A Deeper Look into RowHammer's Sensitivities: Experimental Analysis of Real DRAM Chips and Implications on Future Attacks and Defenses

Lois Orosa* ETH Zürich
A. Giray Yağlıkçı* ETH Zürich
Haocong Luo ETH Zürich
Ataberk Olgun ETH Zürich, TOBB ETÜ
Jisung Park ETH Zürich
Hasan Hassan ETH Zürich
Minesh Patel ETH Zürich
Jeremie S. Kim ETH Zürich
Onur Mutlu ETH Zürich

ABSTRACT

RowHammer is a circuit-level DRAM vulnerability where repeatedly accessing (i.e., hammering) a DRAM row can cause bit flips in physically nearby rows. The RowHammer vulnerability worsens as DRAM cell size and cell-to-cell spacing shrink. Recent studies demonstrate that modern DRAM chips, including chips previously marketed as RowHammer-safe, are even more vulnerable to RowHammer than older chips such that the required hammer count to cause a bit flip has reduced by more than 10X in the last decade. Therefore, it is essential to develop a better understanding and in-depth insights into the RowHammer vulnerability of modern DRAM chips to more effectively secure current and future systems.

Our goal in this paper is to provide insights into fundamental properties of the RowHammer vulnerability that are not yet rigorously studied by prior works, but can potentially be *i*) exploited to develop more effective RowHammer attacks or *ii*) leveraged to design more effective and efficient defense mechanisms. To this end, we present an experimental characterization using 248 DDR4 and 24 DDR3 modern DRAM chips from four major DRAM manufacturers demonstrating how the RowHammer effects vary with three fundamental properties: 1) DRAM chip temperature, 2) aggressor row active time, and 3) victim DRAM cell's physical location. Among our 16 new observations, we highlight that a RowHammer bit flip 1) is very likely to occur in a bounded range, specific to each DRAM cell (e.g., 5.4% of the vulnerable DRAM cells exhibit errors in the range 70 °C to 90 °C), 2) is more likely to occur if the aggressor row is active for longer time (e.g., RowHammer vulnerability increases by 36% if we keep a DRAM row active for 15 column accesses), and 3) is more likely to occur in certain physical regions of the DRAM module under attack (e.g., 5% of the rows are 2x more vulnerable than the remaining 95% of the rows). Our study has important practical implications on future RowHammer attacks and defenses. We describe and analyze the implications of our new findings by proposing three future RowHammer attack and five future RowHammer defense improvements.

*These authors equally contributed to this work.

MICRO '21, October 18–22, 2021, Virtual Event, Greece

ACM ISBN 978-1-4503-8557-2/21/10.
https://doi.org/10.1145/3466752.3480069

CCS CONCEPTS

• **Security and privacy → Hardware attacks and countermeasures**; • **Hardware → Dynamic memory**.

KEYWORDS

RowHammer, Characterization, DRAM, Memory, Reliability, Security, Safety, Temperature, Testing

ACM Reference Format:
Lois Orosa, A. Giray Yağlıkçı, Haocong Luo, Ataberk Olgun, Jisung Park, Hasan Hassan, Minesh Patel, Jeremie S. Kim, and Onur Mutlu. 2021. A Deeper Look into RowHammer's Sensitivities: Experimental Analysis of Real DRAM Chips and Implications on Future Attacks and Defenses. In *MICRO'21: 54th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO '21), October 18–22, 2021, Virtual Event, Greece.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3466752.3480069

1 INTRODUCTION

To maintain competitive DRAM prices, manufacturers focus on reducing the cost-per-bit of DRAM via DRAM circuit designs and manufacturing process technology that improve DRAM storage density, which in turn reduces DRAM cell size and cell-to-cell spacing. Unfortunately, these reductions have been shown to negatively impact DRAM reliability [89, 94] and expose vulnerabilities such as RowHammer [69, 70, 97]. RowHammer is an error mechanism that is caused by *hammering*, or opening and closing (i.e., *activating* and *precharging*), a DRAM row (i.e., *aggressor row*) many times, which can cause bit flips in physically-nearby rows (i.e., *victim rows*) [28, 56, 70, 95, 97, 106, 107, 123, 145, 157–159]. RowHammer has gained attention in both academia and industry, and various attacks have exploited the RowHammer vulnerability to escalate privilege, leak private data, and manipulate critical application outputs [1, 10, 13, 20, 21, 25, 26, 32, 33, 37, 41, 47, 54, 75, 82, 95, 97, 114, 115, 118, 125, 127, 138, 142, 143, 148, 151, 160, 164].[1] To make matters worse, recent experimental studies [20, 26, 69, 70, 95, 97] have found that the RowHammer vulnerability is becoming more severe in newer DRAM chip generations. For example, as shown in [69], chips manufactured in 2020 can experience RowHammer bit flips after an order of magnitude fewer row activations compared to the chips manufactured in 2014 [70]. As the RowHammer vulnerability worsens, ensuring RowHammer-safe operation becomes more expensive in terms of performance overhead, energy consumption, and hardware complexity [69, 108, 156]. Therefore, it is critical to understand RowHammer in greater detail with in-depth insights

[1] A survey of RowHammer studies and attacks can be found in [97].

into how the RowHammer vulnerability varies under different conditions in order to develop more effective and efficient solutions for the security and reliability of current and future DRAM-based computing systems.

