

Towards Physical Unclonable Functions On Analog Neuromorphic Hardware

Sören Jost

Studienarbeit

Ruprecht-Karls-Universität Heidelberg

Fakultät für Ingenieurwissenschaften

ZITI

Electronic Visions - Prof. Dr. J. Schemmel

May 20, 2026

Contents

1	Introduction	3
1.1	Structure of the report	4
2	Background	5
2.1	Physical Unclonable Functions	5
2.2	BrainScaleS-2	6
2.3	Spike (Dis)Similarity Metrics	6
2.3.1	Victor-Purpura Distance	6
2.3.2	van Rossum Distance	7
3	Methods	8
3.1	Neuron Variability Experiment	8
3.2	PUF-SNN	9
4	Results	12
4.1	Neuron Circuit Variations	12
4.1.1	Selection cost parameters	12
4.1.2	Measured Distances	12
4.2	PUF Metric Training on a SNN	15
5	Discussion	19
6	Conclusion	20
7	Outlook	21
	Acronyms	23
	Bibliography	24

1 Introduction

General-purpose, embedded, cyber-physical and other von Neumann-based systems all come with the same downside, all have the need for security. Each system has the goal of securing the integrity of its data. As secure keys for cryptographic procedures are only as safe as the system itself, hardware security can provide further security to the whole systems stack (Akter et al. 2023; Chang et al. 2017). The growing sophistication and prevalence of software-based attacks and fault injection attacks (FIAs) at all abstraction layers of a system increasingly challenge the effectiveness of conventional security primitives (Chang et al. 2017). To date, root-of-trust circuits provide a dependable source of entropy by exploiting the inherent randomness of physical devices to instantiate security primitives, while being resistant to attacks (Chatterjee et al. 2025).

However, conventional von Neumann-based systems are not only challenged from a security perspective but also face fundamental limitations in performance and energy efficiency. The end of Dennard scaling together with the increasing demand for compute- and data-intensive workloads such as neural networks and machine learning (ML) exposes the inefficiencies of traditional architectures (Gasarch et al. 2021). As a result, alternative computing paradigms are being explored.

Neuromorphic systems are one such alternative, inspired by biological neural structures. By using spiking neural networks (SNNs), event-based computation and in-the-loop (ITL) based training, these systems offer advantages in parallelism and energy efficiency (Cramer et al. 2022). Due to the sparse and event-driven nature of neuronal activity, energy is mainly consumed when neurons spike, resulting in overall low power consumption (Pehle et al. 2022; Hadis Takaloo and Ahmadi 2022).

As these systems are further developed and may become part of future computing platforms, it is important to ensure that they provide support for established security mechanisms. While hardware-based security primitives are common practice in conventional von Neumann-based systems, their implementation in neuromorphic hardware is still largely unexplored. Considering security aspects early in the development of such systems is essential for their possible later use in security-critical environments.

One class of such hardware security primitives are physical unclonable functions (PUFs). These circuit based security primitives exploit inherent physical variations introduced during manufacturing to generate device-specific and unpredictable responses. This makes them suitable for applications such as device authentication and cryptographic key generation (Gao et al. 2020; Chang et al. 2017). So far, only a limited number of implementations of PUFs on neuromorphic systems have been presented.

For example Neuron-PUF by Elshamy and Stratigopoulos (2021) which use integrate and fire spiking neurons based on the axon hillock model. The spikes of the neuron are used to produce enable signals for a transmission gate to control the output of a linear feedback shift register. Containing a pre-selected seed the register outputs a response bit, depended on the input current of the neuron circuit. Furthermore Hadis Takaloo and Ahmadi (2022) proposed a modified octopus retina neuron circuit which reacts to an excitation voltage producing a unique spiking output which can be directly encoded into a response bit structure. Further

responses of different neuron cells can be compared and the result be used as a final response, extending the entropy. In both examples the PUF architectures are coming close to 50% uniformity, a metric depicting the stability of entropy within a PUF, showing the probable use case of neuron cells as entropy sources.

Such an entropy source could be the BrainScaleS-2 system (Pehle et al. 2022), providing a neuromorphic platform with 512 analog neurons per chip. Although all neurons are based on the same circuit design, neuron-to-neuron variations should lead to differences in their spiking behavior to identical input stimuli (Pehle et al. 2022). This variability makes the system a promising candidate for the implementation of PUFs. A typical use case of BrainScaleS-2 is the implementation of SNNs. The software stack enables an easy definition of a SNN that can be deployed directly on the analog hardware (Müller et al. 2022). Allowing the use of a SNN structure while exploiting the neuron-to-neuron differences as an chip identity source.

In this work, the neuron variability on BrainScaleS-2 is inspected and a new approach is investigated in which PUF-related metrics are incorporated during the training of a SNN. The goal is to stabilize the responses of the system while maintaining sufficient uniqueness, enabling the realization of reliable PUFs instances on neuromorphic hardware.

1.1 Structure of the report

The remainder of this report is structured as follows. Chapter 2 introduces Physical Unclonable Functions, describes the quality metrics required for their evaluation, presents BrainScaleS-2, and explains spike similarity metrics for spike train evaluation.

Next, Chapter 3 introduces the experiments conducted in this work for evaluating neuron variability and the suitability of SNNs for PUFs. Chapter 4 then presents the results of the conducted experiments, while Chapter 5 discusses the obtained results and their implications. Finally, Chapter 6 concludes this work, and Chapter 7 outlines directions for future work within this project.

2 Background

2.1 Physical Unclonable Functions

PUFs are similar to a unique fingerprint as devices contain inherent randomness introduced at fabrication of the system which is unique for a given device. This randomness is used to produce secret keys in each device. As the randomness exists within standard analog components, the utilization of it is rather inexpensive. Additionally the reproduction of these measurable variations such as in a simulation is considered impossible (Gao et al. 2020). Transforming this randomness into usable secret keys most PUF systems use challenge response pairs (CRPs), a set of unique challenges with an associated response. Between multiple PUFs the response is unique enough to differentiate the devices while the same device answers reliably with the same response. Various technologies, such as FPGA-based arbiter or ring-oscillator PUFs and SRAM-based PUFs (Mathew et al. 2025; Gao et al. 2020), currently offer stable and reliable PUF implementations.

Furthermore, PUFs are classified either as a strong or weak. While strong PUFs can be used in cryptographic authentication procedures due to their key space covering multiple orders of magnitude, weak PUFs are common as device unique keys or seeds for conventional encryption (Gao et al. 2020; Chang et al. 2017). In addition PUFs can be categorized into subclasses such as reconfigurable physical unclonable functions (rPUFs) that can reuse the same devices by remapping the CRP space or public physical unclonable functions (pPUFs) which stem from the idea of asynchronous encryption using the physical instance as a private key and a model instance as the public key (Gao et al. 2020).

