.e.*, the i-th bits of $\boldsymbol{d}$ and $\boldsymbol{u}$), respectively. When $p_{1,i} \geq p_{0,i}$, the bias of d_i (*i.e.*, $q'_{1,i}$) is given by

$$\begin{aligned} q'_{1,i} &= p_{0,i} + q_{1,i} - p_{0,i}q_{1,i} \\ &= p_{0,i} + p_{1,i}q_{1,i} \\ &= 2p_{0,i}. \end{aligned} \tag{19}$$

As we accept x_i only if $d_i = 1$, $p'_{1,i}$ and $p'_{0,i}$ are given by the conditional probabilities of $x_i = 1$ and $x_i = 0$ under $d_i = 1$ as follows:

$$\begin{aligned} p'_{1,i} &= \Pr(x_i = 1 | d_i = 1) \\ &= \frac{\Pr(x_i = 1, d_i = 1)}{\Pr(d_i = 1)} \\ &= \frac{p_{1,i}q_{1,i}}{q'_{1,i}} = \frac{1}{2}, \end{aligned} \tag{20}$$

$$\begin{aligned} p'_{0,i} &= \Pr(x_i = 0 | d_i = 1) \\ &= \frac{\Pr(x_i = 0, d_i = 1)}{\Pr(d_i = 1)} \\ &= \frac{p_{0,i}}{q'_{1,i}} = \frac{1}{2}, \end{aligned} \tag{21}$$

respectively. These formulas indicate that, even if the PUF has a nonnegligible cell-wise bias, the probabilities of "1" and "0" values for the accepted u_i are given by $1/2$ for all i; therefore, $\boldsymbol{u}$ should have no bias and be uniformly distributed. Note that the biases are again given by $p'_{1,i} = p'_{0,i} = 1/2$ as well in the case that $p_{1,i} \leq p_{0,i}$.

Algorithm 3 is the algorithmic description of RRS operation extended to the cell-wise bias model. Note that ACE operation is the same as the previous one in Alg. 2. In contrast to the RRS operation based on global bias model, the extended RRS operation performs the operations in a bit-wise manner in addition to rejection sampling. This indicates that the extended RRS operation essentially has the computational cost same as the original one, regardless of the fact that we require random numbers with various biases and a vectorized operation is unavailable for the above bit-wise operations.

The security of the extended AR-based FE can be proven in the same manner as Theorem 1 because we assume only that $\boldsymbol{u}$ is uniformly distributed in the proof. Accordingly, the extended AR-based FE can realize the σ-bit key generation using a σ-bit random seed, as in the original case.

The PUF size required for successful enrollment is determined based on the cmf of the binomial (*i.e.*, Eq. (16)), similarly to the above. As the acceptance probability of the i-th bit is given by $q'_{1,i} = \min(2p_{0,i}, 2p_{1,i})$, the expected acceptance probability $\mathbb{E}[p_{\text{accept}}]$ is equal to $\mathbb{E}_i[q'_{1,i}] = \mathbb{E}_i[\min(2p_{0,i}, 2p_{1,i})]$. Therefore, in order to guarantee that the expected failure probability of an enrollment is smaller than $P^{(\text{en})}_{\text{thre}}$, we should determine the PUF size m such that

$$l \leq \text{cmf}^{-1}_{\text{bino}}\left(P^{(\text{en})}_{\text{thre}}; m, \mathbb{E}_i[\min(2p_{0,i}, 2p_{1,i})]\right). \tag{22}$$

To demonstrate the advantage of debiasing scheme based on the cell-bias model, let us consider a simple example of PUFs in Fig. 2, where $p_{1,i} = 0.8$ for $0 \leq i \leq n/2 - 1$ and

Algorithm 3 RRS operation based on cell-wise bias model

```
Input: PUF response $\boldsymbol{x}$
Output: Pair of debiased bit string and ACL data $(\boldsymbol{u}, \boldsymbol{d})$ or $\perp$
 1: parameter $m, l, p_{0,i}, p_{1,i}$;
 2: function ExtendedRRSOperation($\boldsymbol{x}$)
 3:     for $i$ from 0 to $m-1$ do
 4:         $q_{1,i} \leftarrow \min(p_{0,i}, p_{1,i}) / \max(p_{0,i}, p_{1,i})$;
 5:         $r_i \leftarrow \mathrm{RNG}_{1,q_{1,i}}$;    ▷ Step(a): $\mathrm{RNG}_{1,q_{1,i}}$ generates one-bit $q_{1,i}$-biased random number
 6:         if $p_{1,i} \geq p_{0,i}$ then
 7:             $x_i \leftarrow x_i \oplus 1$;    ▷ Compute $\bar{x}_i$
 8:         end if
 9:         $d_i \leftarrow x_i \vee r_i$;    ▷ Step (b): Generation of ACL date $d_i$ from $\bar{x}_i$ or $x_i$ and $r_i$
10:         if $d_i = 1$ then    ▷ Step (c): Generation of debiased bit string by rejection sampling
11:             $u_i = x_i$;
12:         else if $d_i = 0$ then
13:             reject $x_i$;
14:         end if
15:     end for
16:     $\boldsymbol{u} \leftarrow \mathrm{Align}(u_0, u_1, \ldots, u_{m-1})$;    ▷ Derive $\boldsymbol{u}$ as a complete bit string without discarded positions
17:     if $|\boldsymbol{u}| \geq l$ then
18:         return $(\boldsymbol{u}, \boldsymbol{d})$;
19:     else if $|\boldsymbol{u}| < l$ then
20:         return $\perp$;    ▷ Return $\perp$ if enrollment fails due to insufficient length of $\boldsymbol{u}$
21:     end if
22: end function
```

$p_{1,i} = 0.2$ for $n/2 \leq i \leq n-1$ with a global bias $p_1 = \mathbb{E}_i[p_{1,i}] = 0.5$. In this case, the conventional debiasing schemes like CC, MD, TD, and BM substantially use the response for the enrollment as it is due to their algorithms, which causes the entropy leakage by the cell-wise bias.[6] In contrast, the proposed scheme soundly derives a debiased bit string which realizes a secure key generation without any entropy leakage. On the other hand, the proposed scheme based on cell-wise bias model discards some bits by bit-wise rejection sampling. Also, more extremely, let us consider another example of PUFs where $p_{1,i} = 1$ for $0 \leq i \leq m/2 - 1$ and $p_{1,i} = 0$ for $m/2 \leq i \leq m-1$. In this example, all PUFs have an identical response of $11 \ldots 100 \ldots 0$, and therefore cannot be used for key generation. Nevertheless, since the global bias is given by $p_1 = 0.5$, the conventional schemes consider them to be unbiased. On the other hand, the proposed scheme rejects all the cells in such PUFs because the acceptance probability of each cell is given by $\min(2p_{0,i}, 2p_{1,i}) = 0$, and returns $\perp$ (*i.e.*, enrollment failure), which shows the soundness of proposed scheme. Generally, if the PUF contains more cells with $p_{1,i}$ equal/close to 1 or 0, the expected debiased bit length $\mathbb{E}_i[\min(2mp_{0,i}, 2mp_{1,i})]$ becomes shorter due to the increase of rejection. However, the accepted cells have no bias independently of the distribution of cell-wise biases.

3.6 Extension to ternary-encoded PUF response

This subsection presents an extension of the AR-based FE to a ternary-encoded PUF response for higher efficiency. Ternary encoding of PUF response was first presented by Yamamoto *et al.* to enhance the entropy and reliability of PUF response using the cell-wise error rate [YSI+11, YSI+13]. In the following, we first briefly describe the ternary encoding of the PUF response, and then describe the extension. We employ the cell-wise bias model, as the global bias model is an instance of the cell-wise bias model.

For a PUF instance, the error rate of response is not the same for all the cells. In other words, each cell has its own error rate,[7] and its distribution is approximated by a

[6] BM may be extended to the cell-wise bias model based on the same strategy as the extension presented in this subsection, as BM is implemented with a mask value generation and bit-parallel operations.

[7] Note that it is sufficient to evaluate the averaged BER for designing ECC to be used in a binary

Gaussian distributions [Mae13]. Therefore, if we repeatedly observe the response of a PUF many times, we can classify the PUF cells into three types: (a) constantly outputting "0," (b) constantly outputting "1," and (c) randomly outputting "0" or "1." Cells of types (a) and (b) are called constant cells, whereas cells of type (c) are called unreliable (or random) cells. It has been shown that most common PUFs contain around 10%–30% random cells [YSI+11, YSI+13, YSI+15, SUHA18]. Unreliable cells are regarded as noise for conventional binary encoding; however, for ternary encoding, unreliable cells are considered to have a stable third value as "random." Ternary encoding is useful for more reliable extraction of PUF response with higher entropy than binary encoding for some types of weak PUFs (*e.g.*, for latch PUFs). As a ternary-encoded PUF response cannot be directly combined with the conventional binary FEs, Kazumori *et al.* presented a ternary FE construction applicable to ternary encoded PUFs, which employs a ternary random seed, a ternary ECC, a ternary helper data, and $GF(3)$ arithmetic for the code-offset [KUH19]. Whereas it sometimes takes nonnegligible cost to derive cell-wise reliability to utilize the ternary PUF response (as well as IBS/C-IBS), usage of ternary response would improve the efficiency of a key generation systems significantly in terms of hardware cost.

Figure 9 shows block diagrams of the ternary AR-based FE, where the new ternary RRS and ACE operations are combined with the ternary FE reported in [KUH19]. The input of ternary RRS operation is a ternary PUF response, and its outputs are a debiased trit (ternary digit) string and a binary ACL data. In the ternary RRS operation, a debiased trit string is derived from a biased 3-categorical distribution[8] via rejection sampling. However, we do not obtain a completely uniform 3-categorical distribution from the ternary PUF response because many PUFs are likely to have far fewer random cells than constant cells. This indicates that we should reject many constant cells to derive a uniform distribution. Instead, in the ternary AR-based FE, we perform binary rejection sampling for constant cells only and accept all random cells. The output trit string of the ternary RRS operation should contain constant cells of "0" and "1" values with the same probability. Thus, we mitigate the entropy loss from the ternary helper data more efficiently than rejection sampling that derives completely uniform distribution. The ternary AR-based FE is designed with an entropy buffer which renders the residual min-entropy greater than σ. The residual min-entropy for ternary FEs is evaluated using the algorithm in [KUH19] (extended algorithm of [DGV+16] to ternary FE).[9]

Let $t_{0,i}$, $t_{1,i}$, and $t_{\mathrm{r},i}$ denote the occurrence probabilities of "0," "1," and "random" values for the i-th cell, respectively ($t_{0,i} + t_{1,i} + t_{\mathrm{r},i} = 1$). Note that, in the global bias model, all cells have an identical bias t_1, which indicates that $t_{1,i} = t_1$ for any i (this is the same for $t_{0,i}$ and $t_{\mathrm{r},i}$). In the ternary RRS operation, along with the binary RRS operation, we first generate an m-bit random number $\boldsymbol{r}$, where the i-th bit r_i has a bias given by

$$q_{1,i} = \frac{\min(t_{0,i}, t_{1,i})}{\max(t_{0,i}, t_{1,i})}. \tag{23}$$

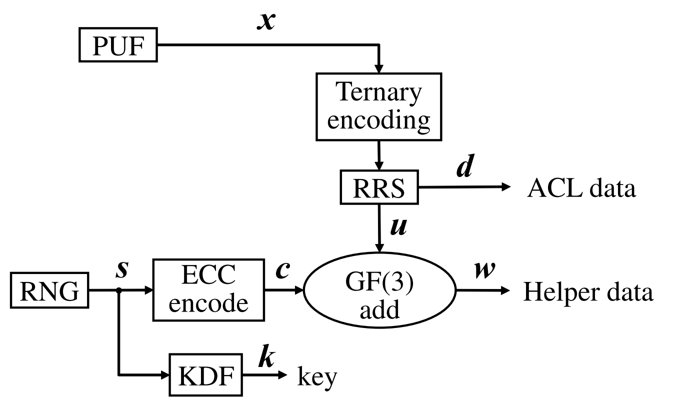
Then, when $t_{1,i} \geq t_{0,i}$, we generate the i-th bit of ACL data $\boldsymbol{d}$ as follows:

$$d_i = \begin{cases} 1 & \text{if } x_i = \text{random}, \\ \bar{x_i} \vee r_i & \text{otherwise}. \end{cases} \tag{24}$$

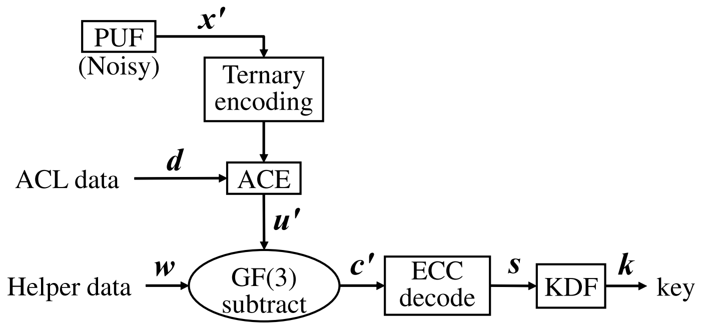
FE [DGSV15].

[8]A θ-categorical distribution is also called a generalized Bernoulli distribution, which is represented by a random variable that can take one of the θ elements. A 2-categorical distribution is a Bernoulli distribution.

[9]Note that the residual min-entropy can be used for the worst-case security evaluation in terms of guess of random variables, although we use Shannon entropy (*i.e.*, average-case security) to explain the proposed method in this paper.



(a) Enrollment



(b) Reconstruction

Figure 9: Block diagram of ternary AR-based FE.