Our goal in this paper is to provide insights into fundamental properties of the RowHammer vulnerability that are not yet rigorously studied by prior works, but can potentially be *i*) exploited to develop more effective RowHammer attacks or *ii*) leveraged to design more effective and efficient defense mechanisms. To this end, we provide a rigorous experimental characterization of 248 DDR4 and 24 DDR3 modern DRAM chips from four major manufacturers to understand how RowHammer vulnerability changes with three fundamental properties of a RowHammer attack: 1) DRAM chip temperature, 2) aggressor row active time, and 3) victim DRAM cell's physical location. This is the first paper that rigorously analyzes these three properties.

Based on our novel characterization results, we make 16 new observations and share 6 key takeaway lessons from our observations. We leverage these observations to propose three RowHammer attack and five RowHammer defense improvements. From our 16 new observations, we highlight three observations that are especially important. First, we find that each vulnerable DRAM cell can experience a RowHammer bit flip only within a bounded temperature range. This range can be as narrow as 5 °C or as wide as 40 °C (in our tested chips). Second, when the aggressor row's active time is longer (e.g., by 5×), 1) more DRAM cells (6.9× on average) experience RowHammer bit flips at a given hammer count and 2) a DRAM row experiences RowHammer bit flips at a smaller hammer count (by 36% on average). Third, a small fraction of DRAM rows in a DRAM module (5%/1%) are *significantly (2.0×/1.6×) more vulnerable* to RowHammer than the rest (95%/99%) of the module.

To study RowHammer effects at the circuit level, we disable RowHammer mitigation mechanisms in the real DRAM chips we characterize. For each experiment, we 1) perform a double-sided RowHammer attack [69, 70, 127], in which both physically-adjacent aggressor rows of the victim row are repeatedly accessed (i.e., hammered), and 2) maintain a high-precision (i.e., error of at most ±0.1 °C) temperature-controlled environment for DRAM. We conduct three main analyses in our characterization study.

First, we investigate the effects of temperature on both 1) the number of bit flips in a DRAM row, referred to as bit error rate (*BER*) and 2) the minimum hammer count value at which the first bit error is observed (HC_{first}) in a victim DRAM row under RowHammer attack. Our *BER* analysis demonstrates that a vulnerable DRAM cell experiences bit flips in a specific and bounded range of temperature, which can be as narrow as 5 °C, or as wide as 40 °C. Our *BER* analysis also shows that the effect of temperature on the *BER* of a DRAM chip highly depends on the DRAM chip manufacturer. For example, DRAM chips of one manufacturer show increasing *BER* with temperature, whereas DRAM chips from another manufacturer show decreasing *BER* with temperature. Our analysis of HC_{first} demonstrates that the RowHammer vulnerability tends to worsen as temperature increases.

Second, we test the sensitivity of RowHammer bit flips to the active time of an aggressor row. To do so, we change the time between an aggressor row activation to the succeeding precharge command from 34.5 ns to 154.5 ns with 30 ns steps while the total hammer count is fixed at a given value. Using this methodology we analyze the variation in both HC_{first} and *BER*. We observe that as the time between the aggressor row activation and precharge command increases, DRAM cells become more vulnerable to RowHammer.

Third, we analyze how RowHammer vulnerability varies based on the *physical location* of a DRAM cell. We observe that HC_{first} significantly varies across rows such that only a small fraction of DRAM rows (5%/1%) exhibit significantly higher RowHammer vulnerability (2.0×/1.6× lower HC_{first} values on average across all four manufacturers) than the rest of the rows (95%/99%).

Based on our new observations, we describe and analyze three (five) improvements to increase the effectiveness of existing RowHammer attacks (defense mechanisms).

We make the following contributions in this work:

- We present the first rigorous experimental study that examines temperature effects on RowHammer bit flips in modern DRAM chips. Our tests using 248 DDR4 and 24 DDR3 modern DRAM chips from four major manufacturers demonstrate that a DRAM cell experiences bit flips in a specific and bounded range of temperature and the RowHammer vulnerability tends to worsen as temperature increases.
- We experimentally demonstrate, for the first time, how RowHammer vulnerability changes with the active time of the aggressor rows. Our results show that as the aggressor row's active time increases (e.g., by 5×), 1) more DRAM cells (6.9× on average) experience RowHammer bit flips at a given hammer count and 2) a DRAM row experiences RowHammer bit flips at a smaller hammer count (by 36% on average).
- We demonstrate that a DRAM cell's RowHammer vulnerability significantly depends on the cell's location. We observe that only a small fraction of DRAM rows (5%/1%) exhibit significantly higher RowHammer vulnerability (2.0×/1.6× lower HC_{first} values) than the rest (95%/99%) of the rows.
- Based on our new observations on RowHammer's sensitivities to temperature, aggressor row's active time, and a victim DRAM cell's physical location in the DRAM chip, we describe and analyze three future RowHammer attack and five future RowHammer defense improvements.