Core properties that such a CRP needs to show in a PUF have been formed and realized in the metrics of uniformity, bit-aliasing, inter-Hamming distance (or commonly known as uniqueness), and the intra-Hamming distance (referred to as reliability) (Sergio Vinagrero Gutierrez and Natale 2026). For this report, uniqueness and reliability play a vital role as metrics, as they characterize the stability and usability of a PUF and the underlying hardware during the development process. Additional metrics will need to be considered at a later stage when inspecting the finalized PUF design. Uniqueness and reliability are defined as follows:

- **Uniqueness** describes the distinctive identity of a device out of a group of devices within a system. Uniqueness can be measured by comparing the response of one device to the responses of the other devices within the group. Usually expressed as the inter-Hamming distance between the response $x(c)$ of device X and the response $y(c)$ of the device Y to the challenge c : $HD(x,y)$
- **Reliability** expresses the reproducibility of a response to the same challenge on one device across different external influences and time. Reliability can be measured by comparing the first response of a device to all following responses. Usually expressed as the intra-Hamming distance between the responses $x(c)$ and $x'(c)$ of device X to the challenge c : $HD(x,x')$

2.2 BrainScaleS-2

BrainScaleS-2 is an accelerated mixed-signal neuromorphic system incorporating a mixed-signal ASIC core with 512 analog neurons based on the adaptive exponential integrate-and-fire (AdEx) model (Pehle et al. 2022). In addition, digital components such as a single instruction multiple data (SIMD) processor are integrated. The analog circuits emulate the neurons as well as synaptic connections in time-continuous matter rather than discrete-time with an acceleration of 10^3 basing on the properties of the silicon substrate. In detail, the ASIC contains 512 neuron circuits, which can be interconnected via synaptic arrays. These arrays are split into four quadrants, each connected to one column, and are complemented by local configuration memory as well as spike routing capabilities (Pehle et al. 2022). Important to note is the capability of configuring singular neuron circuits such as the exponential or adaptive current to transform the AdEx neuron into simpler models or countering device-specific deviations from the manufacturing process (Müller et al. 2022). All further handling of spike events, configuration data, on-chip calibration, hybrid plasticity rule application and the analog readout for ITL is part of the digital system (Pehle et al. 2022). Furthermore, a higher ordered off-chip FPGA manages the translation from the executing time-continuous system to the experiment defining time-discrete host system. Providing memory buffers for incoming data the FPGA extends the systems limited on-chip memory with higher-latency off-chip memory (Müller et al. 2022). This allows the use of the system for neuro-scientific modeling and neuro-inspired machine learning. Even though the system is highly balanced, the integrated neuron circuits exhibit neuron-to-neuron variations, which require calibration of the neuron parameters (Pehle et al. 2022). These parameters are adjusted via the capacitive memory (CapMem), providing a precision of 10 bits for each parameter (Hock et al. 2013). However, the CapMem itself also exhibits small variations between its circuits. These neuron variations can be exploited for the purpose of entropic response generation.

2.3 Spike (Dis)Similarity Metrics

Focusing on the determination of the suitability of BrainScaleS-2 by inspecting the uniqueness and reliability of its neuron circuits for later the implementation of a neuromorphic PUF, the common evaluation metrics presented in Section 2.1 are difficult to deploy in this setting. Spike trains are not binary vectors that can be easily compared, and there is currently no intention to encode the resulting spike trains into binary responses in order to compare raw responses directly. As a result, alternative metrics must be considered for evaluating the system’s performance with respect to uniqueness and reliability. For this purpose, two established spike similarity metrics were chosen to analyze the (dis)similarity of two or more spike trains, enabling the assessment of similarity in the case of reliability and dissimilarity in the case of uniqueness.

2.3.1 Victor-Purpura Distance

Victor and Purpura introduced in Victor and Purpura (1997) a new method to analyze the temporal structure of spike trains. Originally, three families of metrics were formed, each describing a metric space defined over spike trains and the distances between their spikes. In this work, the focus is on the metric family $D^{spike}[q]$, which operates on spike times and is later referred to as Victor-Purpura distance (VPD). Simplified, $D^{spike}[q]$ is defined by the parameters q , representing the cost (in s^{-1}), and Δt , representing the time steps. The

distance of a spike train A to the spike train B is then determined by finding the minimal costs for transforming A into B by using the following operations:

- **Shifting** a spike of A to fit onto the next spike of B by the amount of Δt with a cost of q adding to the total cost $q|\Delta t|$.
- **Remove** or **Insert** a spike in either spike train with a cost of 1. Choosing to **Remove** and **Insert** a spike for a total cost of 2 if the cost to **shift** exceeds $|\Delta t| > 2/q$

In case $q = 0$ the shifting of spikes is free and the VPD is formed back to a spike count metric. Further very large values of q will lead to q acting as a dial for sensitivity treating very similar spikes as completely different making the VPD usable to mark minor differences.

2.3.2 van Rossum Distance

Following to Victor and Purpura another important distance metric was formed by van Rossum in (Rossum 2001). Instead of removing, inserting or shifting a singular spike from one spike train, van Rossum proposed to transform the spike trains into continuous-time functions by convolving each spike with an exponential kernel shown in Equation (2.2). Summing these kernels results in a filtered function $f(t)$ as shown in Equation (2.1). The squared distance D^2 is defined as the integral of the squared difference between two filtered trains, scaled by $\frac{1}{\tau}$ (Equation (2.3)). The squared norm of an exponential kernel scales with τ , and the van Rossum distance compensates for this via a $\frac{1}{\tau}$ normalization factor. Simply said if both spike trains contain a spike at identical time steps they cancel each other out. In case they are shifted the resulting area translates to the distance. Choosing τ affects the behavior of the distance metric: a very small τ allows the van Rossum distance (vRD) to act as a coincidence detector, while a very large τ transforms it into a spike count difference measure. Choosing τ in a smaller size can then be helpful to classify the similarity of two spike trains.

$$f(t) = \sum_i k(t - t_i) \quad (2.1)$$

$$k(t) = \begin{cases} e^{-t/\tau} & t \geq 0 \\ 0 & t < 0 \end{cases} \quad (2.2)$$

$$D^2 = \frac{1}{\tau} \int_0^\infty (f(t) - g(t))^2 dt \quad (2.3)$$

3 Methods

To inspect the suitability of BrainScaleS-2 for PUFs, two experiments were defined. The first focuses on the general variations of the neuron circuits by comparing the resulting output spike metrics of reliability and uniqueness in statically configured neuron populations. The second involves training a SNN based on the loss calculated from the same metrics used in the first experiment.

Both experiments are based on the metrics already established in Section 2.1. The main challenge here is that the metrics in the literature are defined for bit vectors, which are the most basic datatype for comparison. However, a SNN or a simple neuron population does not directly produce bit vectors that can be used to calculate these metrics. While encoding spiketrains into bit vectors is possible, this would defeat the purpose of a direct, raw comparison of spiketrains for variability evaluation. Therefore, in the following experiments, uniqueness and reliability are interpreted differently from their definitions in the literature.

For our use case, we define uniqueness and reliability as distance metrics based on either the VPD or vRD similarity metrics:

- **Uniqueness** – the distance between the first spiketrains of all neurons.
- **Reliability** – the distance between all spiketrains produced by a single neuron.

3.1 Neuron Variability Experiment

Before directly implementing and training a SNN on the distance metrics, the overall neuron variation of BrainScaleS-2 needs to be analyzed. The neuron variability experiment is implemented using the *PyNN* API, which allows a high-level Python implementation.

The experiment uses a simple network structure consisting of a single spike source connected to 512 neurons. The connection is realized as an all-to-all projection using identical synaptic weights. This setup ensures that all neurons receive the same input, allowing differences in the output spiketrains to be attributed to neuron variability rather than input variations.