Here, the bias of d_i (denoted by $q'_{1,i}$) is given by

$$\begin{aligned} q'_{1,i} &= \Pr((\bar{x}_i \vee r_i) = 1 | x_i \neq \text{random}) + \Pr(x_i = \text{random}) \\ &= t_{0,i} + q_{1,i}(1 - t_{\mathrm{r},i}) - t_{0,i}q_{1,i} + t_{\mathrm{r},i} \\ &= 2t_{0,i} + t_{\mathrm{r},i}. \end{aligned} \tag{25}$$

When $t_{1,i} < t_{0,i}$, d_i is given in the same manner except that x_i is not inverted when $x_i \neq$ random, and $q'_{1,i}$ is given by alternating $t_{0,i}$ and $t_{1,i}$ in Eq. (25) (*i.e.*, $q'_{1,i} = 2t_{1,i} + t_{\mathrm{r},i}$ for $t_{1,i} < t_{0,i}$). Note that the ACL data $\boldsymbol{d}$ of the ternary RRS operation is still binary because the ACL data represents the acceptance/rejection of the corresponding cells. Finally, we extract the debiased trit string from the cells with $d_i = 1$.

Algorithm 4 is the algorithmic description of ternary RRS operation based on the cell-wise bias model. This algorithm is derived by replacing the generations of $\bar{x}_i$ and d in Alg. 3 with the corresponding ternary operation. The ternary ACE operation is performed using Alg. 2, which extracts (*i.e.*, accept) the cells with $d_i = 1$.

Let us consider the biases of the ternary RRS operation outputs. Let $t'_{0,i}$, $t'_{1,i}$, and $t_{\mathrm{r},i}$ be the biases (*i.e.*, the occurrence probabilities of "0," "1," and "random" values) for the i-th trit in the debiased string $\boldsymbol{u}$, respectively. Similar to the binary RRS operation, the biases are represented as the conditional probabilities of x_i given d_i. Hence, when

Algorithm 4 Ternary RRS operation based on cell-wise bias model

```
Input: Ternary PUF response $\boldsymbol{x}$
Output: Pair of debiased bit string and ACL data $(\boldsymbol{u}, \boldsymbol{d})$ or $\perp$
 1: parameter $m, l, t_{0,i}, t_{1,i}, t_{\mathrm{r},i}$;
 2: function ExtendedRRSOperation($\boldsymbol{x}$)
 3:     for $i$ from 0 to $m-1$ do
 4:         $q_{1,i} \leftarrow \min(t_{0,i}, t_{1,i}) / \max(t_{0,i}, t_{1,i})$;
 5:         $r_i \leftarrow \mathrm{RNG}_{1,q_{1,i}}$;   ▷ Step(a): $\mathrm{RNG}_{1,q_{1,i}}$ generates one-bit $q_{1,i}$-biased random number
 6:         if $t_{1,i} \geq t_{0,i}$ then
 7:             if $x_i \neq$ random then
 8:                 $x_i \leftarrow x_i \oplus 1$;   ▷ Compute $\bar{x}_i$
 9:             end if
10:         end if
11:         if $x_i =$ random then   ▷ Step (b): Generation of ACL date $d_i$ based on Eq. (24)
12:             $d_i = 1$
13:         else
14:             $d_i \leftarrow x_i \vee r_i$;
15:         end if
16:         if $d_i = 1$ then   ▷ Step (c): Generation of debiased trit string by rejection sampling
17:             $u_i = x_i$;
18:         else if $d_i = 0$ then
19:             reject $x_i$;
20:         end if
21:     end for
22:     $\boldsymbol{u} \leftarrow \mathrm{Align}(u_0, u_1, \ldots, u_{m-1})$;   ▷ Derive $\boldsymbol{u}$ as a complete trit string without discarded positions
23:     if $|\boldsymbol{u}| \geq l$ then
24:         return $(\boldsymbol{u}, \boldsymbol{d})$;
25:     else if $|\boldsymbol{u}| < l$ then
26:         return $\perp$;   ▷ Return $\perp$ if enrollment fails due to insufficient length of $\boldsymbol{u}$
27:     end if
28: end function
```

$t_{1,i} \geq t_{0,i}$, the resulting biases $t'_{0,i}$, $t'_{1,i}$, and $t_{\mathrm{r},i}$ are given by

$$\begin{aligned} t'_{1,i} &= \Pr(x_i = 1 | d_i = 1) \\ &= \frac{\Pr(x_i = 1, d_i = 1)}{\Pr(d_i = 1)} \\ &= \frac{t_{1,i} q_{1,i}}{q'_{1,i}} = \frac{t_{0,i}}{2t_{0,i} + t_{\mathrm{r},i}}, \end{aligned} \tag{26}$$

$$\begin{aligned} t'_{0,i} &= \Pr(x_i = 0 | d_i = 1) \\ &= \frac{\Pr(x_i = 0, d_i = 1)}{\Pr(d_i = 1)} \\ &= \frac{t_{0,i}}{q'_{1,i}} = \frac{t_{0,i}}{2t_{0,i} + t_{\mathrm{r},i}}, \end{aligned} \tag{27}$$

$$\begin{aligned} t'_{\mathrm{r}} &= \Pr(x_i = \mathrm{random} | d_i = 1) \\ &= \frac{\Pr(x_i = \mathrm{random}, d_i = 1)}{\Pr(d_i = 1)} \\ &= \frac{t_{\mathrm{r},i}}{q'_{1,i}} = \frac{t_{\mathrm{r},i}}{2t_{0,i} + t_{\mathrm{r},i}}, \end{aligned} \tag{28}$$

respectively. The biases when $t'_{1,i} < t'_{0,i}$ are similarly given as $t_{1,i} = t_{0,i} = t_{1,i}/(2t_{1,i} + t_{\mathrm{r},i})$. Equations (26)–(28) indicate that there is no bias between "0" and "1," although the occurrence probability of the "random" value is less than those of "0" and "1."

The security of the ternary AR-based FE can be explained in the same manner as for the binary cases, because the ACL data d also represents acceptance/rejection of the corresponding cell as well. However, as the ternary string from the ternary RRS operation is not completely uniform, $\mathbb{H}(W|D,S)$ should not be equal to $\mathbb{H}(S)$ in contrast to the binary case in Eq. (12).[10] Thus, the residual entropy $\mathbb{H}(S|W,D)$ (and $\mathbb{H}(X|W,D)$) should be less than $\mathbb{H}(S)$, which yields an entropy leakage through helper data. In other words, we cannot realize a ς-trit key generation (*i.e,* a σ-bit key generation where $\sigma = \varsigma \log_2 3$) using a ς-trit random seed. Therefore, to realize a σ-bit key generation using the ternary AR-based FE, we should determine the PUF size using an entropy buffer (*i.e.*, using a random seed sufficiently longer than σ) such that $\mathbb{H}(S|W,D) \geq \sigma$ (or $\mathbb{H}(X|W,D) \geq \sigma$). More concretely, in the design of the ternary AR-based FE, we first derive the bias of ternary string from the RRS operation (*i.e.*, $t'_{0,i}$, $t'_{1,i}$, and $t'_{\mathrm{r},i}$), and then evaluate the residual entropy using the algorithm presented in [KUH19] in order to compute the required seed length $|\boldsymbol{s}|$ and the corresponding $|\boldsymbol{c}|$ $(= l)$ for σ-bit key generation. Note here that, although the algorithm in [KUH19] cannot be directly used for evaluating $\mathbb{H}(X|W,D)$, it is sufficient to evaluate $\mathbb{H}(U_{0:l-1}|W)$ because $\mathbb{H}(U_{0:l-1}|W) \leq \mathbb{H}(X|W,D)$.

After determining l required to satisfy $\mathbb{H}(U_{0:l-1}|W) \geq \sigma$, we determine the PUF size using Eq. (16) such that the enrollment failure probability should be less than a threshold $P^{(\mathrm{en})}_{\mathrm{thre}}$. As $\mathbb{E}[p_{\mathrm{accept}}]$ is given by $\mathbb{E}_i[\min(2t_{0,i}, 2t_{1,i}) + t_{\mathrm{r},i}]$, we should determine the PUF size m using the following bound:

$$l \leq \mathrm{cmf}^{-1}_{\mathrm{bino}}\big(P^{(\mathrm{en})}_{\mathrm{thre}}; m, \mathbb{E}_i[\min(2t_{0,i}, 2t_{1,i}) + t_{\mathrm{r},i}]\big). \tag{29}$$

In the global bias model, $\mathbb{E}[p_{\mathrm{accept}}]$ should be substituted with $\min(2t_0, 2t_1) + t_{\mathrm{r}}$. Thus, if the PUF contains many random cells, the ternary AR-based FE can achieve higher efficiency, and hence, the effectiveness of ternary encoding is exploited.

4 Performance Evaluation

4.1 Binary FEs

4.1.1 Implementation cost of FEs

This subsection reports an analytical comparison of the implementation costs of the proposed AR-based FE with those of conventional binary FEs under global bias model, as summarized in Tab. 3. In the table, “Retained entropy” denotes the expected debiased bit length extracted from an m-bit p_1-biased PUF response (for $p_1 \geq 0.5$ for TD and this work); “NVM size” denotes the NVM size required for storing helper data and other additional data for the debiasing scheme; “Reusability” indicates whether or not the FE can perform enrollment with different seeds using an identical PUF without nonnegligible entropy leakage; and “Candidates of computational bottleneck” denotes operations that can act as bottlenecks. Note that all FEs have ECC encoding and decoding (denoted by ECC enc. and ECC dec., respectively) as one of candidates at enrollment and reconstruction phases, respectively. We assume that any previous helper data including additional data for debiasing is not available for the reuse of a PUF. In Tab. 3, we consider three VNC-based FEs: the classic von Neumann (CVN), the 2-pass tuple-output von Neumann (2P-TO-VN), and the pair-output von Neumann with erasures (ϵ-2O-VN). CVN is the simplest VNC-based FE. 2P-TO-VN is an efficient variant of VNC-based FE from the perspective of PUF size, which handles the debiased bit as an ECC codeword comprising “01” and “10” and applies the VNC repeatedly to bit pairs discarded in the previous pass (*i.e.*, “00” and “11”). In addition, ϵ-2O-VN is another variant that offers the reusability by replacing the discarded cell with an erasure symbol (or a placeholder). For the NVM size of MD, $\mathcal{O}(2^{\mu})$ denotes the order of 2^{μ} and Mapping eval. indicates the evaluation of many-to-one mapping.

[10] In the ternary FE, $\mathbb{H}(S) = l \log_2 3$ because $\boldsymbol{s}$ is a trit random vector.

Table 3: Comparison of implementation costs for binary FEs

	Retained entropy	NVM size*	Reusability	Candidates of computational bottleneck	
				Enrollment	Reconstruction
No debias	N/A	$l\ (= m)$	Yes	ECC encode	ECC decode
CVN [MLSW15]	mp_0p_1	$l + \lceil m/2 \rceil$	No	ECC encode	ECC decode
2P-TO-VN [MLSW15]	mp_0p_1 $+mE_{p_0,p_1}$ †	$2m$	No	ECC encode	ECC decode
ϵ-2O-VN [MLSW15]	mp_0p_1	$5m/2$	Yes	ECC encode	ECC decode
MD [AWSO17]	m/μ	$l + \mathcal{O}(2^\mu)$	Yes	Mapping eval. and ECC encode	Mapping eval. and ECC decode
BM [USH19]	N/A	$l\ (= m)$	No	RNG and ECC encode	ECC decode
CC [HO17]	N/A	$l\ (= m)$	Yes	ECC encode	ECC decode
TD [Š17]	$2mp_0 - 2$	$l\lceil \log_2 m \rceil$ $+l$	No	DRSG$_l$ and ECC encode	List construction, set manipulation, DRSG$_l$, and ECC decode
This work	$2mp_0$	$l + m$	No	RNG and ECC encode	ECC decode

* NVM size indicates the total memory size for storing helper data including addition data for debiasing scheme (*e.g.*, ACL data). Here, m and l differ depending on the efficiency of retained entropy and BER during ECC decoding. The smaller retained entropy and higher BER yield the greater m and l.

† $E_{p_0,p_1} = \dfrac{p_0^2 p_1^2}{2(p_0^2 + p_1^2)}$.

In Tab. 3, the retained entropy (*i.e.*, the expected debiased bit length) roughly indicates the efficiency as a debiasing scheme. The expected debiased bit length of the AR-based FE is given by $\mathbb{E}[\min(2mp_0, 2mp_1)]$ because it should be equal to the expected number of "1" values in $\boldsymbol{d}$ (*i.e.*, q_1'). The VNC-based FEs extract the entropy of mp_0p_1 $(+mE_{p_0,p_1})$ from an m-bit PUF response, because the VNC extracts one bit (tuple) per two-bit response with a probability of $2p_0p_1$. Here, the retained entropy of ϵ-2O-VN would equal to CVN as the number of cells in ϵ-2O-VN which contribute to entropy is the same as CVN [DGV+16]. The MD-based FE extracts one bit from a μ-bit block, where μ denotes the length of many-to-one mapping and exceeds 3 in the most cases, according to [AWSO17]. The retained entropy cannot be evaluated for BM and CC in the same manner because those FEs are different in the approach. That is, BM adds entropy to PUF response using artificial noise and CC reduces entropy loss by discarding some bits of decoded random seeds. The advantages of the proposed AR-based FE over these schemes can be demonstrated using experimental simulations of 128-bit key generation in Section 4.2. TD and the AR-based FE in this work have greater retained entropy than the other conventional schemes. Note that the expected retained entropy of TD and the proposed scheme is the same in the order because all PUF cells outputting a minor value of "0" or "1" are used for helper data generation in both approaches. In contrast, the actually required PUF size, NVM size, computational cost, and applicable bias model are definitely different because of the difference in the underlying principles (*i.e.*, condense-then-fuzzy-extract and rejection sampling), as demonstrated below. In addition, as the TD-based scheme was originally proposed for the syndrome construction, there are some limitations to the adoptable ECC and decoding method, which may yield less-than-optimal PUF and NVM sizes. Note again that the conventional schemes including TD cannot be applied to the cell-wise bias model.

