# Bit Selection Algorithm Suitable for High-Volume Production of SRAM-PUF

**Kan Xiao[1], Md. Tauhidur Rahman[1], Domenic Forte[1], Yu Huang[2], Mei Su[2] and Mohammad (Mark) Tehranipoor[1]**
[1]ECE Department, University of Connecticut, CT, USA
[2]Synokey LLC, MA, USA

***Abstract*—Physically Unclonable Functions (PUFs) are impacted by environmental variations and aging which can reduce their acceptance in identification and authentication applications. Prior approaches to improve PUF reliability include bit analysis across environmental conditions, better design, and post-processing error correction, but these are of high cost in terms of test time and design overheads, making them unsuitable for high volume production. In this paper, we aim to address this issue for SRAM PUFs with novel bit analysis and bit selection algorithms. Our analysis of real SRAM PUFs reveals (i) critical conditions on which to select stable SRAM cells for PUF at low-cost (ii) unexplored spatial correlation between stable bits, i.e., cells that are the most stable tend to be surrounded by stable cells determined during enrollment. We develop a bit selection procedure around these observations that produces very stable bits for the PUF generated ID/key. Experimental data from real SRAM PUFs show that our approaches can effectively reduce number of errors in PUF IDs/keys with fewer enrollment steps.**

## I. Introduction

With the rapid development of information technology, electronic devices (smart phone, GPS, etc.) are playing a larger role in our daily lives. More and more emerging applications for these devices require access to remote servers, which require device identification and authentication. Inauthentic products, such as counterfeit and cloned devices, can deteriorate service quality and result in customer loss. Traditional tools for uniquely identifying devices, such as product IDs and barcodes, are exposed to everyone and hence can be easily duplicated. Alternatively, a device identifier (ID) may also be digitally stored in the non-volatile memory (NVM) or fuses. Storing the ID using these methods is not only expensive, but also vulnerable to both physical attacks, such as non-invasive, semi-invasive, or invasive as well as software attacks, such as viruses and API-attacks [1]. Securing the ID is of utmost importance since leakage of even a single device ID could be exploited by an adversary to produce pirated devices.

Physical Unclonable Functions (PUFs) have been proposed as a more secure alternative to conventional ID storage because of their unclonability and built-in tamper evidence [2]. PUFs generate unique IDs by exploiting the uncontrollable process variations associated with modern IC fabrication [3]. An ideal PUF takes an input (challenge) and gives a random output (response) which is unique for every device [4]. Many PUFs have been proposed and implemented over the past decade, such as Arbiter PUF, RO-PUF, SRAM PUF, Butterfly PUF, etc. [5] [6] [7]. These different PUFs have various pros and cons, but among them, SRAM PUFs [8][9][10][11] are quite popular for several reasons: (i) SRAM is a standard component available in many technology nodes; (ii) SRAM is already utilized in many systems as a volatile memory so designers do not need to add more dedicated hardware for the PUF; (iii) Typical SRAMs have large amount of space and can easily provide sufficient PUF outputs to generate ID/keys long enough for devices produced at high-volume.

Despite its advantages, however, the stability of SRAM PUF output is heavily impacted by environmental variations and aging effects. One popular approach for dealing with this reliability issue is to correct the unreliable output bits using Error Correcting Code (ECC) [12]. However, ECC leaks information and results in high design costs that ultimately limit the use of PUFs in resource-constrained products. To reduce the overheads, an alternative is to reduce sensitivity to temporal variations. For instance, a reliability enhancement [10] was developed to reinforce the preferred value of SRAM cell by inducing accelerated aging. But performing burn-in stress for 120 hours for one SRAM PUF is prohibitively expensive, time-consuming, and in turn degrades the SRAM. Another proposed approach modifies the VDD ramp-up time to make cells more reliable [9], but this approach requires special circuitry not present in standard SRAM.

In this paper, we take a new approach that selects the most stable bits for PUF IDs/keys with the goal of reducing or possibly removing the need for ECC altogether. A random approach might use exhaustive testing at all environmental conditions/corners and after aging to choose the bits that are stable, but this is ill-suited for mass-produced devices where test time and cost are major concerns. We make the following contributions:

- We conduct preliminary experiments to gather large amounts of real data from an SRAM. We analyze the impact of environmental variations, temporal variations (aging by burn-in), and spatial correlation (neighborhood analysis) on the stability of PUF bits. Our neighborhood analysis reveals that the most stable bits from enrollment depend on the stability of their neighbors, i.e. those surrounded by other stable bits flip less over time.
- Based on our observations, we develop low-cost enrollment tests that only require two corner conditions (high-temperature low-voltage and low-temperature low-voltage) and no burn-in tests. It is more efficient for stable cell selection and is relatively low-cost comparing to exhaustive tests at all conditions.
- We compare the uniqueness, reliability, etc. of our PUF IDs/keys for fresh and aged SRAM chips with IDs/keys from typical approaches. Our results show that our IDs have less bit errors over time, thereby requiring much less ECC overheads.

