Efficient Helper Data Reduction in SRAM PUFs via Lossy Compression

Ye Wang and Michael Orshansky

Department of Electrical and Computer Engineering
The University of Texas at Austin, Austin, TX, USA, 78712
{lhywang, orshansky}@utexas.edu

ABSTRACT

Fuzzy extractors used in PUF-based key generation require storage of helper data in non-volatile memory (NVM). The challenge of using SRAM PUF-based key generation on FPGAs is that high-capacity NVM, such as Flash, is not available on chip. Only expensive one-time-programmable (OTP) memory with limited capacity, such as e-fuses, can be utilized to store helper data.

Our work allows a significant reduction of helper data size (HDS) through two innovative techniques. The first uses bit-error-rate (BER)-aware lossy compression: by treating a fraction of reliable bits as unreliable, it effectively reduces the size of the reliability mask. Considering practical costs of error characterization, the second technique permits across-temperature HDS minimization strategies based on bit-selection (with or without subsequent compression) using room-temperature only characterization. The method is based on stochastic concentration theory and allows efficiently forming confidence intervals for true worst-case BER. We use it to enable lossy compression and key reconstruction with success arbitrarily close to certainty.

Results show that compared to maskless alternative, the proposed algorithm achieves an up to $4.5X$ HDS reduction with only 60% raw bits. Compared to lossless compression, we achieve a further 25% total HDS reduction, at the cost of doubling the number of raw PUF bits, for a 128-bit key. When bit-specific across-temperature characterization is not possible, our method achieves a significant $2.4X$ helper data reduction compared to the maskless alternative for extracting a 128-bit key and a $3X$ reduction for a 256-bit key.

1. INTRODUCTION

Silicon physical unclonable functions (PUFs) are widely used in emerging hardware security applications such as device identification, authentication, and cryptographic key generation [20]. Among existing PUFs [20, 14, 17, 15], SRAM PUFs that produce unique signatures with power-up states are appealing because SRAMs are widely available standard components in a multitude of existing ICs.

However, the major challenge of utilizing silicon PUFs, including SRAM PUFs, lies in the reliability of responses. The standard method to ensure PUF reliability in the key derivation context is to use a fuzzy extractor [10] that utilizes a linear error correction code. The fuzzy extractor algorithm encodes raw PUF responses and compute the code syndrome, to be used as helper data for reconstruction. The helper data is assumed to be public and can be stored in non-volatile memory (NVM) [6].

For FPGA platforms, however, the area- and cost-efficient NVM, such as Flash, is typically not available on chip. For example, most Xilinx and Altera FPGAs and SoCs do not provide on-chip Flash. Off-chip Flash is not always an acceptable solution when security applications are managed by the original semiconductor manufacturers rather than their system customers. In these settings, the helper data need to be stored on-die, much before the board-level details are defined. While these limitations do not extend to all uses of PUFs, they do represent an important class of PUF-driven security protocols. Further, re-writable Flash presents the risk of helper data manipulation attacks [8]. Thus, for semiconductor companies interested in PUF-based keys, the only practical NVM for helper data storage is the one-time-programmable (OTP) memory, such as e-fuses. The problem is that e-fuses are expensive in terms of silicon area and thus their bit-capacity is severely limited, and is significantly below that of Flash. Most chips may have only hundreds to a few kilo-bits of e-fuse. The goal of this paper is to allow a significant reduction in the amount of required helper data, thus reducing the cost of PUF-key deployment dramatically.

A natural tool for reducing HDS for code syndrome is dark-bit masking: identifying bits with high BER and avoid their use [1, 13, 21]. However, that requires representing and storing the reliability bitmap (mask) whose size needs to also be treated as helper data. Thus, we distinguish helper data due to (1) the syndrome and (2) the reliability mask. Lossless compression techniques have been proposed to reduce HDS for both code syndrome [12] and reliability mask [13]. The "1-out-of-n" method [20] is able to decrease the overall BER and thus HDS by sacrificing a large number of raw bits. This can be considered as an early heuristic of lossy compression but is far from optimal in terms of HDS, as we show later.

This paper proposes two innovative techniques based on lossy compression to reduce the total HDS dramatically and demonstrate the optimality. We show how sacrificing a certain fraction of reliable bits, by treating them as unreliable, leads to effective compression and large HDS reduction. First, the paper focuses on HDS minimization when complete bit-specific error information is available across a range of operating conditions. To achieve that, we develop a two-phase joint BER minimization and reliability mask compression algorithm. A derived codebook is used to select a better solution that minimizes the overall average error of selected bits. This is done via the formulation using integer linear programming problem (ILP). By utilizing the Lagrangian relaxation heuristics, the proposed ILP formulation can be decomposed into small separable sub-problems with efficient greedy-based linear-time solutions. Moreover, the proposed lossy compression algorithm

978-3-9819263-0-9/DATE18/©2018 EDAA

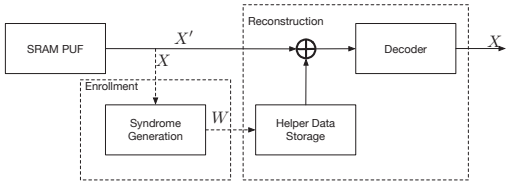

Figure 1: Architecture of a syndrome-based fuzzy extractor.

does not leak any information about the key material.