The NVM size, which indicates the total memory size for storing helper data including additional data for the debiasing scheme (*e.g.*, ACL data), is also important for implementation of PUF-based systems on resource-constrained devices. Note here that the values of the l and m terms in the NVM size values of Tab. 3 are heavily dependent on the efficiency of the debiasing schemes themselves. All the schemes require at least an l-bit NVM for storing a common helper data. It seems that BM and CC (and No debias) may be optimal in terms of NVM size, because they do not generate any additional data to be stored in NVM and require only an l-bit NVM for the helper data. However, the PUF sizes of these FEs significantly increases with the PUF bias; therefore, the NVM sizes of these FEs are not always small, as evaluated in Section 4.2. ϵ-2O-VN requires a larger NVM than other VNC-based FE because the helper data of ϵ-2O-VN is given in ternary (*i.e.*, 0, 1, and erasure symbol). Hence, a $2m$-bit memory is required for storing an m-trit helper data. MD requires an additional memory of $\mathcal{O}(2^{\mu})$ for computation of a 2^{μ}-to-one mapping. For TD, $l\lceil\log_2 m\rceil$ corresponds to the memory size for π, which is a random permutation of $(l\lceil\log_2 m\rceil)$-bit integers. Moreover, it may be necessary for TD to implement and perform DRSG_l, (*i.e.*, deterministic generator of random subset with size of l) which would incur a nonnegligible overhead on the memory size on client devises. Thus, the proposed scheme has an advantage in terms of the NVM size, as the retained entropy is greater than those of CVN and 2P-TO-VN.

A reusable FE can enroll a PUF many times with different seeds, with negligible secrecy leakage compared to a single enrollment. Here, only ϵ-2O-VN is proven to be reusable among these debiasing-based FEs. The MD- and CC-based FEs may also be reusable because they always extract a fixed length of debiased bit string given m and p_1 in a deterministic manner, but there have been no proof on their reusability to the best of authors' knowledge. The reusability of FE is one of an important desirable properties that broadens the applicability of the key generation system. However, as apparent from in Tab. 3, the debiasing schemes extracting debiased bits in a stochastic manner cannot offer the reusability; the proposed method is among those schemes. Thus, extension of the proposed method for reusability remains a topic for future work.

Each debiasing scheme requires unique additional operations, some of which can sometimes constitute a computational bottleneck. First, as the reconstruction is usually performed on a (resource-constrained) client device with a PUF, it is desirable for the reconstruction to have no additional operations. For MD- and TD-based FEs, non-trivial operation(s) should be performed for the debiasing in addition to the ECC decoding. The evaluation of 2^{μ}-to-one mapping in MD can be implemented as a table integrated in the NVM. For TD, DRSG_l, list construction, and set manipulation should induce a nonnegligible overhead in terms of both time and space complexity, as Škorić noted in [Š17] that these operations can constitute a computational bottleneck separate from that of the ECC decoding. In contrast, other methods, including the proposed AR-based FE, can be implemented without any additional resource-consuming operations in their reconstructions. Second, in the enrollment phase, the proposed method (and BM) requires an RNG in order to perform rejection sampling (and to generate an artificial noise). TD also employs a DRSG_l in the enrollment phase to add entropy as a fake permutation. However, the computational cost of these operations is not necessarily a critical issue, because the enrollment does not involve real-time processing. The amount of (biased) random numbers required to perform the RRS operation per PUF is equal to the PUF size m, which would be sufficiently practical.

4.1.2 Simulation of 128-bit key generation

The performances of the proposed and conventional FEs are evaluated for 128-bit key generation, where the averaged BER $\mathbb{E}[e]$ and bias p_1 are virtually determined to simulate a PUF response. In this simulation, we use and evaluate the residual min-entropy for a stronger security notation, utilizing the algorithms in [DGV+16]. Whereas the PUF size has been mainly focused in the performance evaluations as in the previous studies involving debiasing schemes, the NVM sizes are also compared in the present evaluation because they would have an impact on the practicality. We employ the global bias model because none of the conventional methods can work with the cell-wise bias model. In this subsection, we focus on the evaluation of the AR-based FE and major counterparts which can be compared in a unified manner (*i.e.*, VNC-, MD-, and BM-based FEs [MLSW15, AWSO17, USH19]). Note that we cannot evaluate ϵ-2O-VN here because of lack of description about how to determine its PUF size. See the appendices for the evaluation and comparison for the rest of conventional methods (*i.e.*, CC- and TD-based FEs [HO17, Š17]).

Tables 4 and 5 report the performance evaluation results of the proposed AR-based FE for $\mathbb{E}[e] = 0.025, 0.050$ and $\mathbb{E}[e] = 0.075, 0.100$, respectively, where "Rep. code" denotes the length of inner repetition code; "# of blks" denotes the number of codeword blocks; "Res. entr." denotes the residual min-entropy; "NVM size" is the NVM size required for storing helper data, including ACL data; and "Failure prob." denotes the expected reconstruction failure probability. The range of PUF bias is given by $0.58 \leq p_1 \leq 0.90$. In this evaluation, according to the evaluation method in [BGS+08], we employ a concatenated code of inner $[\nu_1, \kappa_1, \delta_1]$ repetition code and outer $[\nu_2, \kappa_2, \delta_2]$ BCH code, which is one of most widely used ECC constructions for PUF-based key generation. BCH codes with a code length $\nu_2 \in \{7, 15, 31, 63, 127, 255\}$ are considered. The concatenated code is designed such that the PUF size is minimized under the condition that the expected failure probabilities of enrollment and reconstruction are smaller than a threshold of 10^{-6}. Additionally, the random seed size is determined such that the PUF size is minimized under the condition that the residual min-entropy exceeds 128.

Tables 4 and 5 also report the simulation results for 128-bit key generation using the conventional debiasing-based FEs with the same conditions. As for the proposed method, the NVM size includes additional data for debiasing in addition to common helper data (*e.g.*, debiasing data in the case of the VNC-based FEs). The input bit length of the mapping used for the MD case is determined such that the resulting PUF size should be minimized.

Figure 10 illustrates the PUF sizes for the key generation using the proposed and conventional FEs. The curves for CVN, 2P-TO-VN, and this work (AR-based FE) form a similar shapes, because these approaches utilize acceptance/rejection to extract uniform distributions. However, the proposed AR-based FE exhibits smaller PUF sizes for the various biases and averaged BERs compared to those of the VNC-based FEs. This is because the proposed FE extracts one debiased bit per bit whereas the VNC-based FEs require two bits to obtain a debiased bit (or tuple). More quantitatively, the expected debiased bit length for the proposed AR-based FE is given by $\mathbb{E}[mq'_1] = \mathbb{E}[2m(1-p_1)]$ when $p_1 > p_0$, whereas those for the VNC-based FEs are $\mathbb{E}[mp_1p_0]$ or $\mathbb{E}[mp_1p_0 + mE_{p_0,p_1}]$. This indicates that the proposed AR-based FE can always obtain a longer debiased bit string than those of the VNC-based FEs.

The PUF size of MD-based FE is not an increment function for p_1. The expected debiased bit length of the MD-based FE is given by $\mathbb{E}[m/\mu]$, where μ (≥ 1) is the input bit length of mapping. The value of μ is an integer greater than two in most cases. Though it can be determined to be small for some specific biases for which a good mapping is found, the proposed FE also achieves a smaller PUF size than that of the MD-based FE even in such specific cases (*i.e.*, $p_1 = 0.74$).

Finally, the BM-based FE cannot be evaluated based on the expected debiased bit length, because the overhead of BM includes an increase in the averaged BER for ECC decoding. The AR-based FE has a smaller PUF size than the BM-based even for $p_1 < 0.64$, and the PUF size for BM significantly increases significantly with the increase in bias owing to the added noise. Thus, we can confirm that the proposed AR-based method has an advantage in terms of PUF size compared to the state-of-the-art FEs for PUFs. For example, the AR-based FE can achieve 43%, 55%, 27%, 51%, and 51% smaller PUF sizes than any other conventional debiasing-based FE for $\mathbb{E}[e] = 0.025$ and $p_1 = 0.58$, 0.66, 0.74,

Table 4: Simulation results of 128-bit key generation using proposed and conventional binary debiasing-based FEs for averaged BER $\mathbb{E}[e] = 0.025$ and 0.050

BER $\mathbb{E}[e]$	PUF bias	FE	Rep. code	Outer code	# of blks	Res. entr.	PUF size	NVM size	Failure prob.
0.025	0.58	CVN	1	$[63, 16, 23]$	8	128.0	2,406	1,707	3.89e-07
		2P-TO-VN	4	$[63, 36, 11]$	4	144.0	1,622	3,244	5.18e-07
		MD	3	$[127, 92, 11]$	3	150.3	1,143	1,143	5.03e-07
		BM	5	$[127, 64, 21]$	2	128.0	1,270	1,270	4.30e-08
		This work	1	$[63, 16, 23]$	8	128.0	**656**	**1,160**	3.89e-07
	0.66	CVN	1	$[63, 16, 23]$	8	128.0	2,624	1,816	3.89e-07
		2P-TO-VN	4	$[63, 36, 11]$	4	144.0	1,870	3,740	5.18e-07
		MD	3	$[127, 85, 13]$	3	155.2	2,286	1,143	5.75e-07
		BM	9	$[127, 64, 21]$	2	128.0	2,286	2,286	6.75e-07
		This work	1	$[63, 16, 23]$	8	128.0	**837**	**1,341**	3.89e-07
	0.74	CVN	1	$[63, 16, 23]$	8	128.0	3,086	2,047	3.89e-07
		2P-TO-VN	4	$[63, 36, 11]$	4	144.0	2,372	4,744	5.18e-07
		MD	3	$[255, 187, 19]$	1	133.0	1,530	765	1.00e-07
		BM	15	$[255, 131, 37]$	1	131.0	3,825	3,825	4.40e-07
		This work	1	$[63, 16, 23]$	8	128.0	**1,122**	**1,626**	3.89e-07
	0.82	CVN	1	$[63, 16, 23]$	8	128.0	4,064	2,536	3.89e-07
		2P-TO-VN	4	$[63, 36, 11]$	4	144.0	3,396	6,792	5.18e-07
		MD	3	$[127, 71, 19]$	3	143.5	4,572	1,143	4.49e-07
		BM	35	$[255, 131, 37]$	1	131.0	8,925	8,925	2.28e-07
		This work	1	$[63, 16, 23]$	8	128.0	**1,654**	**2,158**	3.89e-07
	0.90	CVN	1	$[63, 16, 23]$	8	128.0	6,748	3,878	3.89e-07
		2P-TO-VN	4	$[63, 36, 11]$	4	144.0	6,128	12,256	5.18e-07
		MD	5	$[255, 163, 25]$	1	132.0	8,925	1,275	1.10e-08
		BM	115	$[255, 131, 37]$	1	131.0	29,325	29,325	9.99e-07
		This work	1	$[63, 16, 23]$	8	128.0	**3,030**	**3,534**	3.89e-07
0.050	0.58	CVN	1	$[127, 22, 47]$	6	132.0	3,536	2,530	1.13e-07
		2P-TO-VN	4	$[255, 131, 37]$	1	131.0	1,640	3,280	5.75e-09
		MD	3	$[127, 57, 23]$	3	140.5	3,429	1,143	1.21e-07
		BM	5	$[255, 131, 37]$	1	131.0	1,275	1,275	1.66e-08
		This work	1	$[127, 22, 47]$	6	132.0	**975**	**1,737**	1.13e-07
	0.66	CVN	1	$[127, 22, 47]$	6	132.0	3,854	2,689	1.13e-07
		2P-TO-VN	4	$[255, 131, 37]$	1	131.0	1,892	3,784	5.75e-09
		MD	3	$[255, 139, 31]$	2	157.8	3,060	1,530	2.44e-07
		BM	9	$[255, 131, 37]$	1	131.0	2,295	2,295	1.39e-07
		This work	1	$[127, 22, 47]$	6	132.0	**1,237**	**1,999**	1.13e-07
	0.74	CVN	1	$[127, 22, 47]$	6	132.0	4,526	3,025	1.13e-07
		2P-TO-VN	4	$[255, 131, 37]$	1	131.0	2,398	4,796	5.75e-09
		MD	3	$[255, 131, 37]$	2	172.7	3,060	1,530	3.73e-08
		BM	17	$[255, 131, 37]$	1	131.0	4,335	4,335	4.85e-07
		This work	1	$[127, 22, 47]$	6	132.0	**1,651**	**2,413**	1.13e-07
	0.82	CVN	1	$[127, 22, 47]$	6	132.0	5,950	3,737	1.13e-07
		2P-TO-VN	4	$[255, 131, 37]$	1	131.0	3,434	6,868	5.75e-09
		MD	3	$[255, 91, 51]$	3	163.1	9,180	2,295	4.30e-07
		BM	39	$[255, 131, 37]$	1	131.0	9,945	9,945	3.11e-07
		This work	1	$[127, 22, 47]$	6	132.0	**2,425**	**3,187**	1.13e-07
	0.90	CVN	1	$[127, 22, 47]$	6	132.0	9,860	5,692	1.13e-07
		2P-TO-VN	4	$[255, 131, 37]$	1	131.0	6,194	12,388	5.75e-09
		MD	5	$[255, 71, 59]$	3	153.5	26,775	3,825	1.88e-07
		BM	129	$[255, 131, 37]$	1	131.0	32,895	32,895	8.52e-07
		This work	1	$[127, 22, 47]$	6	132.0	**4,430**	**5,192**	1.13e-07

0.82, and 0.90, respectively.

In addition, whereas BM and MD sometimes have smaller NVM sizes for some smaller biases and averaged BERs, the AR-based FE has the smallest NVM size for a wider range of biases and averaged BERs because of its efficiency. As a result, the AR-based FE can achieve the smallest hardware/implementation cost in total.

