In-DRAM True Random Number Generation Using Simultaneous Multiple-Row Activation: An Experimental Study of Real DRAM Chips

İsmail Emir Yüksel[§] Ataberk Olgun[§] F. Nisa Bostancı[§] Oğuzhan Canpolat[§]
Geraldo F. Oliveira[§] Mohammad Sadrosadati[§] A. Giray Yağlıkçı[§Γ] Onur Mutlu[§]

[§]ETH Zürich [Γ]CISPA

In this work, we experimentally demonstrate that it is possible to generate true random numbers at high throughput and low latency in commercial off-the-shelf (COTS) DRAM chips by leveraging simultaneous multiple-row activation (SiMRA) via an extensive characterization of 96 DDR4 DRAM chips. We rigorously analyze SiMRA's true random generation potential in terms of entropy, latency, and throughput for varying numbers of simultaneously activated DRAM rows (i.e., 2, 4, 8, 16, and 32), data patterns, temperature levels, and spatial variations. Among our 11 key experimental observations, we highlight four key results. First, we evaluate the quality of our TRNG designs using the commonly-used NIST statistical test suite for randomness and find that all SiMRA-based TRNG designs successfully pass each test. Second, 2-, 8-, 16-, and 32-row activation-based TRNG designs outperform the state-of-the-art DRAM-based TRNG in throughput by up to 1.15×, 1.99×, 1.82×, and 1.39×, respectively. Third, SiMRA's entropy tends to increase with the number of simultaneously activated DRAM rows. For example, for most of the tested modules, the average entropy of 32-row activation is 2.51× higher than that of 2-row activation. Fourth, operational parameters and conditions (e.g., data pattern and temperature) significantly affect entropy. For example, increasing temperature from 50°C to 90°C decreases SiMRA's entropy by 1.53× for 32-row activation. To aid future research and development, we open-source our infrastructure at `https://github.com/CMU-SAFARI/SiMRA-TRNG`.

1. Introduction

True random number generators (TRNGs) harness entropy from random physical phenomena (e.g., electrical and thermal noise [1–3]) to generate unpredictable and irreproducible random bitstreams (i.e., true random numbers). True random numbers are crucial in many applications, including cryptography, scientific simulations, machine learning, and recreational entertainment [4–24]. These applications often require a high-throughput TRNG [24–27]. A dedicated hardware TRNG (i.e., an integrated circuit dedicated to true random number generation) can satisfy the throughput requirements of such applications. Unfortunately, *not* all computing systems have dedicated hardware TRNGs.

To alleviate the need for dedicated TRNG hardware, using DRAM [28] as the entropy source for generating true random numbers (i.e., DRAM-based TRNG) is a promising approach due to the prevalence of DRAM throughout almost all modern computing systems, ranging from microcontrollers to supercomputers, and systems that employ Processing-in-Memory (PIM) [29–46]. Several prior works [47–53] demonstrate that by leveraging the analog operational properties of DRAM, it is possible to generate true random numbers in commercial off-the-shelf (COTS) DRAM chips.

Recent works [35, 53–57] experimentally demonstrate a new phenomenon in COTS DRAM chips that could potentially be used as a source of entropy: *simultaneous multiple-row activation (SiMRA)*. These works [35, 53–57] carefully engineer a sequence of DRAM commands that allows the DRAM chip to activate (i.e., open) multiple (i.e., up to 32) DRAM rows simultaneously [56, 57]), which is fundamentally different from the conventional single-row DRAM activation.

We hypothesize that SiMRA could be used as the entropy source of DRAM-based TRNGs. This is because manufacturing process variations could introduce high unpredictability in the charge-sharing process when multiple rows are activated simultaneously. Therefore, SiMRA could potentially provide high entropy and thereby be leveraged for high-throughput and low-latency true random number generation. To design efficient and high-throughput DRAM-based TRNGs, it is important to develop a better understanding of the entropy that SiMRA can provide.

In this work, our *goal* is to experimentally study and understand the potential of SiMRA as a TRNG substrate in COTS DRAM chips. To this end, we provide two main analyses in our study. First, we experimentally characterize the randomness (quantitatively measured using Shannon entropy [58], described in §3.2) that SiMRA provides in 96 COTS DDR4 DRAM chips while varying a large number of parameters: the number of simultaneously activated rows, DRAM chip density & revision, data pattern, temperature, and spatial variation across DRAM subarrays. Second, we evaluate the quality (using the standard NIST statistical test suite for randomness [59]), throughput, and latency of new TRNG designs that leverage SiMRA for varying numbers of simultaneously activated rows.

Based on our real DRAM chip experiments, we make 11 new empirical observations and share 5 key takeaway lessons. We summarize four major results from our analyses. First, we observe that a TRNG that leverages SiMRA (SiMRA-TRNG) is a high-quality TRNG. We show that the random bitstreams generated by SiMRA-TRNG pass all the standard NIST statistical test suite randomness tests [59]. Second, we observe that SiMRA-

TRNG generates true random numbers at high throughput and low latency. Our results show that across all tested DRAM chips, 2-, 8-, 16-, and 32-row activation-based SiMRA-TRNGs achieve up to 1.15×, 1.99×, 1.82×, and 1.39× (and on average 0.72×, 1.16×, 1.00×, and 0.78×) higher throughput than the state-of-the-art DRAM-based TRNG [53], respectively. Third, the number of activated rows has a significant effect on entropy, and entropy increases with the number of activated rows in most of the tested DRAM chips. We observe that in 64 DRAM chips, *increasing* the number of activated rows from 2 to 32 *increases* the entropy by 2.51× on average. However, in the remaining 32 DRAM chips, we do *not* observe a clear trend (i.e., the entropy can either increase or decrease with the number of activated rows). Fourth, operating parameters and conditions (i.e., data pattern, temperature, and spatial variation) significantly affect the entropy of SiMRA. For example, our results for 32-row activation show that increasing the temperature from 50°C to 90°C decreases the entropy on average across all tested DRAM chips by 1.53×. Our work is openly and freely available at https://github.com/CMU-SAFARI/SiMRA-TRNG.

We make the following key contributions:

- We provide the first experimental characterization of the ability of COTS DRAM chips to generate true random numbers via simultaneous multiple-row activation (SiMRA) for varying numbers of simultaneously activated rows.
- We rigorously characterize 96 COTS DDR4 chips under various parameters and conditionsin terms the entropy provided by SiMRA: the number of simultaneously activated rows, DRAM chip density & die revision, data pattern, temperature, and spatial variation across DRAM subarrays.
- We demonstrate that i) SiMRA-TRNG generates high-quality and high throughput true random numbers; and ii) 2-, 8-, 16-, and 32-row activation-based SiMRA-TRNG improves throughput over the state-of-the-art DRAM-based TRNG [53].

2. Background

We provide a brief background on DRAM, simultaneous multiple-row activation, and true random number generators.

2.1. Dynamic Random Access Memory (DRAM)

DRAM Organization. Fig. 1 depicts the hierarchical organization of DRAM-based main memory. The memory controller in a system is connected to multiple DRAM modules via multiple DRAM channels. A DRAM module has one or more ranks, each of which consists of multiple DRAM chips that operate in lock-step. The memory controller can interface with multiple DRAM ranks by time-multiplexing the channel's I/O bus between the ranks. Each chip has multiple DRAM banks, each consisting of multiple DRAM cell arrays (called subarrays [60–63]). Within a subarray, DRAM cells are organized as a two-dimensional array of DRAM rows and columns. A DRAM cell stores one bit of data in the form of an electrical charge in a capacitor, which can be accessed through an access transistor. A wire called *wordline* drives the gates of all DRAM cells' access transistors in a DRAM row. A wire called *bitline* connects all DRAM cells in a DRAM column to a common differential sense amplifier. Therefore, when a wordline is asserted, each DRAM cell in the DRAM row is connected to its corresponding sense amplifier. The sense amplifier compares the voltage level of the activated cell's bitline and the reference voltage (VDD/2) and amplifies the voltage difference as logic-1 or logic-0.

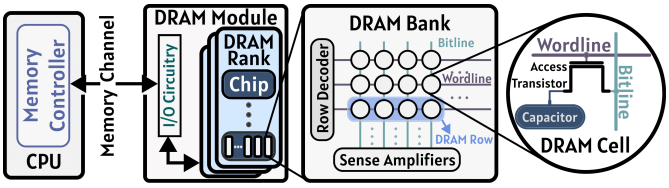

Figure 1: DRAM organization. Adapted from [64].

DRAM Commands. To serve main memory requests and to maintain data integrity, the memory controller issues DRAM commands, e.g., row activation (*ACT*), bank precharge (*PRE*), data read (*RD*), data write (*WR*), and refresh (*REF*). To perform a read or write operation, the memory controller first needs to open a row, i.e., copy the data of the cells in the row to the sense amplifiers. To open a row, the memory controller issues an *ACT* command to a bank by specifying the address of the row to open. After row activation completes, the memory controller issues either a *RD* or a *WR* command to read or write a DRAM word within the activated row. To access data from another DRAM row in the same bank, the memory controller must first close the currently open row by issuing a *PRE* command. The memory controller also periodically issues *REF* commands to prevent data loss due to charge leakage.

DRAM Cell Operation & Timing. We describe DRAM cell operations and DRAM timing by explaining the steps in activating a cell.[1] First, to activate the cell, the memory controller issues an ACT command. The row decoder asserts the wordline and thus connects the cell capacitor to the bitline. Second, the cell capacitor charge begins to perturb the bitline, known as the charge-sharing process. Charge sharing continues until the bitline voltages reach a level that the sense amplifier can safely amplify (e.g., VDD/2+ε or VDD/2-ε). Third, the sense amplifier then kicks in to amplify the difference between the bitline voltage and the reference voltage (VDD/2). Depending on the cell's charge, the bitline becomes either VDD or GND. To return a cell to its precharged state (i.e., to restore the cell's charge), the voltage in the cell must first be fully restored, which requires waiting for t_{RAS} timing parameter (i.e., the timing parameter between ACT and PRE commands). Fourth, once the cell is restored, the memory controller can issue a PRE command. The cell returns to the precharged state, where the memory controller can reliably issue an ACT command after waiting for timing parameter t_{RP} (i.e., the timing parameter between PRE and ACT commands).

[1] We describe DRAM cell operation, but all the described operations happen at row granularity (i.e., to all cells in the activated row).

