Multi-Valued Physical Unclonable Functions based on Dynamic Random Access Memory

Sven Müelich*
Martin Bossert
Robert F.H. Fischer
sven.mueelich@uni-ulm.de
martin.bossert@uni-ulm.de
robert.fischer@uni-ulm.de
Ulm University, Institute of Communications Engineering
Ulm, Germany

Chirag Sudarshan*
Christian Weis
Norbert Wehn
sudarshan@eit.uni-kl.de
weis@eit.uni-kl.de
wehn@eit.uni-kl.de
Technische Universität Kaiserslautern, Microelectronic Systems Design Research Group
Kaiserslautern, Germany

ABSTRACT

Physical unclonable functions (PUF) are hardware primitives that are very attractive for cryptographic applications to generate and store secure keys. Dynamic random access memory (DRAM) has recently become a promising technology for the construction of PUFs. In this work, multi-valued PUFs (MVPUFs) based on DRAM retention patterns are constructed and assessed for the first time. We propose a holistic solution by considering the complete PUF processing chain consisting of physical PUF source, PUF readout, digitization, channel coding, and the application of a hash function. In contrast to previous DRAM PUFs, a 4-ary symbol instead of a binary one is derived from each PUF cell. This results in an increase of entropy and thus in an improved error correction performance. Unlike previous PUFs based on DRAM retention patterns, we do not require any debiasing algorithms. To the best of our knowledge, this is the first proposal of a multi-valued PUF based on DRAM. Its properties are an average inter-response distance of ≈ 0.48, an average intra-response distance of ≈ 0.04 under stable environmental conditions, and an entropy of up to 0.99 bit.

CCS CONCEPTS

• **Security and privacy → Embedded systems security**.

KEYWORDS

DRAM, physical unclonable functions, multi-valued PUF, key generation, key storage, error correction

ACM Reference Format:
Sven Müelich, Martin Bossert, Robert F.H. Fischer, Chirag Sudarshan, Christian Weis, and Norbert Wehn. 2020. Multi-Valued Physical Unclonable Functions based on Dynamic Random Access Memory. In *The International Symposium on Memory Systems (MEMSYS 2020), September 28-October 1, 2020, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3422575.3422787

*Both authors contributed equally to this research.

MEMSYS 2020, September 28-October 1, 2020, Washington, DC, USA

ACM ISBN 978-1-4503-8899-3/20/09…$15.00
https://doi.org/10.1145/3422575.3422787

1 INTRODUCTION

Physical unclonable functions (PUFs) are hardware primitives, which exploit intrinsically present randomness to generate random but reproducible sequences, referred as PUF responses, that are unique to the device. The randomness in devices with the same circuitries exists due to variations in the manufacturing process and is also present across devices produced by the same manufacturer. Since responses are unique to the device, they are excellent candidates to generate secure cryptographic keys that are random, unique and unpredictable. Often, these properties are not fulfilled from keys generated by *pseudo random number generators (PRNGs)*, e.g., [17]. Usually, cryptographic keys are stored in a non-volatile memory. Even when a protected memory is used, there exist various kinds of attacks to leak the key, e.g., [32]. Also, a protected non-volatile memory requires additional chip area and increases the costs. Since in PUFs, responses can be reproduced on demand, the need for storing a key is eliminated. Instead the key is regenerated, when required by the cryptosystem. However, the reproducibility (one of the key properties) of PUF responses is highly sensitive to variations in environmental factors, such as temperature or supply voltage. Thus, some symbols of responses are erroneously regenerated, while error-free responses are indispensable to derive a key. To guarantee error-free responses, error-correcting codes have to be applied for post-processing of PUF responses. Extensive information about PUFs and their properties can, e.g., be found in [18].

Recently, Dynamic Random Access Memory (DRAM) has been used to generate PUFs. DRAM-based PUFs are a promising alternative to other constructions, since DRAM is intrinsically present in most mobile and embedded devices. Hence, in contrast to other well-known PUF constructions like for example Ring Oscillator PUFs (ROPUFs), no additional circuitry or modifications of the hardware design are required to add PUF functionality to a given device. Previous literature on DRAM-based PUFs (cf. Sec. 2.3) presented various techniques to extract the PUF responses. Most of these prior studies focus only on showcasing unclonability and uniqueness of DRAM-based PUFs. However, critical issues like biased responses

and noise that affects uniformity and reproducibility (cf. Sec. 2.1) are not emphasized. However, the uniformity and reproducibility properties are particularly significant when PUFs are employed for cryptographic key generation.

Biased Responses—During the response extraction process, DRAM-based PUFs classify cells that flip their original value as "0" (so-called weak cells) and cells that retain their original value as "1" (so-called strong cells). The raw bit vectors extracted from DRAM-based PUFs are heavily biased, i.e., they contain either a predominant number of "0s" or "1s". For example, the row hammer based DRAM PUF (cf. Sec. 2.3) presented in [27] has less than 5% of cells that flipped its original value after 120 s of hammer time. Consequently, such a PUF has a low entropy. The secret key generation from such a non-uniform source has a major impact on the entropy of the generated keys and highly relies on the superiority of the employed debiasing/post-processing techniques.

Noise—DRAM cells are highly sensitive to variations in environmental conditions. Hence, some cells do not consistently flip their original values at every response extraction instance. This phenomenon induces an error pattern in the extracted PUF responses. Literature on DRAM-based PUFs shows, that the best-case intra-Jaccard index that represents the stability of the PUF responses is approximately 0.9. Noisy responses cannot directly serve as a reliable cryptographic key. For example, consider a PUF with an average number of bit flips of 32.904 and an intra-Jaccard index of 0.9662 [27]. The absolute number of noisy bit flips for such a PUF is as high as 1.112. Hence, standard literature on PUFs applies a helper data algorithm with an error-correcting code to counteract the noise. However, for PUF responses that suffer under a large bias, the helper data used to generate the key might in turn leak information about the key unless the responses are effectively debiased [15].

In this work, we present a novel *multi-valued PUF (MVPUF)* that extracts a 4-ary symbol from each weak DRAM cell. The proposed MVPUF has three major advantages: first, PUF responses have a negligible bias. Second, extracting a 4-ary symbol from each PUF cell increases the entropy from at most 1 bit to at most 2 bits. Third, it is more robust to noise since the available error correction techniques for a standard non-binary data (i.e., higher-order alphabet) is more powerful. This work models the MVPUF responses as a noisy communication channel and presents the complete PUF processing chain (i.e., from response to key generation) that addresses the aforementioned challenges. Our definition of MVPUF has to be distinguished from some approaches in the literature, which group bits extracted from several PUF cells to one multi-valued symbol, e.g., [33, 34]. Our approach enables better noise handling (cf. Sec. 3.5) and allows the extraction of a longer sequence of symbols from the available PUF cells.

We summarize the new contributions of this paper:

- To the best of our knowledge, this is the first proposal of a multi-valued PUF based on DRAM. The properties of the proposed PUF are an average inter-response distance of $\approx$ 0.48, an average intra-response distance of $\approx$ 0.04 under stable environmental conditions, and an entropy of up to 0.99 bit.
- No extra debiasing step is required.
- We provide a holistic view by proposing a complete PUF processing chain consisting of the following elements: physical PUF source, PUF readout, digitization, channel coding, hash function.

The paper is structured along the components of the PUF processing chain. Ch. 2 describes the DRAM as physical PUF source, which is the true physical unclonable entity that contains intrinsic randomness due to variations in manufacturing processes of physical objects. Ch. 3 explains how the randomness in the physical PUF source is measured and how the digitized PUF readout is generated. In Ch. 4, PUF responses are generated by applying a Gray labeling to the PUF readout. Uniqueness and reproducibility of the resulting PUF responses are evaluated. Ch. 5 deals with error-correction, which is required since PUF responses are influenced by environmental conditions (e.g., temperature), and hence, might be reproduced erroneously. To generate robust keys, the error-free reproduction of PUF responses has to be guaranteed by applying an error-correcting code. Ch. 6 tackles the problem that responses are not uniformly distributed, which is usually solved by hashing the responses to the key-length κ, that is required by the application. Ch. 7 discusses the obtained results and Ch. 8 summarizes related work. Ch. 9 finally concludes the paper. Fig. 1 visualizes the components of the PUF processing chain together with the corresponding input and output data.