4.2 FEs exploiting cell-wise reliability

4.2.1 Implementation cost of FEs

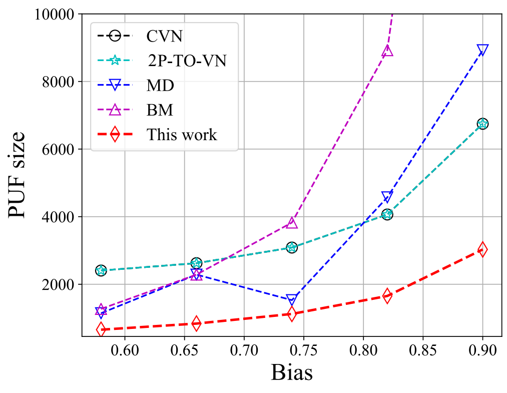
The following subsection reports a comparison of the ternary AR-based FE with its counterparts, that is, the conventional FEs that exploit the cell-wise reliability. As mentioned above, such FEs incur a cost in order to obtain the ternary response or cell-wise reliability for PUF instances and they are not always available (depending on scenario and

Table 5: Simulation results of 128-bit key generation using proposed and conventional binary debiasing-based FEs for averaged BER $\mathbb{E}[e] = 0.075$ and 0.100

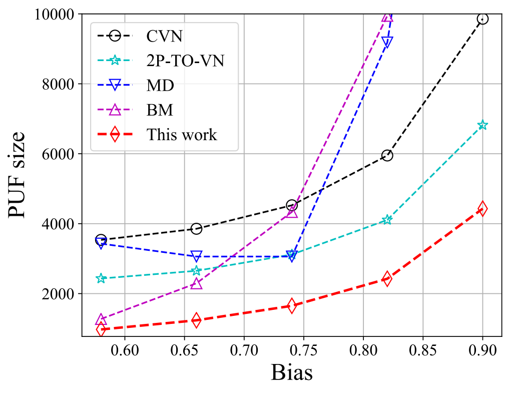
BER $\mathbb{E}[e]$	PUF bias	FE	Rep. code	Outer code	# of blks	Res. entr.	PUF size	NVM size	Failure prob.
0.075	0.58	CVN	1	$[255, 45, 87]$	3	135.0	3,551	2,540	6.18e-07
		2P-TO-VN	6	$[127, 64, 21]$	2	128.0	2,386	4,772	4.81e-09
		MD	3	$[255, 131, 37]$	3	184.0	2,295	2,295	1.40e-07
		BM	7	$[127, 64, 21]$	2	128.0	1,778	1,778	1.84e-07
		This work	1	$[255, 45, 87]$	3	135.0	**978**	**1,743**	6.18e-07
	0.66	CVN	1	$[255, 45, 87]$	3	135.0	3,871	2,700	6.18e-07
		2P-TO-VN	6	$[127, 64, 21]$	2	128.0	2,746	5,492	4.81e-09
		MD	3	$[255, 107, 45]$	3	165.6	4,590	2,295	9.59e-07
		BM	11	$[255, 131, 37]$	1	131.0	2,805	2,805	2.04e-08
		This work	1	$[255, 45, 87]$	3	135.0	**1,241**	**2,006**	6.18e-07
	0.74	CVN	1	$[255, 45, 87]$	3	135.0	4,543	3,036	6.18e-07
		2P-TO-VN	6	$[127, 64, 21]$	2	128.0	3,474	6,948	4.81e-09
		MD	3	$[255, 87, 53]$	3	154.4	4,590	2,295	3.19e-07
		BM	21	$[255, 131, 37]$	1	131.0	5,355	5,355	2.24e-08
		This work	1	$[255, 45, 87]$	3	135.0	**1,657**	**2,422**	6.18e-07
	0.82	CVN	1	$[255, 45, 87]$	3	135.0	5,973	3,751	6.18e-07
		2P-TO-VN	6	$[127, 64, 21]$	2	128.0	4,964	9,928	4.81e-09
		MD	3	$[255, 47, 85]$	6	133.3	18,360	4,590	4.22e-08
		BM	43	$[255, 131, 37]$	1	131.0	10,965	10,965	7.68e-07
		This work	1	$[255, 45, 87]$	3	135.0	**2,434**	**3,199**	6.18e-07
	0.90	CVN	1	$[255, 45, 87]$	3	135.0	9,897	5,713	6.18e-07
		2P-TO-VN	6	$[127, 64, 21]$	2	128.0	8,944	17,888	4.81e-09
		MD	9	$[127, 29, 43]$	8	141.7	64,008	9,144	7.45e-07
		BM	145	$[255, 131, 37]$	1	131.0	36,975	36,975	8.40e-07
		This work	1	$[255, 45, 87]$	3	135.0	**4,446**	**5,211**	6.18e-07
0.100	0.58	CVN	5	$[127, 64, 21]$	2	128.0	5,734	4,137	3.24e-08
		2P-TO-VN	6	$[255, 131, 37]$	1	131.0	2,396	4,792	3.93e-08
		MD	3	$[127, 29, 43]$	6	131.8	6,858	2,286	1.28e-07
		BM	7	$[255, 131, 37]$	1	131.0	1,785	1,785	8.37e-08
		This work	5	$[127, 64, 21]$	2	128.0	**1,597**	**2,867**	3.24e-08
	0.66	CVN	5	$[127, 64, 21]$	2	128.0	6,246	4,393	3.24e-08
		2P-TO-VN	6	$[255, 131, 37]$	1	131.0	2,756	5,512	3.93e-08
		MD	3	$[127, 29, 43]$	12	138.3	9,144	4,572	5.35e-07
		BM	13	$[255, 131, 37]$	1	131.0	3,315	3,315	1.20e-08
		This work	5	$[127, 64, 21]$	2	128.0	**2,016**	**3,286**	3.24e-08
	0.74	CVN	5	$[127, 64, 21]$	2	128.0	7,322	4,931	3.24e-08
		2P-TO-VN	6	$[255, 131, 37]$	1	131.0	3,486	6,972	3.93e-08
		MD	3	$[255, 47, 85]$	6	134.0	9,180	4,590	5.46e-10
		BM	23	$[255, 131, 37]$	1	131.0	5,865	5,865	1.45e-07
		This work	5	$[127, 64, 21]$	2	128.0	**2,678**	**3,948**	3.24e-08
	0.82	CVN	5	$[127, 64, 21]$	2	128.0	9,610	6,075	3.24e-08
		2P-TO-VN	6	$[255, 131, 37]$	1	131.0	4,982	9,964	3.93e-08
		MD	7	$[127, 29, 43]$	14	136.7	49,784	12,446	5.46e-09
		BM	49	$[255, 131, 37]$	1	131.0	12,495	12,495	7.09e-07
		This work	5	$[127, 64, 21]$	2	128.0	**3,921**	**5,191**	3.24e-08
	0.90	CVN	5	$[127, 64, 21]$	2	128.0	15,888	9,214	3.24e-08
		2P-TO-VN	6	$[255, 131, 37]$	1	131.0	8,976	17,952	3.93e-08
		MD	15	$[255, 47, 85]$	6	131.0	160,650	22,950	8.92e-08
		BM	165	$[255, 131, 37]$	1	131.0	42,075	42,075	6.77e-07
		This work	5	$[127, 64, 21]$	2	128.0	**7,140**	**8,410**	3.24e-08

the properties of used PUF). When available, these FEs can achieve higher efficiency than the corresponding binary FEs.

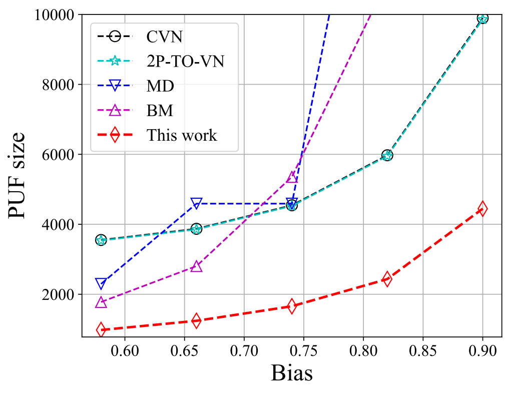
Table 6 summarizes the implementation costs of the FEs in the global bias model in a similar manner to Tab. 3. Here, we assume that $t_1 \geq t_0$ for this work (*i.e.*, the AR-based FE). In Tab. 6, No debias denotes the ternary FE in [KUH19] without debiasing. The ternary CVN (TCVN), the 2-passes tuple-output ternary von Neumann (2P-TO-TVN), the pair-output ternary von Neumann with erasures (ϵ-2O-TVN) denote the VNC-based FEs extended to the ternary PUF response in [SUHA17, SUHA18], namely the ternary VNC (TVNC)-based FEs. Note that the NVM size in Tab. 6 is given in bits, whereas the helper data of No debias and this work is given as a vector consisting of ternary values. In addition, ϵ-2O-TVN also utilizes a ternary helper data consisting of "0," "1," and "erasure." Accordingly, No debias, ϵ-2O-TVN, and this work require a $2l$-bit NVM for storing an l-trit vector $\boldsymbol{w}$. In contrast, the NVM sizes of TCVN and 2P-TO-TVN are given in a manner



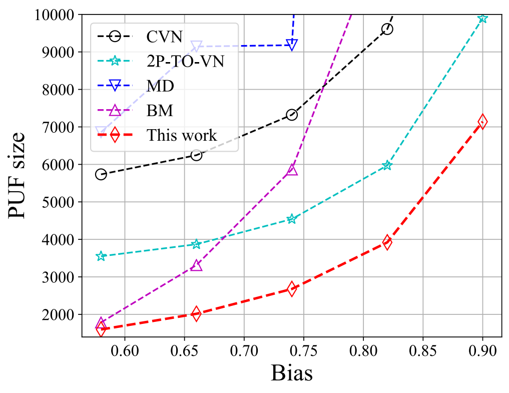
(a) $\mathbb{E}[e] = 0.025$



(b) $\mathbb{E}[e] = 0.500$



(c) $\mathbb{E}[e] = 0.075$



(d) $\mathbb{E}[e] = 0.100$

Figure 10: Comparison of AR-based FE with conventional debiasing-based FEs for binary PUF response.

similar to the binary version, because their helper data (and ECC) are given in binary.

The ternary FEs (*i.e.*, No debias, TCVN, 2P-TO-TVN, ϵ-2O-VN, and this work (the AR-based FE) in Tab. 6) should detect random cells in order to utilize the ternary PUF response (denoted by "detecting rand. cells") before performing the enrollment and reconstruction. Whereas the detection can be implemented with only two delay flip flops (D-FFs), a NAND gate, and an OR gate per PUF cell [YSI$^+$11], a nonnegligible latency is sometimes required for reliable detection of the random cells; that is, for extraction of the ternary response with a lower trit-error rate (TER) in the ECC decoding [SUHA18]. C-IBS should derive the cell-wise reliability at the enrollment, and no additional procedure is required for the reconstruction. Deriving the cell-wise reliability with a statistical means would basically take a larger cost than random cell detection, which is sometimes considered to be a nonnegligible overhead [LPS12]. In summary, the enrollment cost is higher for the C-IBS, whereas the TVNC- and ternary AR-based FEs require a latency overhead at reconstruction.

Table 6: Comparison of implementation costs for FEs exploiting cell-wise reliability

	Retained entropy	NVM size*	Reusa-bility	Candidates of computational bottleneck	
				Enrollment	Reconstruction
No debias [KUH19]	N/A	$2l$	Yes	Detecting rand. cells and ECC enc.	Detecting rand. cells and ECC dec.
TCVN [SUHA18]	$mT_{t_0,t_1,t_\mathrm{r}}$ †	$l + \lceil m/2 \rceil$	No	Detecting rand. cells and ECC enc.	Detecting rand. cells and ECC dec.
2P-TO-TVN [SUHA18]	$mT_{t_0,t_1,t_\mathrm{r}}$ $+mE'_{t_0,t_1,t_\mathrm{r}}$ †‡	$2m$	No	Detecting rand. cells and ECC enc.	Detecting rand. cells and ECC dec.
ϵ-2O-TVN [SUHA18]	$mT_{t_0,t_1,t_\mathrm{r}}$ †	$5m/2$	Yes	Detecting rand. cells and ECC enc.	Detecting rand. cells and ECC dec.
C-IBS [HMSS12]	$m\nu_1/\lambda$	$\lceil \log_2 \lambda \rceil/\lambda$ $\times m\nu_1$	No	Deriving cell-wise reliability and ECC enc.	ECC dec.
This work	$m(2t_0 + t_\mathrm{r})$	$2l + m$	No	Detecting rand. cells RNG, and ECC enc.	Detecting rand. cells and ECC dec.

* NVM size indicates the total memory size for storing helper data including addition data for debiasing scheme (*e.g.*, ACL data). Here, m and l differ depending on the efficiency of retained entropy and BER/TER during ECC decoding. The smaller retained entropy and higher BER/TER yield the greater m and l.

† $T_{t_0,t_1,t_\mathrm{r}} = t_0 t_1 + t_0 t_\mathrm{r} + t_\mathrm{r} t_1$.

‡ $E'_{t_0,t_1,t_\mathrm{r}} = \dfrac{t_0^2 t_1^2 + t_0^2 t_\mathrm{r}^2 + t_\mathrm{r}^2 t_1^2}{2(t_0^2 + t_1^2 + t_\mathrm{r}^2)}$.

4.2.2 Simulation of 128-bit key generation

Tables 7 and 8 report the performance evaluation results of the TVNC-based FEs, ternary AR-based FE, and the FE with C-IBS in the same manner as Tabs. 4 and 5, respectively.[11] We cannot evaluate ϵ-2O-TVN here as well as ϵ-2O-VN. Here, the ternary AR-based FE employs the ternary Reed-Solomon (RS) code over $GF(3)$, as in the ternary FE of [KUH19], whereas the C-IBS and TVNC-based FEs employ a binary BCH code. Note that, for the ternary AR-based FE, the random seed is given by a trit string with a sufficient length so that the residual min-entropy is greater than 128. The "λ or t_r" column in each table denotes the index block size for C-IBS or the random cell ratio for ternary AR- and TVNC-based FEs. The column of ν_1 denotes the length of inner code, that is, the length of C-IBS codeword for C-IBS and the repetition code length for TVNC-based FEs and this work (AR-based FE). We evaluated the FEs when $t_\mathrm{r} = 0.20$ and 0.30 for the ternary FEs. The row of PUF bias denotes the bias in binary response (*i.e.*, p_1). We assume that $p_0 = t_0 + p_0 t_\mathrm{r}$ and $p_1 = t_1 + p_1 t_\mathrm{r}$ for conversion of (p_0, p_1) to (t_0, t_1, t_r). Furthermore, recall that $\mathbb{E}[e]$ also denotes the averaged BER for the binary PUF response; thus, we use the empirical noise model presented by Maes [Mae13] and the formulas in [SUHA18] to convert $\mathbb{E}[e]$ to averaged TER for evaluation of the ternary FEs. Figure 11 illustrates the PUF sizes required for 128-bit key generation with the these FEs.

