J Electron Test manuscript No.
(will be inserted by the editor)

Aging-Resilient SRAM-based True Random Number Generator for Lightweight Devices

Wendong Wang [1] · **Ujjwal Guin**[1] · **Adit Singh**[1]

Received: date / Accepted: date

Abstract A random number generator (RNG) is an important building block for cryptographic operations primarily to generate random nonces and secret keys. The power-up value of an SRAM array has been widely accepted as an entropy source for generating random numbers. However, only a few cells of the SRAM are truly random upon repeated power-ups; the vast majority of cells display a distinct bias from manufacturing process variations. Consequently, a relatively large SRAM array is required to obtain sufficient entropy for generating random numbers. Earlier research has proposed the use of controlled device aging at pre-deployment stage to enhance the initial entropy of an SRAM array. However, aging in the field can adversely affect the entropy and degrade randomness; we show here that any initial aging to increase SRAM entropy can even be counter productive. Instead, we propose an SRAM-based random number generation approach, which continually manipulates device aging during operation to constantly maximize entropy for the entire deployment period. The key idea is to continually stress the SRAM cells in their power-up states at regular intervals. This helps counteract the aging caused by the random memory states that occur during operation. Silicon results are presented to validate our proposed approach.

Keywords TRNG · NBTI · process variation · entropy

1 Introduction

With the advancement of ubiquitous distributed computing under the broad umbrella of the Internet of Things (IoT), the number of connected devices (edge devices) is expected to grow exponentially in the coming decades. wide use of these edge devices, which can be expected to include a large share of Cyber-Physical Systems (CPS), in critical applications (*e.g.* smart home, smart city, automated transportation etc.) will present unique security challenges. Due to the resource constraints of low cost, low power hardware, the majority of edge devices today are unable to use standard cryptographic protocols to communicate securely. Trappe et al. have shown how the power constraint in IoT edge devices limits the encryption functionality of the sensor nodes, which leads to poorly encrypted communication, or often no encryption at all [33]. In particular, the inability to incorporate cryptographic primitives in low cost IoT edge devices results in significant vulnerability in their communications [11, 13]. Consequently, reliably authenticating the devices and securely controlling access to them, maintaining confidentiality for all communications, and also ensuring data integrity, all become very challenging.

Clearly, in order to facilitate large scale adoption in diverse IoT/CPS applications, the IoT devices must be of low-cost. This requires that any added security features reuse available hardware resources to the greatest extent possible. It is widely recognized that power-up state of the available SRAM memory in the IoT edge devices can been exploited as a low cost entropy source for implementing a hardware-based random number generator (RNG). However, such a RNG must operate reliably, without degradation in output quality, over long periods in harsh conditions since IoT nodes may be deployed over a diverse range of physical locations and challenging environments. This paper addresses the prob-

---

Wendong Wang
E-mail: wendong@auburn.edu

Ujjwal Guin
E-mail: ujjwal.guin@auburn.edu

Adit Singh
E-mail: adsingh@auburn.edu

[1] Department of Electrical and Computer Engineering
Auburn University

lem of ensuring the randomness and quality of SRAM-based RNGs in the face of device stress and long term aging.

Random number generators (RNG) have extensive practical applications. Besides cryptographic algorithms and secure protocols, random number also play a crucial role in IP piracy and IC overproduction [12, 22], countermeasures against side channel attacks [9, 29], generating nonces or random seeds [19,21,30], and anti-counterfeit measures [10]. Consequently, much research is focused on developing "true" RNGs (TRNGs) with provable randomness properties as specified in certification tests, such as those recommended by National Institute of Standards and Technology (NIST). In [2, 25, 26, 37], the authors proposed a ring oscillator based TRNG by taking advantage of process variations and dynamic temperature variations. Whereas, the authors utilize metastable mode of digital components to implement TRNGs-[32,36]. Several researchers have also implemented TRNGs utilizing memories. In [39], the author proposed a flash memory based TRNG by taking advantage of random telegraph noise in the intrinsic circuit. The DRAM based TRNG is the one of direction. In [8], the author proposed using startup value of DRAM to implement random number generator. In [31], the authors use the data remanence effect of DRAM to implement a TRNG. As the static random-access memory (SRAM) is one of basic memory component in computing systems, many researchers have tried to use it to develop a TRNG.

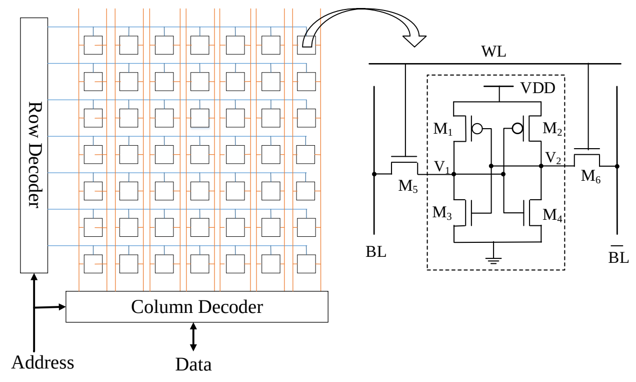
Researchers have developed TRNG structures utilizing SRAM as an entropy source [5,14,18]. By combining the entropy source with a pseudo random number generator (PRNG), these random number generators can achieve a relatively high throughput. However, limited entropy from an SRAM array poses a potential risk for any SRAM based TRNG as majority of cells are stable at either 0 or 1. Herrewege *et al.* presented a PRNG structure based on the power-up state of an SRAM array by implementing in commercial off-the-self microcontrollers [35]. The author showed that the SRAM present in these microchips (*e.g.* PIC16F1825) cannot securely seed the PRNG. The authors in [17, 35] showed that the entropy of an SRAM array decreases with the decrease of temperature. This may cause freezing attacks, where an attacker can expose an SRAM chip to a low temperature. These prior works indicate that entropy of power-up state of SRAM is very crucial parameter for creating a secure PRNG. Consequently, researchers have introduced new methods to improve and preserve the entropy of the SRAM array. In [6] and [23], the authors proposed a minor modification to the conventional SRAM array to improve the instability of SRAM cells. In [16], the authors exploit device aging to improve the initial entropy of the SRAM array at deployment.

