Advanced Analysis of Cell Stability for Reliable SRAM PUFs

Alison Hosey, Md. Tauhidur Rahman, Kan Xiao, Domenic Forte, and Mohammad Tehranipoor
ECE Department, University of Connecticut
{ aph09003, tauhid, kanxiao, forte, tehrani}@engr.uconn.edu

Abstract

A Physically Unclonable Function (PUF) is a structure that when issued a challenge, it produces a unique and reliable response which can be used as an identifier or a cryptographic key. SRAM PUFs create unique responses upon power up as certain SRAM cells output a 1 or 0 with high probability due to uncontrollable process variations. A current challenge in SRAM PUFs is their sensitivity to temperature and voltage variations as well as aging. By creating algorithms that isolate stable bits quickly and with minimal testing, the use of SRAM PUF should become more practical. In this paper, we explore the selection of stable bits through enrollment under different conditions (temperature, voltage, and aging) and also by exploiting previously undiscovered interactions between neighboring SRAM cells. We develop metrics that analyze the impact of each neighboring cell and each enrollment condition. Our metrics can be used to identify the best cells and conditions for stable bit selection. We have analyzed data from Spartan 3 FPGA and our metrics identify the best neighborhood size (16 stable neighbors) and best enrollment condition pair (high temperature, high voltage and low temperature).

I. Introduction

Physical unclonable function is an emerging potential security block for generating volatile secret keys in cryptographic applications [5], [6], [7], [8], [9], [10], [11]. Authentication, identification, counterfeit detection and cryptographic key are the main applications of PUFs [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11]. PUFs offer a high level of protection in cryptographic applications with strong volatile key storage. PUFs are unclonable and inexpensive by nature. They create a unique identifier by utilizing the uncontrollable process variations that affect integrated circuits. PUFs are issued a challenge and (ideally) produce a unique and reliable response in return. Since this response is unique to the device, it can therefore be used as a device ID or key. Unlike previous methods, PUFs are less vulnerable to attacks, and also require no additional manufacturing steps. A variety of different types of PUFs have been explored recently, including Arbiter PUF [5], ring-oscillator (RO)-PUF [6], [14], SRAM PUF [7], [10], Latch PUF [8], and many more. This paper explores utilizing the popular SRAM PUF for identification and cryptographic key generation. SRAM PUF offers the convenience of using commonly available and integrated SRAM (instead of including a dedicated hardware in the circuit) as well as the capacity to provide large enough outputs for identifier/key generation/storage [7], [10].

Although SRAM PUF offers many appealing features, there are two main challenges to its current application. First, SRAM PUF is quite sensitive to noise generated by temperature and voltage level variations [10] [11] [13] This sensitivity is not unique to SRAMs and can be seen in a variety of electronics due to physical phenomena which alter threshold voltages and other properties [9], [12], [14], [16], [17]. Second, SRAM PUF also experiences the effects of aging on the reliability of the output [10]. Previously, error correcting code (ECC) [9], [10], [17] had been used to repair errors in the SRAM output prior to use as a cryptographic key. Unfortunately, ECC creates a substantial amount of overhead while also increasing the likelihood of secure information being leaked [9], [10], [15], [16], [17]. Alternatively, exhaustive measurements of the SRAM PUF can be used to identify the most reliable cells [10], [11]. However, these are costly and do not extend well for devices produced at high volume.

In our previous work, we have identified that neighboring SRAM cells can be used to determine the more reliable SRAM PUF cells [10]. Based on our observation, we presented a simple bit selection algorithm. While the SRAM PUF reliability improved dramatically compared to random bit selection, our approach was mostly operating in a blind fashion. Here we perform more advanced analysis in order to identify the best cells and conditions. Our main contributions include:

- Development of three new metrics to analyze the relationships between neighboring SRAM cells in one dimension – 1D (i.e., assuming that the SRAM data is organized as one long array) and the influence of environmental conditions.
- Utilization of these proposed metrics, real SRAM data (over 1 billion measurement data) are examined in greater detail and the effect of different temperature, voltage, and aging conditions on reliability can be more easily categorized. We determine the optimal window and threshold sizes. We also identify the most efficient measurement conditions for initial enrollment.