2.2. Multiple-Row Activation in DRAM

Activating multiple DRAM rows (i.e., multiple-row activation) is a key technique to perform massively parallel in-DRAM computation, as initially shown in [34, 62, 65–70]. Prior works [32–35, 37, 38, 40, 54, 55, 57, 62, 65, 67–73] perform 1) in-DRAM data copy & initialization (as in [62]) and 2) in-DRAM bitwise operations (as in [34, 67]).

In-DRAM Data Copy & Initialization. RowClone [62] enables data movement within DRAM at row granularity without incurring the energy and execution time costs of transferring data between the DRAM and the computing units. An intra-subarray RowClone operation [62] works by issuing two back-to-back *ACT* commands: the first *ACT* copies the contents of the source row *A* into the sense amplifiers. The second *ACT* connects the DRAM cells in the destination row *B* to the bitlines. Because the sense amplifiers have already sensed and amplified the source data by the time row *B* is activated, the data in each cell of row *B* is overwritten by the data stored in the sense amplifiers (i.e., row *A*'s data).

In-DRAM Bitwise Operations. Prior works [34, 67] introduce the concept of simultaneously activating three rows in a DRAM subarray (i.e., triple-row activation) through modifications to the DRAM circuitry. When three rows are concurrently activated, three cells connected to each bitline share charge simultaneously and contribute to the perturbation of the bitline [34, 67]. Upon sensing the perturbation of the three simultaneously activated rows, the sense amplifier amplifies the bitline voltage to V_{DD} or 0 V if at least two of the three DRAM cells are charged or discharged, respectively. As such, simultaneously activating three rows results in a Boolean majority-of-three operation (MAJ3).

SiMRA in COTS DRAM Chips. Many operations envisioned by these works [34, 67] can *already* be performed in *real unmodified* commercial off-the-shelf (COTS) DRAM chips. Recent works experimentally demonstrate that COTS DRAM chips can simultaneously activate multiple DRAM rows (called SiMRA [57]) by issuing an $ACT \rightarrow PRE \rightarrow ACT$ (*APA*) command sequence with violated t_{RAS} and t_{RP} timing constraints [35, 53–57, 71, 72, 74]. By doing so, COTS DRAM chips can perform 1) bitwise operations (i.e., MAJ, AND, OR, NOT, NAND, and NOR) [35, 54–57, 71], 2) data copy & initialization [54, 56, 57, 72], and 3) true random number generation [53, 72].

2.3. True Random Number Generators (TRNGs)

A true random number generator (TRNG) samples random physical processes (e.g., thermal noise) to construct a random bitstream. Harvesting randomness from a physical phenomenon *often* produces bits that are biased or correlated [75, 76]. Post-processing methods can eliminate the bias and correlation and provide protection against environmental changes and adversary tampering [13, 75, 76]. Well-known post-processing techniques are the von Neumann corrector [77] and cryptographic hash functions such as SHA-256 [78].

DRAM-based TRNGs. Prior works propose DRAM-based mechanisms to implement true random number generators (TRNGs) by violating timing parameters [50, 51, 53], using data retention failures [47, 79], and using startup values [48, 49]. The state-of-the-art DRAM-based TRNG [53] generates true random numbers by simultaneously activating four rows (called QUAC).

3. Experimental Methodology

3.1. COTS DRAM Testing Infrastructure

We conduct commercial off-the-shelf (COTS) DRAM chip experiments using DRAM Bender [71, 80] (built upon SoftMC [81, 82]), an FPGA-based DDR4 DRAM testing infrastructure that provides precise control of the DRAM commands issued to a DDR4 DRAM module and DRAM timing parameters. Fig. 2 shows our experimental setup that consists of four main components: a) a thermocouple-based temperature sensor and a pair of heater pads pressed against the DRAM chips that heat up the DRAM chips to a desired temperature, b) an FPGA development board (Xilinx Alveo U200 [83], programmed with DRAM Bender to execute our test programs, c) a host machine that generates the test program and collects experimental results, and d) a PID temperature controller (MaxWell FT200 [84]) that controls the heaters and keeps the temperature at the desired level. Fig. 3 shows our laboratory comprising many DDR4 and HBM2 testing platforms.

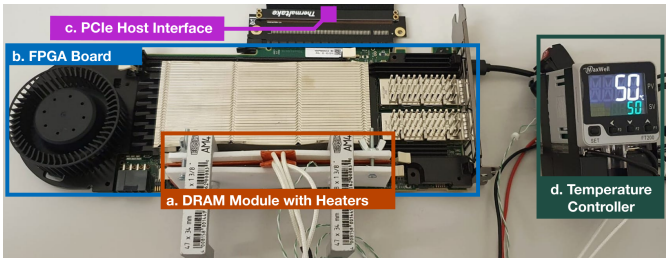

Figure 2: Our DRAM Bender [71] based experimental setup.

Figure 3: Our laboratory for real DRAM chip experiments.

COTS DDR4 DRAM Chips Tested. Table 1 provides the tested 96 (12) COTS DDR4 DRAM chips (modules) that we focus our analysis on along with the chip manufacturer (Chip Mfr.), module manufacturer (Module Mfr.), the number of tested modules (#Modules), the number of tested chips (#Chips), die revision (Die Rev.), chip density, chip organization (Chip Org.), and DRAM module's manufacturing date in the form of year-week (Date).

To investigate whether our characterization study applies to different DRAM technologies, designs, and manufacturing processes, we test a total of 152 (22) COTS DDR4 DRAM chips (modules) from all three major manufacturers (i.e., SK Hynix, Samsung, and Micron) with different die densities and die revisions from each DRAM chip manufacturer. While we observe successful simultaneous multiple-row activation (SiMRA)-based random number generation in *all* tested SK Hynix modules, we observe no successful simultaneous multiple-row activation (and thus random number generation) in Samsung and Micron chips. As described in [57], the reason we do not observe successful SiMRA operation may be that some DRAM chips contain internal circuitry that either ignores the *PRE* command or suppresses the second *ACT* command when the timing parameters (t_{RP} and t_{RAS}) are significantly violated. Thus, we focus our analysis on 96 (12) COTS DDR4 DRAM chips (modules) from SK Hynix, listed in Table 1.

Table 1: Summary of DDR4 DRAM chips tested.

Chip Mfr.	Module Mfr.	#Modules (#Chips)	Die Rev.	Chip Density	Chip Org.	Date[∇] year-week
SK Hynix	TimeTec	4 (32)	A	4Gb	x8	N/A
	TeamGroup	4 (32)	M	4Gb	x8	N/A
	SK Hynix	4 (32)	A	8Gb	x8	18-43

[∇] We report "N/A" if no date is marked on the label of a module.

3.2. Testing Methodology

DRAM Subarray Boundaries. Understanding which rows are simultaneously activated in the same subarray requires reverse engineering DRAM subarray boundaries. We follow the methodology used in prior works to identify subarray boundaries [32, 33, 35, 53, 54, 56, 57, 63, 72, 85]. We leverage the observation that COTS DRAM chips can copy the content of a DRAM row (i.e., source row) to another DRAM row (i.e., destination row) if and only if these rows are in the *same* subarray. We perform the in-DRAM copy operation for every possible source and destination row address in each tested bank and reverse engineer the DRAM subarray boundaries.

Simultaneously Activated Rows with APA. Prior works [35, 53, 57, 74] demonstrate that issuing an $ACT \rightarrow PRE \rightarrow ACT$ (*APA*) sequence allows DRAM chips to activate 2, 4, 8, 16, and 32 DRAM rows simultaneously. To uncover which rows are simultaneously activated by issuing an *APA* sequence, these works [35, 53, 57, 74] show that issuing an *APA* sequence followed by a WRITE (WR) command overwrites the simultaneously activated rows with the data sent with the WR command. We follow the same methodology and reverse engineer the simultaneously activated rows with *APA* sequence for every possible row address in all tested subarrays. We call a group of simultaneously activated rows as a simultaneously activated row (*SAR*) group.

Characterization Methodology. Our characterization experiment consists of three key steps. First, we initialize rows in the tested *SAR* group with a predefined data pattern. Each bit in the data pattern is used to initialize one row in the *SAR* group. Fig. 4 shows two example data layouts for 2-row activation with data patterns `01` (a) and `10` (b). We initialize the first activated row with the left most bit value (e.g., for `01`, all cells in the first row have logic-0) and second row with the right most bit value. Second, we issue the *APA* command sequence with reduced timings to simultaneously activate these rows. Third, we read the data in the tested subarray's sense amplifiers (a total of 65536 sense amplifiers) by respecting the timing parameters. We repeat this experiment for each *SAR* group for 1000 times and record the results in the host machine.

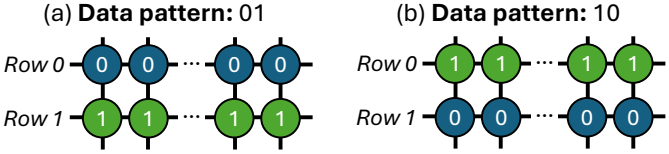

Figure 4: Two example data layouts for 2-row activation with (a) data pattern `01` and (b) data pattern `10`.

Entropy Metric. We measure the randomness in DRAM sense amplifiers using Shannon entropy [58]. Shannon entropy ($H(x)$) of a bitstream generated by sampling a random source, which is sense amplifier (x) in our experiments, is calculated as $H(x) = -\sum_{i=0}^{1} P(x_i)\log_2 P(x_i)$, where $P(x_0)$ and $P(x_1)$ represent the probabilities of observing logic-0 and logic-1, respectively. We measure the Shannon Entropy by repeatedly performing the characterization experiment 1000 times for a tested *SAR* group. We analyze the entropy at the granularity of 512-bit blocks (also known as cache blocks), as the *RD* and *WR* operation granularity of a DRAM module attached to a CPU is at cache block (e.g., 64B) granularity. To do so, we calculate the entropy of each cache block by aggregating the entropy values of the bitstream generated by all 512 sense amplifiers corresponding to that cache block. We define a metric called *average cache block entropy*: the average entropy across all cache blocks in a row of sense amplifiers (i.e., 128 cache blocks) in a *SAR* group. For example, for a *SAR* group with two cache blocks having entropies of 25 and 75 each, the average cache block entropy group is 50.