This paper is focused on bias temperature instability (BTI) based degradation of SRAM-based RNGs due to operational stress over time. BTI is mostly observed to impact PMOS transistors as Negative BTI (NBTI) in traditional bulk technologies [24, 27], but the discussion equally applies to Positive PBTI (PBTI), if there is significant aging degradation in the NMOS transistors. The primary contributions of the paper are two-fold. First, we demonstrate that any initial enhancement of the entropy of an SRAM array through controlled and selective aging of devices prior to deployment (as proposed, for example, in [16]) is not retained for very long during actual use, and can have an adverse effect on security if relied upon at design time. This is because, with many different random data patterns stored in a functional SRAM during normal operation, all devices get maximally aged from stress over time as the increase in threshold voltages of transistors due to NBTI saturates. All transistors now experience nearly equal elevation in threshold voltage. Consequently the net imbalance in threshold voltages in each SRAM cell, which decides the power-up state of the cell, returns to what it was before any compensating aging stress was applied, i.e., as at the time of manufacture. Thus aging cannot be reliably used to increase long term entropy in the SRAM states at power-up if the memory is simultaneously also used for functional purposes, as is common in low cost applications. In actual fact, SRAM entropy can often degrade below it's initial level at manufacture during operation due to uneven aging stress caused by repeated patterns in the data stored in memory. The resulting loss of randomness in the RNG can present a significant security risk. This motivates the second contribution of this paper: the development of a controlled periodic aging strategy during operation that prevents any significant degradation of entropy in the SRAM throughout its operational life, and thereby ensures that the RNG always satisfies its randomness specifications. (Illustrated later in Figure 2.) The concepts and methodologies presented in this paper have been validated through extensive silicon experiments as discussed later Sections.

The rest of the paper is organized as follows. Section 2 introduces the modeling of power-up state for an SRAM, and how it is impacted by the aging. Section 3 discusses the our proposed approach for generating true random numbers. Experimental results are results are given in Section 4. Finally, we conclude our paper in Section 5.

2 Background: Randomness and the Power Up-state of SRAMs

The most significant device parameters influencing the power-up state of an SRAM cell are the threshold voltages (v_{th}) of the MOS transistors. Note that a conventional SRAM cell consists of six transistors as shown in Figure 1. Here four of the transistors (M_1, M_2, M_3 and M_4) form a bistable latch that stores the 1-bit of data in the cell. BL, and $\overline{BL}$ are the complementary bit lines that provide access to the stored bit through the access transistors M_5 and M_6.



(a) SRAM array (b) 6T SRAM cell

Fig. 1: Simplified architecture of an SRAM array and a six-transistor SRAM cell.

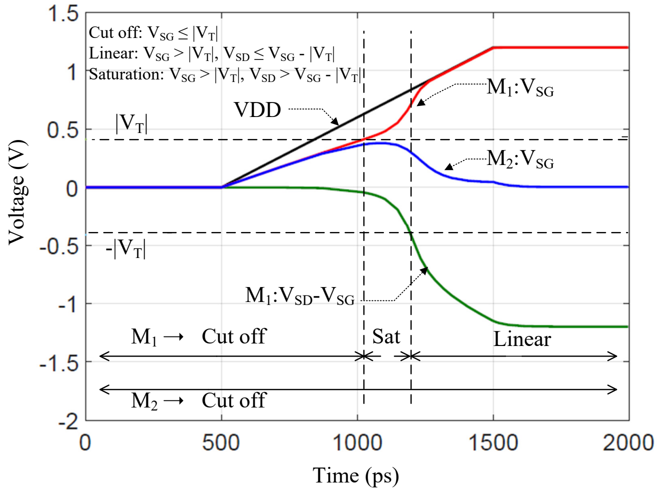


Fig. 2: Timing diagram of internal nodes of an SRAM Cell during the power-up.

2.1 Effect of Process Variations on the Power-up State

When an SRAM array is powered-up, initially each individual memory cell randomly acquires either a logic 0 or logic 1 value. During design, the MOS transistors in each matched pair (M_1 and M_2,M_3 and M_4, and M_5 and M_6) in Figure 1 are carefully layed out and fabricated to be completely identical, including all their layout related parasitic components. The perfect symmetry of the memory latch in a SRAM cell maximizes noise margins during operation. Consequently, each SRAM cell should ideally have a 50% chance of acquiring either logic 0 or logic 1 when powered up, the actual value decided by random unbiased noise. However, in practice most MOS transistor pairs will not be perfectly matched due to random manufacturing process variations, the most significant of which are the small variations in the threshold voltages in each MOSFET. v_{th} differences in the PMOS and NMOS transistor pairs can either both cause a cell bias in the same direction or in opposite directions; the net imbalance decides the overall bias towards either 1 or 0 at power-up. A larger net imbalance in the transistor pairs result in a more skewed SRAM cell. If the net v_{th} difference is small, the power-up values may be still be somewhat random, with a bias towards either 0 or 1. On the other hand, for relatively large net v_{th} imbalances, the power-up state may be stable and always the same over multiple power-up cycles. Based on this behavior at power-up, the SRAM cells have been divided into three categories in the literature – no-skewed cells, partially-skewed cells, and fully-skewed cells [7]. Experience shows that 80-90% of the cells in an SRAM are typically fully-skewed or stable cells at power-up, 5-15% are partially skewed unstable cells that sometimes display random behavior, while less than 5% display highly random behavior with nearly equal probability of powering-up in either state.