The rest of the paper is organized as follows. In Section II, we describe the details of SRAM PUF application in previous work. The proposed metrics for neighboring SRAM cell analysis in 1D are described in Section III. Section IV will provide the results of our experimental test setup. The conclusions drawn from this work as well as intended future work will be given in Section V.

II. Background

In recent years, there has been much investigation into SRAM PUF as a simple but effective form of hardware-based identification and key generation/storage. The usage of SRAM as the PUF medium is appealing for a variety of reasons. Most notably, SRAM is commonly available in most systems and therefore does not require additional hardware. The uniqueness of output from one SRAM compared to another is also the largest among existing PUFs [11] [12] Each SRAM is composed of cells which each contain a pair of cross-coupled inverters. Due to uncontrollable variations in the manufacturing process, different CMOS devices have different physical parameters (e.g. doping-levels, transistor oxide thickness, etc.). When an SRAM is powered-up, these variations affect the power-up state of their associated cells. It has been observed that certain cells have a strong preference to power-up to a 1 or 0 state. Cells that have no preference are deemed neutral and power-up at random depending upon the influences of environmental noise [13]. The more useful cells for PUF output are the ones that strongly prefer 0 or 1.

By examining the power-up state of an SRAM, a unique identifier can be created because the process variations and resultant preferences are truly random (being entirely dependent upon a physical anomaly). An SRAM PUF identification system would be similar to the fingerprint security measures found in biometrics. When fingerprints are used for identification, the fingerprint is a unique identifier which is compared to other fingerprints in a database and subsequently accepted or rejected based upon its authenticity. The SRAM PUF output (which is ideally a unique and reliable response) would then be compared to a known SRAM PUF response database and verified.

To ensure the reliability of the SRAM PUF response, the output needs to either be error corrected in post-processing or analyzed in such a way that only stable cells are used. ECC uses specialized encoding and decoding processes to ameliorate data instability. Unfortunately, such processes create considerable overhead for implementation. For example, [14] shows ECC requires additional $\sim 87k$ gates for 128-bit reliable key. For the latter, a bit selection algorithm was recently proposed [10]. The basis of this algorithm is to identify the most stable bits based upon the performance of their neighboring cells in a series of enrollment tests (testing under some extreme corner conditions). The algorithm finds the cells which consistently only output a value of '0' or '1' and then ranks their overall stability using a weighting algorithm. The weighting algorithm takes into account the number of stable and unstable neighbors a cell has. Essentially, the farther away a cell is from the nearest unstable cell (weight 0), the higher weighted value it is assigned. As shown in Fig. 1, stable bits (labeled S) and unstable bits (labeled U) are assigned based upon these calculated stability weights. The cells that have a weight greater than a predefined threshold are selected as the most stable ones. While the measurements used in the algorithm rely only upon fresh (un-aged) SRAM, results showed that the algorithm was most successful in identifying cells that were robust against aging (stable over the lifetime of the device). Further, since the algorithm only requires measurements at a few corner conditions, it allows for a faster enrollment as well especially when considered for high-volume products. Finally, identification of the most stable cells to reduce the size of ECC, thereby decreasing cost and secret leakage.

III. SRAM Cell Stability Analysis with Proposed Metrics

In this section, we explain the proposed metrics for capturing the relationships between SRAM cells. While intuition tells us that cells should be most influenced by their neighbors, we do not necessarily know the physical locations of all the cells (proprietary information) in a memory. Hence, we shall assume 1D analysis for simplicity.

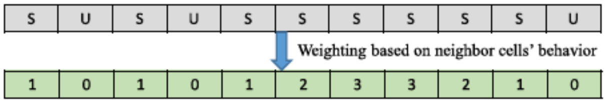
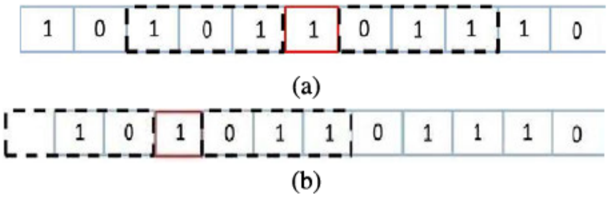


Figure 1: Bit Selection Weighting Algorithm (Top: Stable (S) and Unstable (U) bits; Bottom: Associated Weighted Value) .