The benefits of the technique are significant. Consider deriving a 128-bit key with failure rate 10^{-6} from SRAM cells with an average BER of 10.22%. A direct implementation of fuzzy extractor requires 1060 raw source bits and 932 OTP bits for helper data. A bit-selection scheme with lossless compression [13] reduces the helper data to 288 bits. After the proposed lossy compression, only 206 bits of HDS are needed, which is a 25% reduction, though at the cost of doubling the number of raw bits.

However, the approach above assumes a possibly expensive BER characterization effort in which the reliability information under all possible conditions for each bit of each production PUF chip is extracted to key reconstruction failure rate. Worst-case BER characterization across all working conditions may be unacceptably costly in a production environment.

The second technique extends lossy HDS minimization to across-temperature operation via stochastic modeling of worst-case BERs. This is enabled by our key observation that across-temperature (worst-case) BERs are correlated with their nominal-temperature one-probabilities, for stable cells. We propose to perform a one-time large sample characterization experiment, predict individual worst-case BERs based on nominal one-probabilities, and drive the HDS reduction algorithm with the estimate of worst-case BERs. By utilizing a stochastic concentration inequality, we show that for SRAM cells with sufficient size, the true average worst-case BER can be accurately estimated. The accuracy of estimation enables the same reliability guarantee of key reconstruction with overwhelmingly high probability.

The result is dramatic: compared to the maskless alternative, our method still results in an overall $2.3X$ helper data reduction for extracting a 128-bit key and a $3X$ reduction for a 256-bit key, while allowing key reconstruction to fail in only 10^{-5} cases.

2. HELPER DATA SIZE IN PUF-BASED KEY EXTRACTORS

In this section, we briefly review existing helper data reduction algorithms in the context of PUF-based key generation. Figure 1 shows a conventional PUF-based key generation flow. In the interest of space, we skip the details about standard framework of key generation using fuzzy extractor, which are contained in [10], but state only the assumptions, models, standard techniques and conclusions we adopt and use in this paper. In the following discussion, vectors and matrices are denoted with bold symbols with dimension on superscript, which can be dropped for convenience in case of no confusion.

First, we adopt the assumption that individual reliabilities are heterogeneous but independent since SRAM cells generate randomness through independent local mismatches [19, 18]. In this model, cell i is defined with the one-probability $p_i(T, V_{dd})$ as the probability of returning a 1 from one random power-up under temperature T and supply voltage V_{dd}. The enrollment responses X_i's are derived via majority voting under nominal voltage and temperature [18]. Moreover, since SRAM PUFs generate randomness from unbiased local device mismatches, X_i's are usually unbiased, as observed by [16] and validated in Section 5. We take this assumption throughout this paper. Then BER $p_{ei}(T, V_{dd})$ can be derived from the one-probability $p_i(T, V_{dd})$ and enrollment response X_i accordingly. To guarantee key reconstruction under all possible environmental conditions, we need to ensure that even with worst-case BERs p_{wi}, defined by

$$p_{wi} = \max_{T \in [T_{min}, T_{max}], V_{dd} \in [V_{min}, V_{max}]} p_{ei}(T, V_{dd}),$$

the key is reproduced reliably. Silicon experiments [4] show that, for SRAM PUFs, the impact of voltage variation is very small compared to that of temperature. Therefore, we focus only on the temperature dependence in this paper. Individual $p_{ei}(T, V_{dd})$'s, X_i's and p_{wi}'s can be extracted via multiple power-up measurements. See [18] for more details on derivation and extraction of reliability information.

Second, we apply concatenated coding with repetition codes as inner code and (shortened) BCH codes as outer code in the implementation of fuzzy extractors. As mentioned in [2], concatenated codes are typically more efficient than single codes in terms of code length and HDS. When using a linear code $\mathcal{C}(n, k, t)$, the code syndrome as helper data $\boldsymbol{W}$ has a length of $(n-k)$. Introduction of helper data W inevitably discloses partial information on the source bits $\boldsymbol{X}$. Typical cryptographic applications pose security requirements on $\boldsymbol{X}$ in terms of conditional min-entropy $H_\infty(\boldsymbol{X}|\boldsymbol{W})$. Although a tighter bound is available [7], we adopt the widely-used conservative left-over entropy bound given by [10]:

$$H_\infty(\boldsymbol{X}|\boldsymbol{W}) \geq H_\infty(\boldsymbol{X}) - (n-k),$$

in order to compare with other existing HDS reduction schemes.