The rest of the paper is organized as follows. In the next section, we discuss background on SRAM PUFs and their use in device authentication. In Section III, we perform our initial experiments and analyze the impact of different conditions, etc.

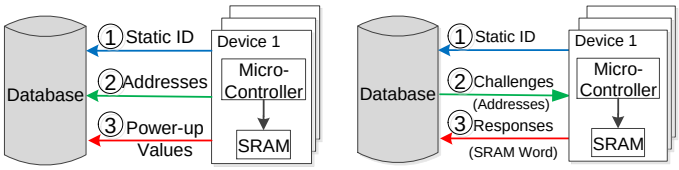
on SRAM cell stability. Section IV discusses our proposed enrollment and bit selection algorithms. We present our results in Section V and conclude with Section VI.

## II. Background and Preliminaries

### A. A Typical Protocol for Systems with Intrinsic SRAM PUFs

Figure 1 illustrates a typical authentication protocol for devices with SRAM PUF. The protocol consists of two phases: (a) *enrollment* and (b) *reconstruction* [13]. An SRAM chip is integrated in the device (on-board or in package). Before the device is delivered to market, the device is "enrolled". Specifically, the SRAM PUF output (or a portion of it) is read by the system owner and then stored in a database. If only a portion of it is stored, then the database will also store the SRAM addresses corresponding to that portion.

In the reconstruction phase, the device starts the protocol by sending its static ID (publicly available) to the database. The database then searches for the corresponding challenges (addresses) associated with the static ID and sends them back to the device. The device collects the SRAM outputs (which may be noisy) and sends them to the database for authentication/identification. The database compares them with the genuine output stored in its database. Compared to the conventional approaches that use bar-codes or embedded IDs/keys into a device, the PUF approach is much more secure [2]. As long as each PUF generates a unique ID/key, each device will be authenticated properly.




(a) (b)

Figure 1: SRAM PUF (a) enrollment and (b) reconstruction

### B. Operation and Salient Features of SRAM PUF

For a standard 6T SRAM, every memory cell is composed of six transistors that are two cross-coupled CMOS inverters and two access transistors. The inverters are designed to be symmetric, match in size, etc., but random variations incurred during manufacturing will result in random mismatches. SRAM PUFs exploit the mismatch which results in each SRAM cell being biased (or skewed) toward a zero or one at power-up.

In addition to process variations, the power-up value can also be influenced by temporal variations, such as voltage supply variation, operating temperature, aging effects (such as negative bias temperature instability (NBTI) and hot-carrier injection (HCI)), etc. The temporal variations can cause cells to start-up in a different state, causing the ID generated by the SRAM PUF to change over time. For illustrative purposes, both process variation and noise are considered as impacting the *skew* of a cell. The skew of a cell is a continuous quantity used to represent the power-up tendency of that cell. Skew at a given power-up is influenced by noise, so the skew of each cell across many power-ups can be described by a probability distribution function, as shown in Figure 2.

Ideally, the SRAM PUF output would be stable and unique if only process variation exists. However, in reality, the ID/keys will change over time because of temporal noise and aging degradation over time. Therefore, if the magnitude of process

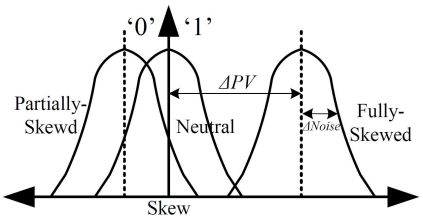



Figure 2: The influence of process variation and noise on cell's skew [8].

variation of one cell is large enough to safely offset temporal noise, this cell would be able to produce a reliable power-up value despite environmental noise and even degradation due to aging, like the fully-skewed cell shown in Figure 2. Since SRAM has a large number of memory cells to use as PUF, there exists sufficient fully-biased cells (large process variation) that could potentially produce a very stable key bit. Our objective then becomes development of techniques that are low-cost in terms of design and test overhead for identifying the most stable SRAM PUF bits.

## III. Preliminary Experimentation and Observations

In this section, we will explore the impact of supply voltage, ambient temperature and aging effects on power-up values (i.e., reliability of the ID/key) using real SRAM measurements. We also analyze the dependence of location on the stability of SRAM cell power-up values. From the observations we make from this data, we will develop low-cost techniques in latter sections for choosing the most stable bits for PUF ID/key.

Experiments are performed on a Xilinx Spartan-3 FPGA board with a 1MB on-board ISSI SRAM. Here, we focus on 6KB of the SRAM, because a small portion of SRAM can provide enough bits to construct a PUF. Additionally, since every SRAM cell is designed to be completely identical, the characteristics of a portion of SRAM cells can be extended to other memory cells in this SRAM as well as others.