As detailed in Tabs. 7 and 8 and Fig. 11, the ternary AR-based FE can achieve the smallest PUF size for cases in which the averaged digit-error rate (*i.e.*, BER or TER) is

[11] In [SUHA18], there is no description about how to determine the required PUF size m for TVNC-based FEs such that the enrollment failure probability is smaller than a threshold. As the acceptance probability of a cell (*i.e.*, $\mathbb{E}[p_\mathrm{accept}]$ in Eq. (16)) is given by $2(t_0 t_1 + t_0 t_\mathrm{r} + t_\mathrm{r} t_1)$, we calculate the PUF size m for the TCVN using the following bound: $l \leq \mathrm{cmf}_\mathrm{bino}^{-1}\left(P_\mathrm{thre}^{(\mathrm{en})}; \left\lfloor \frac{m}{2} \right\rfloor, 2(t_0 t_1 + t_1 t_r + t_r t_0)\right)$ with $P_\mathrm{thre}^{(\mathrm{en})} = 10^{-6}$. Similarly to TCVN, the PUF size m for the 2P-TO-TVN is also determined by replacing the acceptance probabilities of the bound for the binary 2P-TO-VN with the ternary ones as follows: $\sum_{a=0}^{l-1} \sum_{b=0}^{\lfloor \frac{m}{4} \rfloor} \mathrm{pmf}_\mathrm{bino}\left(\frac{a-4b}{2}; \left\lfloor \frac{m}{2} \right\rfloor, 2(t_0 t_1 + t_1 t_r + t_r t_0)\right) \cdot \mathrm{pmf}_\mathrm{bino}\left(b; \left\lfloor \frac{m-(a-4b)}{4} \right\rfloor, \frac{2(t_0^2 t_1^2 + t_1^2 t_r^2 + t_r^2 t_0^2)}{(t_0^2 + t_1^2 + t_r^2)^2}\right) < P_\mathrm{thre}^{(\mathrm{en})}$, where $\mathrm{pmf}_\mathrm{bino}(l; m, p)$ denotes a pmf for binomial distribution having l-times successes from m-times challenges with a probability p.

Table 7: Simulation results of 128-bit key generation using C-IBS and ternary debiasing-based FEs for averaged BER $\mathbb{E}[e] = 0.025$ and 0.050

BER $\mathbb{E}[e]$	PUF bias	FE	λ or t_r	ν_1	Outer code	# of blks	Res. entr.	PUF size	NVM size	Failure prob.
0.025	0.58	C-IBS	6	3	$[127, 64, 21]$	2	128.0	1,524	3,810	8.63e-08
		TCVN	0.20	1	$[63, 36, 11]$	4	144.0	958	731	7.17e-09
		2P-TO-TVN		4	$[31, 26, 3]$	5	130.0	1,158	2,316	7.81e-07
		This work		1	$[26, 12, 15]$	4	146.2	**397**	**709**	1.38e-07
		TCVN	0.30	1	$[63, 36, 11]$	4	144.0	914	709	2.33e-09
		2P-TO-TVN		4	$[31, 26, 3]$	5	130.0	1,104	2,208	3.66e-07
		This work		1	$[26, 14, 13]$	3	183.2	**295**	**529**	1.31e-07
	0.66	C-IBS	7	3	$[127, 64, 21]$	2	128.0	1,778	3,810	1.93e-07
		TCVN	0.20	1	$[63, 36, 11]$	4	144.0	1,002	753	9.04e-09
		2P-TO-TVN		4	$[31, 26, 3]$	5	130.0	1,210	2,420	9.14e-07
		This work		1	$[26, 12, 15]$	4	146.2	**482**	**794**	1.38e-07
		TCVN	0.30	1	$[63, 36, 11]$	4	144.0	944	724	2.77e-09
		2P-TO-TVN		4	$[31, 26, 3]$	5	130.0	1,142	2,284	4.11e-07
		This work		1	$[26, 14, 13]$	3	183.2	**352**	**586**	1.31e-07
	0.74	C-IBS	8	3	$[255, 131, 37]$	1	131.0	2,040	3,825	3.36e-07
		TCVN	0.20	1	$[63, 36, 11]$	4	144.0	1,084	794	1.36e-08
		2P-TO-TVN		4	$[15, 11, 3]$	12	132.0	1,500	3,000	6.52e-07
		This work		1	$[26, 12, 15]$	4	146.2	**599**	**911**	1.38e-07
		TCVN	0.30	1	$[63, 36, 11]$	4	144.0	1,000	752	3.72e-09
		2P-TO-TVN		4	$[31, 26, 3]$	5	130.0	1,208	2,416	5.02e-07
		This work		1	$[26, 14, 13]$	3	183.2	**423**	**657**	1.31e-07
	0.82	C-IBS	12	3	$[255, 131, 37]$	1	131.0	3,060	3,825	4.95e-08
		TCVN	0.20	1	$[63, 36, 11]$	4	144.0	1,220	862	2.53e-08
		2P-TO-TVN		4	$[15, 11, 3]$	12	132.0	1,684	3,368	9.94e-07
		This work		1	$[26, 12, 15]$	4	146.2	**774**	**1,086**	1.38e-07
		TCVN	0.30	1	$[63, 36, 11]$	4	144.0	1,088	796	5.78e-09
		2P-TO-TVN		4	$[31, 26, 3]$	5	130.0	1,312	2,624	6.75e-07
		This work		1	$[26, 14, 13]$	3	183.2	**522**	**756**	1.31e-07
	0.9	C-IBS	20	3	$[255, 131, 37]$	1	131.0	5,100	3,825	6.02e-07
		TCVN	0.20	1	$[63, 36, 11]$	4	144.0	1,444	974	6.30e-08
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	2,082	4,164	3.83e-12
		This work		1	$[26, 12, 15]$	4	146.2	**1,070**	**1,382**	1.38e-07
		TCVN	0.30	1	$[63, 36, 11]$	4	144.0	1,222	863	1.07e-08
		2P-TO-TVN		4	$[15, 11, 3]$	12	132.0	1,686	3,372	5.56e-07
		This work		1	$[26, 14, 13]$	3	183.2	**668**	**902**	1.31e-07
0.050	0.58	C-IBS	6	3	$[127, 64, 21]$	2	128.0	1,524	3,810	1.39e-07
		TCVN	0.20	1	$[127, 64, 21]$	2	128.0	966	737	1.57e-08
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	1,390	2,780	3.54e-08
		This work		1	$[80, 28, 53]$	2	190.2	**788**	**1,428**	3.69e-07
		TCVN	0.30	1	$[127, 64, 21]$	2	128.0	920	714	3.80e-09
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	1,326	2,652	1.18e-08
		This work		1	$[80, 24, 57]$	1	131.0	**397**	**717**	3.59e-12
	0.66	C-IBS	7	3	$[127, 64, 21]$	2	128.0	1,778	3,810	1.56e-07
		TCVN	0.20	1	$[127, 64, 21]$	2	128.0	1,010	759	2.29e-08
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	1,452	2,904	4.76e-08
		This work		1	$[80, 28, 53]$	2	190.2	**948**	**1,588**	3.69e-07
		TCVN	0.30	1	$[127, 64, 21]$	2	128.0	952	730	5.07e-09
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	1,370	2,740	1.47e-08
		This work		1	$[80, 24, 57]$	1	131.0	**470**	**790**	3.59e-12
	0.74	C-IBS	8	3	$[255, 131, 37]$	1	131.0	2,040	3,825	9.04e-08
		TCVN	0.20	1	$[127, 64, 21]$	2	128.0	1,092	800	4.45e-08
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	1,568	3,136	8.03e-08
		This work		1	$[80, 28, 53]$	2	190.2	**1,168**	**1,808**	3.69e-07
		TCVN	0.30	1	$[127, 64, 21]$	2	128.0	1,008	758	8.34e-09
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	1,448	2,896	2.17e-08
		This work		1	$[80, 24, 57]$	1	131.0	**563**	**883**	3.59e-12
	0.82	C-IBS	11	3	$[255, 131, 37]$	1	131.0	2,805	3,825	3.36e-07
		TCVN	0.20	1	$[127, 64, 21]$	2	128.0	1,228	868	1.23e-07
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	1,762	3,524	1.79e-07
		This work		1	$[80, 28, 53]$	2	190.2	**1,499**	**2,139**	3.69e-07
		TCVN	0.30	1	$[127, 64, 21]$	2	128.0	1,096	802	1.75e-08
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	1,574	3,148	3.86e-08
		This work		1	$[80, 24, 57]$	1	131.0	**692**	**1,012**	3.59e-12
	0.9	C-IBS	18	3	$[255, 131, 37]$	1	131.0	4,590	3,825	5.08e-07
		TCVN	0.20	1	$[127, 64, 21]$	2	128.0	1,456	982	5.38e-07
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	2,082	4,164	5.82e-07
		This work		1	$[80, 28, 53]$	2	190.2	**2,062**	**2,702**	3.69e-07
		TCVN	0.30	1	$[127, 64, 21]$	2	128.0	1,230	869	4.91e-08
		2P-TO-TVN		4	$[63, 45, 7]$	3	135.0	1,764	3,528	8.67e-08
		This work		1	$[80, 24, 57]$	1	131.0	**885**	**1,205**	3.59e-12

Table 8: Simulation results of 128-bit key generation using C-IBS and ternary debiasing-based FEs for averaged BER $\mathbb{E}[e] = 0.075$ and 0.100