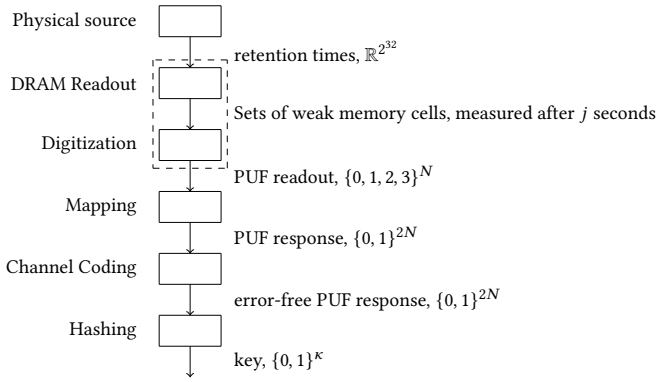


Figure 1: PUF processing chain visualizing all steps required to derive a cryptographic key from the randomness intrinsically present in a physical source.

2 PHYSICAL PUF SOURCE

In this chapter, we first provide basic information about physical unclonable functions (PUFs) and DRAMs.

2.1 Physical Unclonable Functions (PUFs)

Originally, the concept of PUFs has been proposed in [23]. PUFs are hardware primitives, which include randomness that is intrinsically present due to variations in manufacturing processes of physical objects. This randomness can be exploited to derive and store keys for various cryptographic applications. Literature distinguishes two types of PUFs, depending on whether the PUF response is derived

solely from the randomness of the hardware or additionally from an input sequence. In the first case, the PUF response is used to derive a cryptographic key, which is inseparably linked to the hardware. In the second case, authentication is a possible application (cf. Ch. 4.3). In both cases, *initialization* and *reproduction* are distinguished. During initialization, so-called helper data are generated, which are used during reproduction in order to reproduce reliable keys. Essentially, there exist two paradigms to construct PUFs. In delay-based PUFs (e.g., Arbiter PUF [16], Ring-Oscillator PUF [8]), uncontrollable delays in integrated circuits are exploited to extract randomness. In memory-based PUFs (e.g., SRAM PUF [9, 11]), randomness occurs from uncontrollable behavior of memory cells.

The core properties of PUFs are uniqueness, reproducibility, and unclonability. Uniqueness is fulfilled, if PUF responses derived from different PUFs are distinguishable. Reproducibility is fulfilled, if we obtain the same PUF response when evaluating a specific PUF several times. A PUF is called unclonable, if it is not possible for the manufacturer to produce an identical copy of a given PUF, i.e., a PUF from which a predetermined response is generated. Different quality measures like Hamming distance or Jaccard index have been used in literature to evaluate uniqueness and reproducibility. Also the symbols in PUF responses are required to be uniformly distributed. An extensive introduction to PUFs is provided by the standard literature, e.g., [18].

2.2 Dynamic Random Access Memory (DRAM)

Dynamic Random Access Memory (DRAM) devices are organized as a set of banks that includes two-dimensional arrays of memory cells. A memory cell consists of a transistor-capacitor (1T1C) pair. The data is stored as a charge in the capacitor. The cells are connected row-wise via wordlines, and column-wise via bitlines. The wordlines are driven by the row decoder that activates a specific row and the bitlines are connected to sense amplifiers (SAs) that identify the logical value stored in the respective memory cells of an activated row. This process is called row opening (or activation), the read and write operation in DRAM can only be performed on an open row. An open row in a memory array has to be closed by an operation called precharge before opening any new row of that memory array.

2.3 PUFs based on DRAM

Essentially four different approaches to construct PUFs from DRAM have been studied in literature.

(1) **Retention Errors [7]:** DRAM PUFs based on retention errors extract randomness by pausing the refresh operation, which is used to periodically recharge the memory cells. Depending on the refresh-off duration and the DRAM temperature, the cells can be categorized as strong cells and weak cells. Strong cells retain their charge after the pause time, while the weak cells lose their charge. These weak cells are unique for a given device and can be used to generate PUF responses. However, there are a set of cells (referred as noisy cells) that changes its behavior over the time. These cells affect the reproducible behavior of PUF and result in noisy PUF responses.
(2) **Reduced Latency [14]:** The randomness observed by reducing the timing parameters, which are used to ensure the proper operation of DRAM device, reveals device-dependent characteristics that can be utilized for the construction of a PUF. For example, by reducing the time required for row activation process, t_{RCD}, we can distinguish cells whose charge is successfully detected by sense amplifiers (i.e., strong cells), and cells for which the sensing fails (i.e., weak cells).
(3) **DRAM Initial Value [31]:** Motivated by the effects that are exploited in other memory technologies (e.g., SRAM) to construct PUFs, [31] extracts PUF responses based on the initialization value of DRAM cells.
(4) **Row Hammer [27]:** The repeated activation and precharging of a specific row in a DRAM array results in a side affect referred as *Row Hammer*. After a certain number of activation and precharging iterations, the weak cells of the adjacent rows lose their charge and these weak cells are unique to the rows and devices.

Throughout this paper, we use retention errors as the source of randomness for generating PUF responses and cryptographic keys, due to the availability of a retention error measurement platform with precise temperature control. However, the proposed processing chain is as well applicable to other sources like reduced latency errors and row hammer errors. The focus of the paper is not the randomness extraction technique but how the extracted randomness from the DRAM can be used to generate consistent cryptographic keys by using the proposed MVPUF processing chain.

3 DRAM READOUT AND DIGITIZATION

During the IC readout stage, the DRAM cells are evaluated for different refresh-off times. Result is a set of addresses of memory cells, that have been classified as weak for any of the used refresh-off times. From each of these weak cells, a 4-ary symbol is derived in the digitization stage, depending on the refresh-off time, for which the cell was classified as weak for the first time. During initialization, all available DRAM cells are evaluated. During reproduction only the cells that behaved as weak cells during initialization are evaluated.

3.1 Measurement Setup

DRAM retention errors are extracted using an FPGA-based DDR3 DRAM characterization platform. The Xilinx memory controller, i.e., memory interface generator (MIG) is customized to control the refresh operation and its timings. This platform is also equipped with a heating module for precise temperature control of the DDR3 SO-DIMM installed on the FPGA board. The operating temperature can be configured in the range of 25°C–90°C with an accuracy of ±2°C. This setup interacts with a host computer using the PCIe interface. The platform is fully automated using a custom software that is capable of writing a specific pattern of data, maintaining the specified temperature, managing the refresh operation (i.e., On/Off) and data read back. For further details about the measurement platform, we refer to [13]. This work uses a 2 GB DDR3 SO-DIMM for extracting retention error. Each 2 GB DDR3 SO-DIMM consists of four 4 Gb devices of which each device is defined to be a PUF. Hence, our studies in this work are based on a set of 4 PUFs. The retention error for a given pause interval is measured as follows:

First, all cells are charged. Second, the refresh operation is paused for a required interval. Third, the cells are read. The addresses of the cells that flipped it original value for a given refresh pause interval are extracted.

3.2 Available DRAM Data

The sample size of our study comprises four PUFs. Each PUF was measured at four different refresh pause durations, namely at 9 s, 10 s, 11 s, and 12 s. For each refresh pause duration 40 measurements were conducted at 40°C. The total number of measured DRAM cells per PUF is 2^{32}. Throughout this paper, let $k \in \{1, 2, 3, 4\}$ be the PUF number, $j \in \{9, 10, 11, 12\}$ be the refresh pause duration of the measurement in seconds, and $i \in \{0, \ldots, 39\}$ be the number of the measurement.

3.3 Data Analysis

To establish a notation, we define sets that contain addresses of memory cells that are classified as weak cells for certain retention times j.