### A. Stability Under Power Voltage Variation

First, the PUF output is collected under supply voltage variations. Since the operation voltage range of the on-board SRAM is $\pm 10\%$ of the nominal voltage 3.3 volt, many measurements are taken at different core voltages from 3.0v to 3.6v. In order to illustrate the changes of power-up values under voltage variations, 20 measurements at low voltage (3.0v) and high voltage (3.6v) are taken (10 from each voltage). For every memory cell, the frequency of producing a '1' is calculated based on 10 measurements at the same voltage. A frequency of 1 (0) indicates these cells are producing stable '1' ('0') in all ten measurements, while other values between 0 and 1 indicate that cells produce some ones and some zeroes in the 10 measurements (i.e., were not stable). Then, we analyzed the change of distribution of cell skew as voltage varied from low to high.

Table I depicts the distribution. The first row (column) in Table I indicate the frequency of producing '1' at high (low) voltage. The fractions in the table represent the percentage of the cells that have a particular combination of frequencies producing '1' at low and high voltages. For example, the number in the fourth row and third column says that about 0.31% cells produced a '1' at HV and LV with frequencies between 0-0.2 and 0.4-0.6 respectively. The table shows that 66.40% (34.06% + 32.34%) of cells generate either a stable '0' or '1' across the measurements. The top right and bottom left elements in Table I

are both 0 which indicates that none of the bits that were stable '0' ('1') at HV became stable one (zero) at LV and vice versa.

Table I: Distribution of cell skew at high and low voltages.

| LV/HV | 0 | 0∼0.2 | 0.2∼0.4 | 0.4∼0.6 | 0.6∼0.8 | 0.8∼1.0 | 1 |
|---|---|---|---|---|---|---|---|
| **0** | 34.06 | 2.57 | 0.17 | 0.02 | 0 | 0 | 0 |
| **0∼0.2** | 1.96 | 3.93 | 1.49 | 0.27 | 0.04 | 0.01 | 0 |
| **0.2∼0.4** | 0.16 | 1.36 | 2.09 | 1.15 | 0.29 | 0.01 | 0.01 |
| **0.4∼0.6** | 0.01 | 0.31 | 1.12 | 1.79 | 1.08 | 0.17 | 0.04 |
| **0.6∼0.8** | 0 | 0.05 | 0.30 | 1.37 | 2.68 | 0.90 | 0.50 |
| **0.8∼1.0** | 0 | 0 | 0.03 | 0.21 | 1.52 | 0.41 | 1.42 |
| **1** | 0 | 0 | 0 | 0.03 | 0.65 | 2.46 | 32.34 |

### *B. Stability Under Temperature Variation*

Similar experiments were conducted by sweeping temperature from 0°C to 80°C using our Themostream system. We use the same approach as above to analyze the distribution changes introduced by temperature variations (see Table II). Our data indicates that 60.33% (30.79%+29.54%) of cells are stable regardless of temperature, and 14.06% of memory cells change to the opposite skew state due to temperature change. Among them, 4.79% (2.24%+2.55%) memory cells change from stable zero to stable one or stable one to stable zero). Our experiments demonstrate that temperature variations have much greater impact on bit stability than supply voltage variations for SRAM PUF.

Table II: Distribution of cell skew across temperature.

| LT/HT | 0 | 0∼0.2 | 0.2∼0.4 | 0.4∼0.6 | 0.6∼0.8 | 0.8∼1.0 | 1 |
|---|---|---|---|---|---|---|---|
| **0** | 30.79 | 1.12 | 0.65 | 0.47 | 0.55 | 0.39 | 2.24 |
| **0∼0.2** | 4.22 | 0.53 | 0.36 | 0.28 | 0.35 | 0.23 | 2.03 |
| **0.2∼0.4** | 2.30 | 0.29 | 0.20 | 0.16 | 0.20 | 0.15 | 1.56 |
| **0.4∼0.6** | 1.77 | 0.30 | 0.15 | 0.18 | 0.21 | 0.20 | 1.75 |
| **0.6∼0.8** | 2.06 | 0.34 | 0.22 | 0.23 | 0.27 | 0.27 | 2.43 |
| **0.8∼1.0** | 1.46 | 0.34 | 0.17 | 0.17 | 0.23 | 0.19 | 2.43 |
| **1** | 2.55 | 0.67 | 0.55 | 0.59 | 0.81 | 0.80 | 29.54 |

### *C. Stability Under Aging Effects*

In this section, we look at the stability of cells as the SRAM ages. To accelerate the aging, we perform burn-in of the SRAM using Thermostream burn-in system. Specifically, we execute write '0', write '1' ,and read operations alternately to/from every memory cell under high temperature (80°C) and high power voltage (3.6) conditions. In Table III, we compare two measurements performed under nominal conditions (room temperature + nominal voltage) on the fresh SRAM and the SRAM after 2 hours of accelerated aging. Table III shows that 59.26% (29.82%+29.44%) of the bits are stable and approximately 1.89% (0.88%+1.01%) of cells change from one stable state to the opposite stable state because of aging effects.