Third, we adopt the binomial approximation of $\mathrm{HD}(X, X')$, the error count between enrollment and reconstruction. For PUF responses with heterogeneous BERs $(p_{e1}, p_{e2}, \ldots, p_{en})$, Hamming distance $\mathrm{HD}(X, X')$ follows a Poisson binomial distribution. Since the number of raw bits n is sufficiently large in practice, a binomial distribution with effective $\bar{p}_e = \sum_{i=1}^{n} p_{ei}/n$ serves as a good approximation of the corresponding Poisson binomial distribution [6]. The key is required to be reconstructed at a given failure probability, P_{Fail}, for example, 10^{-6}. For a given error correction code (concatenated or single), an equivalent average BER p_t can be found to guarantee reconstruction with specified probability P_{Fail}. We call p_t the target probability of the code. Note that as long as $\bar{p}_e \leq p_t$, the key will be reconstructed with the required success probability $1 - P_{\mathrm{Fail}}$.

The rapid increase in HDS $(n-k)$ required for code syndrome as a function of BER is illustrated in Figure 2 for the case of a 128-bit key.

One way to reduce HDS for code syndrome is dark-bit masking [13]: lowering BER p_e by masking (avoiding) highly unreliable bits. The simplest masking strategy is to define a BER threshold p_{th} and label cells with $p_{ei} \leq p_{th}$ as reliable to obtain a reliability mask $\boldsymbol{M}$. Denote the fraction of reliable bits for a given p_{th} as C. Notice that C is also the probability of one cell being labelled as reliable under the assumption of independent SRAM cells. Then, the expected HDS of the resulting mask $\boldsymbol{M}$ is n/C.

One way to reduce HDS for $\boldsymbol{M}$ is to use standard lossless compression algorithms, such as run-length encoding (RLE)

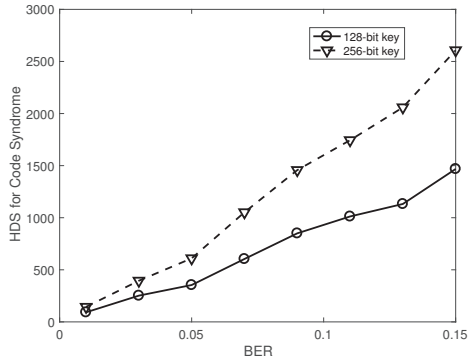

Figure 2: HDS to store BCH code syndrome for different BER.

[12] and Huffman encoding [5], to compress the binary bitmap. The optimal HDS is $nH(C)/C$, where $H(\cdot)$ is Shannon entropy. However, this scheme does not help when C is close to 0.5: the compressed mask takes a size of $2n$, the same as dark-bit masking.

In contrast to the global BER threshold, [20] proposes to divide PUF responses into length-n segments and record the indices of the most reliable cell in each segment. Then each selected reliable bit requires $\log_2(n)$ bit for masking. As we mentioned above, this can be considered as an early heuristic of lossy compression on reliability masks, since it sacrifices $n-1$ raw bits for the overall HDS reduction. But as we demonstrated in Section 5, this scheme is far away from optimal.

3. LOSSY COMPRESSION FOR HELPER DATA REDUCTION

In this section, we describe the joint HDS reduction and BER minimization algorithm. **Our essential idea is a lossy compression: extra reliable bits can be traded for reduced HDS on reliability masks.**

The proposed algorithm, shown in Algorithm 1, proceeds by partitioning a sequence of SRAM cells with size n into k segments, each of length $l = n/k$. It consists of two phases: (1) a codebook construction step enabling compact expression of reliability masks, and (2) an error minimization step that maps the reliability mask of each segment of SRAM cells to one codeword from the codebook.

Our lossy compression algorithm defines a lossy ratio D ($0 < D < 1$) as the target ratio of further discarded reliable bits. For example, $C = 0.5$ and $D = 0.5$ means that 50% raw bits are considered reliable and among all reliable bits 50% of them are discarded, namely, 25% of the original raw bits are finally utilized.

The codebook $\mathcal{C}$ is designed to contains an all-zero vector $\mathbf{0}^l$ and $(2^{lR(D)} - 1)$ random sampled vectors of length l, in which $R(D)$ is defined as:

$$R(D) = H(C) - (1 - C + CD)H(\frac{1-C}{1-C+CD}).$$

$lR(D)$ bits are able to represent a codeword via indexing since the codebook $\mathcal{C}$ has the cardinality of $|\mathcal{C}| = 2^{lRD}$. Therefore, significant reduction of HDS for reliability mask requires that $R(D) \ll 1$. Each coordinate $\mathcal{C}[j]_k$ of each non-zero codeword $\mathcal{C}[j]$ follows an i.i.d. binary distribution that

$$\Pr(\mathcal{C}[j]_k = 1) = 1 - C + CD.$$

Similar to masks constructed via a global BER threshold in Section 2, a "1" in a codeword means that the corresponding SRAM cell is masked as reliable and selected for the key generation.