2 BACKGROUND

We provide a brief background on DRAM organization, DRAM access timings, and RowHammer vulnerability. For more detailed background on these, we refer the reader to many prior works [16–19, 26, 29, 35, 36, 38, 45, 59–62, 66–72, 76, 78, 79, 83, 84, 86, 93, 98, 99, 104, 110–112, 117, 128–131, 136, 146, 147, 163].

2.1 DRAM Organization

Fig. 1 depicts the hierarchical organization of DRAM-based main memory. The *memory controller* in a system is typically connected to multiple DRAM *modules* via multiple DRAM *channels*. Each channel operates independently. A DRAM *module* has one or more ranks, each of which consisting 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. Because the I/O bus is shared, the memory controller serializes accesses to different ranks in the same channel. A DRAM chip is organized into multiple DRAM *banks*. DRAM banks in a DRAM chip share a common I/O circuitry.

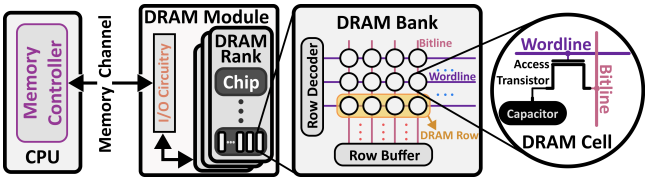

Fig. 1: DRAM organization.

DRAM *cells* in a DRAM bank are laid out in a two-dimensional structure of *rows* and *columns*. Each DRAM cell on a DRAM row is connected to a common *wordline* via *access transistors*. A *bitline* connects a column of DRAM cells to a DRAM sense amplifier to access and manipulate data. The two-dimensional array structure of DRAM cells is typically partitioned into multiple DRAM *subarrays* [17, 72, 128]. Each subarray is connected to sense amplifiers that enable sensing of data, called *local row buffers*.

2.2 DRAM Access Timings

The memory controller accesses DRAM locations via three major steps. First, the memory controller issues an *ACT* command to activate a specific row within a bank, which prepares the row for a column access. Second, the memory controller issues a *RD* or *WR* command to read or write to a column in the row, respectively. Third, once all column operations to the active row are complete, the memory controller issues a precharge (*PRE*) command, which closes the row and prepares the bank for a new activation.

To guarantee correct DRAM operation, the memory controller must observe standardized timings between consecutive commands, called *timing parameters* [72, 77, 78]. Timing parameters ensure that the internal DRAM circuitry has sufficient time to perform the operations required by the command. In this work, we deal with two key timings: 1) the minimum time that a row should stay active before a precharge command is issued to the bank (t_{RAS}) and 2) the minimum time a precharge command needs to complete before a row is activated in the same bank (t_{RP}). t_{RAS} ensures that the DRAM sense amplifiers have enough time following a row activation to correctly restore the charge in all cells in the open row before the row is closed. t_{RP} ensures that all bitlines in the subarray are fully precharged to their idle reference voltage (typically $V_{DD}/2$) before the next row is activated.

2.3 The RowHammer Vulnerability

Modern DRAM chips suffer from an error mechanism, called RowHammer [69, 70, 97] that happens when a DRAM row (i.e., aggressor row) is repeatedly activated enough times before its neighboring rows (i.e., victim rows) get refreshed [28, 56, 69, 70, 95, 97, 106, 107, 123, 145, 157–159]. Due to the aggressive reduction in manufacturing process technology node size, DRAM cells become smaller and closer to each other, exacerbating the RowHammer vulnerability. Therefore, as DRAM manufacturers continue to increase DRAM storage density, DRAM chips' vulnerability to RowHammer increases [20, 26, 69, 70, 95–97].

The RowHammer vulnerability can be used to reliably induce bit flips in main memory using various system-level security attacks [1, 10, 13, 20, 21, 25, 26, 32, 33, 37, 41, 47, 54, 75, 82, 95, 97, 114, 115, 118, 125, 127, 138, 142, 143, 148, 151, 160, 164]. Prior work demonstrates that inducing bit flips via a RowHammer attack is practical for privilege escalation [32, 33, 54, 82, 118, 127, 142, 151], denial of service [32, 82], leaking confidential data [75], and manipulating a critical application's correctness [41, 160]. Thus, it is necessary to rigorously understand the RowHammer vulnerability of modern DRAM chips, project future attacks, and develop effective RowHammer defense mechanisms in modern systems that use DRAM. Through characterization [69, 70, 106, 107] and modeling [28, 56, 107, 119, 123, 145, 157–159], past research shows that circuit-level capacitive coupling [56, 119] and trap-assisted leakage [159] have a significant effect on RowHammer bit flips [145].

Based on the understanding provided by prior characterization and modeling research, a large body of research proposes various RowHammer defenses [2–7, 14, 23, 28, 30, 35, 52, 57, 64, 70, 73, 80, 108, 123, 132, 134, 143, 155–158, 162]. DRAM manufacturers 1) implement RowHammer prevention mechanisms, generally called Target Row Refresh (TRR) [26, 51, 52], which perform proprietary operations within DRAM to prevent RowHammer bit flips (without success, as shown by [26, 37]) and 2) enhance DRAM communication protocols with a new feature called refresh management (*RFM*) [50, 53]. *RFM* requires the memory controller to count the number of activations at DRAM bank granularity and issue a command when the activation count reaches a threshold value. By doing so, it provides an on-DRAM-die RowHammer defense mechanism (e.g., Silver Bullet [23, 155]) with the necessary time to refresh victim rows. Despite efforts to contain and defend against RowHammer, the vulnerability still exists and is expected to worsen in the future [20, 26, 37, 69, 95–97], as clearly demonstrated by recent work [26, 37, 69].