(a)

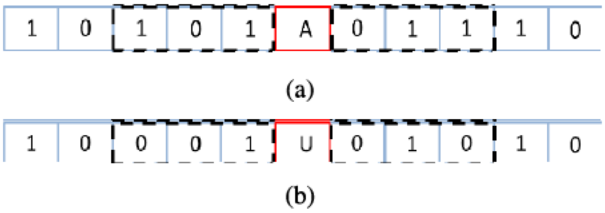
(b)

Figure 2: Target cell with (a)(red X) with $w = 6$ (dashed outline) and (b) extends beyond the array (target cell X).

1D analysis means that the SRAM cells are assumed to be ordered in one long array based on their logical address. One benefit of our metrics is that they may end up exposing the physical locations based on the correlations between cells they capture.

We begin by defining some notation and important variables. Let C represent the conditional probability of a bit not flipping; w is the the window size used to assess the neighborhood sum; and t is the threshold weight of a neighborhood sum (Fig. 2). The window size w is the number of cells examined on either side of the target cell. This value can only be of even value as we only analyze neighbors around the target cell in a symmetric fashion. For example, if $w = 6$, then the neighborhood of the target cell would include 3 cells on the left of the target cell and 3 cells on the right of the target cell. This is shown in 2(a). In cases where the window size around a target cell expands beyond the array of cells (2(b)), the analyses from that target cell are ignored for consistency in analysis.

In determining the most stable cells, the number of stable neighbors must be greater than or equal to t. Consider window size $w = 6$. If $t = 4$, then we only consider the target cell to be acceptably stable if at least 4 of its 6 neighbors analyzed are stable. In 3(a), this condition is true and therefore the target cell would be accepted by our bit selection algorithm. In 3(b), the target cell is not accepted because it does not have enough stable neighbors.



(a)

(b)

Figure 3: Window includes (a) 4 stable cells (if T = 4, target cell is accepted (A)) and (b) 2 stable cells (if T = 4, target cell is not accepted (U)).

Based upon these variables, conditional probability can be used to concisely describe the dependence of an SRAM cell upon the stability of its neighbors. In essence, conditional probability $P(A|B)$ is the likelihood of an event A occurring given that a separate event B has occurred. Conditional probability can be written as

$$P(A|B) = \frac{P(A,B)}{P(B)} \tag{1}$$

where $P(A,B)$ denotes the probability of events A and B occurring while $P(B)$ represents the probability of only B occurring. In our analyses, we define conditional probability for the probability of a cell being stable (i.e. not flipping) given that the neighbors analyzed are stable (i.e. do not flip). Event A is therefore the probability of a target cell not flipping and event B is the probability that the target cells neighbors do not flip. This relationship can be modified so that event B allows for only a certain percentage, x, of the neighboring cells to flip (i.e. if N is the number of neighbors, then $x = N/t$).

We investigate three metrics based on conditional probability: (i) *total neighborhood analysis* that captures the impact the number of stable neighbors in a window; (ii) *neighborhood pairs analysis* which examines the influence of particular cells in the window; (iii) *environmental analysis* that captures the effect of isolated environmental conditions on the relationships of all cells in a window.

A. Total Neighborhood Analysis

This form of analysis calculates the stability of the target cell based upon all of its neighbors within a specified window size w. We define the neighborhood based probability as

$$C_N = \frac{P(A,B;w,t)}{P(B;w,t)} \tag{2}$$

The notation P(X;y) means that the probability of a random event X depends on the variable y. In this case, y is not random, but chosen by the user. In Eqn. (2), event A is the probability of the target cell not flipping while event B is the probability of t or more neighbors within the window size, w, (i.e., if t or more cells within a distance, $d = w/2$ from the target cell) don't flip. This metric focuses on the changes in conditional probability for different window sizes and threshold values. In our prior work [10], we assumed (based on intuition) that the larger the threshold, the better the result. However, the number of stable bits selected is lesser as t increases. The benefit of this proposed metric is that the ideal combination of w and t for optimal stability results can be determined for an SRAM and provide trends which can be extended to the examination of all SRAMs of the same type. If there exists some $t < w$ that is better or just as good as $t = w$, then we will be able to identify more stable bits for the PUF key than in [10].