The goal of this experiment is to enable an easy comparison of up to all 512 on-chip neurons via a high-level interface. The range starts from a single neuron to inspect its reliability and extends to all available neurons to examine both their reliability and the uniqueness between them. The focus is on recording spiketrains and calculating the required distance-based metrics.

The experiment is parameterized by the number of neurons to inspect, the number of spikes in the input spike train, the number of different spike trains to test, and the number of repetitions, which are important for reliability measurements. All 512 neurons are recorded at once and the recordings saved. This allows offline data processing of the dataset or subsets without requiring direct access to BrainScaleS-2.

The experiment procedure follows a sequential pipeline. First, input spike trains are generated. These spike trains are then applied to the neuron population. Each spike train is

repeated for a defined number of repetitions, allowing multiple responses of the same neuron to identical inputs to be recorded.

This repetition enables the calculation of reliability, as the responses of a single neuron can be compared across repetitions. In contrast, uniqueness is evaluated by comparing the responses between different neurons. For this purpose, typically the first repetition is used, although a user-defined repetition or multiple repetitions can also be selected to provide a broader view. Computing uniqueness across all repetitions is possible, but leads to a significant increase in computational cost due to the complexity of the distance calculations.

After recording, reliability and uniqueness are computed separately using the *elephant* library (Denker et al. 2018), which provides implementations of the VPD and vRD metrics. To handle the computational complexity, the calculations are parallelized using multiprocessing, distributing the workload across neurons or repetitions.

Furthermore, the neuron parameters can be manipulated via the *PyNN* API, which provides high-level access to the hardware parameters and allows systematic variation of neuron configurations.

Overall, this setup allows the separation of variability across repetitions (reliability) from variability across neurons (uniqueness).

The variability of BrainScaleS-2 neurons will be analyzed under three scenarios:

1. A run over all neurons with standard calibration.
2. A run where a single parameter is deviated from the standard calibration, which is expected to produce distinct neuron dynamics. This should demonstrate how parameter variations can generate unique outputs, which are required for the second experiment.
3. A run where two parameters are deviated from the standard calibration, which is expected to produce even bigger distinctive neuron dynamics. This demonstrates combinatorial use of parameter dependent variations.

3.2 PUF-SNN

The design of PUF CRP spaces can require a long search for suitable challenge inputs and the stabilization of hardware responses. Such tasks can be supported by ML approaches like SNNs. The benefit of using SNNs on BrainScaleS-2 is that the neurons used within the network can also serve as a source of entropy for the PUF. This is the core idea of the following experiment: using a SNN to resemble a PUF entity while defining a loss function based on PUFs metrics, leading to a network that produces unique but reliable responses.

The physical unclonable function spiking neural network (PUF-SNN) is implemented in *hx-torch*, allowing a Torch-like implementation of a neural network on BrainScaleS-2 using surrogate gradients. This setup demonstrates the applicability of SNNs for PUF stabilization.

The objective is to train a SNN using the defined PUF metrics such that it produces high uniqueness and high reliability, where reliability corresponds to a low distance between repeated responses. Although this may initially seem contradictory, in terms of distance metrics this corresponds to maximizing uniqueness distance while minimizing reliability distance.

For this purpose a simple PUF-SNN is defined as seen in Figure 3.1. Consisting of one single leaky-integrate-fire (LIF) neuron with one input and one recurrent connection weighted over the parameters W_I and W_R . To demonstrate the learnability of the metrics based on gradient optimization on the networks parameters, multiple PUF-SNN instances are needed

for comparison of their output spiketrains. Both networks receive the same input spike train and share the weight parameters W_I and W_R . The optimizer adjusts these shared parameters during backpropagation to enforce reliable yet distinct output spike trains for both neurons.

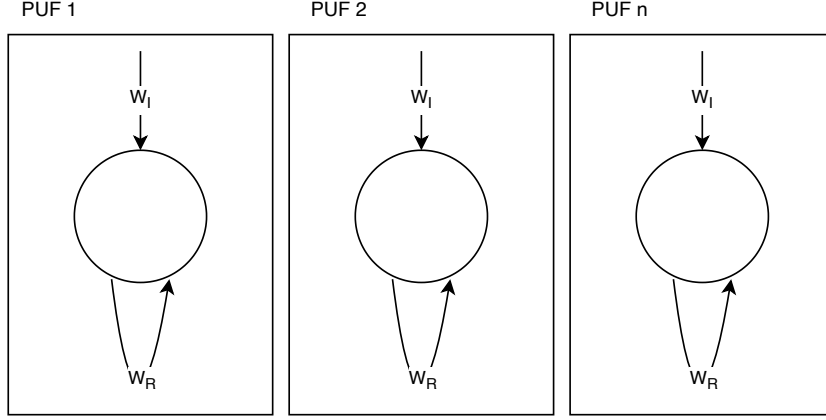


Figure 3.1: Simple PUF-SNN with one LIF neuron and one input and one recurrent connection weighted by parameters W_I W_R . Two or more PUF-SNN instances, receiving the same input spike train, are compared to produce PUF metrics while sharing the weight parameters

The loss function in Equation (3.1) is designed to reflect this objective. The uniqueness should be maximized, which is achieved by minimizing its reciprocal. The reliability should be minimized directly. A hyperparameter α is introduced to control the balance between both terms, allowing a stronger focus on either reliability or uniqueness. The choice of α is discussed in Chapter 4.

$$L = ((1 - \alpha) \times \frac{1}{Uniqueness}) + (Reliability \times \alpha) \quad (3.1)$$

The implementation of the model and its training pipeline follows a standard approach with some modifications. After loading the calibration for the chip, neuron parameters can be adjusted on the hardware level (e.g., *leak_i_bias*), introducing controlled variations in neuron responses.

A fixed input spike train is generated and used throughout the entire experiment. This ensures that observed differences in the output originate from neuron variability rather than changes in the input. The input spike train is repeated with a batch size equal to the number of repetitions, allowing multiple responses to identical inputs.

The forward pass performs the forward computation of the networks, followed by the calculation of uniqueness and reliability using a custom vRD implementation that supports gradient backpropagation. The resulting distance values are aggregated and passed to the loss function.

Since uniqueness and reliability can differ by orders of magnitude, both metrics are normalized using a method based on Welford’s online algorithm. This enables continuous estimation of mean and variance. A z-score normalization followed by a sigmoid function maps the values into the range $[0, 1]$, resulting in a more stable loss and smoother parameter updates during training.

Under ideal conditions, such as in mock mode where only the software model is used, both uniqueness and reliability distances should approach zero, as no variability is present. When running on hardware with standard calibration, variations in neuron responses are expected due to inherent hardware differences.

Further increasing the variability by modifying hardware parameters after calibration should amplify these differences, highlighting the impact of controllable parameter changes on the behavior of the network.

4 Results

4.1 Neuron Circuit Variations

All conducted neuron variability experiments were performed on all 512 neurons, with 250 repetitions for the reliability analysis. The uniqueness was computed based on the first repetition performed in each experiment. Furthermore, three different spike trains were tested, each containing 25 spikes within a simulation time of $150\ \mu\text{s}$. This setup provides a broad view of the variations of neuron circuits within BrainScaleS-2.