Definition 1. *a) For a specific PUF, let $\mathcal{A}_{j,i}$ denote the set of addresses of memory cells that was classified as weak during measurement i with a refresh pause duration of j seconds. For example, for a specific PUF, $\mathcal{A}_{9,0}$ denotes the set of weak cells that was determined during measurement $i = 0$ for a refresh pause duration of 9 s.*

b) Let the set

$$\mathcal{A}^{j,i} = \bigcup_{\ell=9}^{j} \mathcal{A}_{\ell,i} \quad (j = 9, \ldots, 12) \tag{1}$$

denote all addresses of memory cells that act as weak cell up to including refresh pause duration j. Note that $\mathcal{A}^{9,i} \subseteq \mathcal{A}^{10,i} \subseteq \mathcal{A}^{11,i} \subseteq \mathcal{A}^{12,i}$. Written in an extensive way, (1) expresses the following relationships:

$$\begin{aligned}
\mathcal{A}^{9,i} &= \mathcal{A}_{9,i} \\
\mathcal{A}^{10,i} &= \mathcal{A}_{9,i} \cup \mathcal{A}_{10,i} \\
\mathcal{A}^{11,i} &= \mathcal{A}_{9,i} \cup \mathcal{A}_{10,i} \cup \mathcal{A}_{11,i} \\
\mathcal{A}^{12,i} &= \mathcal{A}_{9,i} \cup \mathcal{A}_{10,i} \cup \mathcal{A}_{11,i} \cup \mathcal{A}_{12,i}
\end{aligned}$$

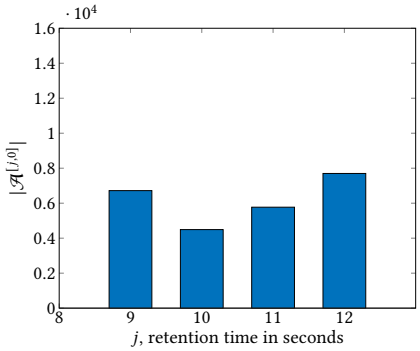
Note that $\mathcal{A}_{j,i} = \mathcal{A}^{j,i}$ due to the subset relationship.

c) Let

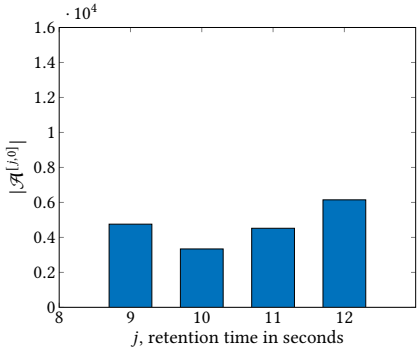
$$\mathcal{A}^{[j,i]} = \mathcal{A}^{j,i} \setminus \mathcal{A}^{j-1,i} \quad (j = 9, \ldots, 12)$$

be the set of addresses of memory cells that were classified as weak for the first time at refresh pause duration j and behaved as strong cell for all refresh pause durations $j' < j$. Note that $\bigcap_{j=9}^{12} \mathcal{A}^{[j,i]} = \emptyset$.

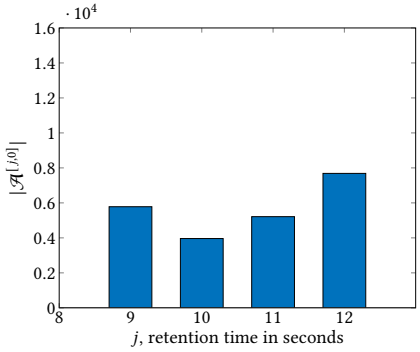
Fig. 2 visualizes, for all four PUFs, the number of weak memory cells that appear for the first time when using refresh pause duration j, where $j = 9, \ldots, 12$. For all four PUFs, measurement $i = 0$ is used for visualization. The sets $\mathcal{A}^{[j,0]}$ are calculated according to Def. 1 c). For $i = 1, \ldots, 39$, the cardinalities of the sets are similar. In the digitization stage (cf. Sec. 3.4), we will derive one symbol from each weak cell, depending on the set $\mathcal{A}^{[j,i]}$ that has this cell as member.



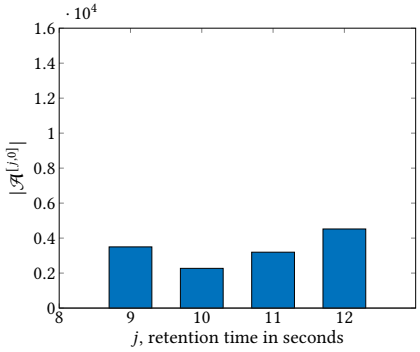
(a) PUF 1



(b) PUF 2



(c) PUF 3



(d) PUF 4

Figure 2: Visualization of the number of memory cells that are classified as weak for the first time at refresh pause duration j and as strong for all refresh pause duration $j' < j$.

Let $\mathcal{A}_{\text{init}}$ be the set of cells that is classified as weak during initialization. Similarly, let $\mathcal{A}_{\text{rep}}$ be the set of cells that is classified as weak during reproduction. When comparing these sets, we observe a large intersection. This is exemplified by the set diagram in Fig. 3. However, there are always cells that are classified as weak during reproduction and as strong during initialization. Those cells are represented by the set $\mathcal{A}_{\text{rep}} \setminus \mathcal{A}_{\text{init}}$. Querying a cell of this set during reproduction yields an insertion error, since we obtain an additional symbol from the source that did not exist during initialization. Insertion errors are avoided by only using cells from the set $\mathcal{A}_{\text{init}}$ during reproduction. Hence, the addresses of cells in $\mathcal{A}_{\text{init}}$ have to be stored. If this is done in ascending order for the union $\bigcup_{j=9}^{12} \mathcal{A}^{[j,i]}$ this does not allow to draw any conclusions from a cell to j without executing a measurement, and hence, security is not decreased. Also, there are always cells that are classified as weak during initialization and as strong during reproduction ($\mathcal{A}_{\text{init}} \setminus \mathcal{A}_{\text{rep}}$). Those cells lead to deletion errors. Such an error is possible to occur only when a cell was initially classified as weak for a retention time of 12 s. Hence, we deal with strong cells that occur during reproduction by categorizing them as weak for $j = 12$. Note that, since the retention time of cells might slightly vary, noise is always present, and hence, some cells are classified different than during initialization.

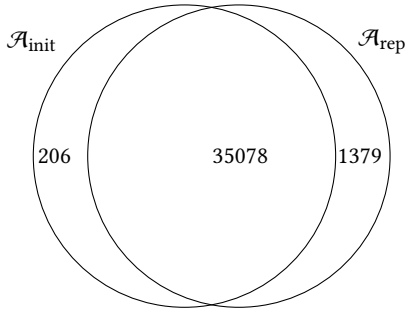


Figure 3: Number of DRAM cells that are classified as weak during initialization ($\mathcal{A}_{\text{init}}$) and during reproduction ($\mathcal{A}_{\text{rep}}$) with exemplary numbers obtained for PUF 1.

3.4 Source

We model the PUF as a 4-ary source Q, where each weak DRAM cell's address corresponds to an output of Q. Let a be the address of a weak DRAM cell. We define the corresponding source symbol u to be

$$u = \begin{cases} 0, \text{ if } a \in \mathcal{A}^{[9]} \\ 1, \text{ if } a \in \mathcal{A}^{[10]} \\ 2, \text{ if } a \in \mathcal{A}^{[11]} \\ 3, \text{ if } a \in \mathcal{A}^{[12]}. \end{cases} \tag{2}$$

For PUF 1, the source symbols occur with the probabilities shown in Table 1, where

$$p = \begin{cases} \frac{|\mathcal{A}^{[9]}|}{|\mathcal{A}^{[9]} \cup \cdots \cup \mathcal{A}^{[12]}|}, \text{ for } u = 0 \\ \frac{|\mathcal{A}^{[10]}|}{|\mathcal{A}^{[9]} \cup \cdots \cup \mathcal{A}^{[12]}|}, \text{ for } u = 1 \\ \frac{|\mathcal{A}^{[11]}|}{|\mathcal{A}^{[9]} \cup \cdots \cup \mathcal{A}^{[12]}|}, \text{ for } u = 2 \\ \frac{|\mathcal{A}^{[12]}|}{|\mathcal{A}^{[9]} \cup \cdots \cup \mathcal{A}^{[12]}|}, \text{ for } u = 3. \end{cases} \tag{3}$$

For the other PUFs the numbers turned out to be similar. The entropy of source Q is $\mathrm{H}(Q) = 1.9426 < \log_2(4) = 2$, which is a relative entropy of

$$\frac{H(Q)}{\log_2(4)} = \frac{1.9426}{2} = 0.9713. \tag{4}$$

Subsequently, we denote a sequence produced by this source as $\boldsymbol{u} \in \{0, 1, 2, 3\}^N$ (where N is the number of considered cells) and call it *PUF readout*.

3.5 PUF Readout Noise Handling

This section highlights the advantages of the proposed MVPUF 4-ary symbol readout in terms of noise handling compared to the 4-ary symbol readout by grouping the binary bits extracted from several PUF cells (as shown in [33, 34]) referred as *group-based 4-ary*.