### *D. Stability of Neighboring Cells*

Prior work has shown that neighboring chips on the same wafer undergo similar processes and correlate to each other. Hence, chips have the same fault-free (or conversely faulty) properties as their neighbors in the same wafer [14]. The spatial correlation of parameters among cells within a wafer has been investigated in [15] [16].

In this section, we investigate whether the same relationship holds for predicting the reliability of SRAM cells for PUFs by performing the following test. We took 10 measurements at random environmental conditions for the entire SRAM. Based on the output, we labeled the cells as "stable" if they produced the same value for all 10 measurements or as "unstable" otherwise.

Table III: Distribution of cell skew for fresh and aged SRAM.

| A/F | 0 | 0∼0.2 | 0.2∼0.4 | 0.4∼0.6 | 0.6∼0.8 | 0.8∼1.0 | 1 |
|---|---|---|---|---|---|---|---|
| **0** | 29.82 | 2.14 | 0.84 | 0.63 | 0.62 | 0.38 | 0.88 |
| **0∼0.2** | 3.94 | 1.27 | 0.622 | 0.56 | 0.58 | 0.40 | 1.03 |
| **0.2∼0.4** | 1.80 | 0.72 | 0.50 | 0.40 | 0.47 | 0.35 | 1.08 |
| **0.4∼0.6** | 1.18 | 0.62 | 0.41 | 0.42 | 0.49 | 0.37 | 1.24 |
| **0.6∼0.8** | 1.08 | 0.66 | 0.47 | 0.49 | 0.64 | 0.51 | 2.05 |
| **0.8∼1.0** | 0.55 | 0.37 | 0.26 | 0.34 | 0.54 | 0.48 | 2.37 |
| **1** | 1.01 | 0.80 | 0.64 | 0.81 | 1.28 | 1.52 | 29.44 |

For each stable and unstable cell, we looked at a 20 cell "window" around the cells (the 9 cells before and 9 cells after it) and counted the number of unstable cells in the window. Figure 3 shows the distribution of the count for stable and unstable cell cases fitted with Gaussian distributions. The distribution for stable cells is closer to zero meaning they have less unstable bits surrounding them. This test shows that the stability of each cell/bit *is* correlated with its neighborhood: the more stable cells/bits in the neighborhood, the more stable/reliable the cell will be as a PUF.

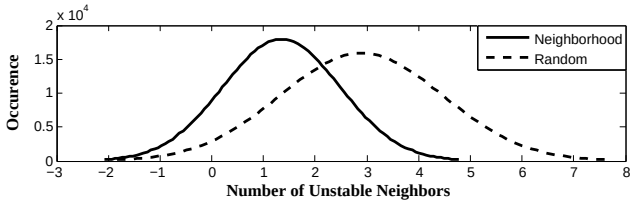



Figure 3: Stable neighbors provides better reliability than random selection.

## IV. Proposed Enrollment and Bit Selection Techniques

There are two primary observations we have obtained from Section III: (i) temperature variation can introduce more flipped bits than voltage variation and aging; (ii) the power-up stability of a cell is related to its neighboring cells. We take advantage of these characteristics to identify very reliable cells with a reduced set of enrollment tests and a neighborhood-based bit selection.

### *A. Critical Test Conditions for Enrollment*

Due to instability of power-up values under environment variations, we want to identify stable cells and exclude two types of unstable cells, partial-skewed cells and neutral-skewed cells. Partial-skewed cells are stable in most of power-up operations and have a small possibility to flip. By changing environment condition to make the probability distribution shift to neutral location, some partial-skewed cells will be much easier to flip. Power voltage and temperature are two major environmental factors we can change in the enrollment tests with relatively low cost. According to the experiment results in Section III, we know that temperature can contribute more to the change of cell's skew.

The impact of temperature on MOSFET devices is well studied in literature. An increase in temperature decreases device threshold voltages:

$$V_{th}(T) = V_{th}(T_0) - \kappa\Delta T$$

Therefore, $V_{th}$ has a linear relationship to temperature. Additionally, an increase in temperature increases the magnitude of thermal noise as well [8]:

$$\sigma^2_{NOISE}(T) = \frac{2K_BT}{C}$$

which could lead to more random power-up values. In order to let both 0-skew and 1-skew cells move closer to the neutral location, we select the maximum and minimum temperatures. Moreover, low supply voltage leave a cell susceptible to noise-induced state changes, while higher voltage makes a cell stable and immune to noise [8]. Thus, for the PUF enrollment tests, high-temperature/low voltage (HTLV) and low-temperature/low voltage (LTLV) are the best condition combination for low cost enrollment tests, which can find unstable bits more efficiently. Additionally, for neutral-skewed cells, the high and low temperature conditions can shift their neutral position left and right, so the added environmental noises make them much easier to flip in a small number of tests. Changing temperature during enrollment is a much more slower process than changing power supply voltage; thus using HTLV and LTLV makes the enrollment process fast and low-cost. We will evaluate the effectiveness of using HTLV and LTLV for enrollment in Section V.