Number of Tested SAR Groups. We observe that a tested DRAM bank has between 64 and 128 subarrays, depending on the DRAM module. We randomly select three subarrays in one bank per DRAM module. Within each subarray, we randomly test 100 *SAR* groups each for 2-, 4-, 8-, 16-, and 32-row activation.

Data Pattern. We analyze entropy using data patterns with a varying number of logic-1 bits. We sweep all possible numbers of logic-1 bits in the data pattern (i.e., from 0 to the number of rows that are activated). For each number of logic-1s, we first calculate the number of distinct data patterns. For example, 2-row activation has 2 distinct data patterns (i.e., `01` and `10`). If the number of data patterns is smaller than 100, we test all possible data patterns. Otherwise, we generate and test 100 random distinct data patterns to limit the larger parameter space (e.g., 32-row activation has more than 4.2B distinct data patterns).

Temperature. We perform our experiments at five temperature levels: 50°C, 60°C, 70°C, 80°C, and 90°C. All experiments are conducted at 50°C unless stated otherwise.

To aid future research and development, we open-source our infrastructure and scripts at https://github.com/CMU-SAFARI/SiMRA-TRNG.

4. Using Simultaneous Multiple-Row Activation as a Source of Entropy

4.1. Key Mechanism

Simultaneously Activating Up to 32 DRAM Rows. Our key idea relies on a fundamental design characteristic in modern DRAM chips: hierarchical design of the row decoder circuitry. Modern DRAM chips have multiple levels of row address decoding stages to reduce latency, area, and power consumption [53, 57, 86–89]. The hierarchical row decoder structure expands a row address into a set of intermediate control signals and issuing an *ACT* command asserts multiple control signals [53,57,86,87,89]. Prior works [56,57] demonstrate that by changing the row addresses targeted by two *ACT* commands in $ACT \rightarrow PRE \rightarrow ACT$ (*APA*), we can control the number and addresses of the simultaneously activated rows in a subarray. These works [56,57] hypothesize that to simultaneously activate 2^N rows, N decoders in the hierarchical row decoder need to have asserted immediate signals (e.g., asserting intermediate signals in two decoders activates $2^2 = 4$ rows simultaneously). In our tested DRAM chips, the hierarchical row decoder circuitry consists of 5 decoders (the same as prior works [56,57]), and thus, we can simultaneously activate up to $2^5 = 32$ rows. Due to space limitations, we refer the reader to these prior works [56,57] for more details on the hypothetical row decoder circuitry in DRAM chips and how this row decoder design allows simultaneous activation of multiple rows.

Key Idea. Fig. 5 shows the command sequence for true random number generation in COTS DRAM chips using SiMRA and eight DRAM cells connected to a bitline. The memory controller issues each command (shown in orange boxes below the time axis) at the corresponding tick mark, and asserted signals are highlighted in red. Initially, four cells (R0, R2, R4, and R6) have a voltage level of VDD, and the remaining four cells (R1, R3, R5, and R7) ground (GND), and the bitline has a voltage level of VDD/2 (❶). To ease understanding, in Fig. 5, we assume that $ACT \rightarrow PRE \rightarrow ACT$ simultaneously activates eight rows (i.e., all rows from R0 to R7).

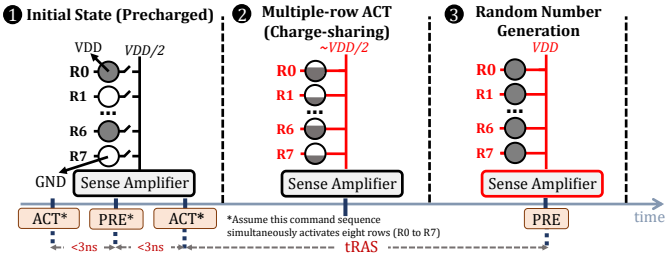

Figure 5: Command sequence for true random number generation in COTS DRAM chips using SiMRA and the state of cells during each related step.

To generate random numbers, we first issue one $ACT \rightarrow PRE \rightarrow ACT$ command sequence (❶). Doing so activates eight cells simultaneously and enables charge-sharing between them and their bitline. Hence, the bitline voltage perturbation is affected by the simultaneously activated eight cells, where four of them try to pull up the bitline voltage and the remaining four try to pull down the bitline voltage. As a result, these opposing contributions of simultaneously activated cells can randomly perturb the bitline from the reference voltage ~VDD/2 (❷). The sense amplifier then kicks in and tries to amplify the voltage on the bitline, which results in sampling a random value based on random perturbations on the bitline voltage [50, 53]. For example, the single depicted bitline is randomly sampled as VDD in this figure (❸). Finally, we send a *PRE* command to complete the process.

4.2. COTS DRAM Chip Characterization Results

Effect of the Number of Simultaneously Activated Rows. Fig. 6 shows the distribution of the average cache block entropy (y-axis) for varying numbers of simultaneously activated rows in a box and whiskers plot.[2] Each subfigure in Fig. 6 represents a different chip density and die revision pair. We make *Observations 1-3* from Fig. 6.

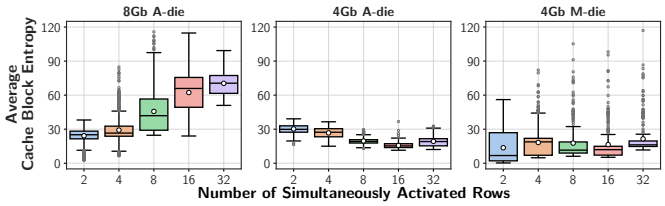

Figure 6: The effect of DRAM architecture (chip density & revision) and the number of simultaneously activated rows on the distribution of average cache block (512-bit) entropy.

Observation 1. SiMRA can generate random values with all tested numbers of simultaneously activated rows.

We observe that across all tested chips, SiMRA can generate 22.87, 25.17, 29.47, 34.54, and 40.41 average cache block entropy with 2-, 4-, 8-, 16-, and 32-row activations, respectively.

Observation 2. Entropy tends to increase with the number of simultaneously activated rows.

We observe that in 8Gb A-die and 4Gb M-die chips, the average cache block entropy increases by 2.51× on average when we increase the number of simultaneously activated rows from 2 to 32. However, in 4Gb A-die chips (middle subplot), there is no strong correlation between the number of activated rows and cache block entropy as the entropy initially decreases until the number of rows reaches 16 and then increases when moving from 16- to 32-row activation.

We hypothesize that this could be due to two reasons. First, as it is impractical to test all possible data patterns for a higher number of row activations (e.g., 16- or 32-row activation), it is possible that we could not find the best data pattern to achieve the highest entropy. However, for a lower number of row activations (e.g., 2-row activation), we explore nearly all data patterns since it is more feasible to explore all data patterns, and thus, it is more likely that we find the best data pattern to

[2] In a box-and-whiskers plot, the box is lower-bounded by the first quartile and upper-bounded by the third quartile. The box size is the distance between the first and third quartiles. Whiskers show the minimum and maximum values.

achieve the highest entropy. Second, many components in the DRAM circuitry could vary for each chip density & die revision pair, resulting in a different interaction between entropy and the number of simultaneously activated rows. For example, the row decoder circuitry may not fully drive the wordlines when more rows are activated, or the sense amplifier's safety margin could be wider in one DRAM architecture compared to others. We call for future circuit-level and device-level research to fundamentally understand and gain more insights into how different DRAM components could affect SiMRA's entropy.

Observation 3. DRAM architecture (especially chip density) significantly affects SiMRA's entropy.

We observe that while 8Gb A-die has an average cache block entropy of 37.62 across all numbers of activated rows, 4Gb A-die and M-die have 25.06 and 16.83, respectively. We hypothesize this could be due to two reasons. First, newer (and denser) chips are more susceptible to read disturbance [64, 74, 85, 90–110], which could increase the noise on bitline voltage and result in increased entropy, compared to chips with lower density. Second, as discussed in *Observation 2*, the significant entropy variation observed across DRAM chip densities could be due to differences in their underlying DRAM microarchitectures (i.e., DRAM chips of different densities could employ different DRAM microarchitectures).

Takeaway 1. *SiMRA's entropy generally increases as the number of simultaneously activated rows increases. Entropy and its interaction with the number of simultaneously activated rows varies across DRAM microarchitectures.*

Data Pattern Dependence. Fig. 7 shows the distribution of average cache block entropy (y-axis) across all tested DRAM chips for different numbers of simultaneously activated rows (subplots) with tested numbers of logic-1s in the data pattern (x-axis). We make *Observation 4* from Fig. 7.

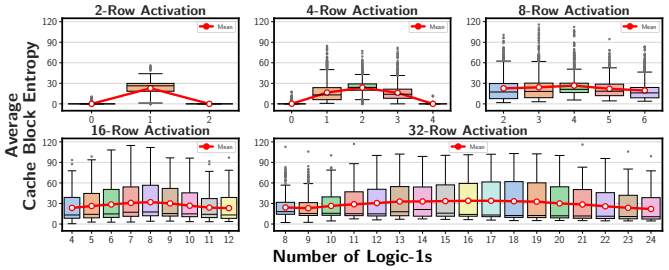

Figure 7: Effect of the number of logic-1s in the data pattern on the distribution of average cache block entropy.

Observation 4. The average cache block entropy significantly varies across tested numbers of logic-1s for every number of row activations.

We observe that 2-, 4-, 8-, and 16-row activations produce the highest average cache block entropy when the data pattern has an equal number of logic-1s and logic-0s, which are 446.19×, 1.44×, 1.10×, and 1.03× higher than the second highest average cache block entropy point, respectively. That is, the closer a data pattern is to the middle on the x-axis, the higher the entropy. For 32-row activation, the highest average cache block entropy is produced at data pattern with 17 logic-1s, 1.01× higher than the data patterns with 16 logic-1s. We hypothesize that data patterns that contain equal numbers of logic-1s and logic-0s (or one more logic-1) result in a small differential voltage. During charge sharing, logic-1 cells attempt to raise the bitline voltage, while logic-0 cells attempt to lower it. Thus, the bitline voltage converges to the reference voltage, and the differential voltage is well below the sense amplifier's reliable sensing margin. Consequently, the sense amplifier fails to reliably sense & amplify, and thus, non-deterministically settles to logic-1 or logic-0.