B. Neighborhood Pairs Analysis

The intention behind this approach is to determine the level of influence neighbors at specific intervals from the target cell have. We define the neighborhood pair based probability as

$$C_{NP} = \frac{P(A,B;w)}{P(B;w)} \tag{3}$$

Here event A is the same as before. Assuming the target cell is located at position i, we define event B as the probability that the cells located at $i - w/2$ and $i + w/2$ dont flip. The data of event B combines the data for the neighborhood at two isolated conditions. In this case, $t = 2$ (both of the two cells being examined must be stable for the target cell to be considered acceptable). The benefits of this analysis are that the influence of neighbors can be ranked. In the original bit selection algorithm (Section II), all neighbors were treated equally. However, with knowledge of the rank/influence of each neighbor, we could actually improve the bit selection. For example, neighbors with larger (smaller) influence can be given larger (smaller) weight in the sum calculation for each target cell. This would be a more accurate way of identifying the most stable cells.

C. Environmental Analysis

In order to assess the effectiveness of environmental conditions in finding the most stable cells, this method considers the stability of a target cell using neighborhood data when only a pair of isolated fresh SRAM conditions are used for enrollment. We define the environmental total neighborhood based probability as

$$C_{EN} = (P(A,B))/(P(B;c_1,c_2)) \tag{4}$$

Here event A remains unchanged with all conditions (for fresh or aged data) being used to determine target cell stability. Conversely, event B examines all of the neighbors in w using only a pair of temperature and voltage conditions (label as c_1 and c_2) and for fresh data only. For example, c_1 could be the high temperature, high voltage (HTHV) corner. While the definition of c_{EN} could be expanded to include any number of conditions, we shall only focus on choosing two conditions at a time to keep enrollment costs low. The time and cost of enrollment is directly related to the number of measurements taken from the PUF. The analysis of all of the possible pairs of condition combinations will demonstrate which condition pairs have the most inter-cell dependency and imply which enrollment tests can be most effectively extrapolated to determine the SRAM cells that are 'most stable' over time.

IV. Results and Analysis

A. Experimental Setup

Our results are based upon experiments conducted using the 1MB on-board SRAM of the Xilinx Spartan-3 FPGA board. The on-board SRAM provided a proper environment for the implementation of SRAM PUF, with the FPGA reading the SRAM and sending the data to a computer to be recorded. The various temperature and voltage conditions were achieved using a Thermostream system and power supply respectively.

Table I shows a list of the environmental conditions next to the abbreviations which will be used throughout the rest of this paper. Extensive burn-in was performed on the SRAM to create an accelerated aging process. We recorded results after 15 hours of accelerated aging to capture the cell stability at different times.

10 trials were performed at each of the 10 temperature/voltage combinations for each age (fresh and aged 15 hours). Each trial results in 4,194,304 bits of data being measured. For all the trials, conditions, etc. for which we

Table I: Temperature/Voltage Conditions versus Abbreviations Used.

Condition	Abbreviation
High Temperature	HT
High Temperature High Voltage	HTHV
High Temperature Low Voltage	HTLV
High Voltage	HV
Low Temperature	LT
Low Temperature High Voltage	LTHV
Low Temperature Low Voltage	LTLV
Low Voltage	LV
Nominal Conditions (Set 1)	N1
Nominal Conditions (Set 2)	N2

took measurements, this resulted in over 1 billion samples for computing conditional probabilities. Such an extensive data set allows for a full exploration of the entire SRAM and overall dependability in the results of the analysis.

Using the dependency relationship demonstrated by conditional probability, we were able to analyze the interactions of SRAM cells in 1D under the wide variety of conditions outlined earlier (voltages, temperatures, and aging). We performed the following experiments:

- Initial results used *total neighborhood analysis* (C_N) to determine the dependency of the target cell on its neighbors at various window sizes around a target cell for $1 \leq t \leq w$ (integer values only). Note, that the threshold can never exceed the window size. All temperature and voltage conditions were used for the fresh and aged 15 hours SRAM data sets to note the reliability of window size and threshold combinations as an SRAM ages. The conditional probability C_N for each combination of window sizes and thresholds was plotted individually for each aging condition examined (fresh, aged 2 hours, and aged 15 hours).
- Next, we wanted to highlight cells with the most influence on the target cell by using the more targeted approach of *neighborhood pairs analysis* (C_{NP}). In this scenario $t = 2$ for all window sizes tested. The range of tested window sizes was $1 \leq w \leq 4096$ (even values only). Conditional probability C_{NP} was plotted against window size and only fresh data was examined using this approach.
- Finally, *environmental analysis* C_{EN} was used to determine the effects of specific pairs of conditions on conditional probability in fresh data only. There were 10 available temperature/voltage environments, which would combine to 45 potential pairs, each with 7 evaluated window sizes $w = 2, 4, 8, 16, 24, 32, 40$.