In order to investigate how the difference in v_{th} influence the power-up state of SRAM in more detailed way, the HSPICE simulation have been performed by using 32 *nm* bulk Predictive Technology Model (PTM). Figure 2 shows the voltage potential of different nodes for a single SRAM cell corresponding to Figure 1.(b). In the design stage, the parameter of all the MOSFETs are intended to be identical (M_1,M_2, M_3,M_4). However, each MOSFET will obtain variability resulted from manufacturing process variation. To explain the power-up behavior, we ignore the process variation of NMOS (M_3 and M_4) transistors (for simplicity and the power-up of an SRAM cell depends of PMOS transistors for fast power supply ramp [38]. The v_{th} of M_2 is assumed to be higher than M_1 (absolute value) for 20%. As depicted from Figure 2, the potential of node 1 (V_1) and node 2 (V_2) increase nearly at the same rate initially. However, at some point of time, M_1 reach the saturation region due to lower V_{th}. In contrast, M_2 remain in cut-off region. Consequently, potential across node 1 (V_1) reaches to VDD as higher current will flow through M_1 compared with M_2, while potential across node 2 (V_2) decreases to zero. In summary, the cell will power up with 1 if the absolute value of v_{th} of M_2 is higher than M_1, and vice versa.

2.2 SRAM-Based Random Number Generation

While the vast majority of bits in any SRAM are fully-skewed and therefore, stable at power-up, a large SRAM still has enough randomness (entropy) to support generation of a random number. This is readily seen by considering the result of an XOR operation over all the bits of the SRAM. The resulting single binary bit will be quite random and unpredictable over multiple power-up instances if even a relatively few bits in the SRAM array are unstable. A multi-bit random number, for example 32-bits, can be easily obtained by partitioning a large SRAM into 32 arbitrary blocks, and then performing XOR operations on each of the individual blocks to generate the 32 random bits, i.e., one bit from each block. In practice, as discussed in the next section, RNGs used in security applications employ more sophisticated methods to extract the entropy (often referred as entropy pool, see Figure 4) instead of the simple XOR operation.

2.3 SRAM Entropy and Aging

Observe that RNG randomness measures can be further improved if there are more unstable bits in the SRAM. Some research, e.g., [6], has proposed modifications to the conventional SRAM array to increase the instability of SRAM cells. However, such a custom approach may not be appropriate for low cost IoT nodes. More recent work has suggested the use of controlled device aging to maximize the number of unstable cells. Recall that PMOS transistors experience NBTI aging under negative bias stress, resulting in an increase in the magnitude of their threshold voltage over time from a negative bias stress at the gate. NMOS devices experience a similar PBTI effect, but this has traditionally been observed to be much smaller. The NBTI stress windows are the periods when the PMOS transistors are ON; similarly NMOS transistors are stressed when they are ON. There is some partial recovery from BTI degradation when the stress is removed, i.e., when the transistors are OFF, but, for random inputs, the average threshold voltage shift due to aging tends to increase over time and finally saturate in the long run as shown in Figure 6.

In newly manufactured SRAMs, where the threshold shifts from aging have not yet occurred and are far from saturation, controlled aging offers a way to counteract the inherent bias in cells from process variations and increase the number of unstable cells. Consider an SRAM cell that is powered-up and acquires a low (state 0) at the left output of the latch (transistors M_1 and M_3 in Figure 1). This cell obviously has an inherent 0 bias which can be caused by the left NMOS transistor in the NMOS transistor pair having a smaller threshold voltage, and/or the right PMOS in the PMOS pair having a smaller (in magnitude) threshold voltage . Lower threshold transistors are the first to turn ON at power up, which in this case would force the left output to 0 and the right output to 1. Notice that after power up these very same transistors, the left NMOS and the right PMOS are ON and therefore under BTI stress. This tends to increase the magnitude of their threshold voltages, thereby reducing the bias in the cell over time because the complementary transistors in each pair are OFF and therefore do not similarly degrade with time. Thus, BTI aging can be used to reduce, and even overcome, the inherent bias in SRAM cells through deferentially aging transistor pairs to create an opposite bias by holding the initial power-up state for a controlled period. This approach has been proposed [16] as a method to increase cell instability and thus to increase the SRAM entropy before it is deployed in the field.

However, the problem with such an methodology is that this increase in entropy, which depends on differential aging of the transistor pairs in the SRAM cells, can only work when the transistors are new and relatively unaged. Over time, once all transistors in the SRAM get significantly aged from the stress of a very large number of random functional data cycles, the threshold voltage degradation in all PMOS transistors saturates at a nearly the same levels, eliminating any differential impact, and hence returning the SRAM cells to their biases at the time of manufacture. Thus, if a RNG is designed with an SRAM assuming the higher level of entropy achieved through selective aging pre-deployment, it may fail randomness specifications over time as the number of unstable bits in the SRAM decrease. This is clearly observed in the silicon results presented in later sections.

From the above discussion, it is clear that RNGs must be designed assuming that SRAM entropy cannot be reliably increased beyond the level inherent at manufacture. On the other hand, our silicon results show that on the SRAM entropy can actually drop below its initial level during the early period of deployment, before device aging saturates. This occurs because not all stress in functional memory during operation is entirely random. For example, memory data statistics show that in some applications, SRAM cells have a distinctly higher likelihood of storing a 0 value [40]. Such a storage pattern will tend to increase cell bias dis-proportionally in one direction, reducing the number of unstable cells. We therefore develop a controlled aging strategy to ensure that the SRAM entropy never drops (within bounds) below its initial entropy at manufacture anytime during deployment. This can be achieved by periodically powering up the SRAM and then holding the power-up state for a calculated window of time to neutralize any built up bias from systematic stress. This period should be small enough to allow only minimum shifts in the threshold voltage, well within the threshold voltage mismatch range acceptable for RNG operation, to ensure that the controlled BTI stress cycle does not itself leave an unacceptable bias in any cell. The power up and hold operation can then be repeated as frequently as needed to ensure proper RNG operation. Notice that any SRAM cell, utilized in the RNG, that for any reason develops an unwanted bias will be stressed back towards neutrality during each controlled BTI stress cycle, with small threshold voltage shifts. In case the threshold shift caused by the BTI stress overshoots neutrality and creates an opposite bias in the cell, then during the next controlled stress cycle, the cell will power up in the complementary state creating BTI stress in the complementary set of transistors, thereby again driv-