3 MOTIVATION AND GOAL

Prior research experimentally demonstrates that RowHammer is clearly a worsening DRAM reliability and security problem [20, 26, 69, 70, 95–97]. Despite all efforts, newer DRAM chips are shown to be *significantly* more vulnerable to RowHammer than older generation chips [69]. Even DRAM chips that have been marketed as RowHammer-free in 2020 experience RowHammer bit flips at *significantly* lower hammer counts (e.g., 9.6K for LPDDR4 chips when TRR protection is disabled [69] and 25K for DDR4 chips when TRR protection is enabled [26]) compared to the DDR3 DRAM chips manufactured in 2014 (e.g., 139K [70]). Many prior works [2–7, 14, 23, 28, 30, 35, 52, 57, 64, 70, 73, 80, 108, 123, 132, 134, 143, 155–158, 162] have proposed RowHammer defense mechanisms to provide RowHammer-safe operation with either probabilistic or deterministic security guarantees. Unfortunately, recent works [69, 108, 156] have demonstrated that many of these defense mechanisms will incur *significant* performance, energy consumption, and/or hardware complexity overheads such that they become prohibitively expensive when deployed in future DRAM chips [69].

To enable RowHammer-safe operation in future DRAM-based computing systems in an effective and efficient way, it is critical to rigorously gain detailed insights into the RowHammer vulnerability and its sensitivities to varying attack properties. Unfortunately, despite the existing research efforts expended towards understanding RowHammer [28, 56, 69, 70, 105–107, 119, 123, 145, 157–159], scientific literature lacks rigorous experimental observations on how the RowHammer vulnerability varies with three fundamental properties: 1) DRAM chip temperature, 2) aggressor row active

time, and 3) victim DRAM cell's physical location. This lack of understanding raises very practical and important concerns as to how the effects of these three fundamental properties can be exploited to improve both RowHammer attacks and defense mechanisms.

Our goal in this paper is to rigorously evaluate and understand how the RowHammer vulnerability of a real DRAM chip at the circuit level changes with 1) temperature, 2) aggressor row active time, and 3) victim DRAM cell's physical location in the DRAM chip. Doing so provides us with a deeper understanding of RowHammer to enable future research on improving the effectiveness of existing RowHammer attacks and defense mechanisms. We hope that these analyses will pave the way for building RowHammer-safe systems that use increasingly more vulnerable DRAM chips. To achieve this goal, we rigorously characterize how the RowHammer vulnerability of 248 DDR4 and 24 DDR3 modern DRAM chips from four major DRAM manufacturers vary with these three properties.

4 METHODOLOGY

We describe our methodology and infrastructure for characterizing the RowHammer vulnerability in real DRAM modules.

4.1 Testing Infrastructure

We experimentally study 248 DDR4 and 24 DDR3 DRAM chips across a wide range of testing conditions. We use two different testing infrastructures: 1) SoftMC [38, 126], capable of precisely controlling temperature and command timings of DDR3 DRAM modules and 2) a modified version of this infrastructure that supports DDR4 chips, also used in [26, 37, 69, 103].

SoftMC. Fig. 2 shows one of our SoftMC setups for testing DDR4 modules (Fig. 2a). We use two types of Xilinx FPGA boards: 1) Alveo U200 [154] (Fig. 2b) to test DDR4 DIMMs [52, 92], and 2) ML605 [152] to test DDR3 SODIMMs. This infrastructure enables precise control over both DDR4 and DDR3 timings at the granularity of 1.25 ns and 2.50 ns, respectively. We use a host machine, connected to our FPGA boards through a PCIe port [113] (Fig. 2c) to 1) perform the RowHammer tests that we describe in §4.2 and 2) monitor and adjust the temperature of DRAM chips in cooperation with the temperature controller (Fig. 2d).

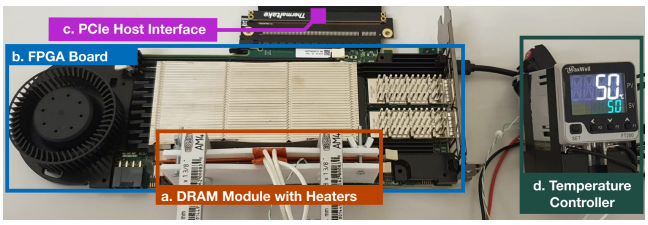

Fig. 2: SoftMC Infrastructure: (a) DRAM module under test clamped with heater pads, (b) Xilinx Alveo U200 FPGA board [154], programmed with a DDR4 version of SoftMC [38], (c) PCIe connection to the host machine, and (d) temperature controller.