BER $\mathbb{E}[e]$	PUF bias	FE	λ or t_r	ν_1	Outer code	# of blks	Res. entr.	PUF size	NVM size	Failure prob.
		C-IBS	6	3	$[127, 64, 21]$	2	128.0	1,524	3,810	6.11e-07
		TCVN		1	$[255, 131, 37]$	1	131.0	969	739	1.86e-08
		2P-TO-TVN	0.20	4	$[31, 16, 7]$	8	128.0	1,790	3,580	7.91e-07
	0.58	**This work**		3	$[80, 52, 29]$	2	255.7	**2,292**	**4,212**	3.65e-07
		TCVN		1	$[255, 131, 37]$	1	131.0	925	717	7.68e-09
		2P-TO-TVN	0.30	4	$[31, 16, 7]$	8	128.0	1,708	3,416	5.08e-07
		This work		1	$[80, 14, 67]$	2	143.5	**770**	**1,410**	5.97e-08
		C-IBS	7	3	$[127, 64, 21]$	2	128.0	1,778	3,810	2.76e-07
		TCVN		1	$[255, 131, 37]$	1	131.0	1,015	762	3.23e-08
		2P-TO-TVN	0.20	4	$[63, 36, 11]$	4	144.0	1,896	3,792	2.25e-09
	0.66	**This work**		3	$[80, 52, 29]$	2	255.7	**2,728**	**4,648**	3.65e-07
		TCVN		1	$[255, 131, 37]$	1	131.0	955	732	1.18e-08
		2P-TO-TVN	0.30	4	$[31, 16, 7]$	8	128.0	1,764	3,528	6.29e-07
		This work		1	$[80, 14, 67]$	2	143.5	**904**	**1,544**	5.97e-08
		C-IBS	8	3	$[255, 131, 37]$	1	131.0	2,040	3,825	5.22e-08
		TCVN		1	$[255, 131, 37]$	1	131.0	1,097	803	8.48e-08
		2P-TO-TVN	0.20	4	$[63, 36, 11]$	4	144.0	2,046	4,092	4.70e-09
0.075	0.74	**This work**		3	$[80, 52, 29]$	2	255.7	**3,335**	**5,255**	3.65e-07
		TCVN		1	$[255, 131, 37]$	1	131.0	1,011	760	2.45e-08
		2P-TO-TVN	0.30	4	$[31, 16, 7]$	8	128.0	1,864	3,728	9.11e-07
		This work		1	$[80, 14, 67]$	2	143.5	**1,076**	**1,716**	5.97e-08
		C-IBS	11	3	$[255, 131, 37]$	1	131.0	2,805	3,825	1.52e-07
		TCVN		1	$[255, 131, 37]$	1	131.0	1,233	871	3.65e-07
		2P-TO-TVN	0.20	4	$[63, 36, 11]$	4	144.0	2,296	4,592	1.46e-08
	0.82	**This work**		3	$[80, 52, 29]$	2	255.7	**4,251**	**6,171**	3.65e-07
		TCVN		1	$[255, 131, 37]$	1	131.0	1,101	805	7.26e-08
		2P-TO-TVN	0.30	4	$[63, 36, 11]$	4	144.0	2,052	4,104	4.17e-09
		This work		1	$[80, 14, 67]$	2	143.5	**1,315**	**1,955**	5.97e-08
		C-IBS	17	5	$[255, 131, 37]$	1	131.0	4,335	6,375	5.63e-07
		TCVN		1	$[127, 43, 29]$	3	129.0	2,099	1,430	2.94e-07
		2P-TO-TVN	0.20	4	$[63, 36, 11]$	4	144.0	2,710	5,420	7.77e-08
	0.9	**This work**		3	$[80, 52, 29]$	2	255.7	**5,813**	**7,733**	3.65e-07
		TCVN		1	$[255, 131, 37]$	1	131.0	1,235	872	3.26e-07
		2P-TO-TVN	0.30	4	$[63, 36, 11]$	4	144.0	2,300	4,600	1.34e-08
		This work		1	$[80, 14, 67]$	2	143.5	**1,672**	**2,312**	5.97e-08
		C-IBS	6	3	$[255, 131, 37]$	1	131.0	1,530	3,825	3.54e-09
		TCVN		1	$[127, 43, 29]$	3	129.0	1,401	1,081	7.47e-07
		2P-TO-TVN	0.20	4	$[63, 36, 11]$	4	144.0	1,816	3,632	1.82e-07
	0.58	**This work**		3	$[80, 40, 41]$	2	157.4	**2,292**	**4,212**	8.44e-07
		TCVN		1	$[127, 43, 29]$	3	129.0	1,337	1,049	7.12e-07
		2P-TO-TVN	0.30	4	$[63, 36, 11]$	4	144.0	1,734	3,468	1.74e-07
		This work		3	$[80, 28, 53]$	1	134.9	**1,141**	**2,101**	3.22e-15
		C-IBS	7	3	$[127, 64, 21]$	2	128.0	1,778	3,810	7.52e-07
		TCVN		1	$[63, 16, 23]$	8	128.0	1,896	1,452	2.09e-07
		2P-TO-TVN	0.20	4	$[63, 36, 11]$	4	144.0	1,896	3,792	2.68e-07
	0.66	**This work**		3	$[80, 40, 41]$	2	157.4	**2,728**	**4,648**	8.44e-07
		TCVN		1	$[127, 43, 29]$	3	129.0	1,381	1,071	9.87e-07
		2P-TO-TVN	0.30	4	$[63, 36, 11]$	4	144.0	1,790	3,580	2.35e-07
		This work		3	$[80, 28, 53]$	1	134.9	**1,333**	**2,293**	3.22e-15
		C-IBS	8	3	$[255, 131, 37]$	1	131.0	2,040	3,825	4.54e-08
		TCVN		1	$[63, 16, 23]$	8	128.0	2,046	1,527	3.99e-07
		2P-TO-TVN	0.20	4	$[63, 36, 11]$	4	144.0	2,046	4,092	5.33e-07
0.100	0.74	**This work**		3	$[80, 40, 41]$	2	157.4	**3,335**	**5,255**	8.44e-07
		TCVN		1	$[63, 16, 23]$	8	128.0	1,892	1,450	3.01e-07
		2P-TO-TVN	0.30	4	$[63, 36, 11]$	4	144.0	1,892	3,784	3.95e-07
		This work		3	$[80, 28, 53]$	1	134.9	**1,582**	**2,542**	3.22e-15
		C-IBS	11	5	$[255, 131, 37]$	1	131.0	2,805	6,375	1.90e-08
		TCVN		1	$[255, 71, 59]$	2	142.0	2,322	1,671	7.16e-11
		2P-TO-TVN	0.20	4	$[127, 64, 21]$	2	128.0	2,312	4,624	3.12e-11
	0.82	**This work**		3	$[80, 40, 41]$	2	157.4	**4,251**	**6,171**	8.44e-07
		TCVN		1	$[63, 16, 23]$	8	128.0	2,052	1,530	6.26e-07
		2P-TO-TVN	0.30	4	$[63, 36, 11]$	4	144.0	2,052	4,104	8.58e-07
		This work		3	$[80, 28, 53]$	1	134.9	**1,927**	**2,887**	3.22e-15
		C-IBS	16	5	$[255, 131, 37]$	1	131.0	4,080	6,375	4.75e-07
		TCVN		1	$[255, 71, 59]$	2	142.0	2,740	1,880	1.82e-09
		2P-TO-TVN	0.20	4	$[127, 64, 21]$	2	128.0	2,730	5,460	5.29e-10
	0.9	**This work**		3	$[80, 40, 41]$	2	157.4	**5,813**	**7,733**	8.44e-07
		TCVN		1	$[255, 71, 59]$	2	142.0	2,324	1,672	2.10e-10
		2P-TO-TVN	0.30	4	$[127, 64, 21]$	2	128.0	2,316	4,632	7.94e-11
		This work		3	$[80, 28, 53]$	1	134.9	**2,445**	**3,405**	3.22e-15

relatively small. This is because the ternary AR-based FE has a larger expected length of debiased trit string than those of the TVNC-based FEs. Notably, if the random cells ratio (*i.e.*, t_{r}) is higher, both the ternary AR- and TVNC-based FEs are become more efficient. This clearly shows the effectiveness of ternary encoding of the PUF response; that is, the random cells can contribute to the entropy. In contrast, they are considered as a noise in the binary encoding, including C-IBS.

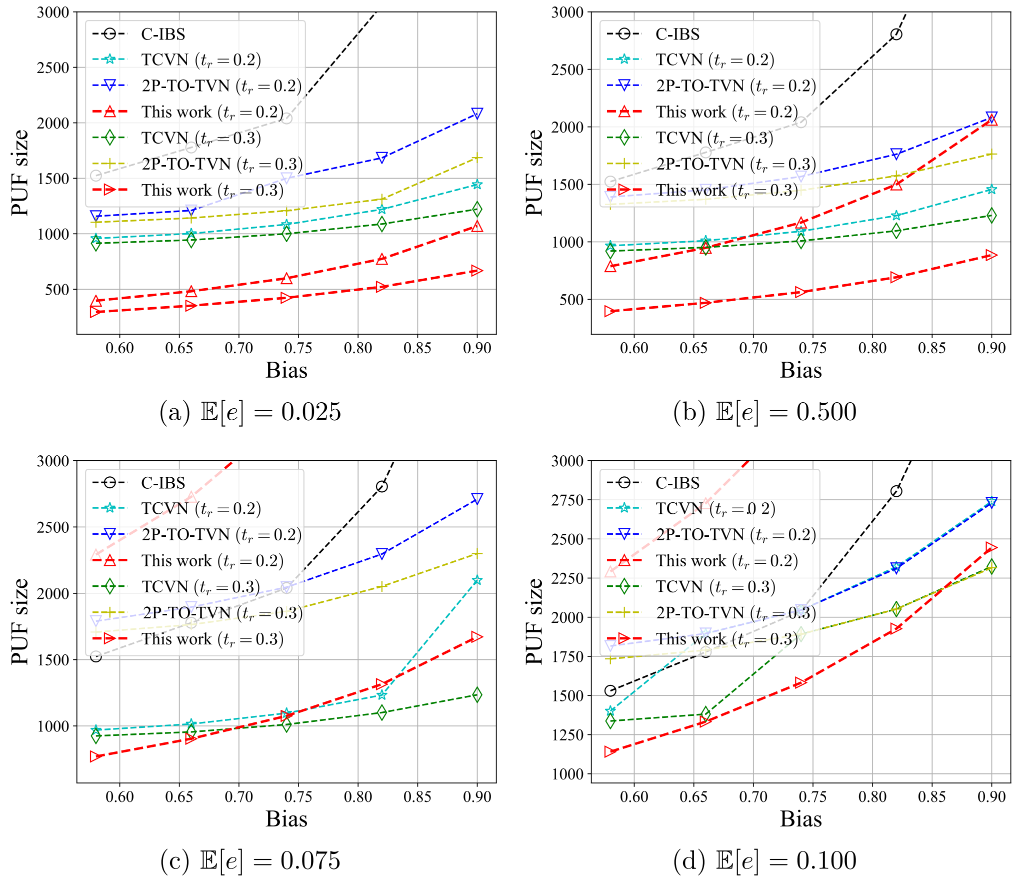
The efficiency of the TVNC-based FE is comparable to or higher than that of the AR-based FE when the averaged digit-error rate is relatively high and $t_{\mathrm{r}} = 0.20$. This is because that the mapping used in the TVNC is equivalent to an ECC decoding with a Hamming erasure distance. Hence, the TVNC-based FE is more robust to an increase in noise, and can, therefore, provide efficient key generation even if the noise is increased. Moreover, if the ratio of random cells is lower, the debiased trit string of ternary AR-based debiasing is far from the uniform distribution, which leads to a nonnegligible min-entropy loss and larger PUF size when the averaged TER is relatively higher. From the perspective of NVM size, we can also confirm that the ternary AR-based FE has a size comparable to conventional methods for PUFs to which the ternary AR-based FE is efficiently applied, whereas the ternary AR-based FE requires a $2l$-bit NVM for storing an l-trit vector. This yields the smaller hardware/implementation cost of the ternary AR-based FE in total. Thus, we can confirm the effectiveness of the ternary AR-based FE for PUFs with some biases and digit-error rates, especially in the cases that the ratio of random cells is high. For example, the ternary AR-based FE can achieve up-to 63% and 57% higher efficiency than its conventional counterparts for $\mathbb{E}[e] = 0.025$ and 0.050, respectively.

5 Discussion

5.1 Required preconditions and assumptions

In this study, we assume that the bias of PUF is known. However, it is sometimes difficult to obtain the PUF biases in practice. To the best of authors' knowledge, there are two major methods for estimating the PUF biases: analog simulation and statistical means. Analog simulation extracts the characteristics of PUF using circuit simulation tools such as SPICE, Spectre, and Nanosim. We can extract an exact bias (and BER) if we use a precise model. One major drawback of such simulation-based methods is the computational cost. It is quite time-consuming to simulate circuits with a precise model. Each PUF characteristics may differ depending on its location in a chip (that is one of the motivations to introduce the cell-wise bias model). Therefore, it would be desirable to simulate whole PUF circuits but not a PUF cell to obtain an exact (cell-wise) bias, which would be infeasible in practice. On the other hand, an estimation method based on a statistical means is to measure PUF responses from many PUF devices and then analyze the obtained PUF response statistically from the viewpoints of, for example, (cell-wise) bias and BER. However, the accuracy/resolution of the estimated characteristics depends on the number of available PUF devices and measurements of PUF responses. For achieving a high accuracy/resolution, we require more PUF devices and measurements, which leads to a non-negligible cost. Note that statistical methods obtain only an approximated bias and BER because they are given by the finite number of PUF devices and measurements. Moreover, the characteristics of PUFs may change depending on the measurement environment including supply voltage and temperature, which makes it more difficult to identify the PUF bias without any ambiguity nor fluctuation.

Thus, it would be sometimes difficult to estimate the exact bias of PUF, which means that the PUF bias is given with limited resolution in practice. If the estimated PUF bias is different from the actual bias, the debiasing schemes cannot extract a uniform distribution, which yields the entropy leakage from the helper data even in the presence of debiasing



(a) $\mathbb{E}[e] = 0.025$ (b) $\mathbb{E}[e] = 0.500$ (c) $\mathbb{E}[e] = 0.075$ (d) $\mathbb{E}[e] = 0.100$

Figure 11: Comparison of FEs exploiting cell-wise reliability.

schemes. However, even in the presence of such a discrepancy, the debiasing schemes with an entropy buffer can realize a secure key generation with sufficient residual (min-)entropy via the worst-case evaluation.

Let Δ be the difference between the estimated global bias and actual bias, which can also be considered as the resolution of bias. We assume that the actual biases are given as $p_1 + \Delta$ and $p_0 - \Delta$ for the probabilities of values "1" and "0," respectively. The remaining biases of $\boldsymbol{u}$ (*i.e.*, p'_1 and p'_0) in the binary AR-based FE can be derived by modifying Eqs. (5) and (6) as follows:

$$\begin{aligned} p'_1 &= \Pr(x_i = 1 | d_i = 1) \\ &= \frac{(p_1 + \Delta) q_1}{q'_1} \\ &= \frac{p_0 p_1 + \Delta p_0}{2 p_0 p_1 - \Delta(p_1 - p_0)}, \end{aligned} \tag{30}$$

$$\begin{aligned} p'_0 &= \Pr(x_i = 0 | d_i = 1) \\ &= \frac{p_0 - \Delta}{q'_1} \\ &= \frac{p_0 p_1 - \Delta p_1}{2 p_0 p_1 - \Delta(p_1 - p_0)}, \end{aligned} \tag{31}$$

when $p_1 \geq p_0$. Note here that q'_1 (*i.e.*, the bias of $\boldsymbol{d}$) in Eqs. (5) and (6) is given by

Table 9: Resulting biases p'_1 for different PUF biases p_1 and resolutions Δ

		p_1							
		0.55	0.60	0.65	0.70	0.75	0.80	0.85	0.90
Δ	0.005	0.505	0.505	0.506	0.506	0.507	0.508	0.510	0.514
	0.010	0.510	0.510	0.511	0.512	0.514	0.516	0.520	0.529
	0.015	0.515	0.516	0.517	0.518	0.529	0.524	0.531	0.545
	0.020	0.520	0.521	0.522	0.524	0.527	0.532	0.541	0.561
	0.025	0.525	0.526	0.528	0.530	0.534	0.541	0.553	0.578
	0.030	0.530	0.532	0.534	0.537	0.542	0.550	0.564	0.596

$q'_1 = (p_0 - \Delta) + q_1 + (p_0 - \Delta)q_1 = 2p_0 - \Delta(1 - p_0/p_1)$ in this case, because $\boldsymbol{d}$ is computed from PUF response $\boldsymbol{x}$ having a bias of $p_1 + \Delta$. When $p_0 > p_1$, the resulting bias is given in a symmetrical manner about p_0 and p_1. Here, Eqs. (30) and (31) represent the worst-case biases of $\boldsymbol{u}$ owing to the difference between the estimated and actual biases of PUF. Therefore, we can evaluate the bounds of the residual (min-)entropy leakage/loss of the AR-based FE(s) according to the algorithms presented by Delvaux *et al.* [DGV$^+$16], assuming that $\boldsymbol{u}$ has the biases of p'_0 and p'_1.

Table 9 and Fig. 12 display the resulting bias p'_1 for different PUF biases p_1 and resolutions Δ. Here, we show the case that $p_1 \geq p_0$ as it should be symmetrical. In the most cases, the resulting bias is less than 0.58. In particular, if the resolution is higher than 0.02 (*i.e.*, $\Delta \leq 0.02$), we found that $0.42 \leq p'_1 \leq 0.58$ for $0.08 \leq p_1 \leq 0.92$. It is known that a 128-bit key generation can be easily realized by buffering entropy when $0.42 \leq p_1 \leq 0.58$ as shown in Tab. 1 and Refs. [MLSW15, DGV$^+$16]. As a resolution of 0.02 would be sufficiently practical for the design and evaluation of PUFs, the problem regarding the resolution of bias is trivial for the AR-based FE in terms of entropy leakage.