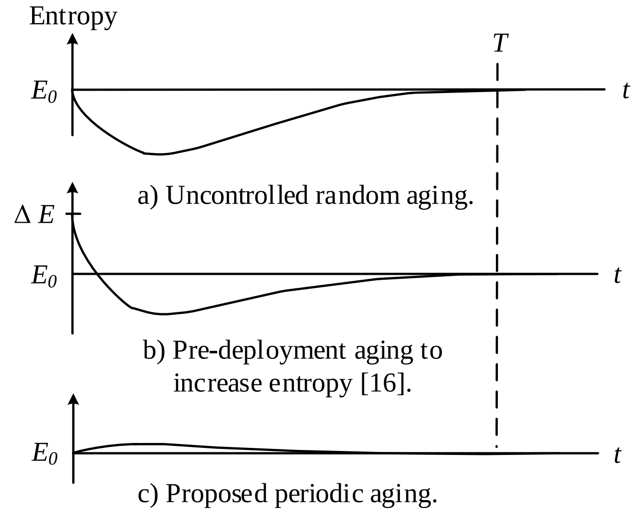


Fig. 3: SRAM entropy versus deployment time for different aging strategies.

ing the cell bias towards neutrality. The proposed methodology is thus fully self-correcting.

The plot in Figure 3.a illustrates the change in entropy from uncontrolled aging during the normal operation of an SRAM in the field. Following deployment, the entropy initially decreases from the biased stress that exists even in the random data, because v_{th} shifts from aging are the greatest in new devices. However, it soon starts recovering as the v_{th} shifts start saturating and therefore equalizing in the SRAM transistor pairs (Figure 3.a). The entropy ultimately recovers to its initial level (E_0) at time T. At this stage, aging degradation has saturated and virtually equalized in all the transistors, bring entropy back to the unaged level. The approach proposed in [16] increases the initial entropy ($E_0 + \Delta E$) with careful pre-deployment aging (Figure 3.b). However, the entropy quickly drops below E_0 after just a short period of deployment in the field, and again slowly recovers to back to E_0. Finally, Figure 3.c shows how we can always maintain (or exceed) the initial entropy using our proposed periodic aging strategy. This guarantees the randomness specifications for the TRN throughout the deployment window.

3 Proposed Approach for Creating an Aging-Resilient SRAM-based RNG

As discussed earlier, the power-up bits generated from an SRAM suffer from low entropy as the majority of the SRAM cells are biased towards 0 or 1 due to process variation. In a standard SRAM array, around 80-90 percent of cells are stable (stable at either 0 or 1) [20]. As a result, we need a large SRAM array to create a true random number. It is often tempting to create an SRAM-based RNG with increased entropy, such that it can be well-suited to resource constrained devices in an IoT or CPS application. In this section, we describe why the initial increase of the entropy of an SRAM array using accelerated aging can create an adverse effect on the bits produced from a RNG while they are operating in the field. We also propose an approach to compensate for any degradation from aging in SRAM-based RNGs.

3.1 Implementation of an SRAM-based RNG

There are two ways for implementing a random number generator (RNG), namely, non-deterministic and deterministic [4]. In a non-deterministic RNG, every bit comes from the physical random source and such a bit is completely random and unbiased. Deterministic RNGs use an algorithm to generate the sequence of true random bits starting with a seed (entropy source) which is random, but whose individual bits may display some bias. In this paper, we treat the SRAM as this entropy source, and then use a hash function to generate the true random number. [3]. Figure 4 shows an implementation of the SRAM-based RNG, where the SRAM array provides the necessary entropy to create the random bits. The entropy pool behaves like an entropy extractor. A secure hash function (SHA-2/SHA-3) is used for extracting the entropy [18]. One can also use AES-CBC-MAC for such purpose [15]. Note that the size of the SRAM array should be large enough to provide the necessary entropy (e.g., entropy should be approximately 256 bits (or more) for a 256-bit random number).

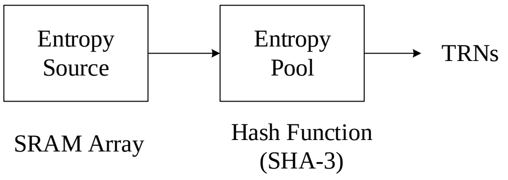


Fig. 4: A typical RNG implementation [4, 15], which uses an SRAM memory.

The entropy, which refers to the disorder or uncertainty, of a source can be calculated using Shannon's formula [28]:

$$E = -\sum_i p(x_i)\log_2 p(x_i) \tag{1}$$

where $p(x_i)$ denotes the probability of observing pattern x_i. For a single bit binary source, the entropy can be calculated as:

$$E_b = -\{p(x_0)\log_2 p(x_0) + p(x_1)\log_2 p(x_1)\} \tag{2}$$

where $p(x_0)$, and $p(x_1)$ denotes the probability of occurring $x = 0$, and $x = 1$, respectively. As an SRAM array consists of multiple SRAM cells, where the power-up state of one SRAM cell is independent of others, we can compute the average entropy per bit using the following equation:

$$E_{b_{avg}} = \frac{1}{S}\sum_{i=1}^{S} E_{b_i} \tag{3}$$

where S represents the SRAM array size, *i.e.*, the number of SRAM cells. To design an SRAM-based RNG, the entropy should be at least equal to or larger than the random bits generated from it.

3.2 Aging-Resilient SRAM-based RNG

An unbiased SRAM cell, which primarily contributes to the entropy of the SRAM array, may become biased due to aging from usage in the field. As a result the overall entropy from an SRAM array can reduce over time. In this section, we explain why pre-deployment initial aging (simple or accelerated) to increase the entropy of an SRAM array can have a negative impact on the randomness of the bits generated from a RNG over time. Instead, we propose an alternate strategy of controlled aging of the SRAM array at regular intervals to compensate for the loss of bias in normal usage and maintain SRAM entropy.