B. Results and Discussion

The results below include the outcomes of all three experiments discussed above.

Total neighborhood analysis : As Fig. 4 demonstrates, the highest conditional probability (i.e. dependency of the target upon its neighborhood) varies for different combinations of window sizes and threshold values. For smaller window sizes, such as 2 through 24, the optimal threshold is equal to the window size. As the window size gets larger, this relationship alters. For these conditions, there is a peak conditional probability located at $t = w$. For larger window sizes (32 and 40), the conditional probability is mainly fixed until around $t = 25$ and $t = 30$ respectively. One possible explanation for this peculiar behavior at larger window sizes could be due to the number of samples. Although we examined a very large data set of over 15 million data points for each window size, we may not have enough samples where target cells are surrounded by 32 or 40 stable cells to draw a significant conclusion. Another thing missing from total neighborhood analysis is the impact of each cell in the neighborhood on the reliability of the target cell. This is covered by neighborhood pairs analysis below.

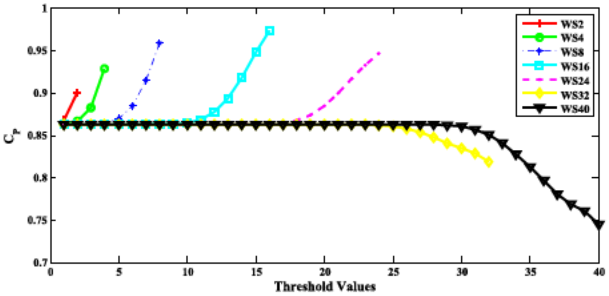


Figure 4: Total Neighborhood Analysis for Fresh Data.

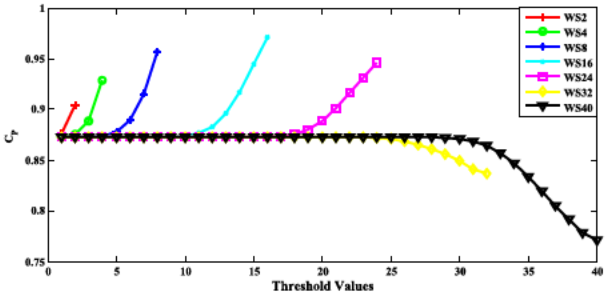


Figure 5: Total Neighborhood Analysis for Aged 15 Hour Data.

For SRAMs aged 15 hours SRAMs (Fig. 5), the trends for the conditional probability are similar to that of the fresh data (Fig. 4), but the conditional probability is higher. This makes intuitive sense. For fresh data, we may call some cells as stable that are not the most stable. After aging, these less stable cells may change value. Such cells would not be included to compute the conditional probabilities for aged data shown in Fig. 5.

Neighborhood pairs analysis : The plot in Fig. 6 demon-

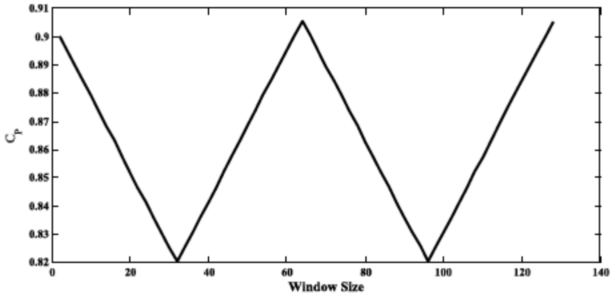


Figure 6: Neighborhood Pairs Analysis for Window Sizes 2-128.