In addition, the PUF size of AR-based FE should be determined regarding the actual bias for a reliable enrollment, which means that we estimate the worst-case estimation of the excepted acceptance probability $\mathbb{E}[p_{\mathrm{accept}}]$ in Eq. (16). In other words, a limited resolution of bias would also contaminate the enrollment reliability. Note that the limited resolution of bias does not contaminate the reconstruction reliability of AR-based FEs because it has no influence on the averaged BER (or TER) at the ECC decode.

In this part of the study, we employed the global bias model. However, even if we employ the cell-wise bias model, the expected remaining bias in $\boldsymbol{u}$ and the worst-case residual (min-)entropy can be evaluated using the expected value of biases, *i.e.*, the global biases. Therefore, the global bias and Eqs. (30) and (31) are used for the worst-case evaluation and for determination of PUF size with regard to the residual (min-)entropy, even for assumption of a cell-wise bias model. Thus, the bias and BER of PUF, their resolutions, and their distribution (*i.e.*, cell-wise bias and reliability) are required as a precondition for design of an FE and a key generation system based on the proposed debiasing scheme (as well as the conventional schemes). These designs would be established for given implementation resources such as PUF and NVM sizes.

Note that all the conventional debiasing schemes should also consider the problem pertaining to the resolution of bias, and determine the PUF and NVM sizes according to the lower bound estimated from the worst-case. Here, depending on the scheme, the limited bias resolution contaminates both/either security and/or reliability. For example, as the VNC-based FEs extract debiased bit strings without parameters depending on bias, and therefore, only the enrollment reliability is contaminated by the resolution of bias in the case of VNC-based FE. In contrast, in the case of any other schemes, the security is compromised if the difference between actual and estimated biases is nonnegligible, because these FEs utilize a PUF bias as a parameter or input of their debiasing algorithm. This induces the entropy leakage due to the remaining bias after debiasing, and therefore these FEs should have an entropy buffer. In addition, since BM and MD calculate the

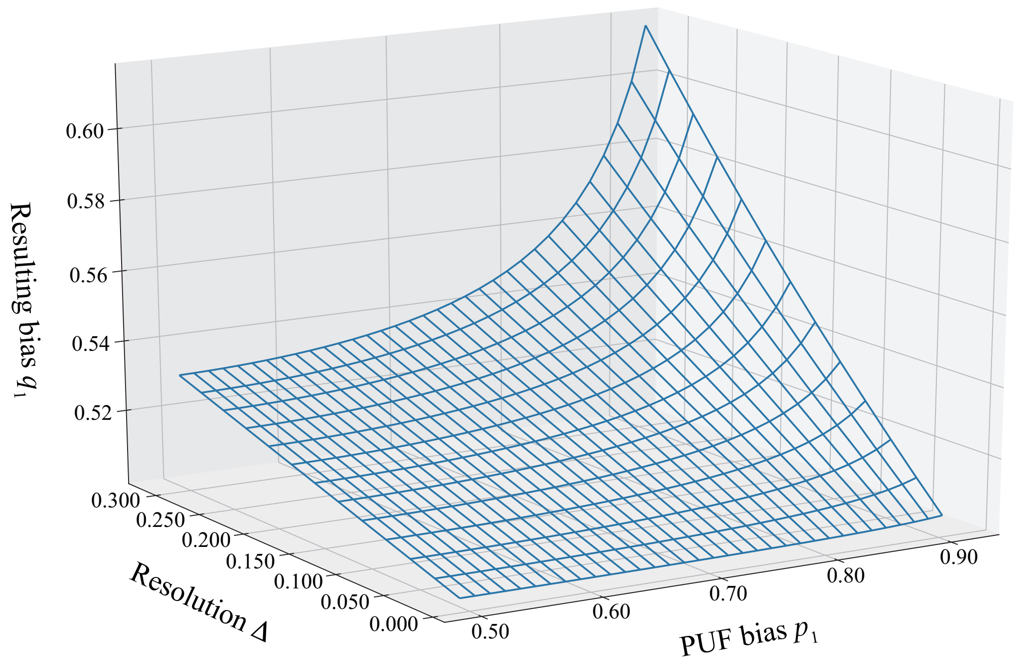


Figure 12: Graphical representation of resulting biases p'_1 corresponding to Tab. 9.

averaged BER at the ECC decode in their reconstruction from the estimated bias, the reconstruction reliability would also be affected for BM and MD.

Another discussion point is the validity/generality of the bias models. The disorder between the model and actual bias induces the entropy leakage. Whereas the cell-wise bias model can represent a wide range of PUF biases (*e.g.*, the area-wise bias in Fig. 2 and the word-wise bias of SRAM PUF), some PUFs are known to have the spatial correlation, spatial proximity, and/or bias in the higher-order statistical moment(s), which are outside the cell-wise bias model [WGP18]. A spatial correlation or proximity indicates an (auto)correlation between cells in a PUF response and PUF response is not i.i.d. Such correlations decrease the (min-)entropy of the PUF response, and induce the (min-)entropy loss. Although we assumed that the PUF response is i.i.d in this study, it may be possible to extend the AR-based debiasing to mitigate such correlations. To achieve this, word-wise rejection sampling depending on the shape of correlation is one of possible extensions of the AR-based FE. As rejection sampling can be basically applied to any type of distribution, rejection sampling with a sufficiently large word length is likely to be used for extraction of the decorrelated response. Its implementation and evaluation would remain as future work.

5.2 Reusability limitation

Reusability is one of the important properties for FEs because it makes the application range of PUF-based systems larger. A reusable FE can enroll random seed many times with a negligible leakage without using the previous helper data. It is clear that the AR-based FEs are not reusable because an attacker knows that the rejected cell has a value of one if $p_1 > p_0$ and *vice versa*, and it may be used in the next enrollment.

Among the debiasing-based FEs mentioned in this paper, ϵ-2O-VN is proven to be reusable. The reusability of ϵ-2O-VN is achieved by replacing the discarded bit with an erasure symbol (or a placeholder). Although we mentioned some similarities between the AR- and VNC-based FEs, such an extension to the reusability is not applicable to the AR-based FE. Therefore, another extension approach would be required to make the AR-based FE reusable. Since Delvaux *et al.* mentioned in [DGV$^+$16] that the reusable VNC-based FE has a considerably larger cost than nonreusable ones, the AR-based FE would require a nonnegligible additional cost as well as VNC-based FE if it is extended to be reusable.

5.3 Potential attacks and risks

Like conventional FEs, the AR-based FE would be potentially targeted by data manipulation attacks [DGSV15, Del17]. In a data manipulation attack, the attacker repeatedly injects an error into helper data and then checks whether key reconstruction fails. Using the key reconstruction result as an oracle, the attacker can then estimate the decoded seed and PUF response with far less complexity than a brute-force effort in some scenarios. To protect the AR-based FE against such data manipulation attacks, a hash function or signature is useful, as it can detect helper data manipulation. For the AR-based FE, the hash or signature should be generated from the ACL data in addition to the helper data and secret seed/key, because the attacker may obtain the information about the PUF response bit with $d_i = 1$ by replacing it with $d_i = 0$ (and *vice versa*). See [DGSV15, Del17] for a concrete construction of this countermeasure and its security.

The transistor-aging effect is also considered as a potential risk for PUF-based systems, because it changes the bit- (or digit-)error rate. To address this issue, the ECC of AR-based FE should be designed with considering the aging effect, as suggested in several previous studies [Del17]. However, although the transistor-aging effect may change the distribution of the PUF response, it will not have impact on the residual (min-)entropy of the AR-based FE. As the helper data is generated at the enrollment before transistor-aging, the attacker cannot exploit the transistor-aging effect through the observation of helper data, as long as the key reconstruction does not fail.

6 Conclusion

This paper presented an efficient FE construction based on a new debiasing method named AR-based FE, which employs the principle of rejection sampling. The proposed FE can derive a debiased bit (or trit) string from biased PUFs more efficiently than conventional schemes. Additionally, the AR-based FE can be simply implemented with an RNG and bit-parallel logic operations in the enrollment server, and it does not require any additional resource-consuming operation at the reconstruction in the client device with PUFs that performs the reconstruction. The AR-based FE can be extended to a ternary-encoded PUF response by combining the proposed scheme with a ternary FE, which can yield a further improvement.

The performances of the proposed FEs were analyzed in comparison with the conventional FEs through experimental 128-bit key generation from PUFs with various biases and averaged digit-error rates. Consequently, we confirmed that the AR-based FE could achieve the smallest PUF and NVM sizes for a wide range of biases and averaged digit-error rates. For example, the AR-based FE requires at least 55% and 51% smaller PUF size or at least 79% and 53% smaller NVM size than any other conventional counterparts for biases of $p_1 = 0.66$ and $p_1 = 0.90$, respectively. In addition, the ternary AR-based FE has up-to 55% higher efficiency than the binary version, and can achieve up-to 63% and 57% higher efficiency than its conventional counterparts for $\mathbb{E}[e] = 0.025$ and 0.050, respectively. These results indicate that the AR-based FEs can achieve secure key generation with minimal hardware cost. Moreover, we demonstrated an extension of the AR-based FE to the cell-wise bias model. Whereas conventional debiasing schemes cannot generally be applied to the cell-wise biases, the extended AR-based FE can realize a secure key generation from PUFs even for cell-wise biases.

Extension of the AR-based FE to mitigate spatial correlation or biases in high-order moments remains as future work. Additionally, the design and implementation of an actual key generation system featuring the proposed method should be conducted in the future.

Acknowledgements

We are grateful to Dr. Stefan Katzenbeisser for shepherding care. This research has been supported by JSPS KAKENHI Grants No. 17H00729 and 20K19765.

References

[AWSO17] Aysin Aysu, Ye Wang, Patrick Schaumont, and Michael Orshansky. A new maskless debiasing method for lightweight physical unclonable functions. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 134–139. IEEE, 2017.

[BBCS91] Charles H. Bennett, Gilles Brassard, Claude Crṕean, and Marie-Hélène Skubiszewska. Practical quantum oblivious transfer. In *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Seience*, pages 351–366, 1991.

[BGS$^+$08] Christoph Bösch, Jorge Guajardo, Ahmad-Reza Sadeghi, Jamshid Shokrollahi, and Pim Tuyls. Efficient helper data key extractor on FPGAs. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, volume 5154 of *Lecture Notes in Computer Science*, pages 181–197. Springer, 2008.

[Boy04] Xavier Boyen. Reusable cryptographic fuzzy extractors. In *ACM conference on Computer and Communication Security (ACM-CCS)*, pages 82–91. ACM, 2004.

[CFP$^+$16] Ran Canetti, Benjamin Fuller, Omer Paneth, Leonid Reyzin, and Adam Smith. Reusable fuzzy extractors for low-entropy distributions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 117–146, 2016.

[Del17] Jeroen Delvaux. *Security Analysis of PUF-Based Key Generation and Entity Authentication.* PhD thesis, KU Leuven, 2017.

[DGSV15] Jeroen Delvuax, Dawn Gu, Dries Schellekens, and Ingrid Verbauwhede. Helper data algorithms for PUF-based key generation: Overview and analysis. *IEEE Transactions on Computer-Aided Design of Integrated and Systems*, 34(6):889–902, 2015.

[DGV$^+$16] Jeroen Delvaux, Dawu Gu, Ingrid Verbauwhede, Matthias Hiller, and Meng-Day Yu. Efficient fuzzy extraction of PUF-induced secrets: Theory and applications. In *International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, volume 9813 of *Lecture Notes in Computer Science*, pages 412–431. Springer, 2016.

[GKST07] Jorge Guajardo, Sandeep S. Kumar, Greet-Jan Schrijen, and Pim Tuyls. FPGA intrisic PUFs and their use for IP protection. In *International Workshop* *on Cryptographic Hardware and Embedded Systems (CHES)*, volume 4727 of *Lecture Notes in Computer Science*, pages 63–80. Springer, 2007.

[HBF08] Daniel E. Holcomb, Wayne P. Burleson, and Kevin Fu. Power-up SRAM state as an identifying fingreprint and source of true random numbers. *IEEE Transactions on Computers*, 58(9):1198–1210, 2008.

[HMSS12] Matthias Hiller, Dominik Merli, Frederic Stumpf, and Georg Sigl. Complementary IBS: Application specific error correction for PUFs. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 1–6. IEEE, 2012.

[HO17] Matthias Hiller and Aysun Gurur Önalan. Hiding secrecy leakage in leaky helper data. In *International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, volume 10529 of *Lecture Notes in Computer Science*, pages 601–619. Springer, 2017.

[HYS16] Matthias Hiller, Meng-Day Yu, and Georg Sigl. Cherry-picking reliable PUF bits with differential sequence coding. *IEEE Transactions on Information Forensics and Security*, 11(9):2065–2076, 2016.

[KLRW14] Patrick Koeberl, Jiangtao Li, Anand Rajan, and Wei Wu. Entropy loss in PUF-based key generation schemes: The repetition code pitfall. In *Hardware-Oriented Security and Trust (HOST)*, pages 44–49. IEEE, 2014.

[KUH19] Kohei Karumori, Rei Ueno, and Naofumi Homma. A ternary fuzzy extractor for efficient cryptographic key generation. In *49th International Symposium on Multiple-Valued Logic (ISMVL 2019)*, pages 49–54. IEEE, 2019.

[LPS12] Vincent van der Leest, Bart Preneel, and Erik van der Sluis. Soft decision error correction for compact memory-based PUFs using a single enrollment. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, volume 7428 of *Lecture Notes in Computer Science*, pages 268–282. Springer, 2012.