4.1.1 Selection cost parameters

The cost parameters for the calculations were evaluated across three step sizes between $\tau = 1e^{-6}\text{ s}$ (or $q = 1e^6\text{ Hz}$), $\tau = 1e^{-5}\text{ s}$ (or $q = 1e^5\text{ Hz}$) and $\tau = 1e^{-4}\text{ s}$ (or $q = 1e^4\text{ Hz}$).

For the vRD, choosing $\tau = 1e^{-5}\text{ s}$ results in an area of $1e^{-5}\text{ s}$ underneath a decaying spike. For the VPD, selecting $q = 1e^5\text{ Hz}$ results in spike shifts below $1e^{-5}\text{ s}$ being cheaper than spike removal or insertion, as

$$q \cdot \Delta t = 1e^5\text{ Hz} \cdot 1e^{-5}\text{ s} = 1.$$

This causes temporal spike shifts to be preferred over insertion or removal operations, providing a more consistent distance measure within the simulation time of $150\ \mu\text{s}$.

However, it is important to consider that the achievable temporal resolution of the distance metric is constrained by the neuron membrane time constant. Under standard calibration, the neuron circuits exhibit a membrane time constant of approximately $\tau_m = 1e^{-5}\text{ s}$, which therefore could act as a practical lower bound for spike-time differentiation. Following Figure 4.1 depicts different calculations of reliability for vRD and Figure 4.2 for VPD under standard calibration without parameter alterations. It can be observed that a cost of $\tau = 1e^{-6}\text{ s}$ (or $q = 1e^6\text{ Hz}$) is too sensitive and overemphasizes distance measurements, while $\tau = 1e^{-4}\text{ s}$ (or $q = 1e^4\text{ Hz}$) is too coarse, showing clear steps in the distribution. This behavior suggests a shift towards spike counting rather than distance-based measurement. At $\tau = 1e^{-5}\text{ s}$ (or $q = 1e^5\text{ Hz}$), the distribution of distances appears smooth, indicating a balanced regime of the cost factors.

4.1.2 Measured Distances

Following, the chip [*hxcube1fpga3chip83-1*] was analyzed under the three situations described in Chapter 3. A standard calibration file in binary format was loaded as the initial configuration of the PyNN experiment. Chosen as cost factors were $\tau = 1e^{-5}\text{ s}$ and $q = 1e^5\text{ Hz}$ showing the best results in the previous tests.

For the first experiment, no parameters were altered after loading the configuration. In the second experiment, *leak_i.bias* = 100 was set. Additionally, the third experiment sets *reset_i.bias* = 1000. Both values were chosen as mean values within the loaded calibration

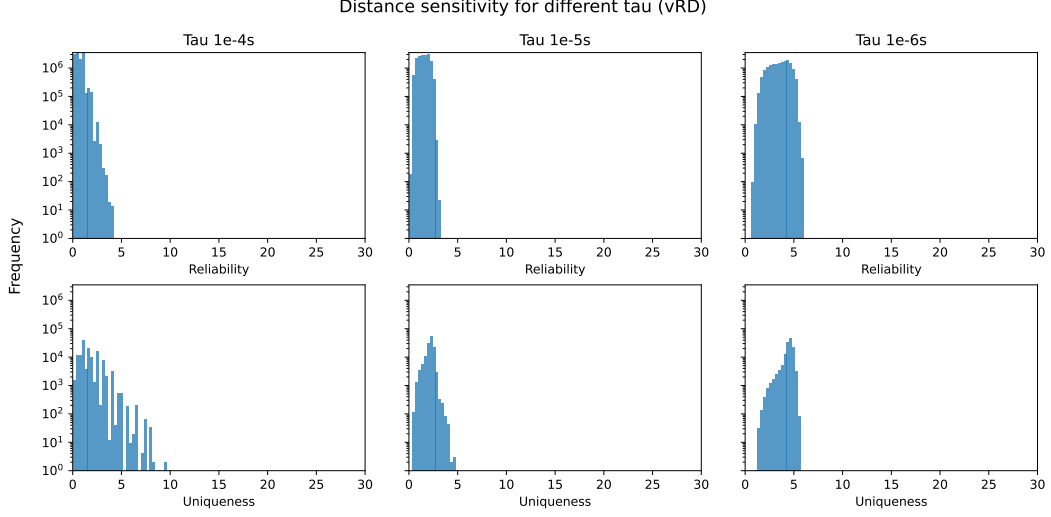


Figure 4.1: Comparison of different τ for reliability and uniqueness measurement with vRD for the first spiketrain tested

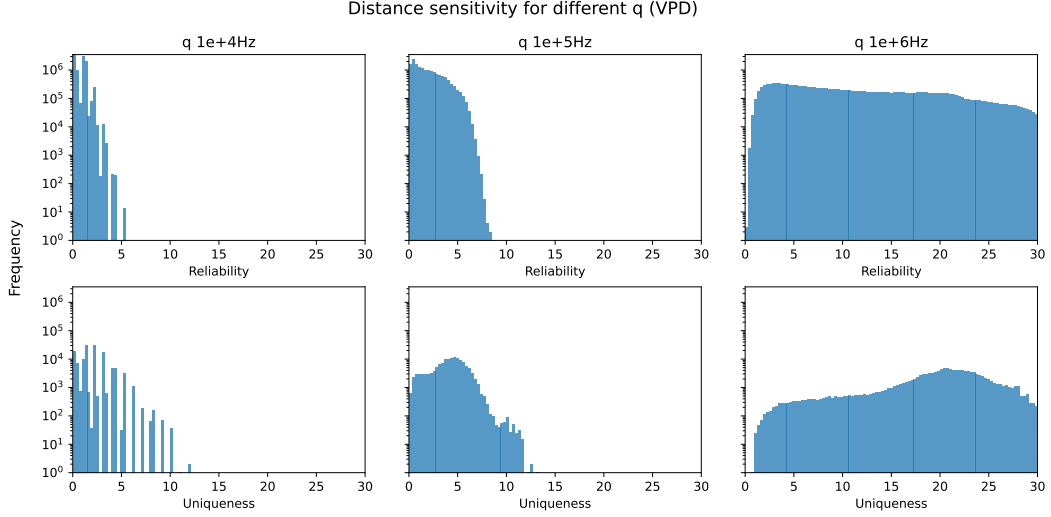


Figure 4.2: Comparison of different q for reliability and uniqueness measurement with VPD for the first spiketrain tested

file. Setting parameters to identical values can still provoke variation within the CapMem, as voltage drops within the circuit may occur.

Figure 4.3 shows the total distance variation of uniqueness and reliability for all three experiments. The upper row indicates that altering $leak_i_bias = 100$, and additionally $reset_i_bias = 1000$ in the third experiment, maintains the reliability trend and may even improve overall reliability. More importantly, in combination with the lower row of Figure 4.3, a clear broadening of the uniqueness distribution can be observed. This demonstrates that altering parameters can be used to achieve more unique neuron behavior.