Let Σ be a finite alphabet of cardinality $|\Sigma| = q$. Further let $c \in \Sigma$ be the transmitted symbol and $y \in \Sigma$ the received symbol. The q-ary symmetric channel is described by the transition probabilities

$$\Pr\{y \mid c\} = \begin{cases} 1 - p \text{ , if } y = c \\ \frac{p}{q-1} \quad \text{, if } y \neq c, \end{cases} \tag{5}$$

where $0 \leq p \leq 1$ is the symbol error probability. For the source with uniformly distributed weak cells and noisy cells, which is the case in DRAM, the group-based 4-ary has channel transitions that resemble a symmetric channel, e.g., the probabilities that a symbol 0 is changed to symbol 1 or 2 or 3 during the regeneration are equal. One or both of the cells that were registered as weak during the enrollment phase may not appear to be weak at the reproduction phase. This means that a given symbol can be transformed into any other symbol in the alphabet due to noise.

Contrary to this, in the proposed extraction any given symbol can at most be transformed to one other symbol in the alphabet. This is because of the DRAM retention behavior. For example, it is unlikely that a cell, that was classified as symbol 0 during enrollment (i.e., belongs to set $\mathcal{A}^{[9]}$), is transformed into symbol 2 (i.e., belongs to set $\mathcal{A}^{[11]}$) during regeneration. Fig. 4 shows the PUF readout behavior modeled as a noisy communication channel. The data analysis shows that such scenarios, i.e., $0 \rightarrow 2$ or $0 \rightarrow 3$ or $1 \rightarrow 3$ are highly unlikely and occur at a very low probability of ≤ 0.002. Likewise, the probability that the symbol, classified as 2 or 3 to be transformed as 1 or 0, respectively, also occurs at a very low probability of $\leq$ 0.001. Consequently, the PUF response in the proposed methodology has half the number of error bits (i.e., worst case) in comparison to group-based 4-ary. This understanding can be leveraged for better error correction.

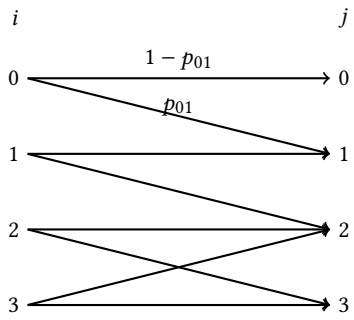


Figure 4: Channel transitions.

Example 1. *Considering the 4-ary PUF readouts*

$$\boldsymbol{u} = 1\,3\,0\,3\,2\,1\,3\,2\,0\,2$$
$$\boldsymbol{u}'_{proposed} = 1\,3\,1\,2\,2\,1\,3\,2\,0\,2$$
$$\boldsymbol{u}'_{group} = 1\,3\,2\,1\,2\,1\,3\,2\,0\,2$$

where $\boldsymbol{u}$ is the arbitrary PUF readout at initialization, $\boldsymbol{u}'_{proposed}$ is proposed readout at reproduction and $\boldsymbol{u}'_{group}$ is the group based readout at reproduction. Applying Gray labeling (cf. Ch. 4) to these symbols, we get the binary sequence

$$\boldsymbol{r} = 01\,10\,00\,10\,11\,01\,10\,11\,00\,11$$
$$\boldsymbol{r}'_{proposed} = 01\,10\,01\,11\,11\,01\,10\,11\,00\,11$$
$$\boldsymbol{r}'_{group} = 01\,10\,11\,01\,11\,01\,10\,11\,00\,11$$

hence, the proposed readout offers better error handling capabilities.

4 DERIVATION OF PUF RESPONSES

The available DRAM data are divided into two disjoint data sets: For each refresh pause duration, we use five out of the 40 measurements to derive the initial response. The other measurements serve as validation group during the reproduction process (cf. Ch. 4.2). Thus, for initialization we consider the sets

$$\mathcal{A}^{[9]} := \bigcap_{i=0}^{4} \mathcal{A}^{[9,i]}, \ \ldots, \ \mathcal{A}^{[12]} := \bigcap_{i=0}^{4} \mathcal{A}^{[12,i]}. \tag{6}$$

We order the addresses of weak DRAM cells in increasing order and randomly choose $N = 1024$ cells that contribute to the PUF readout. We apply Gray labeling to the PUF readouts $\boldsymbol{u} \in \{0,1,2,3\}^N$ and represent each of the four PUF symbols by two bits. The mapping is done as represented in Tab. 1. For comparison, a natural labeling is also given in Tab. 1, which is simply the binary enumeration of the symbols from the 4-ary PUF alphabet.

Table 1: Gray labeling

Symbol u	0	1	2	3
Probability p	0.2545	0.1659	0.2186	0.3609
Gray labeling $\mathcal{S}$	00	01	11	10
Natural labeling	00	01	10	11
Single bit	0	1	0	1

We get two advantages from using the Gray labeling. First, one symbol error, which changes a symbol to one of the two adjacent symbols, only results in one bit error. For example, assume a certain memory address that acts for the first time as weak cell during initialization when using refresh pause duration 9 s. Further assume, that in the reproduction phase this cell acts as weak cell for the first time when using refresh pause duration 10 s. This leads to a change from symbol 1 to symbol 2 in the 4-ary PUF readout. Using the Gray labeling, however, we observe only one bit error (cf. Tab. 1). In contrast, using natural labeling, we would observe two bit errors in some cases, for example when symbol 1 is transformed to symbol 2 during reproduction.

Second, we obtain a distribution of zeros and ones, that is closer to a uniform distribution, since with probability $0.2545 + 0.2186 \approx 0.4732$ we obtain 00 or 11 in the Gray labeled sequence, and with probability $0.1659 + 0.3609 \approx 0.5268$ we obtain 01 or 10.

Consequently the entropy (and relative entropy) of a Gray labeled PUF response is $\mathrm{H} = 0.9979 \leq \log_2(2)$. A Gray labeled sequence will subsequently be denoted with $\boldsymbol{r} \in \{0,1\}^{2N}$ and will be called *PUF response*. We define $n := 2N$. We provide Example 2 to motivate the benefit of extracting symbols from an higher-order alphabet.

Example 2. *Consider the last row of Tab. 1, and assume that we directly extract a binary sequence of length $n = 1024$, where a "0" occurs with probability $p_0 = 0.4732$ and a "1" occurs with probability $p_1 = 0.5268$. This also yields an entropy of 0.9979, but only half as many bits. Recall, that from the physical perspective this means: if during reproduction an address is classified in a neighbored set (e.g., $\mathcal{A}^{[11]}$ instead of $\mathcal{A}^{[12]}$), this results in a bit flip. Using the 4-ary alphabet with Gray labeling only $\frac{1}{2}$ of a block is affected in such a case, since a block has the double length.*

4.1 Uniqueness of PUF Responses

We study the uniqueness of PUFs by considering the relative inter-response distance.

Definition 2. *Let the vectors $\boldsymbol{r}_a = (r_{a,1}, \ldots, r_{a,n})$ and $\boldsymbol{r}_b = (r_{b,1}, \ldots, r_{b,n})$ denote the initial responses of two PUFs a and b. The* relative inter-response distance *of these PUFs is*

$$\mathrm{dist}_{\mathrm{H}}^{\mathrm{inter}} = \frac{1}{n} \cdot |\{\iota : r_{a,\iota} \neq r_{b,\iota}, \iota = 1, \ldots, n\}|.$$

Desired is an average inter-response distance close to 0.5. The average pairwise relative inter-response distances when using Gray labeling and considering all PUFs is 0.48, while we instead obtain an average value of 0.73 when comparing the 4-ary PUF readouts instead. We explain the plausibility of the results by providing a simple example.

Example 3. *Considering the 4-ary PUF readouts*

$$\boldsymbol{u} = 1\,3\,0\,3\,2\,1\,3\,2\,0\,2$$
$$\boldsymbol{u}' = 3\,2\,1\,2\,0\,2\,2\,2\,0\,2$$

we observe a relative Hamming distance of $\frac{7}{10} = 0.7$. Applying Gray labeling on $\boldsymbol{u}$ and $\boldsymbol{u}'$ we get the binary PUF responses

$$\boldsymbol{r} = 01\,10\,00\,10\,11\,01\,10\,11\,00\,11$$
$$\boldsymbol{r}' = 10\,11\,01\,11\,00\,11\,11\,11\,00\,11$$

and hence, a relative Hamming distance of $\frac{9}{20} = 0.45$.