Takeaway 2. *Data pattern significantly affects SiMRA's average cache block entropy.*

Temperature. In this experiment, we use a single *SAR* group with the configuration (defined by the combination of test parameters used, e.g., data pattern) that achieves the highest cache block entropy for varying numbers of row activations in three tested modules, each from a different chip density & die revision. We select each module that achieves the highest average cache entropy among all chips of the same chip density & die revision and test each *SAR* group 100 times.

Fig. 8 shows the average cache block entropy, with error bars representing the range across all tested DRAM modules for five DRAM chip temperature levels: 50°C, 60°C, 70°C, 80°C, and 90°C. The figure clusters boxes into five groups based on the number of simultaneously activated rows (x-axis). In each cluster of boxes, temperature increases from left to right. We make *Observation 5* from Fig. 8.

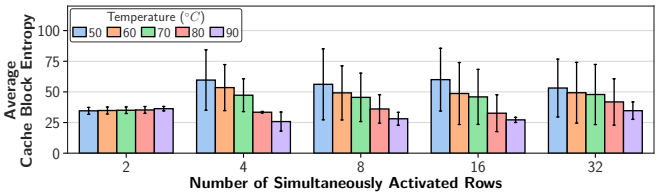

Figure 8: Average cache block entropy at five temperature levels as a function of number of simultaneously activated rows.

Observation 5. Entropy decreases as temperature increases for almost every number of row activations.

For example, for 32-row activation, as the temperature increases from 50°C to 90°C, average cache block entropy decreases by 1.53×. We hypothesize that a high temperature could reduce the threshold voltage of access transistors and, thus, change the behavior of the charge-sharing operation (e.g., by making cells share their charge more and faster) [57, 111, 112]. As a result, the best parameters (e.g., data pattern, timing parameters) to achieve the highest entropy could vary for different temperature levels.

Takeaway 3. *DRAM chip temperature significantly affects SiMRA's average cache block entropy.*

Spatial Variation Across DRAM Subarrays. In this experiment, we categorize each tested subarray into three groups based on its location in a bank: *Beginning* (i.e., the first 1/3 subarrays of a bank), *Middle* (i.e., the second 1/3 subarrays of a

bank), and *End* (i.e., the last 1/3 subarrays of a bank).[3]

Fig. 9 shows the distribution of average cache block entropy (y-axis) across subarrays for varying numbers of simultaneously activated rows (x-axis) based on the subarray's location in a bank (hue). We make *Observations 6 and 7* from Fig. 9.

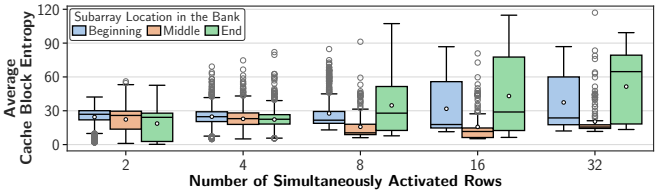

Figure 9: Distribution of average cache block entropy based on the location of a DRAM subarray in a DRAM bank.

Observation 6. The average cache block entropy significantly varies based on the subarray's location in a bank.

We observe that the subarray's location in a bank can lead to variations in the average cache block entropy. The average point of each bar's average cache block entropy (i.e., the average of each bar, white circles in Fig. 9) varies up to 1.31× for 2-row activation, 1.11× for 4-row activation, 2.20× for 8-row activation, 2.74× for 16-row activation, and 2.51× for 32-row activation.

Observation 7. Each number of row activations has a different average cache block entropy variation trend.

We observe that in 2- and 4-row activations, when we get closer to the end of the bank, average cache block entropy consistently decreases. However, this trend is different for 8-, 16-, and 32-row activation: subarrays in the middle of the bank have the lowest average entropy while subarrays at the end of the bank have the highest average entropy.

Takeaway 4. *Average cache block entropy is significantly affected by spatial variation across DRAM subarrays.*

5. SIMRA-TRNG

We demonstrate the potential of using simultaneous multiple-row activation (SiMRA) as the entropy source for a TRNG by developing and evaluating a new TRNG design, SiMRA-TRNG.

SiMRA-TRNG Design. SiMRA-TRNG generates true random numbers using four DRAM banks in five steps: 1) selecting a high-entropy *SAR* group based on our characterization data, 2) initializing DRAM rows by performing the RowClone operation [54, 62, 72] in quick succession in four DRAM banks, 3) performing *APA* on the selected *SAR* to generate random bits in sense amplifiers in four DRAM banks concurrently, 4) reading from four DRAM banks until 256 bits of Shannon Entropy is obtained,[4] and 5) post-processing the extracted bitstream using the SHA-256 cryptographic hash function [78] to eliminate the bias and correlation and provide protection against environmental changes and adversarial tampering [13, 75, 76] (see §2.3).

Quality. We evaluate the quality of SiMRA-TRNG using the NIST Statistical Test Suite (NIST STS) tests [59]. To maintain a reasonable testing time, we test random bitstreams from three DRAM modules, each from a different chip density and die revision pair. For each number of simultaneously activated rows (2-, 4-, 8-, 16-, and 32-row activation), we extract a 1 Gb bitstream from the highest-entropy *SAR* group, partition the bitstream into 1 Mb sequences, and test 1024 sequences per *SAR* group.

Observation 8. Random numbers generated with SiMRA and post-processed with the SHA-256 function pass all *NIST STS tests. We conclude that SiMRA-TRNG reliably produces high-quality true random bitstreams for all tested numbers of simultaneously activated rows.*

Latency. We evaluate the latency of true random number generation by measuring the time required for three key operations: 1) performing N RowClone operations to initialize the N simultaneously activated rows (e.g., for 2-row activation-based SiMRA-TRNG, we perform two RowClone operations) in four DRAM banks, 2) executing the *APA* command sequence to activate a *SAR* group that has the highest entropy in four DRAM banks, and 3) reading random values from the sense amplifiers and post-processing them with SHA-256.

Observation 9. To generate first 256-bits of random numbers, the latencies for SiMRA-TRNG designs based on 2-, 4-, 8-, 16-, and 32-row activation are 164.9*ns,* 254.9*ns,* 434.9*ns,* 794.9*ns, and* 1514.9*ns, respectively.*

Throughput. We analytically model the throughput of SiMRA-TRNG for varying numbers of simultaneously activated rows based on two key factors: 1) N_{block}, the number of 256-bit Shannon entropy blocks available in the highest-entropy *SAR* group,[4] and 2) L_{TRNG}, the latency of true random number generation in four DRAM banks. SiMRA-TRNG's throughput per DRAM module for each number of row activations is equal to $(256 \times N_{block})/L_{TRNG}$ bits per second.

Fig. 10 shows the throughput of SiMRA-TRNG, with error bars representing the range across all tested DRAM modules. Each module's throughput for varying numbers of activated rows is normalized to the state-of-the-art DRAM-based TRNG, QUAC-TRNG (i.e., 4-row activation) [53]. We make two key observations from Fig. 10.

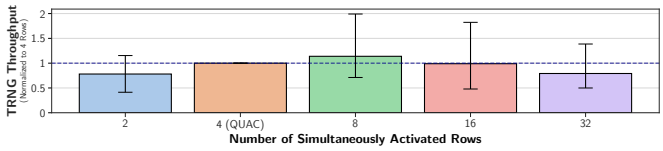

Figure 10: Throughput of generating true random numbers using SiMRA, normalized to state-of-the-art DRAM-based TRNG, QUAC-TRNG (i.e., 4-row activation) [53].

Observation 10. SiMRA-TRNG designs based on 2-, 8-, 16-, and 32-row activation outperform the state-of-the-art DRAM-based TRNG (QUAC-TRNG [53], which uses 4-row activation).

SiMRA with 2-, 8-, 16-, and 32-row activations achieve up

[3] DRAM manufacturers use mapping schemes to translate logical (memory-controller-visible) addresses to physical row addresses [60, 91, 113–126]. To account for in-DRAM row address mapping, we reverse engineer the physical row address layout in all chips, following the prior works' methodology [64, 90, 92, 99].

[4] We observe that 256-bit Shannon entropy is required as an input for the SHA-256 function for generating a 256-bit true random number.

to 1.15×, 1.99×, 1.82×, and 1.39× higher throughput than the state-of-the-art, QUAC-TRNG [53], respectively. We also observe that 2-, 4-, 8-, 16-, and 32-row activation-based SiMRA-TRNG designs achieve average throughputs of 9.90 *Gbps*, 13.81 *Gbps*, 16.05 *Gbps*, 13.94 *Gbps*, and 10.81 *Gbps*, respectively.

Observation 11. There is a tradeoff between TRNG latency and the entropy produced by SiMRA.

While simultaneously activating more rows tends to generate more entropy than activating fewer rows, it also increases the TRNG latency. For example, in all tested DRAM chips, 32-row activation achieves higher average cache block entropy than 16-row activation. However, the higher entropy observed with 32-row activation *does not* lead to a higher throughput as the latency of initializing the simultaneously activated rows is also doubled compared to 16-row activation (i.e., initializing 16 rows vs. 32 rows).

Takeaway 5. *SiMRA-TRNG outperforms the state-of-the-art DRAM-based TRNG in terms of throughput and offers a promising approach to generating true random numbers with high throughput.*

6. Related Work

This is the first work to characterize the use of the simultaneous multiple-row activation for varying numbers of simultaneously activated rows as a TRNG substrate in COTS DRAM chips.

DRAM-based TRNGs [27, 47–53, 79]. These works propose DRAM-based TRNGs using retention failures [47, 79], DRAM start-up values [48, 49], timing failures [50, 51], the unpredictability in DRAM command schedules [52]. The state-of-the-art DRAM-based TRNG [53] generates true random numbers in COTS DDR4 chips by simultaneously activating four rows. Our study generalizes this finding and shows that simultaneously activating 2, 8, 16, and 32 rows generates high-quality true random numbers and can achieve higher throughput than the state-of-the-art. DR-STRaNGe [27] provides end-to-end system design for DRAM-based TRNGs.

SRAM- and FLASH-based TRNGs [127–141]. SRAM-based TRNG designs exploit randomness in startup values [127–135, 137, 138] and randomness caused by reduced supply voltage when reading data [136]. Flash-based TRNGs use random telegraph noise in flash memory devices as an entropy source (up to 1 Mbit/s) [139–141].