Algorithm 1 Lossy compression for HDS reduction

```
procedure LossyCompress(p_w, M, n, N, l, C, D)
    Sample codebook C;
    L ← 0, R ← 1, n_λ ← 0;
    while (n_λ ≠ n) do
        λ ← (L + R)/2;
        n_λ ← 0
        for (i = 1; i ≤ N/l; i++) do
            for (j = 1; j = 2^{lR(D)}; j++) do
                S_ij = Σ_{k=1}^{l} p_{w,ik} C[j]_k;
            end for
            j* = arg min_t((S_it − λ HW(C[t])));
            M[i] ← C[j*];
            n_λ ← n_λ + HW(C[j*]);
        end for
        if (n_λ < n) then
            L ← λ;
        end if
        if (n_λ > n) then
            R ← λ;
        end if
    end while
end procedure
```

After the codebook construction, the second phase takes individual BERs as input and pursues an optimal mapping of reliability masks for each segment of SRAM cells, in order to achieve the overall BER minimization. We define the binary decision variables x_{ij}'s, $\forall i \in [k], j \in [|\mathcal{C}|]$ to indicate the encoding of the mask $\boldsymbol{M}[i]$ from the i-th segment of SRAM cells:

$$x_{ij} = 1, \text{ iff } \mathrm{Enc}(\boldsymbol{M}[i]) = \mathcal{C}[j].$$

As mentioned above, encoding $\boldsymbol{M}[i]$ with a j-th codeword $\mathcal{C}[j]$ provides $\mathrm{HW}(\mathcal{C}[j])$ reliable output bits with the corresponding sum of BERs S_{ij} as

$$S_{ij} = \sum_{k=1}^{l} p_{w,ik}\mathcal{C}[j]_k,$$

in which $p_{w,ik}$ denotes the worst-case BER of the k-th source bit in the i-th segment $\boldsymbol{M}[i]$.

The optimization seeking the optimal average BER under the constraint of a sufficient number of reliable output bits is formulated as an integer linear program (ILP):

$$\begin{aligned} \min_{x_{ij}} \quad & \sum_{i,j} S_{ij}x_{ij} \\ \text{s.t.} \quad & \sum_{i,j} \mathrm{HW}(\mathcal{C}[j])x_{ij} \geq n, \\ & \sum_{j} x_{ij} = 1, \ \forall i \in [k] \\ & x_{ij} \in \{0, 1\}. \end{aligned}$$

The objective function $\sum_{i,j} S_{ij}x_{ij}$ effectively represents the average worst-case BER across all reliable bits. To see that, consider that since we are constraining on the total number of reliable bits $\sum_{i,j} \mathrm{HW}(\mathcal{C}[j])x_{ij}$, optimizing over the sum of

BERs for reliable bits is the same as optimizing over the average BER. The constraint $\sum_j x_{ij} = 1$ implies that only one x_{ij} is assigned to 1, namely, each segment $\boldsymbol{M}[i]$ decodes to a unique codeword $\mathcal{C}_j$. For given C, D, l and p_w's, the proposed codebook construction and ILP formulation achieves near-optimal HDS reduction of mask $\boldsymbol{M}$.

A greedy heuristic can be applied to efficiently solve the proposed ILP problem via Lagrangian relaxation. The major challenge of the ILP formulation above lies in the constraint

$$\sum_{i,j} \mathrm{HW}(\mathcal{C}[j]) x_{ij} \geq n,$$

which couples all decision variables x_{ij}'s. The standard technique in constrained optimization problems is to introduce Lagrangian regularizers to move complex constraints onto the objective function. After introducing a non-negative λ, the proposed ILP formation becomes

$$\begin{aligned} \min_{x_{ij}} \quad & \sum_{i,j} S_{ij} x_{ij} - \lambda(\sum_{i,j} \mathrm{HW}(\mathcal{C}[j]) x_{ij} - n) \\ \text{s.t.} \quad & \sum_j x_{ij} = 1, \ \forall i \in [k] \\ & x_{ij} \in \{0,1\}. \end{aligned}$$

We denote $L(\lambda)$ to be the optimal objective value with Lagrangian regularizer λ. By grouping terms with respect to x_{ij}'s, solving for $L(\lambda)$ can be decomposed into small sub-problems $L_i(\lambda)$'s only with decision variables $\{x_{i1}, x_{i2}, \ldots\}$ inside the i-th segment,

$$\begin{aligned} L_i(\lambda) \quad = \quad \min_{x_{ij}} \quad & \sum_j (S_{ij} - \lambda \, \mathrm{HW}(\mathcal{C}[j])) x_{ij} \\ \text{s.t.} \quad & \sum_j x_{ij} = 1, \\ & x_{ij} \in \{0,1\}. \end{aligned}$$