### B. Neighborhood-based Bit Selection

In this section, we propose a heuristic bit selection algorithm that takes the spatial correlation we discovered in Section III-D.into account. By doing so, we expect to reduce the errors due to environmental and temporal noise (aging) in the SRAM PUF. Since aging has not been covered by the above critical enrollment tests, this bit selection is quite important.

The details of our algorithm are as follows. We develop a metric to assign value (weight) to the stability of SRAM cells. Basically, we have observed that the cells that are "most stable" across environmental conditions are surrounded by more stable cells during enrollment. A stable cell surrounded by more stable cells has a tendency to become more stable because its neighboring cells are likely to experience similar aging stress and operating conditions. In our metric, we give such cells a higher weight. After determining the weight of each cell, we use a heuristic algorithm that greedily chooses cells for the PUF ID/key with weight greater than a threshold.

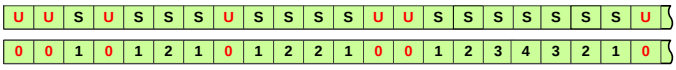



Figure 4: Weight assignment in the bit selection algorithm.

Since we do not know the physical locations of memory cells, the heuristic bit selection algorithms have been proposed under the assumption that a cell is affected by adjacent cells only (however, we expect better results if the physical 2D structure is known, i.e. vertical and adjacent neighbors of each cell). Every word (16 bits) from each memory address is concatenated in a line, as shown in Figure 4. The cells have been weighted based on the adjacent cells along the line. First, all stable and unstable cells are marked as "stable" or "unstable" according to the enrollment tests, as shown in Figure 4. Then all unstable cells are replaced with zeroes, and they will not be selected as ID/key. Next, we focus on some stable blocks which are composed of continuous stable cells. For example, a stable block with $2p-1$ cells, the middle cell is weighted to $p$. Then, the closest two neighbors, one from each side, are weighted to $p-1$ and so on. Finally, the last two cells, from both sides are weighted to 1, because one of its neighbors is unstable bit too. The pseudo-code of the bit selection algorithm is shown as Algorithm 1. An array $W(i)$ is used to record the weight of each bit in the bit selection algorithm, and a k-bit ID/key will be generated in an array *FinalCells*. Taking the 15th to 21st bits in Figure 4 as another example, in a stable block with length of 7, the middle cell is weighted to 4 as that cell is surrounded by six stable cells, three on each side. The other six cells are weighted according to distance to unstable neighbor(s).

Our algorithm uses a threshold to choose the most stable bits for use as the PUF ID/key. The higher the threshold, the larger number of stable cells a cell must have to be selected for the ID/key. However, by increasing the threshold, the number of qualified bits reduces exponentially. Hence, an appropriate threshold should be decided based on key size, number of stable cells, neighborhood characteristics, etc. A lower threshold is required for a higher-sized key from the same number of stable cells.

**Algorithm 1** Neighborhood-based Bit Selection

```
1:  procedure BitSelection (M, T, k)
2:      M ← Total number of bits
3:      T ← Threshold
4:      k ← A k-bit ID/key needs to be generated
5:      for i = 1 to M do
6:          if (The cell i is unstable during enrollment) then
7:              W(i) = 0
8:          else
9:              Weighting the cell i
10:             if (W(i) ≥ T) then
11:                 The cell i is a qualified bit
12:             end if
13:         end if
14:     end for
15:     Q ← Sort weight of qualified bits in descending order
16:     FinalCells ← Q[1 : k] First k bits are selected
17:     return FinalCells[1...k]
18: end procedure
```

## V. Experimental Results and Discussion

### A. Experiment Setup

In this section, our proposed approaches are evaluated on an experiment platform, the Xilinx Spartan-3 FPGA board. Its 1MB on-board SRAM is explored as the SRAM PUF. The whole platform is an intrinsic SRAM PUF system. The FPGA chip acts a microcontroller, which reads the SRAM data and sends it to a PC via UART port. A power supply and Themostream system are employed to adjust the power voltage ($\pm 10\%$) and temperature (0°C-80°C) respectively. The experiment setup is shown in Figure 5.