Temperature Controller. To regulate the temperature in DRAM modules, we use silicone rubber heaters pressed to both sides of the DRAM module (Fig. 2a). We use a thermocouple, placed on the DRAM chip to measure the chip's temperature (similar to JEDEC standards [48]). A Maxwell FT200 temperature controller [88] (Fig. 2d) 1) monitors a DRAM chip's temperature using a thermocouple, and 2) keeps the temperature stable by heating the chip with heater pads. The temperature controller 1) communicates with our host machine via an RS485 channel [149] to get a reference temperature and to report the instant temperature, and 2) controls the heater pads using a closed-loop PID controller. In our tests using this infrastructure, we measure temperature with an error of at most ±0.1 °C. We believe that our temperature measurements from the DRAM package's surface accurately represent the DRAM die's real temperature because the temperature of the DRAM package and the DRAM internal components are strongly correlated [91].

4.2 Testing Methodology

Disabling Sources of Interference. Our goal is to directly observe the circuit-level bit flips such that we can make conclusions about DRAM's vulnerability to RowHammer at the circuit technology level rather than at the system level. To this end, we minimize all possible sources of interference with the following steps. First, we disable all DRAM self-regulation events (e.g., DRAM Refresh [38, 52, 153]) except calibration related events (e.g., ZQ calibration for signal integrity [38, 52]). Second, we ensure that all RowHammer tests are conducted within a relatively short period of time such that we do not observe retention errors [60, 83, 89, 112, 117]. Third, we use the SoftMC memory controller [38, 126] so that we can 1) issue DRAM commands with precise control (i.e., our commands are not impeded by system-issued accesses), and 2) study the RowHammer vulnerability on DRAM chips without interference from existing system-level RowHammer protection mechanisms (e.g., [3, 5–7]). Fourth, we test DRAM modules that do *not* implement error correction codes (ECC) [12, 21, 34, 39, 65, 120]. Doing so ensures that neither on-die [44, 100, 109–111] nor rank-level [21, 65] ECC can alter the RowHammer bit flips we observe and analyze. Fifth, we prevent known on-DRAM-die RowHammer defenses (i.e., TRR [50, 53, 81, 90]) from working by not issuing refresh commands throughout our tests [26, 69].

RowHammer. All our tests use double-sided RowHammer [69, 70, 127], which activates, in an alternating manner, each of the two rows (i.e., aggressor rows) that are physically-adjacent to a victim row. We call this victim row a double-sided victim row. We define single-sided victim rows as the rows that are hammered in a single-sided manner by the two aggressor rows (i.e., rows with +2 or -2 distance from victim row). We define one hammer as a pair of activations to the two aggressor rows. We perform double-sided hammering with the maximum activation rate possible within DDR3/DDR4 command timing specifications [49, 52]. Prior works report that this is the most effective access pattern for RowHammer attacks on DRAM chips when RowHammer mitigation mechanisms are disabled [20, 26, 69, 70, 127].[2] We use 150K hammers (i.e., 300K activations) in our *BER* experiments.[3] We use up to 512K hammers (i.e., the maximum number of hammers so that our hammer tests run for less than 64ms) in our HC_{first} experiments. Due to time limitations, we repeat each test five times, and we study the effects of the RowHammer attack on the 1) first 8K rows, 2) last 8K rows, and 3) middle 8K rows of a bank in each DRAM chip (similar to [70]).

[2]Our analysis of aggressor row active time uses a different access sequence that introduces additional delays between row activations. See §6 for details.
[3]We find that 150K hammers is low enough to be used in a system-level RowHammer attack in a real system [26], and it is high enough to provide a large number of bit flips in all DRAM modules we tested.

Logical-to-Physical Row Mapping. DRAM manufacturers use DRAM-internal mapping schemes to internally translate memory-controller-visible row addresses to physical row addresses [8, 20, 42, 46, 59, 61, 63, 70, 76, 83, 110, 130, 133, 138, 156], which can vary across different DRAM modules. We reverse-engineer this mapping, so that we can identify and hammer aggressor rows that are physically adjacent to a victim row. We reconstruct the mapping by 1) performing single-sided RowHammer attack on each DRAM row, 2) inferring that the two victim rows with the most RowHammer bit flips are physically adjacent to the aggressor row, and 3) deducing the address mapping after analyzing the aggressor-victim row relationships across all studied DRAM rows.

Data Pattern. We conduct our experiments on a DRAM module by using the module's worst-case data pattern (*WCDP*). We identify the *WCDP* for each module as the pattern that results in the largest number of bit flips among seven different data patterns used in prior works on DRAM characterization [16, 19, 60–63, 69, 76, 77, 83, 112], presented in Table 1: colstripe, checkered, rowstripe, and random (we also test the complements of the first three). For each RowHammer test, we write the corresponding data pattern to the victim row (V in Table 1), and to the 8 previous ($V - [1...8]$) and next ($V + [1...8]$) physically-adjacent rows.

Table 1: Data patterns used in our RowHammer analyses.