Multiple-Row Activation in COTS DRAM Chips [35, 53–57, 63, 71, 72, 74]. Several experimental works on real DRAM chips demonstrate that COTS DRAM chips can activate more than one DRAM row simultaneously or sequentially in quick succession by violating specific DRAM timing parameters. These works show that COTS DRAM chips can 1) perform bulk bitwise operations (e.g., AND and NOT) operations [35, 54–57, 71], 2) copy one row's content into one or more rows [54, 56, 57, 72] and 3) can concurrently refresh two rows in two subarrays [63]. These works [35, 53–57, 63, 71, 72, 74] are inspired by and based on RowClone [62] and Ambit [34, 67]. RowClone [62] and Ambit [34, 67] demonstrated using circuit theory, first principles, and simulation that DRAM chips can perform data copy & data initialization and bulk bitwise operations via multiple-row activation.

Experimental Characterization and Analysis of COTS DRAM Chips. Various prior works [64, 71, 74, 85, 90–110, 117, 119, 121, 125, 142–158] analyze different phenomena in COTS DRAM chips using FPGA-based infrastructures like DRAM Bender [71, 80] & SoftMC [81, 82]. Understanding the interaction of these effects with SiMRA-TRNG or the use of these effects for better TRNG can be avenues for future work.

7. Conclusion

We experimentally demonstrated and analyzed the potential and benefits of using the simultaneous multiple-row activation (SiMRA) phenomenon in COTS DRAM chips to generate true random numbers. Through an extensive characterization using 96 DDR4 chips, we showed that 1) using SiMRA as a TRNG substrate achieves higher throughput than the state-of-the-art DRAM-based TRNG and 2) the number of simultaneously activated rows, data patterns, temperatures, and spatial variations significantly affect the SiMRA's entropy. To facilitate future research in using DRAM for novel purposes and discovering new capabilities of DRAM chips, we freely open source our infrastructure at `https://github.com/CMU-SAFARI/SiMRA-TRNG`.

Acknowledgments

We thank the anonymous reviewers of DSN Disrupt 2024, VTS 2025, IMW 2025, and ICCD 2025 for their feedback. We thank the SAFARI Research Group members for providing a stimulating intellectual and scientific environment. We acknowledge the generous gifts from our industrial partners, including Google, Huawei, Intel, and Microsoft. This work, and our broader work in Processing-in-Memory, is supported in part by the Semiconductor Research Corporation (SRC), the ETH Future Computing Laboratory (EFCL), and the AI Chip Center for Emerging Smart Systems (ACCESS).

References

[1] Lishuang Gong, Jianguo Zhang, Haifang Liu, Luxiao Sang, and Yuncai Wang. True Random Number Generators Using Electrical Noise. *IEEE Access*, 2019.
[2] Min Huang, Anbang Wang, Pu Li, Hang Xu, and Yuncai Wang. Real-Time 3 Gbit/s True Random Bit Generator Based on a Super-Luminescent Diode. *Optics Communications*, 2014.
[3] Suresh Srinivasan, Sanu Mathew, Rajaraman Ramanarayanan, Farhana Sheikh, Mark Anders, Himanshu Kaul, Vasantha Erraguntla, Ram Krishnamurthy, and Greg Taylor. 2.4 GHz 7mW All-Digital PVT-Variation Tolerant True Random Number Generator in 45nm CMOS. In *VLSI*, 2010.
[4] Darrell Whitley. A Genetic Algorithm Tutorial. *Statistics and Computing*, 1998.
[5] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *JMLR*, 2010.
[6] W. F. Schmidt, M. A. Kraaijveld, and R. P. W. Duin. Feedforward Neural Networks with Random Weights. In *ICPR*, 1992.
[7] Le Zhang and P.N. Suganthan. A Survey of Randomized Algorithms for Training Neural Networks. *Information Sciences*, 2016.
[8] Y. Miché, B. Schrauwen, and A. Lendasse. Machine Learning Techniques based on Random Projections. In *ESANN*, 2010.
[9] Vittorio Bagini and Marco Bucci. A Design of Reliable True Random Number Generator for Cryptographic Applications. In *CHES*, 1999.
[10] Mohammed Bakiri, Christophe Guyeux, Jean-François Couchot, and Abdelkrim Kamel Oudjida. Survey on Hardware Implementation of Random Number Generators on FPGA: Theory and Experimental Analyses. *CSR*, 2018.
[11] Andrea Röck. Pseudorandom Number Generators for Cryptographic Applications. Master's thesis, Paris-Lodron-Universität Salzburg, 2005.

[12] Xiongfeng Ma, Xiao Yuan, Zhu Cao, Bing Qi, and Zhen Zhang. Quantum Random Number Generation. *Quantum Inf.*, 2016.

[13] Mario Stipčević and Çetin Kaya Koç. True Random Number Generators. In *Open Problems in Mathematics and Computational Science*. 2014.

[14] Mahmood Barangi, Joseph S Chang, and Pinaki Mazumder. Straintronics-Based True Random Number Generator for High-Speed and Energy-Limited Applications. In *IEEE Trans. Magn*, 2016.

[15] Sha Tao and Elena Dubrova. TVL-TRNG: Sub-Microwatt True Random Number Generator Exploiting Metastability in Ternary Valued Latches. In *ISMVL*, 2017.

[16] Zvi Gutterman, Benny Pinkas, and Tzachy Reinman. Analysis of the Linux Random Number Generator. In *SP*, 2006.

[17] Vincent von Kaenel and Toshinari Takayanagi. Dual True Random Number Generators for Cryptographic Applications Embedded on a 200 Million Device Dual CPU SOC. In *CICC*, 2007.

[18] J. Kim, H. Nili, N. D. Truong, T. Ahmed, J. Yang, D. S. Jeong, S. Sriram, D. C. Ranasinghe, S. Ippolito, H. Chun, and O. Kavehei. Nano-Intrinsic True Random Number Generation: A Device to Data Study. *TCAS*, 2019.

[19] Milos Drutarovsky and Pavol Galajda. A Robust Chaos-based True Random Number Generator Embedded in Reconfigurable Switched-Capacitor Hardware. In *Radioelektronika*, 2007.

[20] Sammy HM Kwok and Edmund Y Lam. FPGA-based High-speed True Random Number Generator for Cryptographic Applications. In *TENCON*, 2006.

[21] Abdelkarim Cherkaoui, Viktor Fischer, Laurent Fesquet, and Alain Aubert. A Very High Speed True Random Number Generator with Entropy Assessment. In *CHES*, 2013.

[22] Teng Zhang, Minghui Yin, Changmin Xu, Xiayan Lu, Xinhao Sun, Yuchao Yang, and Ru Huang. High-speed True Random Number Generation Based on Paired Memristors for Security Electronics. *Nanotechnology*, 2017.

[23] Quintessence Labs. Random Number Generators White Paper, 2015.

[24] Patrick J Clarke, Robert J Collins, Philip A Hiskett, Paul D Townsend, and Gerald S Buller. Robust Gigahertz Fiber Quantum Key Distribution. *Applied Physics Letters*, 2011.

[25] Yonggang Wang, Cong Hui, Chong Liu, and Chao Xu. Theory and Implementation of A Very High Throughput True Random Number Generator in Field Programmable Gate Array. *Review of Scientific Instruments*, 2016.

[26] XiaoMing Lu, LiJun Zhang, YongGang Wang, Wei Chen, DaJun Huang, Deng Li, Shuang Wang, DeYong He, ZhenQiang Yin, Yu Zhou, et al. FPGA Based Digital Phase-Coding Quantum Key Distribution System. *Science China Physics, Mechanics & Astronomy*, 2015.

[27] F. Nisa Bostanci, Ataberk Olgun, Lois Orosa, A. Giray Yaglikci, Jeremie S. Kim, Hasan Hassan, Oguz Ergin, and Onur Mutlu. DR-STRaNGe: End-to-End System Design for DRAM-based True Random Number Generators. In *HPCA*, 2022.

[28] Robert H. Dennard. Field-Effect Transistor Memory, 1968. U.S. Patent 3,387,286.

[29] Saugata Ghose, Amirali Boroumand, Jeremie S Kim, Juan Gómez-Luna, and Onur Mutlu. Processing-in-Memory: A Workload-Driven Perspective. *IBM JRD*, 2019.

[30] Onur Mutlu, Saugata Ghose, Juan Gómez-Luna, and Rachata Ausavarungnirun. A Modern Primer on Processing in Memory. In *Emerging Computing: From Devices to Systems — Looking Beyond Moore and Von Neumann*. Springer, 2022.

[31] Onur Mutlu, Saugata Ghose, Juan Gómez-Luna, and Rachata Ausavarungnirun. Processing Data Where It Makes Sense: Enabling In-Memory Computation. *MicPro*, 2019.

[32] Onur Mutlu, Ataberk Olgun, Geraldo F Oliveira, and Ismail E Yuksel. Memory-Centric Computing: Recent Advances in Processing-in-DRAM. In *IEDM*, 2024.

[33] Onur Mutlu, Ataberk Olgun, and Ismail Emir Yuksel. Memory-Centric Computing: Solving Computing's Memory Problem. In *IMW*, 2025.

[34] Vivek Seshadri, Donghyuk Lee, Thomas Mullins, Hasan Hassan, Amirali Boroumand, Jeremie Kim, Michael A Kozuch, Onur Mutlu, Phillip B Gibbons, and Todd C Mowry. Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology. In *MICRO*, 2017.

[35] Ismail Emir Yuksel, Yahya Can Tugrul, Ataberk Olgun, F. Nisa Bostanci, A. Giray Yaglikci, Geraldo F. de Oliveira, Haocong Luo, Juan Gomez Luna, Mohammad Sadrosadati, and Onur Mutlu. Functionally-Complete Boolean Logic in Real DRAM Chips: Experimental Characterization and Analysis. In *HPCA*, 2024.

[36] Juan Gómez-Luna, Yuxin Guo, Sylvan Brocard, Julien Legriel, Remy Cimadomo, Geraldo F Oliveira, Gagandeep Singh, and Onur Mutlu. Evaluating Machine LearningWorkloads on Memory-Centric Computing Systems. In *ISPASS*, 2023.

[37] Nastaran Hajinazar, Geraldo F Oliveira, Sven Gregorio, João Dinis Ferreira, Nika Mansouri Ghiasi, Minesh Patel, Mohammed Alser, Saugata Ghose, Juan Gómez-Luna, and Onur Mutlu. SIMDRAM: A Framework for Bit-Serial SIMD Processing Using DRAM. In *ASPLOS*, 2021.