Due to the binary constraints on x_{ij}, the i-th sub-problem $L_i(\lambda)$ identifies the index j with minimum $(S_{ij} - \lambda \, \mathrm{HW}(\mathcal{C}[j]))$:

$$x_{ij} = \begin{cases} 1 & j = \arg\min_t((S_{it} - \lambda \, \mathrm{HW}(\mathcal{C}[t]))), \\ 0 & \text{otherwise.} \end{cases}$$

The optimal choice of λ can be find via efficient binary search in interval $[0, 1]$. Therefore the greedy-based heuristic run in near-linear time.

Finally we formally analyze the information leakage via lossy compression. The mutual information $I(\boldsymbol{X}; \boldsymbol{Z})$ between PUF responses $\boldsymbol{X}$ and the compressed mask $\boldsymbol{Z}$ characterizes the amount of entropy leakage through the compressed mask $\boldsymbol{Z}$. The obtained compressed mask $\boldsymbol{Z}$ can be fully determined by the compression algorithms applied on the reliability information p_w's. $\boldsymbol{X}$ and $\boldsymbol{p_w}$ are independent as long as reliability distributions for cells enrolled to "1" and "0" are indistinguishable. This is true as we shown in Section 5. Therefore $\boldsymbol{Z}$ and $\boldsymbol{X}$ are also independent, which implies a 0 leakage.

4. EFFICIENT METHOD FOR ACROSS TEMPERATURE HDS REDUCTION

The approach in Section 3 leads to an optimal HDS reduction but assumes a possibly expensive characterization effort in which the reliability information for each bit of every production PUF chip is extracted across all conditions.

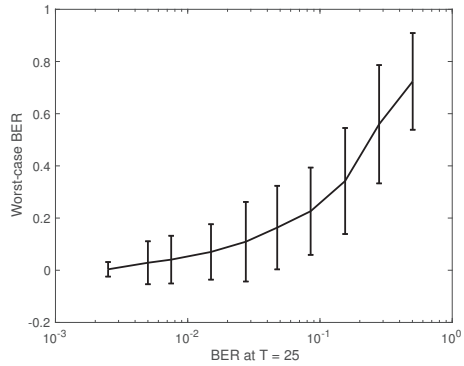

Figure 3: Individual cell's maximum across-temperature BERs plotted against nominal-temperature one-probabilities.

In this section, we present a technique based on stochastic modeling of worst-case BERs to drive the lossy compression via measurements performed only under room temperature, while allowing key reconstruction with overwhelmingly high probability. For every instance with obtained average BER $\bar{p}_w = \sum_{i=1}^{n} p_{wi}/n \leq p_t$, the reconstruction probability guarantee is met, as argued in Section 2. Specifically, we show that for a practical number of cells involved, using the estimate of $\bar{p}_w$, this technique ensures that

$$\Pr(\bar{p}_w \leq p_t) \geq 1 - \delta,$$

in which $\delta \ll 1$ is the fraction of PUF chips for which reliable reconstruction cannot be guaranteed. We show that δ can be set to a value that ensures almost-perfect reconstruction rate, for example, $\delta = 10^{-5}$.

This technique relies on our observation in Figure 3 that **the maximum (worst-case) p_w's over the entire temperature range are correlated with their nominal temperature p's for most reliable bits**. Now we present the flow. (1) First, a population-based analysis of SRAM cells is performed at the technology development phase, to get for each cell the nominal one-probability p_i and worst-case p_{wi} with respect to enrollment under nominal temperature. (2) For cells under each fixed nominal one-probability p, the average μ_p and standard deviation σ_p of worst-case p_w's are extracted. (3) At the production phase, we measure individual one-probabilities p_i's under nominal temperature for each chip. (4) For cells with nominal p, we use μ_p to estimate p_w. Namely, $\{\mu_{p_1}, \mu_{p_2}, \ldots\}$ are sent into the proposed HDS minimization algorithm (described in Section 3), instead of $\{p_{w1}, p_{w2}, \ldots\}$, to select SRAM bits as input to fuzzy extractor. (5) For the set of selected cells, we estimate $\bar{p}_w$ by $\bar{\mu}_p = \sum_{i=1}^{n} \mu_{p_i}/n$. (6) $\Pr(\bar{p}_w \leq p_t)$ is evaluated to indicate whether another iteration of this flow by increasing n is needed.