Row Address	Colstripe†	Checkered†	Rowstripe†	Random
$V^* \pm [0, 2, 4, 6, 8]$	0x55	0x55	0x00	random
$V^* \pm [1, 3, 5, 7]$	0x55	0xaa	0xff	random

*V is the physical address of the victim row
†We also test the complements of these patterns

Metrics. We measure two metrics in our tests: 1) the minimum hammer count value at which the first bit error is observed (HC_{first}) and 2) the number of bit flips in a DRAM row, referred to as bit error rate (*BER*). A *lower* HC_{first} or *higher BER* value indicates higher RowHammer vulnerability. To quickly identify HC_{first}, we perform a binary search where we use an initial hammer count of 256k. We repeatedly increase (decrease) the hammer count by Δ if we observe (do not observe) bit flips in the victim row. The initial value is $\Delta = 128k$, and we halve it for each test until it reaches $\Delta = 512$ (i.e., we identify HC_{first} with an accuracy of 512 row activations).

Temperature Range. To study the effects of temperature, we test DRAM chips across a wide range of temperatures, from 50 °C to 90 °C, with a step size of 5 °C.

4.3 Characterized DRAM Chips

Table 2 summarizes the 248 DDR4 and 24 DDR3 DRAM chips we test from four major manufacturers. We use a diverse set of modules with different chip densities, die revisions and chip organizations. We share analyses of additional modules separately in [124].

Table 2: Summary of DDR4 (DDR3) DRAM chips tested.

Mfr.	DDR4 #DIMMs	DDR3 #SODIMMs	#Chips	Density	Die	Org.
Mfr. A	9	1	144 (8)	8Gb (4Gb)	B (P)	x4 (x8)
Mfr. B	4	1	32 (8)	4Gb (4Gb)	F (Q)	x8 (x8)
Mfr. C	5	1	40 (8)	4Gb (4Gb)	B (B)	x8 (x8)
Mfr. D	4	–	32 (–)	8Gb (–)	C (–)	x8 (–)

5 TEMPERATURE ANALYSIS

We 1) provide the first rigorous experimental characterization of the effects of temperature on the RowHammer vulnerability using real DRAM chips and 2) present new observations and insights based on our results.

5.1 Impact of Temperature on DRAM Cells

We analyze the relation between temperature and the RowHammer vulnerability of a DRAM cell using the methodology described in Section 4.2. To do so, we first cluster vulnerable DRAM cells by their *vulnerable temperature range* (i.e., the minimum and maximum temperatures within which a cell experiences at least one RowHammer bit flip across all experiments). Second, we analyze *how* the RowHammer bit flips of DRAM cells manifest within their vulnerable temperature range. Table 3 shows the percentage of vulnerable cells that flip in *all* temperature points of their vulnerable temperature ranges.

Table 3: Percentage of vulnerable DRAM cells that flip in all temperature points within the vulnerable temperature range of the cell.

Mfr. A	Mfr. B	Mfr. C	Mfr. D
99.1%	98.9%	98.0%	99.2%

Obsv. 1. *A DRAM cell is, with a very high probability, vulnerable to RowHammer in a continuous temperature range specific to the cell.*

For example, only 0.9% of the vulnerable DRAM cells in Mfr. A do *not* exhibit bit flips in at least one temperature point within their vulnerable temperature range. Hence, our experiments demonstrate that a cell exhibits bit flips with very high probability in a continuous temperature range that is specific to the cell.

To analyze the diversity of vulnerable temperature ranges across DRAM cells, we cluster all vulnerable DRAM cells according to their vulnerable temperature ranges. Fig. 3 shows each cluster's size as a percentage of the full population of vulnerable cells. The x-axis (y-axis) indicates the lower (upper) bound of the vulnerable temperature range. Because we do not test temperatures higher (lower) than 90 °C (50 °C), the vulnerable temperature ranges with an upper (lower) limit of 90 °C (50 °C) include cells that also flip at higher (lower) temperatures. For example, 5.4% of the vulnerable DRAM cells in Mfr. A fall into the range 70 °C to 90 °C, which includes cells with *actual* vulnerable temperature ranges of 70 °C to 95 °C, 70 °C to 100 °C, etc.

Obsv. 2. *A significant fraction of vulnerable DRAM cells exhibit bit flips at all tested temperatures.*

We observe that between 9.6% and 29.8% of the cells (x-axis=50 °C, y-axis=90 °C in Fig. 3) are vulnerable to RowHammer across *all* tested temperatures (50 °C to 90 °C) for the four DRAM manufacturers. We also verify (not shown) that Obsv. 2 holds for the three SODIMM DDR3 modules described in Table 2.

Obsv. 3. *A small fraction of all vulnerable DRAM cells are vulnerable to RowHammer only in a very narrow temperature range.*

For example, 0.4% of all vulnerable DRAM cells of Mfr. A, are only vulnerable to RowHammer at 70 °C (i.e., a single tested temperature value). Note that inducing even a single bit flip can be critical for system security, as shown by prior works [25, 32, 118, 151]. Our experimental results show that 2.3%, 1.8%, 2.4%, and 1.6% of all tested DRAM cells for Mrfs. A, B, C, and D, respectively, experience

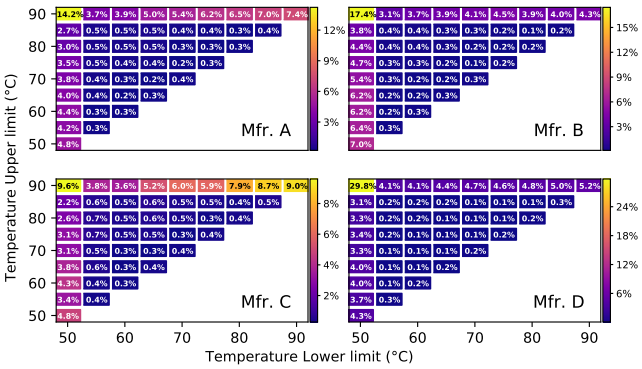

Fig. 3: Population of vulnerable DRAM cells, clustered by vulnerable temperature range.

a RowHammer bit flip within a temperature range as narrow as 5 °C. We conclude that some DRAM cells experience RowHammer bit flips at localized and narrow temperature ranges.