The entropy of an SRAM array results from the balanced cells, where the threshold voltages of transistor pairs is virtually equal at manufacturing and the power-up state is equally likely to be 0 or 1. However, more than 80% of the SRAM cells are usually biased towards either 0 or 1 due to manufacturing process variations. Kiamehr *et al.* [16] proposed a solution by aging an SRAM array just after manufacturing to increase the number of unbiased cells in the array. The scheme exploits NBTI aging to selectively increase v_{Thu} of the PMOS in one inverter by stressing the cell in the power-up state. It can thereby obtain, on average, more balanced cells and higher entropy. The approach is particularly effective because v_{Thu} shifts from aging are faster and larger in a newly manufactured part. However, this scheme creates a vulnerability when an SRAM is deployed in the field. As the other PMOS transistor still remains fresh after this initial aging, it can age at a much faster rate when random data is stored in the SRAM, quickly neutralizing any initial increase in entropy.

Figure 5 shows a simplified schematic of the SRAM cell (depicted in Figure 1.b) to analyze it's behavior under aging. We can model the output node capacitance by adding two large lumped capacitors, C_1 and C_2 at the node 1 (output of inverter 1) and node 2 (output of inverter 1), respectively. For the simplicity, we have removed the access transistors (M_5, and M_6). At the design stage, both transistor pairs are balanced such that $M_1 - M_2$, and $M_3 - M_4$ have the same parameters, and all parasitics are the same for both inverters.

Figure 6 shows the change in (the absolute value of) v_{th} due to post-deployment aging on a cell where controlled pre-deployment aging is used to balance the cell and increase cell entropy. We use HSPICE MOSRA [34] to simulate the aging. An SRAM cell is designed using 32*nm* PTM technology [1], where the threshold voltages (in absolute value) are $v_{t1} = 0.4$ and $v_{t2} = 0.408$. If we initially age the cell while storing 0, the transistor M_1, with the lower v_{th} (in magnitude) will experience aging due to NBTI and its threshold voltage increases to $v^*_{t1} (> v_{th})$ (see Figure 5). After the initial aging (t_1), the v_{th} for both the transistors will become equal and the SRAM cell becomes unbiased. However, once the chip is used in the field, the SRAM sees random data. To simulate this part, we age the SRAM cell with random data. At time t_1, the rate of v_{th} change for both the transistors is not equal. The transistor M_2 ages much faster, and the initial bias quickly returns. As a result, any initial compensation becomes ineffective.

We next analyze how the aging with the power-up state affects the bias of an SRAM cell. Assume that initially the

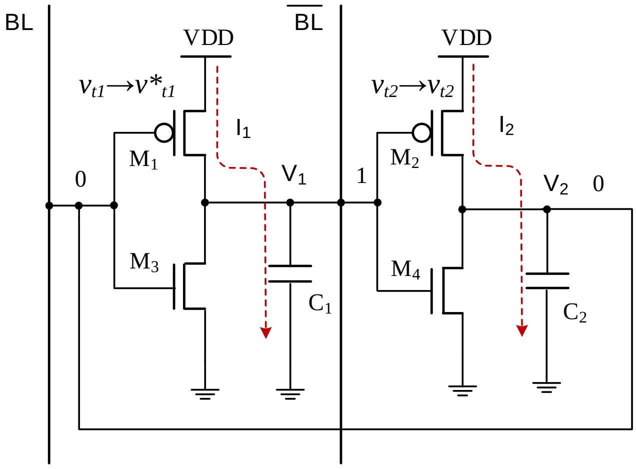


Fig. 5: Simplified schematic of an SRAM cell to explain the power-up behavior.

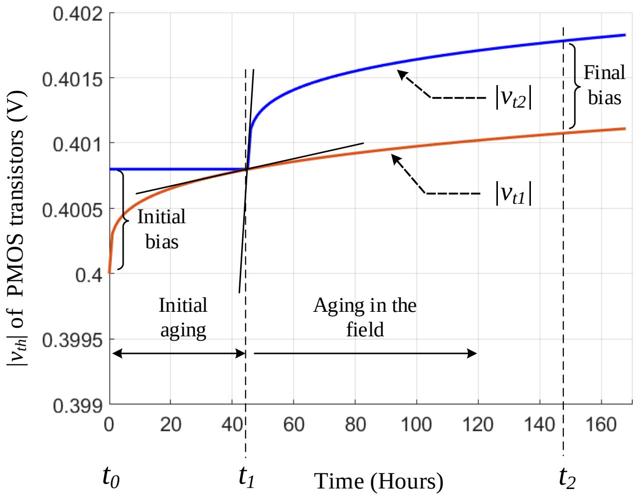


Fig. 6: Controlled pre-deployment aging effect on SRAM cell.

threshold voltage of M_2 (v_{t2}) is higher than the threshold voltage of M_2 (v_{t1}) (see Figure 5). For simplicity, the effect of noise is omitted in the discussion. The cell will power-up with 0 as M_1 will quickly reach to saturation, and the node 1 (V_1) will pull up to VDD. If we keep this state, transistor M_1 will be NBTI stressed as its gate is negative and v_{t1} will be increased in magnitude to v^*_{t1}. On the other hand, transistor M_2 will remain fresh. If we repeatedly power up the SRAM array and hold the state each PMOS transistor will age alternatively and remain unbiased if the v_{th}s of both the PMOS transistor's are the same. If they are not, v_{t1} will be increased incrementally to reach $v_{t2} (> v_{th1})$ (or vice versa), and the cell will then stay unbiased.

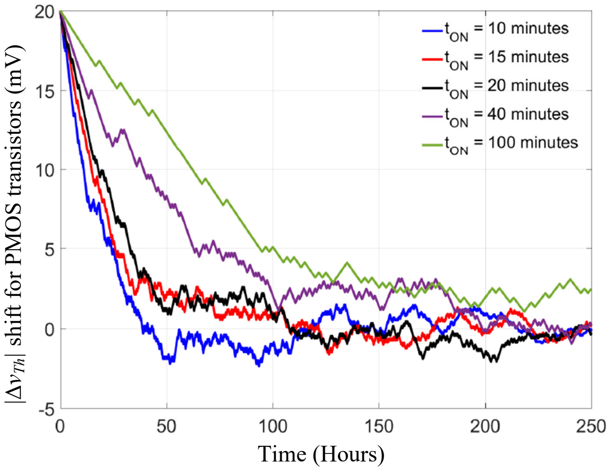


Fig. 7: Simulation for Δv_{th} shift for the PMOS pairs in periodic aging.