We now prove the correctness and effectiveness by formally demonstrating that (1) estimating $\bar{p}_w$ by $\bar{\mu}_p$ is accurate for a sufficiently large n. (2) a small offset ν can be efficiently found that guarantees that a PUF with $\bar{\mu}_p \leq p_t - \nu$ will meet its reconstruction guarantee (except for δ fraction of cases).

The first step is to prove that individual p_w's of selected cells are independent. The only assumption required is that the only, or the dominant, source of variation between the individual SRAM cells exhibits no spatial correlation, as we

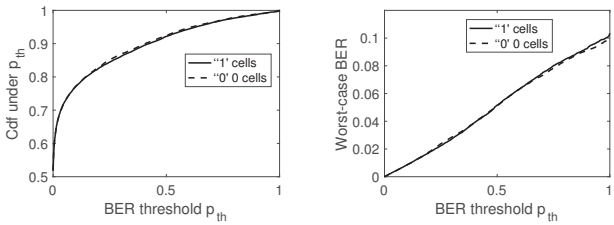

Figure 4: CDF and average BER under given threshold.

Method	# raw bits	ECC		Target BER	HDS		
		Outer	Inner		Mask	Syn	Total
Maskless	1060	[212,128,11]	[5,1,2]	10.22%	0	932	932
1-out-of-n	704	[176,128,6]	N/A	0.25%	352	48	400
Dark-bit	256	[160,128,4]	N/A	0.085%	256	32	288
Loseless	256	[160,128,4]	N/A	0.085%	244	32	276
Lossy	638	[160,128,4]	N/A	0.085%	174	32	206

Table 1: HDS reduction with exact worst-case BERs for a 128-bit key.

mentioned in Section 2.

Then, the independence property allows us to access the quality of estimating $\bar{p}_w$ by $\bar{\mu}_p$ using stochastic concentration theory, and specifically, the Bennett's inequality [3]:

THEOREM 1. *Let $X_1, X_2, \ldots, X_n$ be independent random variables with zero mean, and assume $X_i \leq 1$ almost surely, $\forall i \in [n]$. Let $\sigma^2 = \sum_{i=1}^{n} \mathrm{Var}(X_i)/n$. Then for any $t > 0$,*

$$\Pr(\sum_{i=1}^{n} X_i > t) \leq \exp(-n\sigma^2 h(\frac{t}{n\sigma^2})),$$

in which $h(u) = (1+u)\log(1+u) - u$ for $u > 0$.

In our setting, $X_i = p_{wi} - \mu_{p_i}$ denotes the deviation of p_{wi} from its predicted alternative μ_{p_i}, and $\mathrm{Var}(X_i) = \sigma_{p_i}^2$. The deviation $\bar{p}_w - \bar{\mu}_p$ is just $\sum_{i=1}^{n} X_i$. Take $t = \phi(n)(\sum_{i=1}^{n} \sigma_{p_i})$ and assume an upper bound for all σ_p of selected SRAM cells to be σ,

$$\begin{aligned}\Pr(\bar{p}_w - \bar{\mu}_p > t) &= \Pr(\sum_{i=1}^{n} X_i > \phi(n)(\sum_{i=1}^{n} \sigma_{p_i})) \\ &\leq \exp(-n\sigma^2 h(\frac{\phi(n)}{\sigma})).\end{aligned}$$

Therefore, we show that with increase of n, $\bar{p}_w$ converges to $\bar{\mu}_p$ exponentially fast.

Second, for given n and δ, we show how to derive ν analytically and drive the flow iteratively by increasing n when the reliability guarantee is not met for current bit-selection. Under the proposed condition $\bar{\mu}_p + \nu \leq p_t$, the event $\bar{p}_w > p_t$ implies that $\bar{p}_w - \bar{\mu}_p > \nu$. Namely,

$$\Pr(\bar{p}_w > p_t) \leq \Pr(\bar{p}_w - \bar{\mu}_p > \nu).$$

In order to satisfy the constraint $\Pr(\bar{p}_w > p_t) \leq \delta$, it is sufficient to set $\nu = \phi(n)\sigma$ and get

$$\Pr(\bar{p}_w - \bar{\mu}_p > \nu) \leq \exp(-n\sigma^2 h(\frac{\phi(n)}{\sigma})) \leq \delta.$$

Optimal values of $\phi(n)$ and ν can be determined via the equation above.

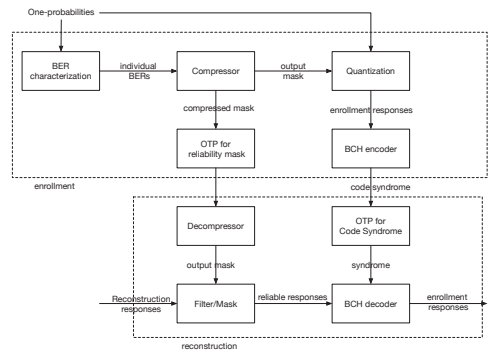

Figure 5: The key generation system with lossy compression.