We exploit Obsvs. 1–3 in §8.

Takeaway 1. *To ensure that a DRAM cell is not vulnerable to RowHammer, we must characterize the cell at all operating temperatures.*

5.2 Impact of Temperature on DRAM Rows

We analyze the relation between a DRAM row's RowHammer vulnerability and temperature in terms of both *BER* and HC_{first}.

BER Analysis. Fig. 4 shows how the *BER* changes as temperature increases, compared to the mean *BER* value across all the samples at 50 °C, for four DRAM manufacturers. In each plot, we use a point and error bar[4] to show the *BER* change for the victim row (i.e., distance from the victim row = 0), and the *BER* change for the two single-sided victim rows (i.e., distance ±2 from the victim row), across all rows we test.

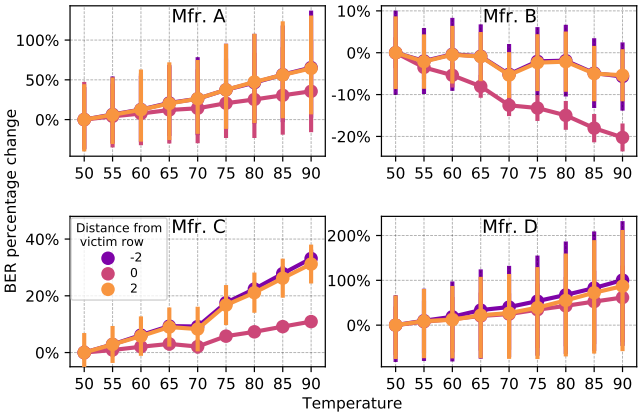

Fig. 4: Percentage change in *BER* (RowHammer bit flips) with increasing temperature, compared to *BER* at 50°C.

Obsv. 4. *A DRAM row's BER can either increase or decrease with temperature depending on the DRAM manufacturer.*

[4]Each point and error bar represent the mean and the 95% confidence interval across the samples, respectively.

We observe that the average *BER* of all three victim rows (one double-sided victim row and two single-sided victim rows), from Mfrs. A, C, and D increases with temperature, whereas the *BER* of rows from Mfr. B decreases as temperature increases. We hypothesize that the difference between these trends is caused by a combination of DRAM circuit design and manufacturing process technology differences (see §5.3).

***HC_{first}* Analysis.** Fig. 5 shows the distribution of the change in HC_{first} (in percentage) when temperature increases from 50 °C to 55 °C, and from 50 °C to 90 °C, for the vulnerable rows of the four manufacturers. The x-axis represents the percentage of all vulnerable rows, sorted from the most positive HC_{first} change to the most negative HC_{first} change. For each curve, we mark the x-axis point at which the curve crosses the y=0% line. This represents the percentile of rows whose HC_{first} increases with temperature; e.g., for Mfr. A, when temperature increases from 50 °C to 90 °C, only 45% (P45) of the tested rows have a higher HC_{first} (indicating reduced vulnerability for that fraction of rows); i.e., most rows from Mfr. A are more vulnerable at 90 °C than at 50 °C. For clarity, we only show two temperature changes (i.e., from 50 °C to 55 °C and from 50 °C to 90 °C), but our observations are consistent across all intermediate temperature changes we tested (i.e., from 50 °C to 50+Δ°C, for all Δ's that are multiples of 5 °C).

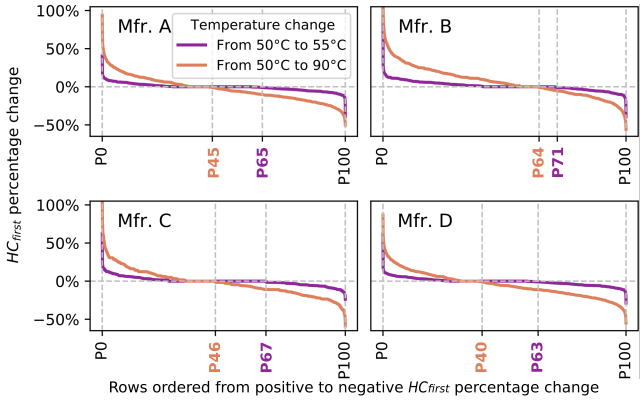

Fig. 5: Distribution of the change in HC_{first} across vulnerable DRAM rows as temperature increases.