[Mae03] Roel Maes. *Physically Unclonable Functions—Constructions, Properties, and Applications.* Lecture Notes in Computer Science. Springer-Verlag Berlin Heidelberg, 2003.

[Mae13] Roel Maes. An accurate probabilistic reliability model for scilicon PUFs. In *Cryptographic Hardware and Embedded Systems (CHES)*, volume 8086 of *Lecture Notes in Computer Science*. Springer, 2013.

[MHV12] Roel Maes, Anthony Van Herrewege, and Ingrid Verbauwhede. PUFKY: A fully functional PUF-based cryptographic key generator. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, volume 7428 of *Lecture Notes in Computer Science*, pages 302–319, 2012.

[MLSW15] Roel Maes, Vincent van der Leest, Erik van der Sluis, and Frans Willems. Secure key generation form biased PUFs. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, volume 9293 of *Lecture Notes in Computer Science*, pages 517–534. Springer, 2015.

[MTV09] Roel Maes, Pim Tuyls, and Ingrid Verbauwhede. Low-overhead implementation of a soft decision helper data algorithm for SRAM PUFs. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, volume 5747 of *Lecture Notes in Computer Science*, pages 332–347. Springer, 2009.

[SD07] G. Edward Suh and Srinivas Devadas. Physical unclonable functions for device authentication and secret key generation. In *ACM/IEEE Design Automation Conference*, pages 9–14, 2007.

[SHO07] Y. Su, J. Holleman, and B. Otis. A 1.6pJ/bit 96% stable chip-ID generating circuit using process variations. In *IEEE International Solid-State Circuits Conference (ISSCC). Digest of Techical Papers*, pages 406–407. IEEE, 2007.

[SUHA17] Manami Suzuki, Rei Ueno, Naofumi Homma, and Takafumi Aoki. Multiple-valued debiasing for physically unclonable functions and its application to fuzzy extractors. In *International Workshop on Constructive Side-Channel Analysis and Secure Design (COSADE)*, volume 10348 of *Lecture Notes in Computer Science*, pages 248–263. Springer, 2017.

[SUHA18] Manami Suzuki, Rei Ueno, Naofumi Homma, and Takafumi Aoki. Efficient fuzzy extractors based on ternary debiasing method for biased physically unclonable functions. *IEEE Transactions on Circuits and Systems I*, 66(2):616–629, 2018.

[USH19] Rei Ueno, Manami Suzuki, and Naofumi Homma. Tackling biased PUFs through biased masking: A debiasing method for efficient fuzzy extractor. *IEEE Transactions on Computers*, 68(7):1091–1104, 2019.

[vN51] John von Neumann. Various techniques used in connection with random digits. *Applied Math Series, Notes by G.E. Forsythe, in National Bureau of Standards*, 12:36–38, 1951.

[Š17] Boris Škorić. A trivial debiasing scheme for helper data systems. *Journal of Cryptographic Engineering*, 8:341–349, 2017.

[WGP18] Florian Wilde, Berndt M. Gammel, and Michael Pehl. Spatial correlation analysis on physical unclonable functions. *IEEE Transactions on Information Forensics and Security*, 13:1468–1480, 2018.

[Wyn75] A. D. Wyner. The wire-tap channel. *Bell Labs Technical Journal*, 54(8):1355–1387, 1975.

[YD10] Meng-Day Yu and Srinivas Devadas. Secure and robust error correction for physical unclonable functions. *IEEE Design & Test of Computers*, 27(1):48–65, 2010.

[YSI$^+$11] Dai Yamamoto, Kazuo Sakiyama, Mitsugu Iwamato, Kazuo Ohta, Takao Ochiai, Masahiko Takenaka, and Kouichi Itoh. Uniqueness enhancement of PUF responses based on the locations of random outputting RS latches. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, volume 6917 of *Lecture Notes in Computer Science*, pages 390–406. Springer, 2011.

[YSI$^+$13] Dai Yamamoto, Kazuo Sakiyama, Mitsugu Iwamoto, Kazuo Ohta, Masahiko Takenaka, and Kouichi Itoh. Variety enhancement of PUF responce using the locations of random outputting RS latches. *Journal of Cryptographic Engineering*, 3(4):197–211, 2013.

[YSI$^+$15] Dai Yamamoto, Kazuo Sakiyama, Mitsugu Iwamoto, Kazuo Ohta, Masahiko Takenaka, Kouichi Itoh, and Naoya Torii. A new method for enhancing variety and maintaining reliability of PUF responses and its evaluation on ASICs. *Journal of Cryptographic Engineering*, 5(2):187–199, 2015.

Appendix

A. Comparison with CC-based FE

The AR-based FE is compared with the CC-based FEs [HO17]. Note, however, that it is difficult to derive the residual min-entropy of a CC-based FEs with a BCH code owing to the lack of description and publicly available data. Therefore, we evaluate the AR- and CC-based FEs with a Reed-Muller (RM) code used in the evaluation of [HO17] as in [USH19]. Table 10 lists the required PUF size, NVM size, and reconstruction failure rate required for key generation using a 128-bit random seed for both the AR- and CC-based FEs with a concatenated code of outer [64, 22, 16] RM and inner repetition codes. Here, "CC bits" indicates the number of bits discarded to mitigate the entropy loss of the CC-based FE. Here, we retrieved the values of the CC-based FE from [USH19], whereas those for the AR-based FE are calculated according to the formulas presented in Section 3. From Tab. 10, we can confirm that the AR-based FE can achieve a smaller PUF size than the CC-based FE under various conditions. If the PUF bias p_1 is large, it is necessary for the CC-based FEs to discard many bits of random seed to mitigate the entropy loss. Moreover, the CC-based FEs cannot mitigate the entropy leak sufficiently for very large biases, as apparent from the entropy leakage results in the table. Note that efficient combination of CC-based debiasing with an FE with DSC [HYS16] was shown by Hiller and Önalan [HO17]; the efficiency was demonstrated through application to an SRAM PUF with $p_1 = 0.54$ and $\mathbb{E}[e] = 0.10$. However, the efficiency of this combination for different biases and averaged BERs is unclear, and we cannot evaluate CC-based FEs with DSC because there is insufficient information on their evaluation. Note that, in this work, we focused on FEs with code-offsets as we aim to fairly compare the debiasing methods. Note also that the AR-based FE would work with various FEs for PUFs, including those featuring DSC.

B. Comparison with TD-based FE

Table 11 lists the evaluation results of the AR- and TD-based FEs [Š17] for 128-bit key generation. To conduct a fair comparison and to clarify the advantages of proposed debiasing scheme, we introduce some assumptions for the evaluation of TD-based FE. That is, we hypothetically apply TD to an FE with code-offset construction, and design a repetition-BCH concatenated code according to averaged BER such that the PUF size is minimized under the condition that the expected reconstruction failure probability is smaller than 10^{-6}. Therefore, each of ECC, number of blocks, residual min-entropy, and the expected reconstruction failure probability are identical for the AR- and TD-based FEs. Subsequently, we determine the required PUF size according to Eq. (16) for the AR-based FE such that the expected enrollment failure probability is smaller than 10^{-6}, and hypothetically determine the same PUF size for the TD-based FE as the AR-based FE. We also consider the fact that the order of retained entropy of TD is the same as the AR-based debiasing. These assumptions and approaches are adopted because we cannot define the parameters and building blocks for the design of TD-based FE due to the lack of description about how to define/determine them for given PUF bias, BER, and implementation resources. Even under the above assumptions, we can confirm that the AR-based FE has 42–79% smaller NVM size than the TD-based FE. Moreover, note again that the AR-based FE can work with the cell-wise bias model and requires no additional resource-consuming operations except for an RNG. In contrast, the TD-based FE cannot mitigate the cell-wise bias and requires additional cost for implementing DRSG_l, constructing lists, and set manipulation. Thus, we can confirm the advantages of the AR-based FE over the TD-based FE.

Table 10: Simulation results of key generation using 128-bit random seed with AR- and CC-based FEs (Values for CC-based FE are derived from [USH19])

BER $\mathbb{E}[e]$	PUF bias	FE	CC bits	Rep. code	# of blks	Entropy leakage	PUF size	NVM size	Failure prob.
0.050	0.52	CC	4	3	8	$\leq$ 7.69	1,536	1,536	2.35e-08
			5	3	8	$\leq$ 4.96	1,536	1,536	2.35e-08
			6	3	8	$\leq$ 3.35	1,536	1,536	2.35e-08
		This work	-	3	6	0	**1,237**	**2,389**	1.41e-07
	0.56	CC	18	3	32	$\leq$ 5.25	6,144	6,144	2.35e-08
			19	3	43	$\leq$ 5.06	8,256	8,256	2.35e-08
			20	3	64	$\leq$ 4.61	12,288	12,288	2.35e-08
		This work	-	3	6	0	**1,377**	**2,529**	1.41e-07
	0.60	CC	19	3	43	$\leq$ 27.24	8,256	8,256	2.35e-08
			20	3	64	$\leq$ 23.63	12,288	12,288	2.35e-08
			21	3	128	$\leq$ 22.35	24,576	24,576	2.35e-08
		This work	-	3	6	0	**1,536**	**2,688**	1.41e-07
0.100	0.52	CC	4	5	8	$\leq$ 7.69	2,560	2,560	8.33e-08
			5	5	8	$\leq$ 4.96	2,560	2,560	8.33e-08
			6	5	8	$\leq$ 3.35	2,560	2,560	8.33e-08
		This work	-	5	6	0	**2,047**	**3,967**	5.00e-07
	0.56	CC	18	5	32	$\leq$ 5.25	10,240	10,240	8.33e-08
			19	5	43	$\leq$ 5.06	13,760	13,760	8.33e-08
			20	5	64	$\leq$ 4.61	20,480	20,480	8.33e-08
		This work	-	5	6	0	**2,268**	**4,188**	5.00e-07
	0.60	CC	19	5	43	$\leq$ 27.24	13,760	13,760	8.33e-08
			20	5	64	$\leq$ 23.63	20,480	20,480	8.33e-08
			21	5	128	$\leq$ 22.35	40,960	40,960	8.33e-08
		This work	-	5	6	0	**2,522**	**4,442**	5.00e-07
0.150	0.52	CC	4	9	8	$\leq$ 7.69	4,608	4,608	3.37e-09
			5	9	8	$\leq$ 4.96	4,608	4,608	3.37e-09
			6	9	8	$\leq$ 3.35	4,608	4,608	3.37e-09
		This work	-	9	6	0	**3,662**	**7,118**	2.02e-08
	0.56	CC	18	9	32	$\leq$ 5.25	18,432	18,432	3.37e-09
			19	9	43	$\leq$ 5.06	24,768	24,768	3.37e-09
			20	9	64	$\leq$ 4.61	36,864	36,864	3.37e-09
		This work	-	9	6	0	**4,042**	**7,498**	2.02e-08
	0.60	CC	19	9	43	$\leq$ 27.24	24,768	24,768	3.37e-09
			20	9	64	$\leq$ 23.63	36,864	36,864	3.37e-09
			21	9	128	$\leq$ 22.35	73,728	73,728	3.37e-09
		This work	-	9	6	0	**4,482**	**7,938**	2.02e-08
0.200	0.52	CC	4	11	8	$\leq$ 7.69	5,632	5,632	8.43e-07
			5	11	8	$\leq$ 4.96	5,632	5,632	8.43e-07
			6	11	8	$\leq$ 3.35	5,632	5,632	8.43e-07
		This work	-	13	6	0	**5,274**	**10,266**	1.08e-07
	0.56	CC	18	11	32	$\leq$ 5.25	22,528	22,528	8.43e-07
			19	11	43	$\leq$ 5.06	30,272	30,272	8.43e-07
			20	11	64	$\leq$ 4.61	45,056	45,056	8.43e-07
		This work	-	13	6	0	**5,809**	**10,801**	1.08e-07
	0.60	CC	19	11	43	$\leq$ 27.24	30,272	30,272	8.43e-07
			20	11	64	$\leq$ 23.63	45,056	45,056	8.43e-07
			21	11	128	$\leq$ 22.35	90,112	90,112	8.43e-07
		This work	-	13	6	0	**6,433**	**11,425**	1.08e-07

Table 11: Simulation results of 128-bit key generation using AR- and TD-based FEs

BER $\mathbb{E}[e]$	PUF bias	FE	Rep. code	Outer code	# of blks	Res. entr.	PUF size	NVM size	Failure prob.
0.025	0.58	TD	1	$[63, 16, 23]$	8	128.0	656	5,544	3.89e-07
		This work						**1,160**	
	0.66	TD					837	5,544	
		This work						**1,341**	
	0.74	TD					1,122	6,048	
		This work						**1, 626**	
	0.82	TD					1,654	6,048	
		This work						**2,158**	
	0.90	TD					3,030	6,552	
		This work						**3,534**	
0.050	0.58	TD	1	$[127, 22, 47]$	6	132.0	975	8,382	1.13e-07
		This work						**1,737**	
	0.66	TD					1,237	9,144	
		This work						**1,999**	
	0.74	TD					1,651	9,144	
		This work						**2,413**	
	0.82	TD					2,425	9,906	
		This work						**3,187**	
	0.90	TD					4,430	10,668	
		This work						**5,192**	
0.075	0.58	TD	1	$[255, 45, 87]$	3	135.0	978	8,415	6.18e-07
		This work						**1,743**	
	0.66	TD					1,241	9,180	
		This work						**2,006**	
	0.74	TD					1,657	9,180	
		This work						**2,422**	
	0.82	TD					2,434	9,945	
		This work						**3,199**	
	0.90	TD					4,446	10,710	
		This work						**5,211**	
0.100	0.58	TD	5	$[127, 64, 21]$	2	128.0	1,597	15,240	3.24e-08
		This work						**2,867**	
	0.66	TD					2,016	15,240	
		This work						**3,286**	
	0.74	TD					2,678	16,510	
		This work						**3,948**	
	0.82	TD					3,921	16,510	
		This work						**5,191**	
	0.90	TD					7,140	17,780	
		This work						**8,410**	