Method	# raw bits	ECC		Target BER	HDS		
		Outer	Inner		Mask	Syn	Total
Maskless	1260	[252,128,18]	[5,1,2]	13.36%	0	1132	1132
1-out-of-n	1340	[335,128,25]	N/A	2.52%	670	207	877
Dark-bit	560	[308,128,21]	N/A	2.03%	560	180	740
Loseless	560	[308,128,21]	N/A	2.03%	512	180	692
Lossy	1000	[308,128,21]	N/A	2.03%	300	180	480

Table 2: HDS reduction without across-temperature characterization for a 128-bit key.

5. EXPERIMENTAL RESULTS

In this section, we test the performance of the proposed HDS reduction algorithms and compare them against the direct fuzzy extractor application, in which reliability masking is not used, the lossless compression for mask data reduction [13] and the "1-out-of-n" method [20].

We characterize 32768 SRAM cells in the on-chip RAM of the HPS (hard processor system) on an Altera Cyclone V SoC. Power-up values of each SRAM cell are measured via 400 transient simulations under three different temperatures: 0°C, 25°C, and 70°C. 50.23% cells are enrolled to 1 under 25°C, which validates the unbiased assumption in Section 2. Reliability distribution for "1" cells and "0" cells are shown respectively in Figure 4, which supports the zero-leakage argument in Section 3. No obvious spatial correlation is observed.

Table 1 compares the performance of different schemes for the case of a 128-bit key with failure rate 10^{-6}. Based on a fuzzy extractor design targeting the average worst-case BER of 10.22%, the direct method uses concatenated codes with the inner code as repetition code $[5, 1, 2]$ and the outer code as a truncated BCH code $[212, 128, 11]$ (shortened from the BCH code $[255, 171, 11]$). It requires 932 bits of helper data (all syndrome bits). The lossless compression requires 276 bits in total for helper data: 32 bits for the BCH syndrome and 244 bits for the reliability mask. The "1-out-of-n" method taking the most reliable bit in length-4 segments requires even more

Method	# raw bits	ECC		Target BER	HDS		
		Outer	Inner		Mask	Syn	Total
Maskless	2270	[454,256,23]	[5,1,2]	12.52%	0	2014	2014
1-out-of-n	1996	[499,256,29]	N/A	2.14%	998	243	1241
Dark-bit	775	[463,256,25]	N/A	1.75%	775	207	982
Loseless	775	[463,256,25]	N/A	1.75%	750	207	957
Lossy	1500	[463,256,25]	N/A	1.75%	450	207	657

Table 3: HDS reduction without across-temperature characterization for a 256-bit key.

Method	# raw bits	ECC	Target BER	HDS		
				Mask	Syn	Total
Maskless	>4000	N/A	10.22%	0	>4000	>4000
1-out-of-n	984	[246,190,7]	0.25%	492	56	548
Dark-bit	360	[224,184,5]	0.085%	360	40	400
Lossless	360	[224,184,5]	0.085%	342	40	382
Lossy	880	[224,184,5]	0.085%	240	40	280

Table 4: HDS reduction with an entropy rate of 0.75 for a 128-bit key.

	Codebook	Syndrome	Compressor	Decompressor	Decode
# LUT	30	217	501	408	2852
# FF	0	281	300	372	2104

Table 5: Hardware utilization results of system implementation in FPGA for a 128-bit key.

bits than the lossless compression. Finally, the lossy compression method achieves a 30% reduction in mask size compared to lossless compression, at the cost of doubling the number of raw bits. Including the syndrome helper data, the algorithms achieves a 25% reduction in total HDS, requiring 206 bits.

Although unbiasedness is not observed in our silicon measurement, some papers show that min-entropy rate is not 1 [11]. We repeat the experiment above with entropy rate 0.75 and show similar advantages achieved by the proposed scheme in Table 4. Without debiasing, the maskless technique using concatenated BCH codes with repetition codes cannot even find a feasible code for the generation of a 128-bit key using 4000 raw bits. Therefore, lower min-entropy rate does not present a challenge to the application of our scheme.

Individual cell's maximum across-temperature BERs with standard deviations, plotted against their BER at nominal temperature, is shown in Figure 3. Comparison between the maskless and the proposed concentration-based approaches is shown in Table 2 and 3. For extracting a 128-bit key, driving the HDS compression algorithm with measurements only under room temperature identifies 308 cells with average predicted BER 0.36% from 1000 raw bits. Applying the concentration inequality with $\phi(n) = 0.6$ allow us to derive the upper bound of 2.03% for the true average worst-case BER so that almost every instantiated PUF (except for a ratio below 10^{-5}) meets the required key reconstruction failure rate. Compared to the maskless alternative, the error-aware lossy compression still achieves a dramatic $2.3X$ HDS reduction. This advantage is more pronounced when a 256-bit key is required: setting it achieves a $3X$ total HDS reduction.