We collected 100 measurements of the entire SRAM memory for fresh and aged SRAMs under different environment conditions: 20 at nominal conditions, 10 at each of following conditions: low voltage (LV), high voltage (HV), low temperature (LT), high temperature (HT), and four corner conditions (LTLV, LTHV, HTLV, and HTHV). Each measurement has 1MB (8,388,608 bits) power-up values of the SRAM. Two phases of accelerated aging sessions were performed on the same SRAM. Two phases of 5-hour aging have been applied to the SRAM chip. After every burn-in, another 100 measurements were performed from the aged SRAM at the same environmental conditions mentioned above.

### B. Effectiveness of Critical Enrollment Tests

In this section, we compare enrollment using the typical nominal conditions (case 1) with our proposed approach from

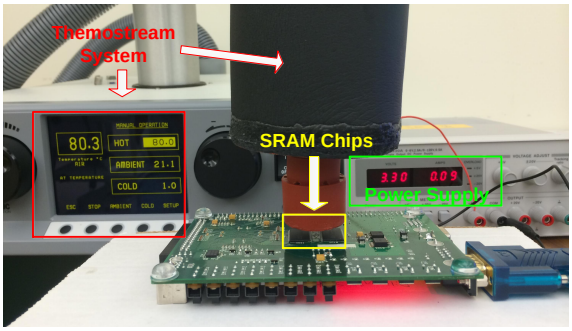
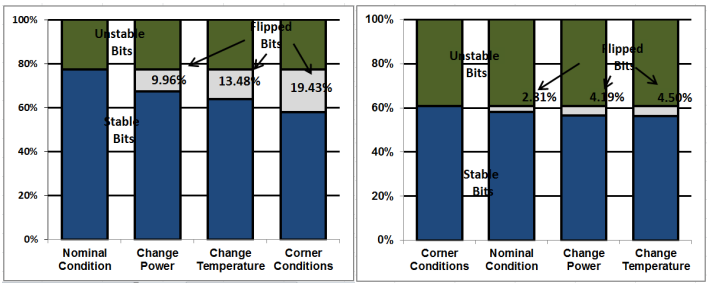



Figure 5: Experiment setup.

Section IV that uses HTLV and LTLV conditions (referred to as case 2). Note that case 1 would be the typical approach in prior work. In both approaches, we determine stable bits as those that do not change for all the corresponding enrollment measurements. Then we calculate how many of these bits change at the other measurement conditions (those not used during enrollment).

The results are shown in Figure 6 where (a) and (b) correspond to case 1 and 2 respectively. From left to right, each new condition is applied resulting in more of the original stable bits determined by enrollment flipping. In Figure 6(a), the original stable bits reduce by about 19.43% as the key is constructed at corner conditions. In Figure 6(b), our proposed approach does a better job of determining the stable bits by focusing on measurements only at critical conditions. The percentage of flipped bits is only 4.5%. By using the stable bits from enrollment using case 2, there will lesser need to correct errors in the ID/key in the reconstruction phase with lesser tests. Therefore, the above experiments illustrate that the choice of conditions used at PUF enrollment is quite significant and the proposed approach is more appropriate than the typical approach.



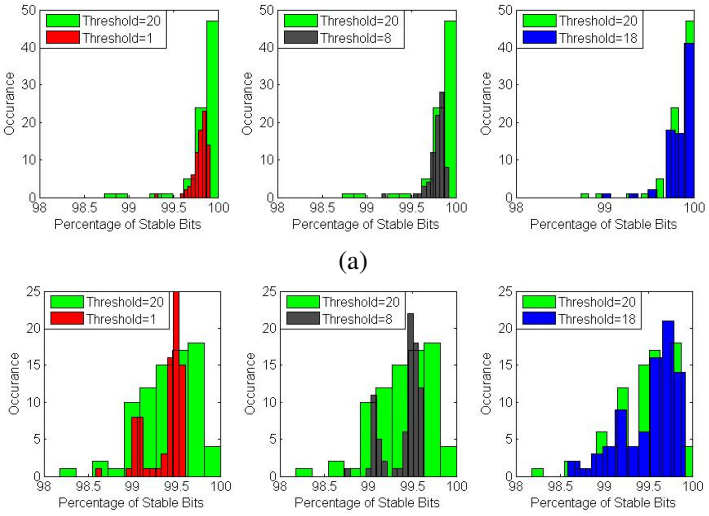

(a) Case 1 (b) Case 2

Figure 6: (a) Case 1: nominal conditions and (b) Case 2: corner conditions for the enrollment tests.

### C. Reliability of SRAM cells

In this experiment, we investigated the whole SRAM using critical condition tests for enrollment as discussed in the previous section. From the reliable bits determined by enrollment, we chose stable bits for analysis according to two approaches: (i) a naive approach that randomly selects the desired number of bits from stable bits obtained during enrollment (called "random" in this paper); (ii) the proposed bit selection approach discussed in Section IV. For the proposed approach, we also varied the threshold to determine its impact. 80 measurements from various environmental conditions (nominal, HTLV, etc.) for each chip state (fresh and aged 10 hours) were used to determine which approach produced fewer bit flips during reconstruction. We calculated the percentage of bits in the 1MB that changed from enrollment in each of the 80 measurements we collected.