Fig. 5 shows the distribution of the pairwise relative inter-response distance of the 4-ary PUF readout as well as for the Gray labeled PUF response. The latter is close to the optimal value.

As further quality measure of uniqueness, we use the relative Hamming weight.

Definition 3. *Let* $\boldsymbol{r} = (r_1, \ldots, r_n)$ *be the binary response of a certain PUF. The relative Hamming weight of* $\boldsymbol{r}$ *is*

$$\mathrm{wt}_{\mathrm{H}}(\boldsymbol{r}) = \frac{1}{n} \cdot |\{\iota : r_\iota \neq 0, \iota = 1, \ldots, n\}|.$$

The optimum value of the average relative Hamming weight is 0.5. Fig. 6 visualizes the relative Hamming weights for the four PUFs as well as the average value.

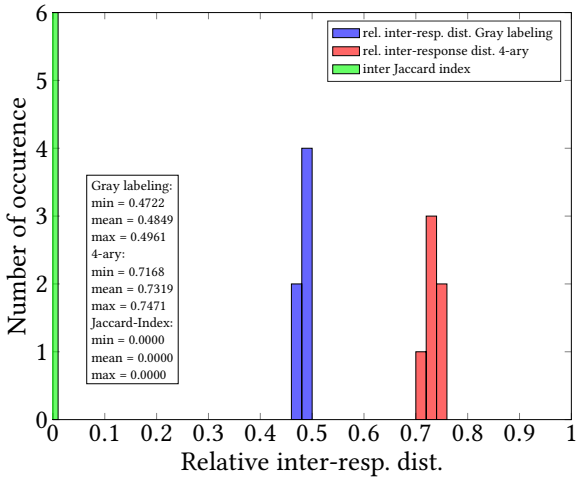


Figure 5: Pairwise relative inter-response distances of the four PUFs.

In contrast to most other PUF constructions, literature on DRAM-based PUFs usually uses the Jaccard index, since this measure is applied to compare sets and memory addresses are represented as sets. For comparability with results from literature on DRAM-based PUFs, we also apply the Jaccard index, in addition the the Hamming distance.

Definition 4. *Let A and B be sets. The* Jaccard index *is defined as*

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}.$$

If we represent the addresses of weak cells of two different PUFs, we result in a Jaccard index of 0, which is the optimal value for the so-called *inter-response Jaccard index* and represents a high level of uniqueness.

4.2 Reproducibility of PUF Responses

In the reproduction phase, for each refresh pause duration $j \in \{9, 10, 11, 12\}$ one measurement is conducted. Then, the sets in

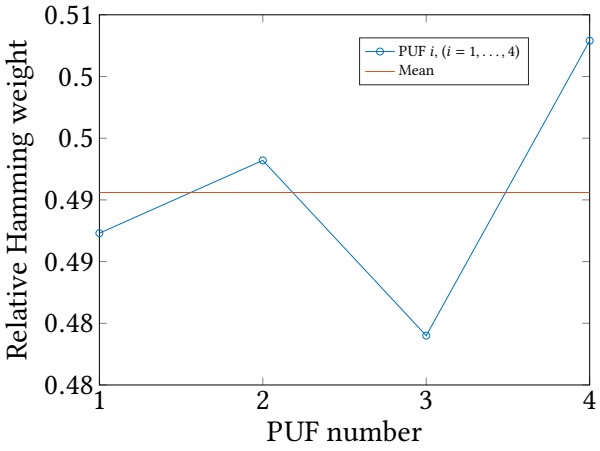


Figure 6: Relative Hamming weights of the four PUFs and the average value.

Def. 1 are generated and responses are derived as explained above. Recall from Ch. 3.3 and Fig. 3 that there exist always memory cells that were classified as weak during initialization, but not during reproduction, i.e., the set $A_{init} \setminus A_{\mathrm{rep}}$. After initialization, we store the set of addresses of cells that were classified as weak cells for any refresh-off time j in ascending order. Since the addresses in $A_{init} \setminus A_{\mathrm{rep}}$ were classified to be weak most often for $j = 12$, we guess that they belong to $\mathcal{A}^{[12]}$. This leads to errors only in rare cases.

We take the 4-ary PUF readouts as well as the Gray labeled PUF responses of the four PUFs and use the validation set to check the quality of reproduced responses. As quality measure for reproducibility, we use the relative intra-response distance.

Definition 5. *Let* $\boldsymbol{r} = (r_1, \ldots, r_n)$ *be the initial response of a certain PUF and* $\boldsymbol{r}'_\mu = (r'_{\mu,1}, \ldots, r'_{\mu,n})$ *be m reproduced responses of the same PUF* ($\mu = 1, \ldots, m$)*. The* relative intra-response distance *of this PUFs is*

$$\mathrm{dist}_{\mathrm{H}}^{\mathrm{intra}} = \frac{1}{m} \sum_{\mu=1}^{m} \left(\frac{1}{n} |\{\iota : r_{\mu,\iota} \neq r'_{\mu,\iota}, \iota = 1, \ldots, n\}| \right).$$

The average relative intra-response distances calculated over all PUFs should be close to "0". Using our measurements, the average relative intra-response distances of the Gray encoded PUF responses is 0.0354, while calculating the distances based on the 4-ary PUF readouts results in 0.0694. Note that all responses are produced under the same environmental (temperature) conditions. We again explain the plausibility of the results by using a simple example.

Example 4. *Considering the 4-ary sequences*

$$\boldsymbol{u} = 0\ 3\ 1\ 2\ 3$$
$$\boldsymbol{u}' = 0\ 2\ 1\ 2\ 3$$

we observe a relative Hamming distance of $\frac{1}{5} = 0.2$. *Applying Gray labeling on* $\boldsymbol{u}$ *and* $\boldsymbol{u}'$ *we get the binary sequences*

$$\boldsymbol{u} = 00\ 10\ 01\ 11\ 10$$
$$\boldsymbol{u}' = 00\ 11\ 01\ 11\ 10$$

and hence, a relative Hamming distance of $\frac{1}{10} = 0.1$.

Fig. 7 shows the distribution of the relative intra-response distances of the 4-ary PUF readout as well as for the Gray labeled PUF responses. The latter is close to the optimal value, such that the remaining instabilities can be treated by applying techniques from the field of error-correcting codes (cf. Ch. 5).

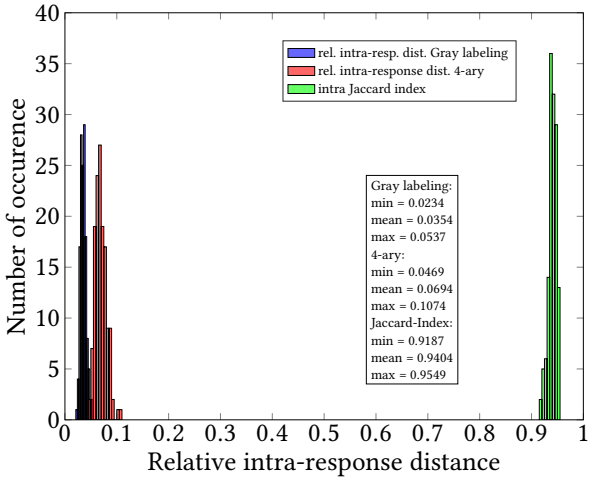


Figure 7: Pairwise relative intra-response distances.

As already done for uniqueness in Ch. 4.1, we calculate the Jaccard index according to Def. 4, to enable a comparison to literature on DRAM-based PUFs. We consider the sets $\mathcal{A}_{\text{init}}$ and $\mathcal{A}_{\text{rep}}$, which include the cells that are classified as weak during initialization and during reproduction, respectively. We obtain an intra-response Jaccard index of ≈ 0.95 under stable environmental conditions, which implies a high robustness.

4.3 PUFs with Challenge-Response Behavior

So far we considered responses that are generated solely based on the hardware's randomness. Another type of PUFs can be challenged with a binary sequence. The response is produced based on that challenge in addition to the hardware's randomness. A challenge and its corresponding response are called a *challenge response pair (CRP)*. For applications, like authentication, two conditions have to be fulfilled. First, it is required to have a large amount of CRPs. Second, based on known CRPs, it has to be impossible to guess responses that correspond to challenges, which do not appear in the set of known CRPs.