Noise can play an important role to create bias in SRAM cells if we perform aging with the power-up state. Stressing a PMOS transistor for long time can create a significant shift in its v_{th}, which can create an increased bias for an SRAM cell. We need to carefully choose aging duration (t_{ON}) such that no significant bias is created. We next report on experiments that analyze how quickly the v_{th} of both PMOS transistor converge. HSPICE simulation is performed, where We choose $v_{t1} = 0.042V$ and $v_{t2} = 0.04V$, and keep the threshold voltages for the NMOS transistor the same for this experiment. Gaussian noise with $\mu = 0$ and $\sigma = 15mV$ (at node 1 and 2 in Figure 5) to emulate the power-up behavior. Figure 7 show the simulation results for Δv_{th} of PMOS transistors versus aging time with different t_{ON} values. A smaller t_{ON} value results in the convergence of both the threshold voltages. We can select this value for compensating the degradation caused during the normal operations of an SRAM chip.

4 Experimental Results from Silicon

Initial aging after manufacturing can increase the overall entropy of an SRAM array, however, over time it can degrade in use and adversely affect the random numbers generated from an SRAM array. This section demonstrates the reduction of entropy from an SRAM array when it gets aged in the field using silicon data. We have selected a commercial off-the-shelf SRAM memory (Microchip 23A640-I/SN - SPI Bus Low-Power Serial SRAM) to perform the experiment. The size of the SRAM chip is 64K bits. Figure 8 shows the experimental setup for reading the power-up state of the SRAM chip. A voltage shifter (Texas Instruments PCA9306 Dual Bidirectional I^2C Bus and SMBus Voltage-Level Translator) is used to interface the Raspberry Pi with the SRAM. The experiment is conducted at the room temperature.

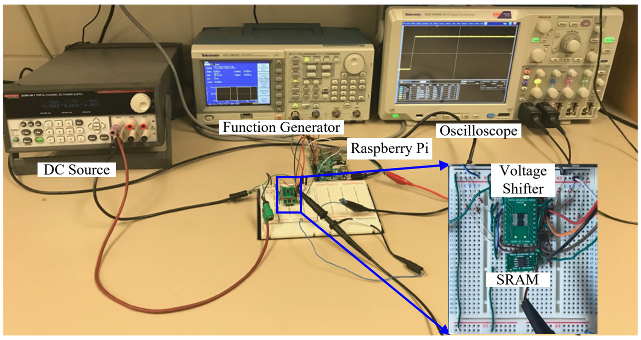


Fig. 8: Experimental set-up to measure SRAM power-up states.

Figure 9 shows the change in average entropy per bit over time. We measure the number of 0s and 1s, and thereby estimate the probability of observing a 0 and 1, at each address location by performing repeated (100) power-up cycles. This allows us to calculate the average entropy per bit (using Equation 3) for the SRAM. Initially, three new SRAM chips were aged for six days at room temperature using periodic aging. We select a controlled aging interval of 15 minutes. We observe that, as expected, the entropy increases quickly initially and then starts saturating. The SRAM chips are then kept on the shelf for few days. Due to some partial recovery of v_{th}, the average entropy per bit decreases during this period. We now emulate the behavior of usage in the field by aging the chip with random patterns, and measure entropy once in a day. We observe a rapid decrease of the entropy which was initially gained through compensation. Over time this starts saturating, but at levels even below the initial entropy. Note that all the three SRAM chips follow a similar trend. (More long duration silicon experiments are underway, but because of lime limitations, will only be available for the final version of this paper, if accepted.)

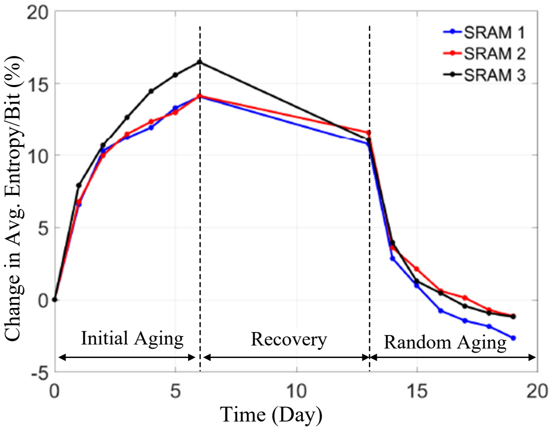


Fig. 9: SRAM entropy versus time.

Figure 10 shows the change in average entropy per bit by using our proposed method. In this experiment, we use two new SRAM chips and the average entropy per bit is calculated by measuring 100 power-up states. First, we measure the entropy of the new SRAM chip. To emulate the usage behavior in the field, these SRAM chips have been aged using random patterns for one day and entropy have been measured. We then perform the periodic aging to compensate the degradation. From the Figure 10, we can observe the entropy start to drop after aging with random patterns. The periodic aging for 3 hours boosts the entropy to its initial value. The dotted line shows the overall trend of entropy change. The silicon data from these two SRAMs clearly shows that our proposed method preserves the initial entropy of the SRAM chips.

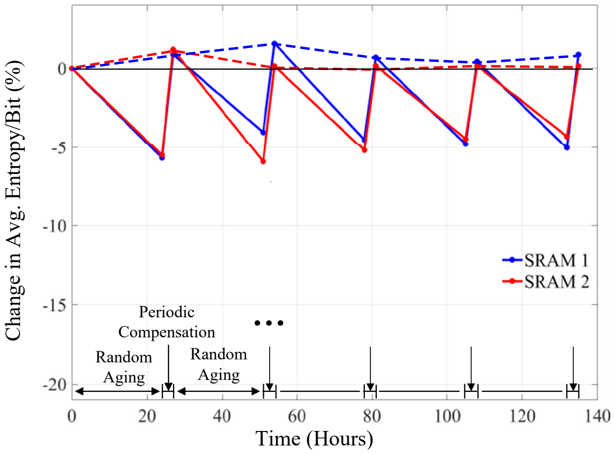


Fig. 10: Periodic compensation by using our proposed method.