Table II: Total Neighborhood Analysis at different operating conditions

Condition Pairs		Maximum Conditional Probablity at Each Window Size							Statistics	
Condition 1	Condition 2	2	4	8	16	24	32	40	Average	Max
HTHV	LT	0.8940	0.9183	0.9475	0.9631	0.9355	0.8630	0.8630	0.9121	0.9631
HTHV	LTHV	0.8939	0.9180	0.9472	0.9628	0.9355	0.8630	0.8630	0.9119	0.9628
HTHV	LTLV	0.8935	0.9174	0.9465	0.9621	0.9349	0.8630	0.8630	0.9115	0.9621
HTHV	N2	0.8927	0.9161	0.9446	0.9603	0.9330	0.8630	0.8630	0.9104	0.9603
HTHV	LV	0.8922	0.9153	0.9438	0.9592	0.9320	0.8630	0.8630	0.9098	0.9592
HTHV	N1	0.8923	0.9153	0.9437	0.9592	0.9320	0.8630	0.8630	0.9098	0.9592
HTHV	HV	0.8921	0.9150	0.9433	0.9588	0.9314	0.8630	0.8630	0.9095	0.9588
HTLV	LT	0.8918	0.9147	0.9429	0.9584	0.9312	0.8630	0.8630	0.9093	0.9584
HTLV	LTHV	0.8916	0.9143	0.9424	0.9579	0.9308	0.8630	0.8630	0.9090	0.9579
HTLV	LTLV	0.8912	0.9136	0.9415	0.9569	0.9298	0.8630	0.8630	0.9084	0.9569
HTHV	HTLV	0.8909	0.9131	0.9409	0.9563	0.9292	0.8630	0.8630	0.9081	0.9563
HT	LT	0.8906	0.9127	0.9403	0.9558	0.9285	0.8630	0.8630	0.9077	0.9558
HT	HTHV	0.8904	0.9122	0.9398	0.9552	0.9282	0.8630	0.8630	0.9074	0.9552
HT	LTHV	0.8904	0.9123	0.9398	0.9552	0.9281	0.8630	0.8630	0.9074	0.9552
HTLV	N2	0.8903	0.9120	0.9394	0.9547	0.9280	0.8630	0.8630	0.9072	0.9547
HT	LTLV	0.8900	0.9115	0.9388	0.9541	0.9271	0.8630	0.8630	0.9068	0.9541
HTLV	N1	0.8898	0.9112	0.9383	0.9537	0.9247	0.8630	0.8630	0.9062	0.9537
HTLV	LV	0.8895	0.9108	0.9377	0.9529	0.9264	0.8630	0.8630	0.9062	0.9529
HTLV	HV	0.8894	0.9105	0.9374	0.9527	0.9263	0.8630	0.8630	0.9060	0.9527
HT	N2	0.8892	0.9103	0.9371	0.9524	0.9255	0.8630	0.8630	0.9058	0.9524
HT	N1	0.8887	0.9095	0.9359	0.9511	0.9245	0.8630	0.8630	0.9051	0.9511
HT	HV	0.8880	0.9081	0.9342	0.9492	0.9228	0.8630	0.8630	0.9040	0.9492
HT	HTLV	0.8878	0.9079	0.9339	0.9490	0.9231	0.8630	0.8630	0.9040	0.9490
HT	LV	0.8810	0.9083	0.9344	0.9494	0.9232	0.8630	0.8630	0.9032	0.9494
LT	N2	0.8872	0.9070	0.9325	0.9472	0.9216	0.8630	0.8630	0.9031	0.9472
LTHV	N2	0.8869	0.9064	0.9317	0.9467	0.9215	0.8630	0.8630	0.9027	0.9467
LT	N1	0.8868	0.9062	0.9315	0.9462	0.9207	0.8630	0.8630	0.9025	0.9462
LTHV	N1	0.8866	0.9057	0.9308	0.9457	0.9207	0.8630	0.8630	0.9022	0.9457
LTLV	N2	0.8864	0.9056	0.9306	0.9453	0.9204	0.8630	0.8630	0.9020	0.9453
LT	LV	0.8862	0.9053	0.9302	0.9446	0.9197	0.8630	0.8630	0.9017	0.9446
HV	LT	0.8862	0.9052	0.9300	0.9445	0.9193	0.8630	0.8630	0.9016	0.9445
LTLV	N1	0.8861	0.9049	0.9296	0.9442	0.9193	0.8630	0.8630	0.9014	0.9442
LTHV	LV	0.8859	0.9046	0.9293	0.9439	0.9195	0.8630	0.8630	0.9013	0.9439
HV	LTHV	0.8859	0.9045	0.9291	0.9436	0.9186	0.8630	0.8630	0.9011	0.9436
LV	N2	0.8854	0.9037	0.9280	0.9424	0.9181	0.8630	0.8630	0.9005	0.9424
HV	N2	0.8853	0.9037	0.9279	0.9424	0.9176	0.8630	0.8630	0.9004	0.9424
LTLV	LV	0.8853	0.9036	0.9279	0.9421	0.9178	0.8630	0.8630	0.9004	0.9421
HV	LTLV	0.8853	0.9036	0.9279	0.9421	0.9173	0.8630	0.8630	0.9003	0.9421
LT	LTHV	0.8852	0.9035	0.9278	0.9420	0.9174	0.8630	0.8630	0.9003	0.9420
N1	N2	0.8851	0.9031	0.9271	0.9416	0.9173	0.8630	0.8630	0.9000	0.9416
LV	N1	0.8850	0.9031	0.9271	0.9414	0.9173	0.8630	0.8630	0.9000	0.9414
HV	N1	0.8850	0.9030	0.9271	0.9413	0.9170	0.8630	0.8630	0.8999	0.9413
LT	LTLV	0.8848	0.9028	0.9268	0.9409	0.9164	0.8630	0.8630	0.8997	0.9409
HV	LV	0.8835	0.9006	0.9235	0.9371	0.9137	0.8630	0.8630	0.8978	0.9371
LTHV	LTLV	0.8843	0.9018	0.9253	0.9294	0.9155	0.8630	0.8630	0.8975	0.9294
Average		0.8882	0.9088	0.9350	0.94977	0.9240	0.863	0.863	0.9045	0.9497
Maximum		0.894	0.9183	0.9475	0.9631	0.9355	0.863	0.863	0.9121	0.9631