[38] Geraldo F Oliveira, Ataberk Olgun, A. Giray Gregorio Yaglikçi, Nisa Bostanci, Juan Gómez-Luna, Saugata Ghose, and Onur Mutlu. MIMDRAM: An End-to-End Processing-Using-DRAM System for High-Throughput, Energy-Efficient and Programmer-Transparent Multiple-Instruction Multiple-Data Processing. In *HPCA*, 2024.

[39] Shuangchen Li, Dimin Niu, Krishna T Malladi, Hongzhong Zheng, Bob Brennan, and Yuan Xie. DRISA: A DRAM-Based Reconfigurable In-Situ Accelerator. In *MICRO*, 2017.

[40] Geraldo F Oliveira, Mayank Kabra, Yuxin Guo, Kangqi Chen, A Giray Yağlıkçı, Melina Soysal, Mohammad Sadrosadati, Joaquin Olivares Bueno, Saugata Ghose, Juan Gómez-Luna, et al. Proteus: Achieving High-Performance Processing-Using-DRAM via Dynamic Precision Bit-Serial Arithmetic. In *ICS*, 2025.

[41] Juan Gómez-Luna, Izzat El Hajj, Ivan Fernandez, Christina Giannoula, Geraldo F Oliveira, and Onur Mutlu. Benchmarking a New Paradigm: Experimental Analysis and Characterization of a Real Processing-in-Memory System. *IEEE Access*, 2022.

[42] Fabrice Devaux. The True Processing in Memory Accelerator. In *Hot Chips*, 2019.

[43] S. Lee, K. Kim, S. Oh, J. Park, G. Hong, D. Ka, K. Hwang, J. Park, K. Kang, J. Kim, J. Jeon, N. Kim, Y. Kwon, K. Vladimir, W. Shin, J. Won, M. Lee, H. Joo, et al. A 1ynm 1.25V 8Gb, 16Gb/s/pin GDDR6-Based Accelerator-in-Memory Supporting 1TFLOPS MAC Operation and Various Activation Functions for Deep-Learning Applications. In *ISSCC*, 2022.

[44] Young-Cheon Kwon, Suk Han Lee, Jaehoon Lee, Sang-Hyuk Kwon, Je Min Ryu, Jong-Pil Son, O Seongil, Hak-Soo Yu, Haesuk Lee, Soo Young Kim, et al. A 20nm 6GB Function-in-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications. In *ISSCC*, 2021.

[45] Dimin Niu, Shuangchen Li, Yuhao Wang, Wei Han, Zhe Zhang, Yijin Guan, Tianchan Guan, Fei Sun, Fei Xue, Lide Duan, et al. 184QPS/W 64Mb/mm^2 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System. In *ISSCC*, 2022.

[46] Liu Ke, Xuan Zhang, Jinin So, Jong-Geon Lee, Shin-Haeng Kang, Sukhan Lee, Songyi Han, Yeongon Cho, Jin Hyun Kim, Yongsuk Kwon, et al. Near-Memory Processing in Action: Accelerating Personalized Recommendation with AxDIMM. *IEEE Micro*, 2021.

[47] Christoph Keller, Frank Gurkaynak, Hubert Kaeslin, and Norbert Felber. Dynamic Memory-based Physically Unclonable Function for the Generation of Unique Identifiers and True Random Numbers. In *ISCAS*, 2014.

[48] Fatemeh Tehranipoor, Wei Yan, and John A Chandy. Robust Hardware True Random Number Generators using DRAM Remanence Effects. In *HOST*, 2016.

[49] Charles Eckert, Fatemeh Tehranipoor, and John A Chandy. DRNG: DRAM-based Random Number Generation Using its Startup Value Behavior. In *MWSCAS*, 2017.

[50] Jeremie S Kim, Minesh Patel, Hasan Hassan, Lois Orosa, and Onur Mutlu. D-RaNGe: Using Commodity DRAM Devices to Generate True Random Numbers with Low Latency and High Throughput. In *HPCA*, 2019.

[51] B. M. S. Bahar Talukder, J. Kerns, B. Ray, T. Morris, and M. T. Rahman. Exploiting DRAM Latency Variations for Generating True Random Numbers. In *ICCE*, 2019.

[52] Changwoo Pyo, Sungil Pae, and Gyungho Lee. DRAM as Source of Randomness. In *IET*, 2009.

[53] Ataberk Olgun, Minesh Patel, A. Giray Yağlıkçı, Haocong Luo, Jeremie S. Kim, Nisa Bostanci, Nandita Vijaykumar, Oguz Ergin, and Onur Mutlu. QUAC-TRNG: High-Throughput True Random Number Generation Using Quadruple Row Activation in Commodity DRAM Chips. In *ISCA*, 2021.

[54] Fei Gao, Georgios Tziantzioulis, and David Wentzlaff. ComputeDRAM: In-Memory Compute Using Off-the-Shelf DRAMs. In *MICRO*, 2019.

[55] Fei Gao, Georgios Tziantzioulis, and David Wentzlaff. FracDRAM: Fractional Values in Off-the-Shelf DRAM. In *MICRO*, 2022.

[56] Ismail Emir Yuksel, Yahya Can Tugrul, F Nisa Bostanci, A Giray Yaglikci, Ataberk Olgun, Geraldo F Oliveira, Melina Soysal, Haocong Luo, Juan Gomez Luna, Mohammad Sadrosadati, and Onur Mutlu. PULSAR: Simultaneous Many-Row Activation for Reliable and High-Performance Computing in Off-the-Shelf DRAM Chips. arXiv, 2023.

[57] Ismail Emir Yuksel, Yahya Can Tugrul, F. Nisa Bostanci, Geraldo F. de Oliveira, A. Giray Yaglikci, Ataberk Olgun, Melina Soysal, Haocong Luo, Juan Gomez Luna, Mohammad Sadrosadati, and Onur Mutlu. Simultaneous Many-Row Activation in Off-the-Shelf DRAM Chips: Experimental Characterization and Analysis. In *DSN*, 2024.

[58] Claude Elwood Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 1948.

[59] Lawrence Bassham, Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Stefan Leigh, M Levenson, M Vangel, Nathanael Heckert, and D Banks. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Special Publication (NIST SP), 2010.

[60] Yoongu Kim, Vivek Seshadri, Donghyuk Lee, Jamie Liu, and Onur Mutlu. A Case for Exploiting Subarray-Level Parallelism (SALP) in DRAM. In *ISCA*, 2012.

[61] Kevin K Chang, Donghyuk Lee, Zeshan Chishti, Alaa R Alameldeen, Chris Wilkerson, Yoongu Kim, and Onur Mutlu. Improving DRAM Performance by Parallelizing Refreshes with Accesses. In *HPCA*, 2014.

[62] Vivek Seshadri, Yoongu Kim, Chris Fallin, Donghyuk Lee, Rachata Ausavarungnirun, Gennady Pekhimenko, Yixin Luo, Onur Mutlu, Phillip B Gibbons, Michael A Kozuch, and Todd Mowry. RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization. In *MICRO*, 2013.

[63] A Giray Yağlikci, Ataberk Olgun, Minesh Patel, Haocong Luo, Hasan Hassan, Lois Orosa, Oğuz Ergin, and Onur Mutlu. HiRA: Hidden Row Activation for Reducing Refresh Latency of Off-the-Shelf DRAM Chips. In *MICRO*, 2022.

[64] Lois Orosa, A Giray Yağlıkçı, Haocong Luo, Ataberk Olgun, Jisung Park, Hasan Hassan, Minesh Patel, Jeremie S. Kim, and Onur Mutlu. A Deeper Look into RowHammer's Sensitivities: Experimental Analysis of Real DRAM Chips and Implications on Future Attacks and Defenses. In *MICRO*, 2021.

[65] Vivek Seshadri and Onur Mutlu. In-DRAM Bulk Bitwise Execution Engine. *arXiv*, 2019.

[66] Vivek Seshadri and Onur Mutlu. Simple Operations in Memory to Reduce Data Movement. In *Advances in Computers*. 2017.

[67] Vivek Seshadri, Kevin Hsieh, Amirali Boroumand, Donghyuk Lee, Michael A. Kozuch, Onur Mutlu, Phillip B. Gibbons, and Todd C. Mowry. Fast Bulk Bitwise AND and OR in DRAM. In *CAL*, 2015.

[68] Vivek Seshadri, Donghyuk Lee, Thomas Mullins, Hasan Hassan, Amirali Boroumand, Jeremie Kim, Michael A Kozuch, Onur Mutlu, Phillip B Gibbons,