An interesting observation is the spike in distance within the third reliability experiment. Since the uniqueness does not show a similar trend, this suggests that an event occurred during the repetitions. Inspecting singular neuron performances neuron 101 showed to have an alteration within its response after the 90th repetition in the experiment. This explains the spike observed in Figure 4.3. The underlying cause is not known. Figure 7.1 in the appendix

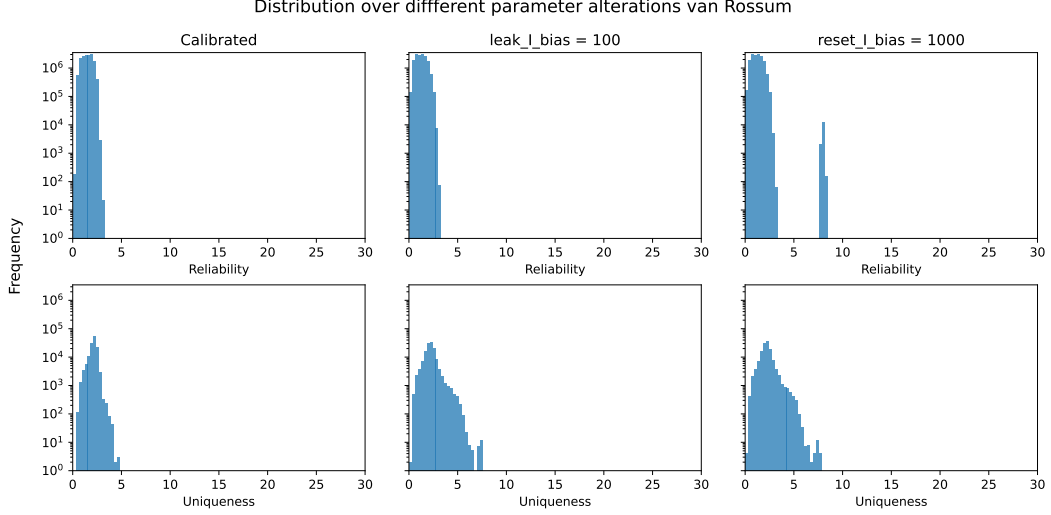


Figure 4.3: Distribution of PUF similarity metrics for vRD with $\tau = 1e^{-5}$ s for the first spiketrain tested. Reliability includes all distances for 512 neurons with 250 repetitions. Uniqueness includes distances of 512 neurons compared to eachother

shows a heatmap comparison of the experiments displaying that neuron 101 displays the maximum distance values of all neurons starting at around the 90th repetition.

Considering a second spike train using the VPD in Figure 4.4, similar observations can be made. Although the reliability distribution widens in the altered experiments, the contrast to the corresponding uniqueness distribution remains acceptable. Compared to the vRD metric in Figure 4.5, the VPD exhibits a wider distribution. This can be attributed to the higher sensitivity of the VPD, where precision may be reduced for larger distances. The VPD produces discrete distances based on spike shifts, whereas the vRD provides a more continuous and precise measure due to the decaying surface calculation of overlapping spike trains.

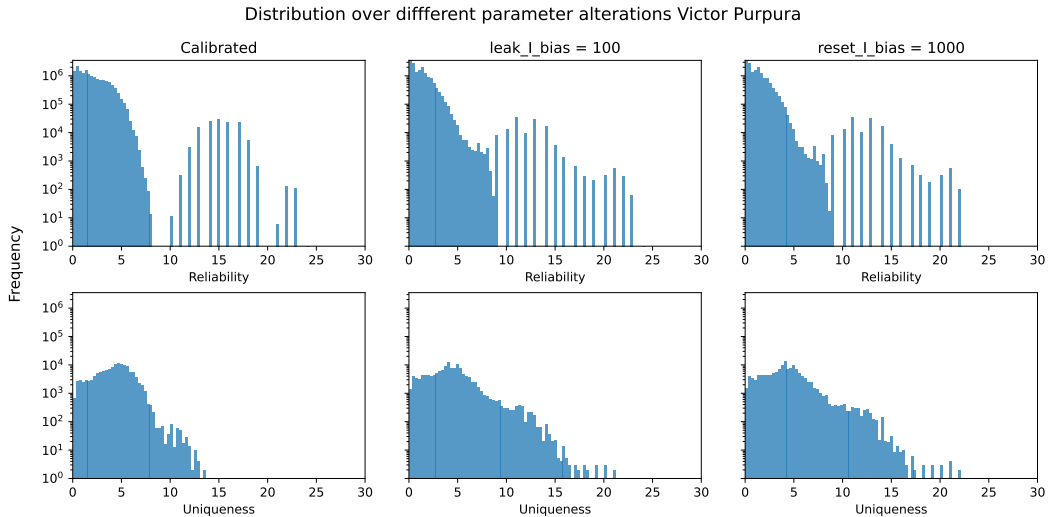


Figure 4.4: Distribution of PUF similarity metrics for VPD with $q = 1e^5$ Hz for the second spiketrain tested. Reliability includes all distances for 512 neurons with 250 repetitions. Uniqueness includes distances of 512 neurons compared to eachother

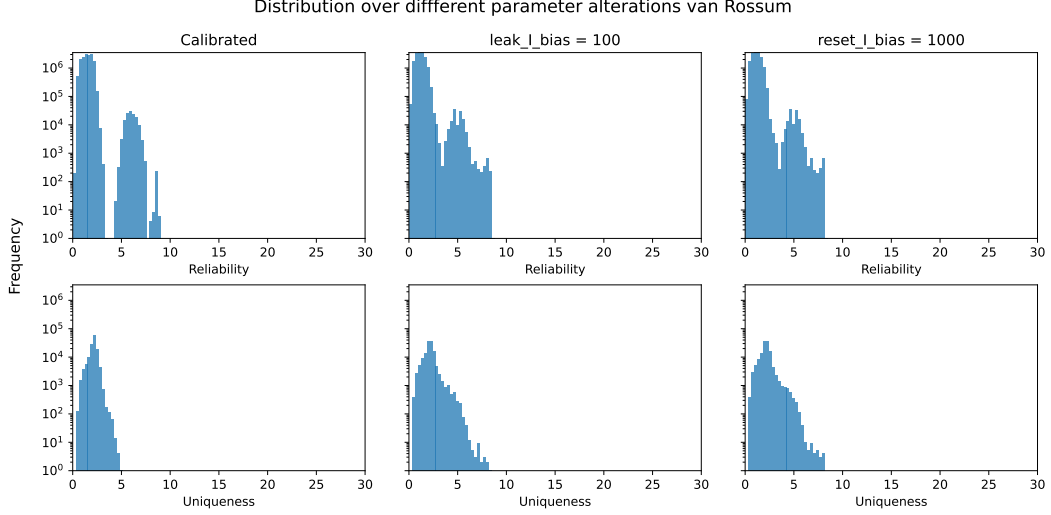


Figure 4.5: Distribution of PUF similarity metrics for vRD with $\tau = 1e^{-5}$ s for the second spiketrain tested. Reliability includes all distances for 512 neurons with 250 repetitions. Uniqueness includes distances of 512 neurons compared to each other

4.2 PUF Metric Training on a SNN

In the following, the distance metrics were applied to the SNN introduced in Chapter 3, which was trained based on the distances of the produced spike trains. It is important to note that the VPD is not suitable for training within neural networks, as the spike removal and insertion operations are not differentiable. For this reason, only the vRD was used.