We outline an authentication scenario, as for example in [29], which can be implemented with challengeable PUFs: Assume that Alice wants to authenticate herself to Bob. During initialization, Bob collects a set of CRPs from Alice and stores them in a trustworthy database. During authentication, Bob chooses a CRP $(\boldsymbol{x}, \boldsymbol{y})$ and sends the challenge $\boldsymbol{x}$ to Alice. Alice challenges her PUF with $\boldsymbol{x}$ and sends the corresponding response $\boldsymbol{y}'$ to Bob. If $\boldsymbol{y} = \boldsymbol{y}'$ Bob knows that $\boldsymbol{y}'$ has to be produced by Alice's PUF. Every CRP may be used only once, since otherwise Eve can collect CRPs and provide correct responses to some of the challenges in the future.

Due to the large amount of DRAM cells that are measured, it is possible to define challenges to a PUF that yield to unique responses. Therefore, we consider the set $\mathcal{A}_{\text{init}}$. Note that the cardinality of $\mathcal{A}_{\text{init}}$ was observed in the interval $[19.808, 35.284]$. For generation of responses, we want to measure a relatively small amount of those cells, e.g., 1.024 (we define a challenge to be one of the 1024-elementary subsets of $\{1, \ldots 15.000\}$). To use a conservative estimation, there are $\approx \binom{15.000}{1.024}$ such responses. This is in the order of $\approx 2^{5.000}$, and hence, supports the requirement that an excessive number of CRPs is desired in order to prevent learning algorithms from setting up a mathematical clone.

We verify the idea by using PUF 1 and study the relative response distances of several CRPs. The ideal value should be close to 0.5. We generate 1.000 CRPs by choosing the challenges randomly followed by evaluating the PUF according to the chosen challenge. Fig. 8 visualizes the results for both, 4-ary PUF readouts and Gray labeled PUF responses. The latter are close to the optimal value, and hence, provide an attacker from guessing the response to an unknown challenge.

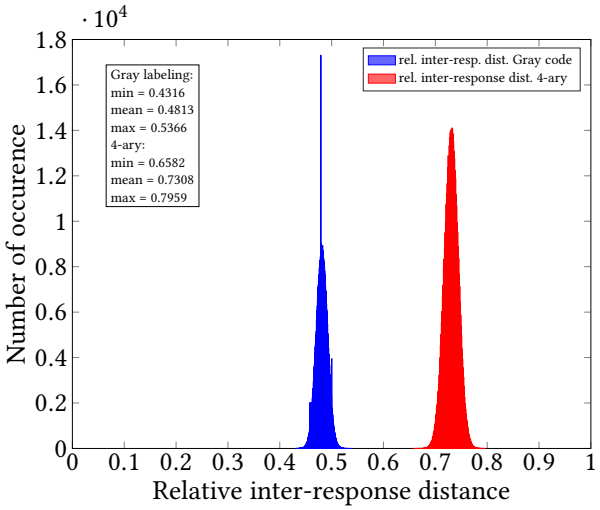


Figure 8: Pairwise relative Hamming distances of responses generated for different challenges using the same PUF.

5 CHANNEL CODING

Error-correcting codes are a very powerful technique to counteract noise that occurs in communication scenarios, when a message is transmitted over a noisy channel. In the context of PUFs, error-correcting codes are used to correct erroneously reproduced PUF responses.

5.1 Preliminaries

A linear code over a finite field $\mathbb{F}_q$ with q elements is a k-dimensional subfield of the vector space $\mathbb{F}_q^n$ and is denoted as $C(q; n, k, d)$. In this notation, n is called the codeword length, k is the dimension, and d the minimum distance. Most often concatenated codes are considered for error-correction in the context of PUFs. Using code concatenation, long codes can be constructed by using two short codes (inner code and outer code). The advantage is, that decoding of the concatenated code is done by (multiple times) decoding the short component codes, which is more efficient. Using $C(n_o, k_o, d_o)$ as outer code and $C(n_i, k_i, d_i)$ as inner code, the concatenated code has the parameters $C(n_o \cdot n_i, k_o \cdot k_i, d_o \cdot d_i)$. The theory of concatenated codes can be found in standard textbooks, e.g. [4].

The following classes of linear codes are used in the design of our concatenated code constructions: Reed-Muller codes [22, 25], denoted as $\mathcal{RM}(q; n, k, d)$, Reed-Solomon codes [26], denoted as $\mathcal{RS}(q; n, k, d)$, and BCH codes [2, 3, 10], denoted as $\mathcal{BCH}(q; n, k, d)$. For construction methods, encoding and decoding of these code classes, we also refer to standard textbooks, e.g., [4].

When designing error-correcting codes for PUFs, several requirements have to be fulfilled. The codeword length has to be equal to the response length, since responses are mapped to codewords. The code dimension k serves as security parameter, since the scheme can be broken by trying all 2^k possible codewords when guessing the correct response. The block error probability obtained when decoding corresponds to the probability that a response is reproduced incorrectly. Usually, a block error probability of 10^{-6} or 10^{-9} is desired when working with PUFs, depending on the underlying hardware.

5.2 Secure Sketches

A secure sketch aims to guarantee PUF responses that are reproduced without errors. For this purpose, a PUF response has to be mapped to a codeword. Since our final PUF response is binary, we can directly apply the code-offset algorithm according to [6, 12]. Fig. 9 visualizes initialization and reproduction of the code-offset algorithm, which we briefly review in this paragraph. During initialization, the initial PUF response $\boldsymbol{r}$ is extracted from the PUF. A random codeword of the used error-correcting code C is chosen and added[1] to $\boldsymbol{r}$. The resulting vector $\boldsymbol{h}$ is stored and serves as helper data in the reconstruction phase. Note that $\boldsymbol{h}$ and a representation of C can be publicly stored in a helper data storage, since an attacker cannot use this information to efficiently reconstruct the initial PUF response $\boldsymbol{r}$.

During reproduction, a response $\boldsymbol{r'}$ is extracted from the PUF. As shown in Fig. 9, a vector $\boldsymbol{y}$ that can be interpreted as received word is obtained when adding $\boldsymbol{h}$ to $\boldsymbol{r'}$. If the distance between $\boldsymbol{r}$ and $\boldsymbol{r'}$ is within the error correction capabilities of C, the decoding result $\hat{\boldsymbol{c}}$ is equal to the codeword $\boldsymbol{c}$, which was chosen during initialization. Hence, adding $\boldsymbol{h}$ to $\hat{\boldsymbol{c}}$ recovers the initial response $\boldsymbol{r}$.

5.3 Channel Model

Studying how the errors that occur during reproduction are distributed over the DRAM cells reveals that they occur relatively

[1] Addition of binary vectors is performed component-wise modulo 2.

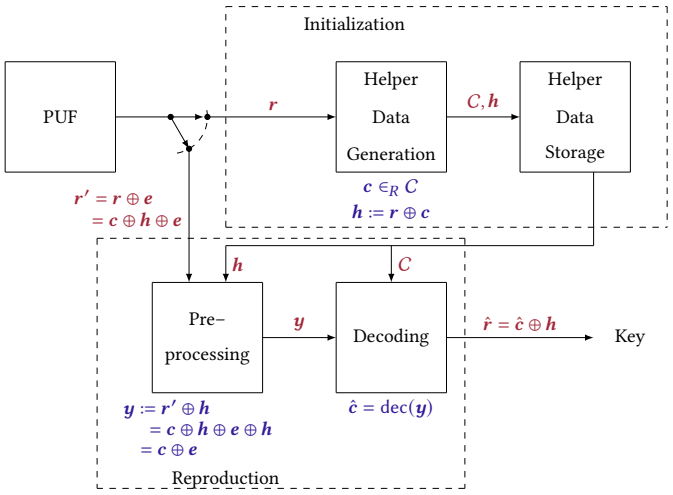


Figure 9: Visualization of the code-offset algorithm for PUFs [20].

uniformly over all cells. Hence, for a first approach, it is reasonable to assume a classical symmetric channel with bit error probability p as channel model. We consider the Gray labeled PUF responses. Since we decided to use a PUF readout of length $N = 1024$ we obtain a response length of $n = 2N = 2048$. The maximum amount of errors that was observed is 0.0537. Due to the small data set (and the stable environmental conditions) we conservatively assume a bit error probability of $p = 0.1$ (this is an established procedure how to estimate the bit error probability of the *Binary Symmetric Channel (BSC)*, for example, compare to [9]). Let the random variable X denote the number of errors in a sequence of length n. The probability that i errors occur in a sequence of length n is

$$\Pr\{X = i\} = \binom{n}{i} \cdot p^i \cdot (1-p)^{n-i}. \tag{7}$$

The expected number of errors is

$$\mathrm{E}\{X\} = n \cdot p, \tag{8}$$

and hence, $p = \frac{\mathrm{E}\{X\}}{n}$. Fig. 10 visualizes the cumulative distribution function of X. $\Pr\{X \leq i\}$ is the probability that at most i errors occur. Hence, $1 - \Pr\{X \leq i\}$ is the probability that more than i errors occur. The reasons for errors in reproduced sequences were discussed in Ch. 3.3.