5 Summary and future work

In this paper, we have presented an approach to preserve the entropy of the power-up states of an SRAM array during deployment. We have shown that SRAM entropy can decrease in deployment from device aging while in use. Consequently, SRAM based RNGs need to manage entropy to ensure high quality random numbers. Unfortunately, any pre-deployment increase in entropy through controlled aging as suggested in prior research cannot be sustained in deployment. Our experimental result shows that we can preserve the initial entropy of COTS SRAM chips using periodic compensation by repeatedly powering up the SRAM chip and then holding its power-up state.

Acknowledgements

This work was supported in parts by the National Science Foundation under Grant Numbers CNS–1755733 and CCF–1527049.

References

1. Predictive Technology Model (PTM), http://ptm.asu.edu/
2. Amaki, T., Hashimoto, M., Onoye, T.: An oscillator-based true random number generator with process and temperature tolerance. In: Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 4–5 (2015)
3. Barker, E., Kelsey, J.: Recommendation for random number generation using deterministic random bit generators. NIST Special Publication **800**, 90A (2015)
4. Barker, E.B., Kelsey, J.M.: Recommendation for random number generation using deterministic random bit generators (revised). National Institute of Standards and Technology (2007)
5. Chen, S., Li, B.: A dynamic reseeding drbg based on sram pufs. In: Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2016 International Conference on, pp. 50–53. IEEE (2016)
6. Clark, L.T., Medapuram, S.B., Kadiyala, D.K.: Sram circuits for true random number generation using intrinsic bit instability. IEEE Transactions on Very Large Scale Integration (VLSI) Systems (99), 1–11 (2018)
7. Cortez, M., Dargar, A., Hamdioui, S., Schrijen, G.J.: Modeling sram start-up behavior for physical unclonable functions. In: Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), pp. 1–6 (2012)
8. Eckert, C., Tehranipoor, F., Chandy, J.A.: Drng: Dram-based random number generation using its startup value behavior. In: 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1260–1263 (2017)
9. Fischer, V.: A closer look at security in random number generators design. In: International Workshop on Constructive Side-Channel Analysis and Secure Design, pp. 167–182. Springer (2012)
10. Guin, U., Bhunia, S., Forte, D., Tehranipoor, M.M.: Sma: A system-level mutual authentication for protecting electronic hardware and firmware. IEEE Transactions on Dependable and Secure Computing **14**(3), 265–278 (2016)
11. Guin, U., Cui, P., Skjellum, A.: Ensuring Proof-of-Authenticity of IoT Edge Devices using Blockchain Technology. In: IEEE International Conference on Blockchain (2018)

12. Guin, U., Shi, Q., Forte, D., Tehranipoor, M.M.: FORTIS: a comprehensive solution for establishing forward trust for protecting IPs and ICs. ACM Transactions on Design Automation of Electronic Systems (TODAES) **21**(4), 63 (2016)
13. Guin, U., Singh, A., Alam, M., Canedo, J., Skjellum, A.: A Secure Low-Cost Edge Device Authentication Scheme for the Internet of Things. In: International Conference on VLSI Design (2018)
14. Holcomb, D.E., Burleson, W.P., Fu, K.: Power-up sram state as an identifying fingerprint and source of true random numbers. IEEE Transactions on Computers **58**(9), 1198–1210 (2009)
15. John, M.: Intel digital random number generator (drng) software implementation guide (2018). Https://software.intel.com/en-us/articles/intel-digital-random-number-generator-drng-software-implementation-guide
16. Kiamehr, S., Golanbari, M.S., Tahoori, M.B.: Leveraging aging effect to improve sram-based true random number generators. In: Proceedings of the Conference on Design, Automation & Test in Europe, pp. 882–885. European Design and Automation Association (2017)
17. Krentz, K.F., Meinel, C., Graupner, H.: Secure self-seeding with power-up sram states. In: 2017 IEEE Symposium on Computers and Communications (ISCC), pp. 1251–1256 (2017)
18. van der Leest, V., van der Sluis, E., Schrijen, G.J., Tuyls, P., Handschuh, H.: Efficient implementation of true random number generator based on sram pufs. In: Cryptography and Security: From Theory to Applications, pp. 300–318. Springer (2012)
19. Li, D., Lu, Z., Zou, X., Liu, Z.: PUFKEY: A high-security and high-throughput hardware true random number generator for sensor networks. Sensors **15**(10), 26251–26266 (2015)
20. Maes, R., Rozic, V., Verbauwhede, I., Koeberl, P., Van der Sluis, E., van der Leest, V.: Experimental evaluation of physically unclonable functions in 65 nm cmos. In: Proceedings of the ESSCIRC, pp. 486–489 (2012)
21. Majzoobi, M., Koushanfar, F., Devadas, S.: Fpga-based true random number generation using circuit metastability with adaptive feedback control. In: International Workshop on Cryptographic Hardware and Embedded Systems, pp. 17–32. Springer (2011)
22. Rahman, M.T., Forte, D., Shi, Q., Contreras, G.K., Tehranipoor, M.: Csst: Preventing distribution of unlicensed and rejected ics by untrusted foundry and assembly. In: 2014 IEEE International symposium on defect and fault tolerance in VLSI and nanotechnology systems (DFT), pp. 46–51 (2014)
23. Rahman, M.T., Forte, D., Wang, X., Tehranipoor, M.: Enhancing noise sensitivity of embedded srams for robust true random number generation in socs. In: 2016 IEEE Asian Hardware-Oriented Security and Trust (AsianHOST), pp. 1–6 (2016)
24. Reddy, V., Krishnan, A.T., Marshall, A., Rodriguez, J., Natarajan, S., Rost, T., Krishnan, S.: Impact of negative bias temperature instability on digital circuit reliability. Microelectronics Reliability (2005)
25. Şarkışla, M.A., Ergün, S.: An area efficient true random number generator based on modified ring oscillators. In: 2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), pp. 274–278 (2018)
26. Şarkışla, M.A., Ergün, S.: Ring oscillator based random number generator using wake-up and shut-down uncertainties. In: 2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST), pp. 104–108. IEEE (2018)
27. Schroder, D.K., Babcock, J.A.: Negative bias temperature instability: Road to cross in deep submicron silicon semiconductor manufacturing. Journal of applied Physics (2003)
28. Shannon, C.E.: Prediction and entropy of printed english. Bell system technical journal (1951)
29. Srinivasan, S., Mathew, S., Ramanarayanan, R., Sheikh, F., Anders, M., Kaul, H., Erraguntla, V., Krishnamurthy, R., Taylor, G.: 2.4 ghz 7mw all-digital pvt-variation tolerant true random number generator in 45nm cmos. In: 2010 Symposium on VLSI Circuits, pp. 203–204. IEEE (2010)
30. Suresh, V.B., Burleson, W.P.: Entropy extraction in metastability-based trng. In: 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 135–140 (2010)
31. Tehranipoor, F., Yan, W., Chandy, J.A.: Robust hardware true random number generators using dram remanence effects. In: Hardware Oriented Security and Trust (HOST), pp. 79–84 (2016)
32. Tokunaga, C., Blaauw, D., Mudge, T.: True random number generator with a metastability-based quality control. IEEE Journal of Solid-State Circuits **43**(1), 78–85 (2008)
33. Trappe, W., Howard, R., Moore, R.S.: Low-energy security: Limits and opportunities in the internet of things. IEEE Security & Privacy **13**(1), 14–21 (2015)
34. Tudor, B., Wang, J., Liu, W., Elhak, H.: Mos device aging analysis with hspice and customsim. Synopsys, White Paper (2011)
35. Van Herrewege, A., van der Leest, V., Schaller, A., Katzenbeisser, S., Verbauwhede, I.: Secure prng seeding on commercial off-the-shelf microcontrollers. In: Proceedings of the 3rd international workshop on Trustworthy embedded devices, pp. 55–64. ACM (2013)
36. Vasyltsov, I., Hambardzumyan, E., Kim, Y.S., Karpinskyy, B.: Fast digital trng based on metastable ring oscillator. In: International Workshop on Cryptographic Hardware and Embedded Systems, pp. 164–180. Springer (2008)
37. Wang, K., Cao, Y., Chang, C.H., Ji, X.: High-speed true random number generator based on differential current starved ring oscillators with improved thermal stability. In: 2019 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1–5 (2019)
38. Wang, W., Singh, A., Guin, U., Chatterjee, A.: Exploiting power supply ramp rate for calibrating cell strength in sram pufs. In: Test Symposium (LATS), 2018 IEEE 19th Latin-American, pp. 1–6 (2018)
39. Wang, Y., Yu, W.k., Wu, S., Malysa, G., Suh, G.E., Kan, E.C.: Flash memory for ubiquitous hardware security functions: True random number generation and device fingerprints. In: IEEE Symposium on Security and Privacy (SP), pp. 33–47 (2012)
40. Wei, D., Deng, L., Zhang, P., Qiao, L., Peng, X.: Nrc: A nibble remapping coding strategy for nand flash reliability extension. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems pp. 1942–1946 (2016)