All experimental runs were performed on the chip `['hxcube1fpga3chip83_1']` with a standard calibration. The hardware parameters `leak_i_bias = 200` and `reset_i_bias = 1000` were altered in all analog neuron circuits used in the experiments. `hxtorch` required scales for model to hardware translation were set to `weight_scale = 57.180171966552734` and `trace_scale = 0.02431616373360157`

The parameters chosen for the experiment were 100 time steps with a resolution of $\Delta t = 2 \mu s$, resulting in a total simulation time of $200 \mu s$. For the reliability calculation, 250 repetitions were used. Instead of performing 250 individual forward passes, a batched approach was applied. Each forward pass processes an input of size $[T, B, N]$, where T represents time steps, B the number of repetitions (batch size), and N the number of neurons. This allows the forward pass to be executed once while collecting all required spike data. The distances are then computed using a custom tensor-based implementation of the vRD, followed by backpropagation to update the trainable parameters.

The parameter τ was selected according to the simulation time and membrane time constant τ_m . Given a total simulation time of $200 \mu s$ and $\tau_m = 1e^{-5}$ s, multiple values of τ were tested under the constraint $\tau < T \cdot \Delta t$, $\tau > 2 \cdot \Delta t$ and $\tau \geq \tau_m$, as issue arises under the chosen time resolution and the membrane time constant. The resulting loss signal of the calculated metrics is too small and sensitive for $\tau \leq \sim 10 \mu s$. Choosing a smaller τ_m in combination with a finer time resolution and even smaller learning rate (LR) could allow the selection of smaller τ for the vRD. Therefore, $\tau = 1e^{-4}$ was selected, as it provides a sufficient loss signal of the distance measure for training.

The weight parameters of the network were initialized based on tested configurations, with $W_I = 0.8044$ and $W_R = 0.2401$. This configuration produces reliable yet distinct spiking behavior from the beginning, avoiding sudden changes such as transitions from silent to active neurons, which were observed in earlier tests.

Avoiding silent neuron behavior is an additional challenge encountered during initial experiments. The silent neuron behaviour is a result seen in early training. The optimization of the network results in stabilizing one PUF instance to reliably produce a spike output while the weights are tuned as that the second PUF instance does not produce spikes at all. While this increases uniqueness and reliability, as only the active PUF instance influences the loss, it is an undesired result. To address this, a spike count-based mean squared error (MSE) loss was introduced. This additional loss term enforces a target spike count for each PUF instance across all batches, guiding the network towards a controlled and usable spiking regime for metric evaluation.

During further testing, it was observed that the MSE term can dominate the overall loss, resulting in limited optimization of uniqueness and reliability. Early improvements in these metrics were primarily a side effect of the MSE pushing the network towards the target spike count. Although the resulting uniqueness and reliability were acceptable, the solution was suboptimal, as the weight parameters tended to saturate towards their maximum values for the given target. To mitigate this, the MSE term was only applied during the initial training phase. After a fixed number of epochs, the loss function was reduced to the formulation described in Chapter 3, allowing direct optimization of uniqueness and reliability.

Finding a stable training configuration required evaluating different values of the hyperparameter α . Figure 4.6 shows two experiments using the extreme values $\alpha = 1.0$ and $\alpha = 0.0$. The learning rate was set to 0.00009 and reduced by an exponential scheduler *StepLR* (*step_size* = 500, $\gamma = 0.95$) every 500 epochs.

For $\alpha = 0.0$, the loss simplifies to $L = \frac{1}{\text{Uniqueness}}$, fully prioritizing uniqueness over reliability. In this case, the uniqueness steadily increases over the epochs. Although the training process is noisy, a clear upward trend can be observed in Figure 4.6, indicating that the loss formulation effectively promotes uniqueness.

In contrast, $\alpha = 1.0$ results in $L = \text{Reliability}$. Here, reliability decreases over time, but the training appears highly unstable. The fluctuations between epochs are pronounced, which is amplified by the logarithmic scale. Small parameter updates can lead to noticeable shifts in spike timing, resulting in large variations in the measured distance. Consequently, the network struggles to consistently improve reliability. Additionally, uniqueness does not show sustained improvement.

Since $\alpha = 0.0$ and intermediate values such as $\alpha = 0.5$ showed more promising behavior improving uniqueness while maintaining acceptable reliability, $\alpha = 0.5$ shows the best trade off. In this setting, the network shows the more balanced optimization, improving uniqueness while simultaneously stabilizing reliability on all consecutive runs. Important to note is the initial improvement for the reliability to be attributed to the MSE term, but its effect diminishes once the MSE is cutoff at the 250th epoch.

To further evaluate this configuration, three independent training runs were performed using $\alpha = 0.5$, a learning rate of 0.00009, a scheduler defined by *StepLR* (*step_size* = 500, $\gamma = 0.95$), and $\Delta t = 2, \mu s$. All runs exhibit consistent training behavior, indicating stable optimization dynamics. Even if run one (left plot) shows not the exact behaviour as the two consecutive ones, the spike output, shown in Figure 4.8, is comparable to the other runs.



Figure 4.6: Evolution of distance metrics and loss over 3000 epochs under equal conditions for different α . Dashed line shows cutoff of MSE in loss



Figure 4.7: Evolution of distance metrics and loss over 5000 epochs in trial-to-trial variation tests under equal conditions for $\alpha = 0.5$. Dashed line shows cutoff of MSE in loss

More importantly, the spike trains observed in the final epoch show highly similar spiking patterns across all runs. This demonstrates the reproducibility of the trained network and indicates that the optimization converges towards a consistent solution under the given configuration.

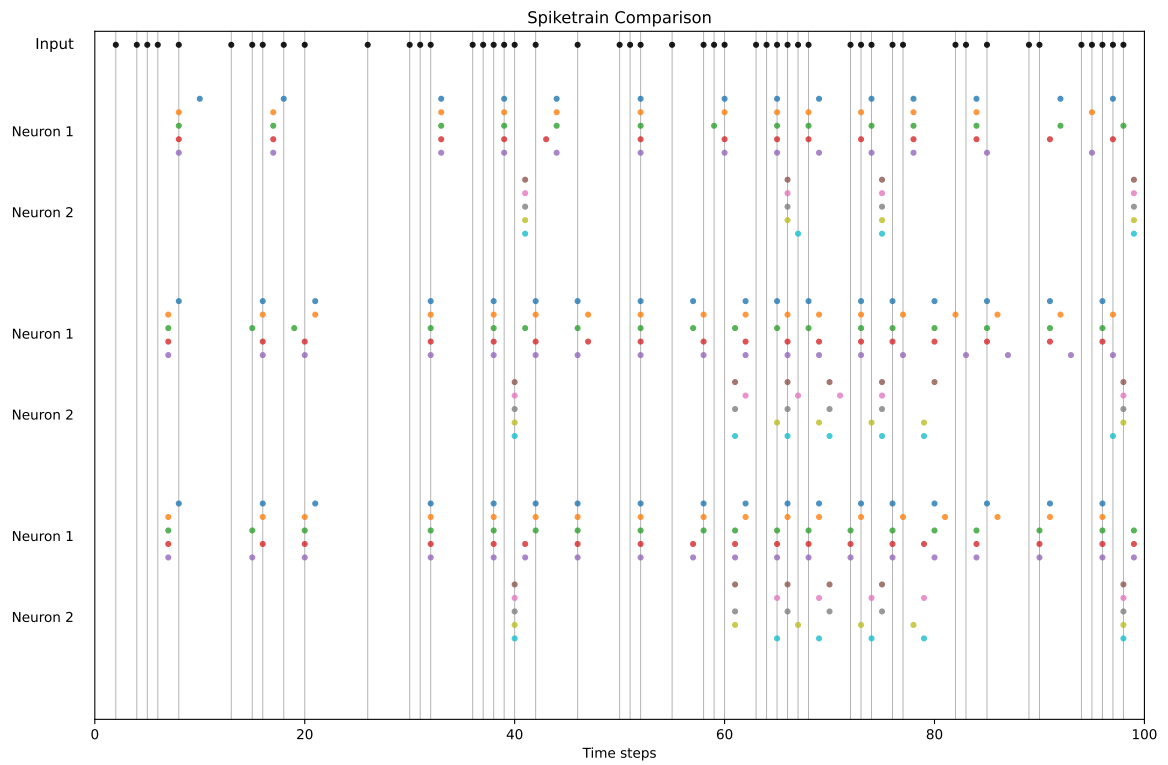


Figure 4.8: Comparison of spike trains produced by neurons in training runs shown in Figure 4.7 over 100 time steps with $dt = 2\mu s$