5.4 Error-Correcting Codes

We consider the Gray labeled PUF responses of length $n = 2048$ as in (7), (8) and Fig. 10. Let the random variable X again denote the number of errors in a response. From the cumulative distribution function (cf. Fig. 10), we conclude that for all $\tau > 260$ we have

$$\begin{aligned} \Pr\{X > \tau\} &= 1 - \Pr\{X \leq \tau\} \\ &= 1 - 1 \\ &= 0. \end{aligned}$$

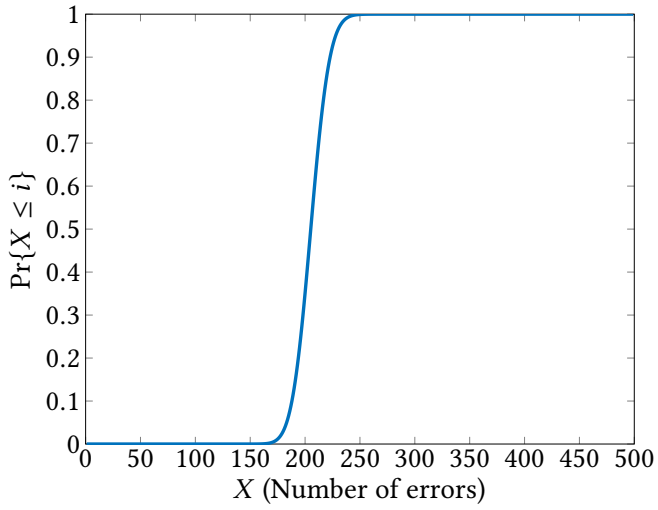


Figure 10: Cumulative distribution function of the number of errors.

Thus, we want to be able to correct $t \leq 260$ errors. This requires an error-correcting code of minimum distance $d = 2t + 1 = 521$.

Assuming a desired key length of 128 bit (e.g., for AES-128 [5]), the requirement is a $C(2; 2048, \geq 128, \geq 521)$. This requirement can be fulfilled by an concatenated code, consisting of an $\mathcal{RS}(2^6; 64, 32, 33)$ outer code, and an $\mathcal{RM}(2; 32, 6, 16)$ inner code. The concatenated code $C_1(2; 2048, 192, \geq 528)$ is obtained.

Assuming a desired key length of 256 bit (e.g., for AES-256 [5]), the requirement is a $C(2; 2048, \geq 256, \geq 521)$. This requirement can be fulfilled by an concatenated code, consisting of an $\mathcal{RS}(2^7; 128, 42, 87)$ outer code, and an extended BCH code[2] $\mathcal{BCH}(2; 16, 7, 6)$ inner code. This yields an overall concatenated code $C_2(2; 2048, 294, \geq 522)$. Note that the small code rates are common in the PUF scenario, e.g., [24]. Of course, code C_2 can also be used for a key of length 128. However, the disadvantage then is the larger field size.

6 HASHING

Note that the final key is always binary in applications. Hence, in this step, a perfectly reconstructed, error free binary PUF response is assumed. This response is hashed to the final key length, for example to length 128 if a key for the *Advanced Encryption Standard (AES) [5]* is generated. This step is rather trivial and usually not considered in the PUF literature. For completeness, we mention that for example the cryptographic hash functions SHA-1 (used in [9]) and SPONGENT (used in [19]) have been applied in the context of PUFs before.

7 COMPARISON

In this section, we compare the new MVPUF processing chain against the binary counterparts proposed in [21] and [28]. For the sake of fair comparison, we consider only those work which uses retention errors as the source of randomness. Table 2 shows the comparison of this work against the aforementioned prior works. The MVPUF responses achieve the highest entropy and requires no additional debiasing algorithm. However, the proposed response extraction technique is more time-consuming compared to binary extraction. Since the proposed MVPUF processing chain can be applied to other sources such as reduced latency and row hammer that consume less time per readout. Hence, the increased readout time will have a negligible effect during the reproduction phase for these sources.

Table 2: Comparison of the proposed MVPUF processing chain against the binary counterparts

	This work	**[21]**	**[28]**
Refresh pause duration	{9s,10s,11s,12s}	10s	10s or 20s
Debiasing Required	No	Yes	Yes
PUF response Entropy	0.99	0.9	-

8 RELATED WORK

In this section, we discuss the PUF response processing techniques adopted in DRAM-based PUF literature for consistent key generation. In [30] a retention error based PUF was proposed. This work employed a lightweight Hamming encoding/decoding algorithm to counteract the noise in PUF responses and generated keys. However, Hamming codes are only capable of correcting single-bit errors and detecting double-bit errors. [21] modeled retention errors extracted from the DRAM as a noisy communication channel and extract channel models to identify the appropriate error-correcting code. [21] proposed to use Temporal Majority Voting (TMV) in combination with one of the 2 algorithms (i.e., choose length and Von Neumann debiasing) to debias the raw bits extracted from the DRAM after refresh off period. [21] proposed to use the helper data scheme (i.e., fuzzy extractor [6]) as a pre-processing scheme to reduce the noise [28], which is then fed to the error-correcting (i.e., BCH code) or decoding block for consistent key generation. [28] is another retention error based PUF that also proposed to employ a fuzzy extractor to reduce the response bit error rate by generating helper data. Similar to [21], [28] also proposed to use choose length debiasing and a linear error-correcting code, such as a BCH code, for error correction. Both, [21] and [28] proposed the processing chain applicable for binary responses. [1] extracted the randomness by exploiting row-hammer errors and proposed to use a similar processing chain as [28]. [14] used the errors occurring due to reduced timing parameters as a source of randomness. [14] proposed a filter mechanism that performs multiple iterations (in orders of 100) of reduced latency reads. Only, the cells that exhibit sensing failure probability greater than a certain threshold are considered as weak cells in the PUF response. This technique is similar to TMV. To the best of our knowledge, this is for the first time that a DRAM-based multi-valued PUF is proposed that extracts non-biased responses and generates keys with high entropy.

[2]Consider the $\mathcal{BCH}(2; 15, 7, 5)$ code (cf. [4]). By extension (which means by appending "0" or "1" to each codeword such that the weight is even) a (n, k, d) code can be transformed into an $(n + 1, k, d + 1)$ code, if d is odd.

9 CONCLUSION

In this work, we proposed a complete PUF processing chain including the first multi-valued PUF based on DRAM. We explained how to extract PUF readouts over a 4-ary alphabet from DRAM retention errors in order to increase entropy and to improve error correction. We applied methods from the fields of channel coding and hash functions to derive secure cryptographic keys that can for example be used for the AES cryptosystem. The constructed PUFs have an average inter-response distance of ≈ 0.48, an average intra-response distance of ≈ 0.04 under stable environmental conditions, and an entropy of up to 0.99 bit. In contrast to binary DRAM PUFs that have been proposed in the literature, no extra debiasing algorithms are required. In future work, robustness under varying temperature conditions will be studied.

ACKNOWLEDGMENTS

This work was funded in parts by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under project FI 982/15-1 and project WE 2442/14-1. The project OPRECOMP acknowledges the financial support of the EU FET programme under grant agreement No.732631.

REFERENCES

[1] Nikolaos Athanasios Anagnostopoulos, Tolga Arul, Yufan Fan, Christian Hatzfeld, André Schaller, Wenjie Xiong, Manishkumar Jain, Muhammad Umair Saleem, Jan Lotichius, Sebastian Gabmeyer, Jakub Szefer, and Stefan Katzenbeisser. 2018. Intrinsic Run-Time Row Hammer PUFs: Leveraging the Row Hammer Effect for Run-Time Cryptography and Improved Security †. *Cryptography* 2 (2018), 13.