strates the dependency of each neighbor on the target cells reliability. The peaks of the plot show the locations (in terms of window sizes) which have the most influence on the target. Surprisingly, these peaks occur at regular intervals of 32. We believe that this is a direct indication of the physical location of the individual cells in relation to one another. Although the cells may be read out in an order where they appear adjacent (assuming 1D), in actuality they may be located in a completely different arrangement. For instance, the peaks in conditional probability at intervals of 32 cells could demonstrate that cells at such a distance in the SRAM read-out are located closer to each other physically. This provides further motivation for multi-dimensional analysis, as some cells which appear close in the 1D SRAM data (i.e. 16 cells away from the target cell) may not actually be located close to the target cell and therefore do not have a profound impact on its power-up state and stability.

Environmental analysis : Table II contains a list of each of the pairs of tested enrollment conditions, arranged in descending order of conditional probability. The last two columns (rows) indicate the average and maximum conditional probability for each row (column). The maximum conditional probability can be seen for the combination HTHV and LT (located at the top of the list). The critical conditions (used in our previous work [10]) appear high on this list (tenth); demonstrating the usefulness of using these conditions to identify cells based upon the results of our metrics. In general, high temperature as condition 1 and either high temperature, nominal temperature, or low temperature at condition 2 works very well for enrollment. At the bottom of the table, we see that use of low temperature as condition 1 and nominal or low temperature for condition 2 works poorly. This makes intuitive sense because SRAMs are known to be affected by thermal noise which increases with temperature. Hence, using high temperature as part of enrollment ensures that we will be able to identify the most stable bits since less stable ones should flip at high temperatures.

Table II also illustrates the impact of different window sizes at different enrollment conditions. The conditional probability appears much more dependent upon window size w as opposed to the specific environmental conditions used in the isolated pair. This is an interesting and unexpected result. From the last two rows it can be observed that the best window size is $w = 16$ (similar to Fig 6). For $w = 2$ and $w = 40$, there is a decrease of 7.14% and 10.4% compared to $w = 16$. This is a much larger decrease than from best (HTHV, LT) to worst (LTHV, LTLV) enrollment conditions, which is clearly visible in the last two columns of Table II).

C. Summary of Results

Overall, we draw three main conclusions from these results for our SRAM:

- The optimal combination of window size and threshold value is not necessarily at the point where $w = t$. Only for small window sizes does $w = t$ works best.
- The most influential neighborhood cells for our SRAM appear to be the cells at distances every 32 bits from the target cell (in 1D). These are likely to be the ones physically located near the target cells in the actual layout.
- Certain pairs of environmental conditions demonstrate higher dependency of the target cell upon its neighbors. Generally, it is better to use high temperature as one of the enrollment conditions.
- The choice of window size w is more important for bit selection than enrollment condition.