We also implement a cryptographic key generator with $C = 0.6$ and $D = 0.5$, utilizing the proposed HDS reduction flow on both FPGA and embedded processor platforms. Both systems comprise a BCH-based fuzzy extractor (encoder and decoder), a hardwired codebook, a compressor and a decompressor, illustrated in Figure 5.

For implementation on FPGA, RTL files are synthesized, placed and routed using Xilinx Vivado Design Suite. The designs are run on the Xilinx AC701 evaluation board with an Artix-7 XC7A200T FPGA chip. RTL files of BCH encoder and decoder are adopted from [9]. Hardware utilization results is given in Table 5. For implementation of embedded processors, source codes in C are compiled and run on an Altera NIOS II/s softcore with 1MB on-chip RAM. Cycle counts of different blocks are shown in Table 6. The overall system is efficient since the compression will only be done once at enrollment.

key length	Compressor	Syndrome	Decompressor	Decode
128 bit	38.52M	1.02M	0.07M	0.56M
256 bit	56.68M	1.41M	0.10M	0.58M

Table 6: Cycle count of system implementation in NIOS II.

6. CONCLUSIONS

This work focus on the trade-off between number of raw SRAM PUF bits and helper data size for both syndrome and reliability mask. We start from presenting a BER-aware lossy compression algorithm for HDS reduction when individual BERs across temperatures are available. Taking into account the economic costs of error characterization, we leverage the tool of stochastic modeling of worst-case BERs to extend the proposed HDS reduction algorithm to all working conditions while maintaining the same reliability guarantee except for a tiny fraction of cases.

7. REFERENCES

[1] M. Bhargava and K. Mai. An efficient reliable puf-based cryptographic key generator in 65nm cmos. In *DATE*, 2014.

[2] C. Bösch, J. Guajardo, A.-R. Sadeghi, J. Shokrollahi, and P. Tuyls. Efficient helper data key extractor on fpgas. *CHES*, 2008.

[3] S. Boucheron, G. Lugosi, and P. Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

[4] M. Cortez, A. Dargar, S. Hamdioui, and G.-J. Schrijen. Modeling sram start-up behavior for physical unclonable functions. In *DFT*, 2012.

[5] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[6] J. Delvaux, D. Gu, D. Schellekens, and I. Verbauwhede. Helper data algorithms for puf-based key generation: overview and analysis. *TCAD*, 2015.

[7] J. Delvaux, D. Gu, I. Verbauwhede, M. Hiller, and M.-D. M. Yu. Efficient fuzzy extraction of puf-induced secrets: Theory and applications. In *CHES*, 2016.

[8] J. Delvaux and I. Verbauwhede. Key-recovery attacks on various ro puf constructions via helper data manipulation. In *DATE*, 2014.

[9] R. Dill, A. Shrivastava, and H. Oh. Optimization of multi-channel bch error decoding for common cases. In *CASES*, 2015.

[10] Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *Eurocrypt*, 2004.

[11] J. Guajardo, S. Kumar, G.-J. Schrijen, and P. Tuyls. Fpga intrinsic pufs and their use for ip protection. *CHES*, 2007.

[12] M. Hiller and G. Sigl. Increasing the efficiency of syndrome coding for pufs with helper data compression. In *DATE*, 2014.

[13] M. Hiller, M.-D. Yu, and G. Sigl. Cherry-picking reliable puf bits with differential sequence coding. *IEEE TIFS*, 2016.

[14] D. E. Holcomb, W. P. Burleson, and K. Fu. Power-up sram state as an identifying fingerprint and source of true random numbers. *IEEE TC*, 2009.

[15] M. Kalyanaraman and M. Orshansky. Novel strong puf based on nonlinearity of mosfet subthreshold operation. In *HOST*, 2013.

[16] S. Katzenbeisser, Ü. Kocabaş, V. Rožić, A.-R. Sadeghi, I. Verbauwhede, and C. Wachsmann. Pufs: Myth, fact or busted? a security evaluation of physically unclonable functions (pufs) cast in silicon. *CHES*, 2012.

[17] S. S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls. The butterfly puf protecting ip on every fpga. In *HOST*, 2008.

[18] R. Maes. An accurate probabilistic reliability model for silicon pufs. In *CHES*, 2013.

[19] R. Maes, P. Tuyls, and I. Verbauwhede. Low-overhead implementation of a soft decision helper data algorithm for sram pufs. In *CHES*. 2009.

[20] G. E. Suh and S. Devadas. Physical unclonable functions for device authentication and secret key generation. In *DAC*, 2007.

[21] M.-D. Yu and S. Devadas. Secure and robust error correction for physical unclonable functions. *IEEE Design & Test of Computers*, 2010.