[2] Raj C. Bose and Dwijendra K. Ray-Chaudhuri. 1960. Further Results on Error Correcting Binary Group Codes. *Information and Control* 3, 3 (1960), 279–290.

[3] Raj C. Bose and Dwijendra K. Ray-Chaudhuri. 1960. On a Class of Error Correcting Binary Group Codes. *Information and Control* 3, 1 (1960), 68–79.

[4] Martin Bossert. 1999. *Channel Coding for Telecommunications.* John Wiley & Sons, Inc.

[5] Joan Daemen and Vincent Rijmen. 2002. *The Design of Rijndael.* Vol. 2. Springer.

[6] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. 2004. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and other Noisy Data. In *International Conference on the Theory and Applications of Cryptographic Techniques.* Springer, 523–540.

[7] Daniel Fainstein, Sami Rosenblatt, Alberto Cestero, Norman Robson, Toshiaki Kirihata, and Subramanian Iyer. 2012. Dynamic Intrinsic Chip ID using 32nm High-K/metal Gate SOI Embedded DRAM. In *Symposium on VLSI Circuits.* IEEE, 146–147.

[8] Blaise Gassend, Dwaine Clarke, Marten Van Dijk, and Srinivas Devadas. 2002. Silicon Physical Random Functions. In *Proceedings of the 9th ACM Conference on Computer and Communications Security.* ACM, 148–160.

[9] Jorge Guajardo, Sandeep S. Kumar, Geert-Jan Schrijen, and Pim Tuyls. 2007. FPGA Intrinsic PUFs and their Use for IP Protection. In *International Workshop on Cryptographic Hardware and Embedded Systems.* Springer, 63–80.

[10] Alexis Hocquenghem. 1959. Codes Correcteurs d'Erreurs. *Chiffres* 2, 2 (1959), 147–56.

[11] Daniel E. Holcomb, Wayne P. Burleson, and Kevin Fu. 2007. Initial SRAM State as a Fingerprint and Source of True Random Numbers for RFID Tags. In *Proceedings of the Conference on RFID Security*, Vol. 7. 2.

[12] Ari Juels and Martin Wattenberg. 1999. A Fuzzy Commitment Scheme. In *Proceedings of the 6th ACM Conference on Computer and Communications Security.* ACM, 28–36.

[13] Matthias Jung, Deepak M. Mathew, Carl C. Rheinländer, Christian Weis, and Norbert Wehn. 2017. A Platform to Analyze DDR3 DRAM's Power and Retention Time. *IEEE Design & Test* 34, 4 (2017), 52–59.

[14] Jeremie S. Kim, Minesh Patel, Hasan Hassan, and Onur Mutlu. 2018. The DRAM Latency PUF: Quickly Evaluating Physical Unclonable Functions by Exploiting the Latency-Reliability Tradeoff in Modern Commodity DRAM Devices. In *Symposium on High Performance Computer Architecture (HPCA).* IEEE, 194–207.

[15] Patrick Koeberl, Jiangtao Li, Anand Rajan, and Wei Wu. 2014. Entropy Loss in PUF-based Key Generation Schemes: The Repetition Code Pitfall. In *International Symposium on Hardware-Oriented Security and Trust (HOST).* IEEE, 44–49.

[16] Jae W. Lee, Daihyun Lim, Blaise Gassend, G. Edward Suh, Marten Van Dijk, and Srinivas Devadas. 2004. A Technique to Build a Secret Key in Integrated Circuits for Identification and Authentication Applications. In *Symposium on VLSI Circuits, 2004. Digest of Technical Papers.* IEEE, 176–179.

[17] Arjen K. Lenstra, James P. Hughes, Maxime Augier, Joppe W. Bos, Thorsten Kleinjung, and Christophe Wachter. 2012. Ron was wrong, Whit is right. Cryptology ePrint Archive, Report 2012/064.

[18] Roel Maes. 2013. *Physically Unclonable Functions: Constructions, Properties and Applications.* Springer Science & Business Media.

[19] Roel Maes, Anthony Van Herrewege, and Ingrid Verbauwhede. 2012. PUFKY: A Fully Functional PUF-based Cryptographic Key Generator. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES).* Springer, 302–319.

[20] Sven Müelich. 2019. *Channel coding for hardware-intrinsic security.* Ph.D. Dissertation. Universität Ulm.

[21] Sven Müelich, Sebastian Bitzer, Chirag Sudarshan, Christian Weis, Norbert Wehn, Martin Bossert, and Robert FH Fischer. 2019. Channel Models for Physical Unclonable Functions based on DRAM Retention Measurements. In *International Symposium Problems of Redundancy in Information and Control Systems (REDUNDANCY).* IEEE, 149–154.

[22] David E. Muller. 1954. Application of Boolean Algebra to Switching Circuit Design and to Error Detection. *Transactions of the IRE Professional Group on Electronic Computers* EC-3, 3 (1954), 6–12.

[23] Ravikanth Pappu. 2001. *Physical One-Way Functions.* Ph.D. Dissertation. Massachusetts Institute of Technology.

[24] Sven Puchinger, Sven Müelich, Martin Bossert, Matthias Hiller, and Georg Sigl. 2015. On Error Correction for Physical Unclonable Functions. In *Proceedings of 10th International ITG Conference on Systems, Communications and Coding (SCC).*

[25] Irving S. Reed. 1954. A Class of Multiple-Error-Correcting Codes and the Decoding Scheme. *IEEE Transactions on Information Theory* (1954).

[26] Irving S. Reed and Gustave Solomon. 1960. Polynomial Codes over Certain Finite Fields. *J. Soc. Indust. Appl. Math.* 8, 2 (1960), 300–304.

[27] André Schaller, Wenjie Xiong, Nikolaos Anagnostopoulos, Muhammad Saleem, Sebastian Gabmeyer, Stefan Katzenbeisser, and Jakub Szefer. 2017. Intrinsic Rowhammer PUFs: Leveraging the Rowhammer Effect for Improved Security. In *Hardware Oriented Security and Trust.* IEEE, 1–7.

[28] A. Schaller, W. Xiong, N. A. Anagnostopoulos, M. U. Saleem, S. Gabmeyer, B. Škorić, S. Katzenbeisser, and J. Szefer. 2019. Decay-Based DRAM PUFs in Commodity Devices. *IEEE Transactions on Dependable and Secure Computing* 16, 3 (2019), 462–475.

[29] G. Edward Suh and Srinivas Devadas. 2007. Physical unclonable functions for device authentication and secret key generation. In *44th ACM Design Automation Conference.* IEEE, 9–14.

[30] Soubhagya Sutar, Arnab Raha, Devadatta Kulkarni, Rajeev Shorey, Jeffrey Tew, and Vijay Raghunathan. 2017. D-PUF: An Intrinsically Reconfigurable DRAM PUF for Device Authentication and Random Number Generation. *ACM Trans. Embed. Comput. Syst.* 17, 1, Article 17 (Dec. 2017), 31 pages. https://doi.org/10.1145/3105915

[31] Fatemeh Tehranipoor, Nima Karimian, Kan Xiao, and John Chandy. 2015. DRAM Based Intrinsic Physical Unclonable Functions for System Level Security. In *Great Lakes Symposium on VLSI.* ACM, 15–20.

[32] Randy Torrance and Dick James. 2009. The State-of-the-Art in IC Reverse Engineering. In *CHES* (Lausanne, Switzerland). 363–381. https://doi.org/10.1007/978-3-642-04138-9_26

[33] Meng-Day Yu, Matthias Hiller, and Srinivas Devadas. 2015. Maximum-Likelihood Decoding of Device-Specific Multi-Bit Symbols for Reliable Key Generation. In *Int. Symposium on Hardware Oriented Security and Trust (HOST).* IEEE, 38–43.

[34] Siarhei S. Zalivaka, Alexander V. Puchkov, Vladimir P. Klybik, Alexander A. Ivaniuk, and Chip-Hong Chang. 2016. Multi-valued Arbiters for Quality Enhancement of PUF Responses on FPGA Implementation. In *Asia and South Pacific Design Automation Conference (ASP-DAC).* IEEE, 533–538.