5 Discussion

The BrainScaleS-2 neuron circuits exhibit inherent variability that can be exploited through parameter manipulation. The results presented in Chapter 4 demonstrate the presence of neuron-to-neuron variability measured in uniqueness while maintaining stable reliability. In particular, uniqueness can be increased without degrading reliability when appropriate parameters are adjusted. This supports the suitability of the neuron circuits for PUF implementations.

Furthermore, the vRD and VPD demonstrate the applicability of spike train similarity metrics for evaluating PUF quality metrics such as reliability and uniqueness. However, their usability differs depending on the context. The VPD is effective for static evaluation but is not suitable for time-continuous learning scenarios such as training SNNs, as its spike insertion and removal operations are not differentiable in backpropagation. In contrast, the vRD can be used for training, provided that its parameters are carefully tuned. In particular, the parameter τ must be chosen in accordance with the membrane time constant. As *hxtorch* operates on time-binned representations of continuous signals, the relationship between τ , the time step size Δt and the membrane time constant becomes critical. Adjusting the membrane time constant, time resolution and the LR could resolve constraints to choose a smaller value of τ .

The observed dynamics of the simple network introduced in Chapter 3 and evaluated in Chapter 4 further demonstrate that training based on these metrics is feasible. However, compared to established ML tasks, the optimization process is significantly more sensitive and noisy. This is due to the fact that the metrics are computed on raw, unencoded spike train data, which inherently contains temporal jitter on the order of a few μs .

This sensitivity is also reflected in the introduction of the MSE loss to mitigate silent neuron behavior. While effective, this additional loss term was observed to dominate the overall optimization process when applied throughout training. Restricting its use to the initial training phase proved to be a more suitable strategy. After its removal, the remaining learning behavior can be attributed to the loss based on the PUF metrics.

Finally, the experiments on the hyperparameter α show that the network can be trained to prioritize either uniqueness or reliability. The results indicate that uniqueness offers greater potential for improvement, whereas reliability tends to operate within a narrower range and exhibits fluctuant behavior as it approaches lower values. This suggests that the system already provides a baseline level of reliability that is difficult to further optimize, while uniqueness remains the more flexible and influential metric during training tuning the networks weight parameters.

6 Conclusion

The results of this work demonstrate that BrainScaleS-2 neuron circuits and SNNs implemented on them can serve as a viable foundation for PUF entities. The combination of inherent neuron-to-neuron variability and controllability through parameter manipulation enables the realization of neuromorphic PUF designs.

It was shown that uniqueness can be increased through parameter adjustment while maintaining stable reliability, highlighting the suitability of the underlying hardware for PUF applications. In addition, spike train similarity metrics were successfully applied to evaluate these properties, and training of a SNN based on such metrics was demonstrated to be feasible under appropriate conditions.

Although the investigated PUF-SNN instance was limited to a simple single LIF neuron and focused on the interaction between two PUF-SNN instances, the results provide a proof of concept for combining neuromorphic systems with PUF principles.

Overall, this work indicates that neuromorphic hardware, even due to its inherent noise and variability, can be systematically utilized for security applications, opening a path towards more complex and scalable neuromorphic PUF designs.

7 Outlook

Building on the successful evaluation of BrainScaleS-2 neuron circuits within a simple SNN structure, future work will extend this approach to larger and more complex network architectures deployed in parallel on the chip. Initial steps have already been taken by implementing a modular test pipeline, enabling the deployment and evaluation of larger networks with dedicated forward passes.

Additionally, the optimization will be extended from solely optimizing weight parameters to also include selected LIF neuron hardware parameters, potentially leading to improved and more stable training performance.

A key next step is the definition of a challenge-response procedure. This includes determining how spike trains produced by the network can be interpreted as responses, as well as defining suitable challenges that reliably induce distinguishable outputs.

In addition, methods for stabilizing the inherently noisy output behavior must be developed. One promising approach is the use of fuzzy extractors, which enable the derivation of stable cryptographic keys from noisy PUF responses.

Finally, a unified neuromorphic PUF design needs to be established. This includes the integration of a well-defined CRP space, response stabilization mechanisms, and a consistent definition of the PUF entity within the system.

APPENDIX

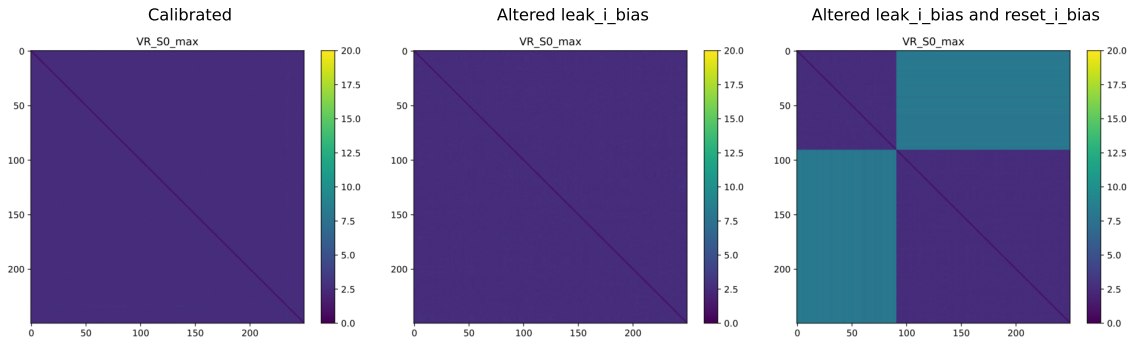


Figure 7.1: Heatmap comparison for maximum values appearing within the reliability measurement. Showing a sudden rise in distance around the 90th repetition. Behaviour is connected to neuron 101 changing the spike train response