and Todd C Mowry. Buddy-RAM: Improving the Performance and Efficiency of Bulk Bitwise Operations Using DRAM. arXiv, 2016.
[69] Vivek Seshadri and Onur Mutlu. The Processing Using Memory Paradigm: In-DRAM Bulk Copy, Initialization, Bitwise AND and OR. arXiv, 2016.
[70] Vivek Seshadri, Yoongu Kim, Chris Fallin, Donghyuk Lee, Rachata Ausavarungnirun, Gennady Pekhimenko, Yixin Luo, Onur Mutlu, Phillip B. Gibbons, Michael A. Kozuch, and Todd C. Mowry. RowClone: Accelerating Data Movement and Initialization Using DRAM. arXiv, 2018.
[71] Ataberk Olgun, Hasan Hassan, A Giray Yağlıkçı, Yahya Can Tuğrul, Lois Orosa, Haocong Luo, Minesh Patel, Oğuz Ergin, and Onur Mutlu. DRAM Bender: An Extensible and Versatile FPGA-based Infrastructure to Easily Test State-of-the-art DRAM Chips. *TCAD*, 2023.
[72] Ataberk Olgun, Juan Gómez Luna, Konstantinos Kanellopoulos, Behzad Salami, Hasan Hassan, Oguz Ergin, and Onur Mutlu. PiDRAM: A Holistic End-to-end FPGA-based Framework for Processing-in-DRAM. *TACO*, 2022.
[73] Vivek Seshadri and Onur Mutlu. Simple Operations in Memory to Reduce Data Movement. In *Advances in Computers*. Elsevier, 2017.
[74] Ismail Emir Yuksel, Akash Sood, Ataberk Olgun, Oğuzhan Canpolat, Haocong Luo, Nisa Bostanci, Mohammad Sadrosadati, Giray Yaglikci, and Onur Mutlu. PuDHammer: Experimental Analysis of Read Disturbance Effects of Processing-using-DRAM in Real DRAM Chips. In *ISCA*, 2025.
[75] Çetin Kaya Koç. About Cryptographic Engineering. In *Cryptographic Engineering*. 2009.
[76] Md Tauhidur Rahman, Kan Xiao, Domenic Forte, Xuhei Zhang, Jerry Shi, and Mohammad Tehranipoor. TI-TRNG: Technology Independent True Random Number Generator. In *DAC*, 2014.
[77] John Von Neumann. Various Techniques used in Connection with Random Digits. *NBS Applied Math Series*, 1951.
[78] FIPS, PUB. 180-2: Secure hash standard (SHS). *US Department of Commerce, National Institute of Standards and Technology (NIST)*, 2012.
[79] Soubhagya Sutar, Arnab Raha, Devadatta Kulkarni, Rajeev Shorey, Jeffrey Tew, and Vijay Raghunathan. D-PUF: An Intrinsically Reconfigurable DRAM PUF for Device Authentication and Random Number Generation. In *TECS*, 2018.
[80] SAFARI Research Group. DRAM Bender — GitHub Repository. https://github.com/CMU-SAFARI/DRAM-Bender, 2022.
[81] Hasan Hassan, Nandita Vijaykumar, Samira Khan, Saugata Ghose, Kevin Chang, Gennady Pekhimenko, Donghyuk Lee, Oguz Ergin, and Onur Mutlu. SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies. In *HPCA*, 2017.
[82] SAFARI Research Group. SoftMC — GitHub Repository. https://github.com/CMU-SAFARI/softmc, 2017.
[83] Xilinx Inc. Xilinx Alveo U200 FPGA Board. https://www.xilinx.com/products/boards-and-kits/alveo/u200.html.
[84] Maxwell. FT20X User Manual. https://www.maxwell-fa.com/upload/files/base/8/m/311.pdf.
[85] A. Giray Yağlıkçı, Geraldo Francisco Oliveira, Yahya Can Tuğrul, Ismail Emir Yuksel, Ataberk Olgun, Haocong Luo, and Onur Mutlu. Spatial Variation-Aware Read Disturbance Defenses: Experimental Analysis of Real DRAM Chips and Implications on Future Solutions. In *HPCA*, 2024.
[86] Fujun Bai, Song Wang, Xuerong Jia, Yixin Guo, Bing Yu, Hang Wang, Cong Lai, Qiwei Ren, and Hongbin Sun. A Low-Cost Reduced-Latency DRAM Architecture With Dynamic Reconfiguration of Row Decoder. *TVLSI*, 2022.
[87] Neil HE Weste and David Harris. *CMOS VLSI design: a circuits and systems perspective*. Pearson Education India, 2015.
[88] Michael A Turi and José G Delgado-Frias. High-performance low-power selective precharge schemes for address decoders. *TCAS-II*, 2008.
[89] Brent Keeth et al. *DRAM Circuit Design. Fundamental and High-Speed Topics*. Wiley-IEEE Press, 2007.
[90] Jeremie S. Kim, Minesh Patel, Abdullah Giray Yağlıkçı, Hasan Hassan, Roknoddin Azizi, Lois Orosa, and Onur Mutlu. Revisiting RowHammer: An Experimental Analysis of Modern Devices and Mitigation Techniques. In *ISCA*, 2020.
[91] Y. Kim, R. Daly, J. Kim, C. Fallin, J. H. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu. Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors. In *ISCA*, 2014.
[92] A. Giray Yağlıkcı, Haocong Luo, Geraldo F De Oliviera, Ataberk Olgun, Minesh Patel, Jisung Park, Hasan Hassan, Jeremie S Kim, Lois Orosa, and Onur Mutlu. Understanding RowHammer Under Reduced Wordline Voltage: An Experimental Study Using Real DRAM Devices. In *DSN*, 2022.
[93] Chulseung Lim, Kyungbae Park, and Sanghyeon Baeg. Active Precharge Hammering to Monitor Displacement Damage Using High-Energy Protons in 3x-nm SDRAM. *TNS*, 2017.
[94] Kyungbae Park, Donghyuk Yun, and Sanghyeon Baeg. Statistical Distributions of Row-Hammering Induced Failures in DDR3 Components. *Microelectronics Reliability*, 2016.
[95] Kyungbae Park, Chulseung Lim, Donghyuk Yun, and Sanghyeon Baeg. Experiments and Root Cause Analysis for Active-Precharge Hammering Fault in DDR3 SDRAM under 3xnm Technology. *Microelectronics Reliability*, 2016.
[96] Seong-Wan Ryu, Kyungkyu Min, Jungho Shin, Heimi Kwon, Donghoon Nam, Taekyung Oh, Tae-Su Jang, Minsoo Yoo, Yongtaik Kim, and Sungjoo Hong. Overcoming the Reliability Limitation in the Ultimately Scaled DRAM using Silicon Migration Technique by Hydrogen Annealing. In *IEDM*, 2017.
[97] Donghyuk Yun, Myungsang Park, Chulseung Lim, and Sanghyeon Baeg. Study of TID Effects on One Row Hammering using Gamma in DDR4 SDRAMs. In *IRPS*, 2018.
[98] Chulseung Lim, Kyungbae Park, Geunyong Bak, Donghyuk Yun, Myungsang Park, Sanghyeon Baeg, Shi-Jie Wen, and Richard Wong. Study of Proton Radiation Effect to Row Hammer Fault in DDR4 SDRAMs. *Microelectronics Reliability*, 2018.
[99] Haocong Luo, Ataberk Olgun, Abdullah Giray Yağlıkçı, Yahya Can Tuğrul, Steve Rhyner, Meryem Banu Cavlak, Joël Lindegger, Mohammad Sadrosadati, and Onur Mutlu. RowPress: Amplifying Read Disturbance in Modern DRAM Chips. In *ISCA*, 2023.
[100] Zhenrong Lang, Patrick Jattke, Michele Marazzi, and Kaveh Razavi. Blaster: Characterizing the blast radius of rowhammer. In *DRAMSec*, 2023.
[101] Hwayong Nam, Seungmin Baek, Minbok Wi, Michael Jaemin Kim, Jaehyun Park, Chihun Song, Nam Sung Kim, and Jung Ho Ahn. Dramscope: Uncovering DRAM Microarchitecture and Characteristics by Issuing Memory Commands. *ISCA*, 2024.
[102] Ataberk Olgun, Majd Osseiran, Abdullah Giray Yaglikci, Yahya Can Tugrul, Haocong Luo, Steve Rhyner, Behzad Salami, Juan Gomez Luna, and Onur Mutlu. An Experimental Analysis of RowHammer in HBM2 DRAM Chips. In *DSN Disrupt*, 2023.
[103] Hwayong Nam, Seungmin Baek, Minbok Wi, Michael Jaemin Kim, Jaehyun Park, Chihun Song, Nam Sung Kim, and Jung Ho Ahn. X-ray: Discovering DRAM Internal Structure and Error Characteristics by Issuing Memory Commands. *IEEE CAL*, 2023.
[104] Yahya Can Tugrul, A. Giray Yaglikci, Ismail Emir Yuksel, Ataberk Olgun, Oguzhan Canpolat, Nisa Bostanci, Mohammad Sadrosadati, Oguz Ergin, and Onur Mutlu. Understanding RowHammer Under Reduced Refresh Latency: Experimental Analysis of Real DRAM Chips and Implications on Future Solutions. In *HPCA*, 2025.
[105] Wei He, Zhi Zhang, Yueqiang Cheng, Wenhao Wang, Wei Song, Yansong Gao, Qifei Zhang, Kang Li, Dongxi Liu, and Surya Nepal. WhistleBlower: A System-level Empirical Study on RowHammer. *TC*, 2023.
[106] Haocong Luo, Ismail Emir Yüksel, Ataberk Olgun, A Giray Yağlıkçı, Mohammad Sadrosadati, and Onur Mutlu. An Experimental Characterization of Combined RowHammer and RowPress Read Disturbance in Modern DRAM Chips. In *DSN Disrupt*, 2024.
[107] Ataberk Olgun, F. Nisa Bostanci, Ismail Emir Yuksel, Oguzhan Canpolat, Haocong Luo, Geraldo F. Oliveira, A. Giray Yaglikci, Minesh Patel, and Onur Mutlu. Variable Read Disturbance: An Experimental Analysis of Temporal Variation in DRAM Read Disturbance. In *HPCA*, 2025.
[108] Abdullah Giray Yaglikci. *Enabling Efficient and Scalable DRAM Read Disturbance Mitigation via New Experimental Insights into Modern DRAM Chips*. PhD thesis, ETH Zürich, 2024.
[109] Haocong Luo, İsmail Emir Yüksel, Ataberk Olgun, A Giray Yağlıkçı, and Onur Mutlu. Revisiting DRAM Read Disturbance: Identifying Inconsistencies Between Experimental Characterization and Device-Level Studies. In *VTS*, 2025.
[110] Ismail Emir Yuksel, Ataberk Olgun, Nisa Bostanci, Haocong Luo, Giray Yaglikci, and Onur Mutlu. ColumnDisturb: Understanding Column-based Read Disturbance in Real DRAM Chips and Implications for Future Systems. In *MICRO*, 2025.
[111] Takeshi Sakata, Kiyoo Itoh, Masashi Horiguchi, and Masakazu Aoki. Subthreshold-Current Reduction Circuits for Multi-Gigabit DRAM's. *JSSC*, 1994.
[112] James Kao, Siva Narendra, and Anantha Chandrakasan. Subthreshold Leakage Modeling and Reduction Techniques. In *ICCAD*, 2002.
[113] Robert T Smith, James D Chlipala, JOHN FM Bindels, Roy G Nelson, Frederick H Fischer, and Thomas F Mantz. Laser Programmable Redundancy and Yield Improvement in a 64K DRAM. *JSSC*, 1981.
[114] Masashi Horiguchi. Redundancy Techniques for High-Density DRAMs. In *ISIS*, 1997.
[115] B. Keeth and R.J. Baker. *DRAM Circuit Design: A Tutorial*. Wiley, 2001.
[116] Kiyoo Itoh. *VLSI Memory Chip Design*. Springer, 2001.
[117] Jamie Liu, Ben Jaiyen, Yoongu Kim, Chris Wilkerson, Onur Mutlu, J Liu, B Jaiyen, Y Kim, C Wilkerson, and O Mutlu. An Experimental Study of Data Retention Behavior in Modern DRAM Devices. In *ISCA*, 2013.
[118] Vivek Seshadri, Thomas Mullins, Amirali Boroumand, Onur Mutlu, Phillip B Gibbons, Michael A Kozuch, and Todd C Mowry. Gather-Scatter DRAM: In-DRAM Address Translation to Improve the Spatial Locality of Non-Unit Strided Accesses. In *MICRO*, 2015.
[119] Samira Khan, Donghyuk Lee, and Onur Mutlu. PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM. In *DSN*, 2016.
[120] Samira Khan, Chris Wilkerson, Zhe Wang, Alaa R Alameldeen, Donghyuk Lee, and Onur Mutlu. Detecting and Mitigating Data-Dependent DRAM Failures by Exploiting Current Memory Content. In *MICRO*, 2017.
[121] Donghyuk Lee, Samira Khan, Lavanya Subramanian, Saugata Ghose, Rachata Ausavarungnirun, Gennady Pekhimenko, Vivek Seshadri, and Onur Mutlu. Design-Induced Latency Variation in Modern DRAM Chips: Characterization, Analysis, and Latency Reduction Mechanisms. In *SIGMETRICS*, 2017.
[122] Andrei Tatar, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi. Defeating Software Mitigations Against Rowhammer: A Surgical Precision Hammer. In *RAID*, 2018.
[123] Alessandro Barenghi, Luca Breveglieri, Niccolò Izzo, and Gerardo Pelosi. Software-Only Reverse Engineering of Physical DRAM Mappings for Rowhammer Attacks. In *IVSW*, 2018.
[124] Lucian Cojocar, Jeremie Kim, Minesh Patel, Lillian Tsai, Stefan Saroiu, Alec Wolman, and Onur Mutlu. Are We Susceptible to Rowhammer? An End-to-End Methodology for Cloud Providers. In *S&P*, 2020.
[125] Minesh Patel, Jeremie Kim, Taha Shahroodi, Hasan Hassan, and Onur Mutlu. Bit-Exact ECC Recovery (BEER): Determining DRAM On-Die ECC Functions by Exploiting DRAM Data Retention Characteristics. In *MICRO*, 2020.