The distributions containing all 80 measurements are shown in Figure 7. Figure 7 (a) and (b) correspond to fresh and 10 hours aging respectively. The first, second, and third columns in Figure 7 compare the distribution for threshold equal to 20 to those with threshold equal to 1,8, and 18. Note that threshold equal to 1 corresponds to the random approach. Looking at the top left plot, one can see that the bit selection approach with threshold equal 20 has more occurrences of 100% than the random approach. As we increase the threshold (more reliable neighbors required for selection as key bit), the occurrences of 100% also tend to improve. For the fresh SRAM, the bit selection algorithm with threshold of 20 has an improvement of 38% than the random selection (Threshold=1). Note that there are some outliers (low values of percentage) for threshold at 20, but these would probably be less significant if we gathered more measurements. The trends clearly show that there is better stability with bit selection and higher threshold values.




(a)

(b)

Figure 7: Distributions of percentage of correct bits using different thresholds for SRAM (a) without aging and (b) with 10 hours aging.

Examining the impact of aging (Figure 7 (b)), we see that the stability of the keys are lower than the fresh chips. This is because the enrollment conditions do not contain aging measurements. Without these measurements, we cannot determine which bits will flip as the chip ages. Comparing the two algorithms (leftmost column of Figure 7), we can see that the percentage of stable measurements at or near 100% is still more for the proposed bit selection algorithm than the random approach. The same trend of improvement as we increase the threshold also still holds for the aged chips. Although the percentage of correct bit reconstruction goes down due to the aging effects, the improvement of bit selection algorithm (at threshold equal 20) compared to the random approach increases to 45%. To some degree, one also can say that the bit selection algorithms compensate well for the lack of aging measurements used in enrollment. This is important because it removes the need for burn-in tests which would have significant costs and device reliability issues for high volume production.

Table IV: Bit flips in the two bit selection methods.

| Reconstruction | | Fresh | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Enroll | Error Bits | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| NC | Random Selection | 338 | 154 | 78 | 39 | 25 | 9 | 5 | 1 | 0 | 0 |
| CC | Random Selection | 550 | 82 | 4 | 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Proposed Selection | 559 | 72 | 7 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Reconstruction** | | **10 hours burn-in** | | | | | | | | | |
| **Enroll** | **Error Bits** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| NC | Random Selection | 118 | 214 | 148 | 91 | 39 | 13 | 11 | 5 | 1 | 2 |
| CC | Random Selection | 347 | 198 | 63 | 24 | 6 | 0 | 1 | 1 | 0 | 0 |
| | Proposed Selection | 438 | 162 | 34 | 5 | 1 | 0 | 0 | 0 | 0 | 0 |

NC: Nominal condition enrollment tests; CC: Critical condition enrollment tests;

### *D. Reliability and Uniqueness of PUF Key*

In these experiments, we evaluate the reliability and uniqueness of keys generated by the proposed bit selection algorithm. We divided the whole 1MB SRAM into 8 PUFs. Each PUF has 128KB and used to generate a 128-bit key. This 128-bit key can uniquely identify 3.4*e*38 devices, so it can apply to most current applications in the market. 20 critical condition tests (HTLV and LTLV) and 20 nominal condition tests were used for enrollment. Another 80 measurements from various environmental conditions for each chip state (fresh and aged 10 hours) were used to reconstruct the key. The proposed bit selection approach and the random bit selection approach that was used in the previous section are investigated and compared in this section.

*1) Bit Error:* To analyze the impact on error correction code (ECC) overheads, we also look at the maximum number of bit flips in each of the 80 measurements for fresh and aged chips for random and neighborhood-based bit selection (threshold equal 18). The number of bits that flip will determine the number of bits we need to correct via ECC and thus is directly related to the ECC overheads. The results are shown in Table IV. Note that for random selection, we consider two different enrollment conditions: nominal conditions (NC) and corner conditions (CC).

Table IV clearly shows that critical enrollment tests (CC) is more effective than nominal condition tests (NC) for selecting stable bits in the enrollment phase. For the fresh chip, NC case has up to 7 simultaneous bit errors. For critical enrollment tests, the maximum number of error bits is 3 for both algorithms so there is improvement in ECC overhead. For fresh chips, we also observe that the average number of errors is also similar between random selection (CC case) and the bit selection algorithm. As the chips age, however, the maximum number of error bits increases at a faster rate for the random selection (CC case) compared to the bit selection algorithm. For 10 hours burn-in, the maximum number of error bits are 9 for the random approach (NC case), 7 for random approach (CC case) but only 4 for bit selection approach. There is about a 22% and 55% decrease in bit flips for the proposed approaches compared to the typical approach (random selection at NC). These improvements represent a nontrivial decrease in overhead since the cost of ECC increases significantly with each bit error.