V. Conclusion

In this paper, we investigated three metrics for analyzing the outputs of SRAM (assuming 1D). Our result showed that the optimal threshold value does not necessarily equal the window size. We also noticed that certain cells set at specific distances from the target cell are more heavily influential than other neighbors within a window. These results lead us to believe that there is much to gain from 2D analysis of SRAM outputs in the future. We also hope to expand our work to consider different SRAM manufacturers (Intel, AMD, etc.) and to develop better bit selection algorithms. Additionally, the process of bit selection and the usage of proposed metrics can be used to develop algorithms that identify the most unstable cells which are optimal for TRNGs.

VI. Acknowledgment

This work was supported in part by the National Science Foundation (NSF) under grant 1359329 and by the NSF and the Semiconductor Research Corporation (SRC) STARSS program under grant CNS 1441750.

References

[1] C. Gorman, " chips on the rise. Spectrum," IEEE 49(6):16-7, 2012.

[2] Md. Tauhidur Rahman et al., "CSST: Preventing Distribution of Unlicensed and Rejected ICs by Untrusted Foundry and Assembly," In IEEE Int. Symposium on Defect and Fault Tolerance Symposium (DFTS), 2014.

[3] M. Rostami, F. Koushanfar, and R. Karri, "A Primer on Hardware Security: Models, Methods, and Metrics," Proceedings of the IEEE, vol. 102, no. 8, pp. 1283-1295, 2014.

[4] G. Contreras, Md. T. Rahman, and M. Tehranipoor, "Secure Split-Test for Preventing IC Piracy by Untrusted Foundry and Assembly," in Int. Symposium on Defect and Fault Tolerance in VLSI Systems (DFT), 2013.

[5] B. Gassend et al., "Silicon physical random functions," in Proceedings of the 9th ACM conference on Computer and communications security, pp. 148-160, 2002.

[6] B. Habib, K. Gaj, and J. Kaps, "FPGA PUF based on programmable LUT delays," Euromicro Conference on Digital system design (DSD), pp. 697, 2013.

[7] D. Holcomb, W. Burleson and K. Fu, "Power-Up SRAM State as an Identifying Fingerprint and Source of True Random Numbers," IEEE Transactions on Computer, 2009.

[8] Y. Su, J. Holleman, and B.P. Otis, "A Digital 1.6pJ/Bit Chip Identification Circuit Using Process Variations," IEEE J. Solid-State Circuits, vol. 43, no. 1, pp. 69-77, 2008.

[9] M. Yu, and S. Devadas, "Secure and Robust Error Correction for Physical Unclonable Functions," IEEE Design & Test of Computers, vol.27, no.1, pp.48-65, 2010.

[10] Kan Xiao et al., "Bit selection algorithm suitable for high-volume production of SRAM-PUF," IEEE Int. Symp. on Hardware-Oriented Security and Trust, pp. 101, 2014.

[11] S. Eiroa et al., "Reducing bit flipping problems in SRAM physical unclonable functions for chip identification," 19th IEEE Int. Conf. on Electronics, Circuits and Systems (ICECS), pp. 392-395, 2012.

[12] R. Maes et al., "Physically Unclonable Functions: a Study on the State of the Art and Future Research Directions," Section 1. Towards Hardware-Intrinsic Security. Springer, 2010.

[13] M. Cortez et al., "Modeling SRAM start-up behavior for physical unclonable functions," IEEE international symposium on Defect and fault tolerance in VLSI and nanotechnology systems (DFT), 2012.

[14] Md. Tauhidur Rahman et al., "ARO-PUF: An Aging-Resistant Ring-Oscillator PUF Design," in proc. Design, Automation, and Test in Europe (DATE), 2014.

[15] D. Lim et al., "Extracting secret keys from integrated circuits," IEEE Trans. VLSI Syst., vol. 13, no. 10, pp. 1200-1205, 2005.

[16] Md. Tauhidur Rahman et al., "TI-TRNG: Technology Independent True Random Number Generator," Proceedings of the The 51st Annual Design Automation Conference on Design Automation Conference (DAC), pp. 179:1–179:6, 2014.

[17] M. Hiller et al., "Complementary IBS: Application specific error correction for PUFs," IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 1-6, 2012.