Acronyms

AdEx adaptive exponential integrate-and-fire. 6

CapMem capacitive memory. 6, 12

CRP challenge response pair. 5, 9, 19

FIA fault injection attack. 3

ITL in-the-loop. 3, 6

LIF leaky-integrate-fire. 9

ML machine learning. 3, 9, 17

MSE mean squared error. 14, 15, 17

pPUF public physical unclonable function. 5

PUF physical unclonable function. 3–6, 8, 9, 13, 14, 17–19

PUF-SNN physical unclonable function spiking neural network. 9

rPUF reconfigurable physical unclonable function. 5

SIMD single instruction multiple data. 6

SNN spiking neural network. 3, 4, 8, 9, 13, 17–19

VPD Victor-Purpura distance. 6–9, 11–13, 17

vRD van Rossum distance. 7–14, 17

Bibliography

- Akter, Sonia, Kasem Khalil, and Magdy Bayoumi (2023). “A Survey on Hardware Security: Current Trends and Challenges”. In: *IEEE Access* 11, pp. 77543–77565. DOI: 10.1109/ACCESS.2023.3288696.
- Chang, Chip-Hong, Yue Zheng, and Le Zhang (2017). “A Retrospective and a Look Forward: Fifteen Years of Physical Unclonable Function Advancement”. In: *IEEE Circuits and Systems Magazine* 17.3, pp. 32–62. DOI: 10.1109/MCAS.2017.2713305.
- Chatterjee, Durba et al. (Aug. 2025). “Hardware Security in the Connected World”. In: *WIREs Data Mining and Knowledge Discovery* 15.3. ISSN: 1942-4795. DOI: 10.1002/widm.70034.
- Cramer, Benjamin et al. (2022). “Surrogate gradients for analog neuromorphic computing”. In: *Proc. Natl. Acad. Sci. U.S.A.* 119 (4) e2109194119 119.4, e2109194119. DOI: 10.1073/pnas.2109194119.
- Denker, M., A. Yegenoglu, and S. Grün (2018). “Collaborative HPC-enabled workflows on the HBP Collaboratory using the Elephant framework”. In: *Neuroinformatics 2018*, P19. DOI: 10.12751/incf.ni2018.0019.
- Elshamy, Mohamed and Haralampos-G. Stratigopoulos (2021). “Neuron-PUF: Physical Unclonable Function Based on a Single Spiking Neuron”. In: *2021 IEEE 27th International Symposium on On-Line Testing and Robust System Design (IOLTS)* —. IEEE. DOI: 10.1109/IOLTS52814.2021.9486716.
- Gao, Yansong, Said Al-Sarawi, and Derek Abbott (Feb. 2020). “Physical unclonable functions”. In: *Nature Electronics* 3, pp. 81–91. DOI: 10.1038/s41928-020-0372-5.
- Gasarch, William et al. (2021). *Alternative Paradigms of Computation*. arXiv: 2111.08916 [cs.ET].
- Hadis Takaloo, Majid Ahmadi and Arash Ahmadi (2022). “Spiking Neurons: A New Entropy Source for Physically Unclonable Functions”. In: *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*. ISSN: 978-1-6654-8485-5/22. DOI: 10.1109/ISCAS48785.2022.993774.
- Hock, M. et al. (Sept. 2013). “An analog dynamic memory array for neuromorphic hardware”. In: *Circuit Theory and Design (ECCTD), 2013 European Conference on*, pp. 1–4. DOI: 10.1109/ECCTD.2013.6662229.
- Mathew, Sanu K., Raghavan Kumar, and Vivek De (2025). “Circuits for Secure Root-of-Trust: TRNG, PUF, Attack-Resistant Cryptographic Accelerators [Feature]”. In: *IEEE Circuits and Systems Magazine* 25.4, pp. 55–63. DOI: 10.1109/MCAS.2025.3595508.
- Müller, Eric et al. (2022). “A Scalable Approach to Modeling on Accelerated Neuromorphic Hardware”. In: *Frontiers in Neuroscience* Volume 16 - 2022. ISSN: 1662-453X. DOI: 10.3389/fnins.2022.884128.
- Pehle, Christian et al. (2022). “The BrainScaleS-2 Accelerated Neuromorphic System with Hybrid Plasticity”. In: *Front. Neurosci.* 16. ISSN: 1662-453X. DOI: 10.3389/fnins.2022.795876. arXiv: 2201.11063 [cs.NE].
- Rossum, M. C. W. van (2001). “A Novel Spike Distance”. In: *Neural Computation* 13.4, pp. 751–763. DOI: 10.1162/089976601300014321.
- Sergio Vinagrero Gutierrez, Elena-Ioana Vatajelu and Giorgio Di Natale (2026). “A Revision of the Quality Metrics of Physical Unclonable Functions”. In: IEEE. DOI: 10.1109/TVLSI.2026.3664178.

Victor, Jonathan D and Keith P Purpura (Jan. 1997). “Metric-space analysis of spike trains: theory, algorithms and application”. In: *Network: Computation in Neural Systems* 8.2, pp. 127–164. ISSN: 1361-6536. DOI: https://doi.org/10.1088/0954-898X_8_2_003.

Funding acknowledgement

The work carried out in this Studienarbeit used systems, which received funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreements Nos. 720270, 785907 and 945539 (Human Brain Project, HBP) and Horizon Europe grant agreement No.

Reproducibility statement

Table 7.1: Repository versions used for reproducibility

Repository	Commit Hash
pynn-brainscales	6d0598b195d3dc9e07617c32d0b02bbfb2a15779
hxcomm	682878fe49d3a4e8a6f4c2f58105abae5edb461d
haldls	de3157189eefcc296403d65e6cf700425007bdfe
grenade	34026da0619d4c90158fda55d796d93617bf32c8
hate	b8c8fd03b6fdc103022848a47bc6838d46b3d3ae
calix	1481598b7989003c008741a533d9600607f07286
code-format	512f1fafb884a7da30a1943793a3cae6441462be
logger	73dadb3ce413c521845ef7d36f818073eee4fefa
sctrltp	7a94faf9af3d38b202276d11de44182e3602b8cd
rant	53199ee94cae1e1c2e4db10e88d570a761b14e0f
hwdb	8877d48a6276f1eade6e8bcabe6c8df2fbcce031a
visions-slurm	b32eaa77766fded68524a51980a119da975a9ac1
flange	522fdd23830d76a072691946e656e6843c43d00d
lib-rcf	469f4b592207902064c1523b3f2df4b501552eba
bss-hw-params	74c6403c4de9a0f85e2f6b3380daf49f553f4e7a
halco	f0fcb56293ff377f3d11cb65e2af13f00e5f0262
fisch	1cfde2f888ec6c95a53adbe73ac90ca63df7bc26
linux	a69d086ba506ac72b60ed10f210be8e613dc2dd4
dapr	b14f8cdde6800d4a10fa2e613a5c7f3e47d328b3
pywrap	5e2af30e9593882b471d3cd02df00b93f13ff479
ztl	c2d4faee05f497010ee55e35bf9c9607eecbf884
lib-boost-patches	136c5b41cb046afe2c726aa4646928bf5190622e
model-hardware-puf	ed599c2eba0873106d6c640e526d41047126830d
hxtorch	aff0192675e6f018131efc359f267a1e6dcee8fd
nir	72f79ef350132beb5658508d682b5ebc7fc9a0de

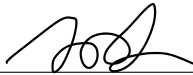
Declaration

I hereby declare that I have written this Studienarbeit independently and that I have not used any sources or aids other than those indicated.

Statement on the Use of AI/GenAI

During the preparation of this Studienarbeit, generative / artificial intelligence-based tools were used in a supportive manner, as listed below.

- Generative AI was used for proofreading, spellchecking, and improving the readability of the written report.
- Generative AI was used for initial code scaffolding for the plots presented in this report.



Heidelberg, May 20, 2026