Wendong Wang received the M.S. degree in Electrical and Computer Engineering from Auburn University, Auburn, AL, USA in 2018. He is pursuing the Ph.D. degree from the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL, USA. His current area of research includes hardware security, cryptography primitives components, such as TRNG and PUF, detecting recycled chip.

Ujjwal Guin received his PhD degree from the Electrical and Computer Engineering Department, University of Connecticut, in 2016. He is currently an Assistant Professor in the Electrical and Computer Engineering Department of Auburn University, Auburn, AL, USA. He received his B.E.degree from the Department of Electronics and Telecommunication Engineering, Bengal Engineering and Science

University, Howrah, India, in 2004 and his M.S.degree from the Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA, USA, in 2010. Dr.Guin has developed several on-chip structures and techniques to improve the security, trustworthiness, and reliability of integrated circuits. His current research interests include Hardware Security & Trust, Supply Chain Security, Cybersecurity, and VLSI Design & Test. He is a co-author of the book *Counterfeit Integrated Circuits: Detection and Avoidance*. He has authored several journal articles and refereed conference papers. He was actively involved in developing a web-based tool, Counterfeit Defect Coverage Tool (CDC Tool), *http://www.sae.org/standardsdev/cdctool/*, to evaluate the effectiveness of different test methods used for counterfeit IC detection. SAE International has acquired this tool from the University of Connecticut. He is an active participant in SAE International's G-19A Test Laboratory Standards Development Committee. He is a member of both the IEEE and ACM.

Adit Singh received the B.Tech. degree from IIT Kanpur, Kanpur, India, and the M.S. and Ph.D. degrees from Virginia Tech,Blacksburg, VA, USA, all in electrical engineering.He was on the faculty at the University of Massachusetts Amherst, Amherst, MA, USA, and the Virginia Polytechnic Institute and State University (Virginia Tech), Blacksburg, VA, USA. He has also held visiting positions during sabbaticals, most recently as a Guest Professorat the University of Freiburg, Freiburg im Breisgau,Germany, in 2012. He is currently a James B. Davis Professor of Electrical and Computer Engineering at Auburn University, Auburn, AL, USA. He has served as a Consultant to many major semiconductor, test, and EDA companies around the world, including as an Expert Witness on patent litigation cases. He has authored over 250 research papers and holds international patents that have been licensed to industry. He is particularly recognized for his pioneering contributions to statistical methods in test and adaptive testing. His current research interests include VLSI technology, in particular integrated circuit test and reliability.

Dr. Singh has had leadership roles as a General Chair/Co-Chair/Program Chair for dozens of international VLSI design and test conferences. He was a Program Co-Chair for the 2014 International Conference on VLSI Design, and a Program Chair of the 2015 Asian Test Symposium. He also served on the editorial boards of several journals, including the IEEE Design and Test, and on the Steering and Program Committees of many of the major IEEE international test and design automation conferences. He served two elected terms as a Chair for the IEEE Test Technology Technical Council (2007-2011), and on the Board of Governors of the IEEE Council on Design Automation (2011-2015)