[126] Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi. TRRespass: Exploiting the Many Sides of Target Row Refresh. In *S&P*, 2020.

[127] Vincent van der Leest, Erik van der Sluis, Geert-Jan Schrijen, Pim Tuyls, and Helena Handschuh. *Efficient Implementation of True Random Number Generator Based on SRAM PUFs*. Springer-Verlag, 2012.

[128] Xu Zhang, Chunsheng Jiang, Gang Dai, Le Zhong, Wen Fang, Ke Gu, Guoping Xiao, Shangqing Ren, Xin Liu, and Sanyong Zou. Improved Performance of SRAM-Based True Random Number Generator by Leveraging Irradiation Exposure. *Sensors*, 2020.

[129] Saman Kiamehr, Mohammad Saber Golanbari, and Mehdi B Tahoori. Leveraging Aging Effect to Improve SRAM-based True Random Number Generators. In *DATE*, 2017.

[130] M Tauhidur Rahman, Domenic Forte, Xiaoxiao Wang, and Mark Tehranipoor. Enhancing Noise Sensitivity of Embedded SRAMs for Robust True Random Number Generation in SoCs. In *AsianHOST*, 2016.

[131] Arindam Sadhu, Kunal Das, Debashis De, and Maitreyi Ray Kanjilal. SSTRNG: Self Starved Feedback SRAM based True Random Number Generator using Quantum Cellular Automata. *Microsystem Technologies*, 2020.

[132] Wendong Wang, Ujjwal Guin, and Adit Singh. Aging-Resilient SRAM-based True Random Number Generator for Lightweight Devices. *Electronic Testing*, 2020.

[133] Po-Shao Yeh, Chih-An Yang, Yi-Hong Chang, Yue-Der Chih, Chrong-Jung Lin, and Ya-Chin King. Self-Convergent Trimming SRAM True Random Number Generation with in-Cell Storage. *JSSC*, 2019.

[134] Daniel E Holcomb, Wayne P Burleson, and Kevin Fu. Initial SRAM State as a Fingerprint and Source of True Random Numbers for RFID Tags. In *RFID*, 2007.

[135] Lawrence T Clark, Sai Bharadwaj Medapuram, and Divya Kiran Kadiyala. SRAM Circuits for True Random Number Generation using Intrinsic Bit Instability. *VLSI*, 2018.

[136] İsmail Emir Yüksel, Ataberk Olgun, Behzad Salami, F Bostancı, Yahya Can Tuğrul, A Giray Yağlıkçı, Nika Mansouri Ghiasi, Onur Mutlu, and Oğuz Ergin. TuRaN: True Random Number Generation using Supply Voltage Underscaling in SRAMs. *arXiv*, 2022.

[137] Rui Wang, Georgios Selimis, Roel Maes, and Sven Goossens. Long-Term Continuous Assessment of SRAM PUF and Source of Random Numbers. In *DATE*, 2020.

[138] Dongfang Li, Zhaojun Lu, Xuecheng Zou, and Zhenglin Liu. PUFKEY: A High-Security and High-Throughput Hardware True Random Number Generator for Sensor Networks. *Sensors*, 2015.

[139] Biswajit Ray and Aleksandar Milenković. True Random Number Generation Using Read Noise of Flash Memory Cells. *TED*, 2018.

[140] Supriya Chakraborty, Abhilash Garg, and Manan Suri. True Random Number Generation from Commodity NVM Chips. *TED*, 2020.

[141] Yinglei Wang, WingKei Yu, Shuo Wu, Greg Malysa, G Edward Suh, and Edwin C Kan. Flash Memory for Ubiquitous Hardware Security Functions: True Random Number Generation and Device Fingerprints. In *SP*, 2012.

[142] Matthias Jung, Éder Zulian, Deepak M. Mathew, Matthias Herrmann, Christian Brugger, Christian Weis, and Norbert Wehn. Omitting Refresh: A Case Study for Commodity and Wide I/O DRAMs. In *MEMSYS*, 2015.

[143] Samira Khan, Donghyuk Lee, Yoongu Kim, Alaa R Alameldeen, Chris Wilkerson, and Onur Mutlu. The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study. In *SIGMETRICS*, 2014.

[144] Samira Khan, Chris Wilkerson, Donghyuk Lee, Alaa R Alameldeen, and Onur Mutlu. A Case for Memory Content-Based Detection and Mitigation of Data-Dependent Failures in DRAM. *CAL*, 2016.

[145] Dae-Hyun Kim, Prashant J Nair, and Moinuddin K Qureshi. Architectural Support for Mitigating Row Hammering in DRAM Memories. *CAL*, 2015.

[146] Justin Meza, Qiang Wu, Sanjeev Kumar, and Onur Mutlu. Revisiting Memory Errors in Large-Scale Production Data Centers: Analysis and Modeling of New Trends from the Field. In *DSN*, 2015.

[147] B. Schroder et al. DRAM Errors in the Wild: A Large-Scale Field Study. In *SIGMETRICS*, 2009.

[148] Christian Weis, Matthias Jung, Peter Ehses, Cristiano Santos, Pascal Vivet, Sven Goossens, Martijn Koedam, and Norbert Wehn. Retention Time Measurements and Modelling of Bit Error Rates of WIDE I/O DRAM in MPSoCs. In *DATE*, 2015.

[149] Minesh Patel, Jeremie S Kim, and Onur Mutlu. The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions. In *ISCA*, 2017.

[150] Seong-Lyong Gong. *Memory Protection Techniques for DRAM Scaling-induced Errors*. PhD thesis, 2018.

[151] Jeremie S Kim, Minesh Patel, Hasan Hassan, and Onur Mutlu. Solar-DRAM: Reducing DRAM Access Latency by Exploiting the Variation in Local Bitlines. In *ICCD*, 2018.

[152] Minesh Patel, Jeremie S Kim, Hasan Hassan, and Onur Mutlu. Understanding and Modeling On-Die Error Correction in Modern DRAM: An Experimental Study Using Real Devices. In *DSN*, 2019.

[153] Hasan Hassan, Yahya Can Tugrul, Jeremie S. Kim, Victor van der Veen, Kaveh Razavi, and Onur Mutlu. Uncovering in-DRAM RowHammer Protection Mechanisms: A New Methodology, Custom RowHammer Patterns, and Implications. In *MICRO*, 2021.

[154] Saugata Ghose, A. Giray Yaglikci, Raghav Gupta, Donghyuk Lee, Kais Kudrolli, William Liu, Hasan Hassan, Kevin Chang, Niladrish Chatterjee, Aditya Agrawal, Mike O'Connor, and Onur Mutlu. What Your DRAM Power Models Are Not Telling You: Lessons from a Detailed Experimental Study. In *SIGMETRICS*, 2018.

[155] Kevin K Chang, A Giray Yağlıkçı, Saugata Ghose, Aditya Agrawal, Niladrish Chatterjee, Abhijith Kashyap, Donghyuk Lee, Mike O'Connor, Hasan Hassan, and Onur Mutlu. Understanding Reduced-Voltage Operation in Modern DRAM Devices: Experimental Characterization, Analysis, and Mechanisms. In *SIGMETRICS*, 2017.

[156] Kevin K Chang, Abhijith Kashyap, Hasan Hassan, Saugata Ghose, Kevin Hsieh, Donghyuk Lee, Tianshi Li, Gennady Pekhimenko, Samira Khan, and Onur Mutlu. Understanding Latency Variation in Modern DRAM Chips: Experimental Characterization, Analysis, and Optimization. In *SIGMETRICS*, 2016.

[157] Jeremie S. Kim, Minesh Patel, Hasan Hassan, and Onur Mutlu. The DRAM Latency PUF: Quickly Evaluating Physical Unclonable Functions by Exploiting the Latency–Reliability Tradeoff in Modern Commodity DRAM Devices. In *HPCA*, 2018.

[158] Donghyuk Lee, Yoongu Kim, Gennady Pekhimenko, Samira Khan, Vivek Seshadri, Kevin Chang, and Onur Mutlu. Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case. In *HPCA*, 2015.