*2) Uniqueness:* The bit selection algorithm would be less useful if it resulted in lower uniqueness of keys/IDs in the population. In this section, we calculated the PUF uniqueness for the random (CC case) and bit selection techniques based on the Hamming distance (HD) between the IDs of different PUFs. For the bit selection, different thresholds are investigated as well (specifically 5, 8, and 12). The occurrence rate of each of 2016 ($\binom{64}{2}$) HDs across the 128 PUF outputs is shown in Table V. Table V contains the mean and variance of the HD for each approach (note that the HD distributions were Gaussian). The average inter-die HD using random bit selection is about 63.78 bits out of 128 or 49.8%. The average inter-die HD of different thresholds are 63.99 (49.99%), 64.03 (50.02%), 63.81(49.85%), which are closer to ideal value 64 out of 128 or 50% than the random bit selection. We conclude that the uniqueness of IDs/keys from our bit selection algorithms is at least as good as the random approach, if not better.

Table V: HD in 128-bit outputs from 64 PUFs.

| | **Random** | **Neigborhood-based Bit Selection** | | |
|---|---|---|---|---|
| | | T=5 | T=8 | T=12 |
| **Mean** | 63.78 | 63.99 | 64.03 | 63.81 |
| **Variance** | 31.96 | 31.35 | 31.94 | 32.07 |

## VI. Conclusion and Future Work

In this paper, we have presented our discovery that critical conditions for enrollment tests can more efficiently identify unreliable bits. We have also developed an effective bit selection algorithm that can assist with key selection (i.e., determine "most stable bits) from the reliable bits obtained from enrollment tests. Our experiments demonstrate the effectiveness of our approaches to reduce ECC overhead with low test cost, thereby making SRAM PUFs more suitable for high-volume production. One limitation in the current paper was that without knowledge of the full physical SRAM structure, we could not develop a neighborhood-based metric/algorithm that includes 2D information. We shall investigate this in future work.

## References


[1] U. Rurmair *et al*, "Modeling Attacks on Physical Unclonable Functions," the 17th ACM conference on Computer and communications security, October 04-08, 2010.

[2] G. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," ACM/IEEE Design Automation (DAC), pp. 9-14, June 2007.

[3] N. Tuzzio, K. Xiao, X. Zhang, and M. Tehranipoor, "A Zero-Overhead IC Identification Technique using Clock Sweeping and Path Delay Analysis," IEEE GLSVLSI, 2012.

[4] B. Gassend *et al*, "Silicon Physical Random Functions," in Proceedings of the 9th ACM conference on Computer and communications security, pp. 148-160, 2002.

[5] M. Tehranipoor and C. Wang, "Introduction to Hardware Security and Trust," Springer, Augest 2011.

[6] R. Maes and I. Verbauwhede, "Physically Unclonable Functions: A Study on the State of the Art and Future Research Directions," Towards Hardware-Intrinsic Security, pp. 3-37, Springer, 2010.

[7] T. Rahman, D. Forte, M. Tehranipoor, and J. Fahrny, "ARO-PUF: An Aging-Resistant Ring-Oscillator PUF Design," Design, Automation, and Test in Europe (DATE), 2014.

[8] 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.

[9] M. Cortez *et al*, "Adapting Voltage Ramp-Up Time for Temperature Noise Reduction on Memory-based PUFs," IEEE intl. Symposium on Hardware-Oriented Security Trust (HOST), 2013.

[10] M. Bhargava, C. Cakir, and K. Mai, "Reliability Enhancement of Bi-Stable PUFs in 65nm Bulk CMOS," IEEE intl. Symposium on Hardware-Oriented Security Trust (HOST), 2012.

[11] Y. Zheng, S. Hashemian and S. Bhunia, "RESP: A Robust Physical Unclonable Function Retrofitted into Embedded SRAM Array," 50th ACM/IEEE Design Automation Conference (DAC), 2013.

[12] 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.

[13] M. Cortez *et al*, "Modeling SRAM Start-Up Behavior for Physical Unclonable Functions," IEEE Symp. Defect and Fault Tolerance in VLSI (DFT), 2012.

[14] S. Sabadc and D. Walker, "Improved Wafer-Level Spatial Analysis for IDDQ Limit Setting," Proceeding of International Test Conf. (ITC), pp. 82-91, 2001.

[15] W. Daasch and K. Cota, "Variance Reduction Using Wafer Patterns in IDDQ Data," Proceeding of International Test Conf. (ITC), pp. 189-198, 2000.

[16] J. Lee and D. Walker "IC Performance Prediction for Test Cost Reduction," IEEE Int. Symp. on Semiconductor Manufacturing Conference, pp. 111-114, 1999.