Obsv. 5. *DRAM rows can show either higher or lower HC_{first} when temperature increases.*

We observe that, for all four manufacturers, a significant fraction of rows can show either higher or lower HC_{first} when temperature increases. For example, when the temperature changes from 50 °C to 55 °C in Mfr. A, 65% of the rows show higher HC_{first}, while 35% of the rows show lower HC_{first}. We conclude that HC_{first} changes differently depending on the DRAM row.

Obsv. 6. *HC_{first} tends to generally decrease as temperature change increases.*

We observe that, for all four manufacturers, fewer rows have a higher HC_{first} when the temperature delta is larger; i.e., the point at which each curve crosses the y=0% point shifts left when the temperature change increases. For example, for Mfr. D, the fraction of vulnerable cells with a higher HC_{first} is much larger when temperature increases from 50 °C to 55 °C (63% of cells) than when

the temperature increases from 50 °C to 90 °C (40% of cells). We conclude that the dominant trend is for a row's HC_{first} to decrease when the temperature delta is larger.

Obsv. 7. *The change in HC_{first} tends to be larger as the temperature change is larger.*

The HC_{first} distribution curve exhibits higher absolute magnitudes when temperature changes from 50 °C to 90 °C, compared to when temperature changes from 50 °C to 55 °C (i.e., the curve generally rotates right and has much higher peaks at its edges when the temperature change increases, i.e., going from orange to purple in the figure). We quantify this observation by calculating the cumulative magnitude change (i.e., the sum of the absolute values of the HC_{first} change from all rows). Our results show that the cumulative magnitude change (not shown in the figure) is 4.2×, 3.9×, 3.8× and 4.3× larger in Mfrs. A, B, C, and D, respectively, when the temperature changes from 50 °C to 90 °C, compared to 50 °C to 55 °C. We conclude that a larger change in temperature causes a larger change in HC_{first}.

Takeaway 2. *RowHammer vulnerability (i.e., both BER and HC_{first}) tend to worsen as DRAM temperature increases. However, individual DRAM rows can exhibit behavior different from this dominant trend.*

5.3 Circuit-level Justification

We hypothesize that our observations on the relation between RowHammer vulnerability and temperature are caused by the non-monotonic behavior of charge trapping characteristics of DRAM cells. Yang et al. [159] show a DRAM charge trap model simulated using a 3D TCAD tool (*without* real DRAM chip experiments). The model shows that HC_{first} decreases as temperature increases, until a temperature inflection point where HC_{first} starts to increase as temperature increases. According to this model, a cell is more vulnerable to RowHammer at temperatures close to its temperature inflection point. We hypothesize that rows within a DRAM chip might have a wide variety of temperature inflection points, and thus the average temperature inflection point of a DRAM chip would determine whether the average RowHammer vulnerability increases or decreases with temperature (Obsvs. 1–7). Park et al. [105, 106] also show an analysis of the relation between HC_{first} and DRAM temperature. Their observations are similar to ours, but they consider only a small number of DDR3 DRAM cells.

Unlike simulations and limited results reported by [105, 106, 159], our comprehensive experiments with 272 DRAM chips show that the temperature inflection points for RowHammer vulnerability are very diverse across DRAM cells and chips.

6 AGGRESSOR ROW ACTIVE TIME ANALYSIS

We provide the first rigorous characterization of RowHammer considering the time that the aggressor row stays in the row buffer (i.e., *aggressor row active time*). Prior works [105, 106, 145] propose circuit models and suggest that RowHammer vulnerability of a victim row can depend on the aggressor row active time based on preliminary data on a very small number of DRAM cells (i.e., only one carefully-selected DRAM row from each manufacturer) [105, 106]. However, none of these works conduct a rigorous analysis of how RowHammer vulnerability varies with aggressor row active time across a significant population of DRAM rows from real off-the-shelf DRAM modules.

Fig. 6 describes the three tests we perform in our experiments: 1) *Baseline Test*, where we use t_{RAS} as the time that an aggressor row stays active, i.e., aggressor row's on-time (t_{AggOn}), and we use t_{RP} as the time that the bank stays precharged, i.e., aggressor row's off-time (t_{AggOff}), 2) *Aggressor On Tests*, where we increase t_{AggOn} before the row is precharged (compared to t_{RAS} in Baseline Test), and 3) *Aggressor Off Tests*, where we increase t_{AggOff} before the aggressor row is activated (compared to t_{RP} in Baseline Test). Therefore, for a given hammer count HC, the overall attack time is $(t_{AggOn} + t_{RP}) \times HC$ and $(t_{RAS} + t_{AggOff}) \times HC$ for Aggressor On and Off Tests, respectively, while it is $(t_{RAS} + t_{RP}) \times HC$ for the baseline tests. Our experiments in this section are conducted at 50 °C on the first 1K rows, the last 1K rows, and the 1K rows in the middle of a bank in our DDR4 chips.

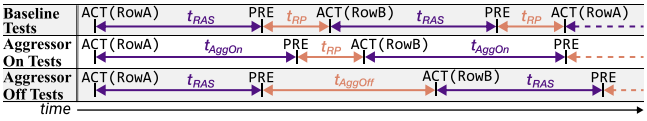

Fig. 6: DRAM command timings for aggressor row active time (t_{AggOn}/t_{AggOff}) experiments. Purple/Orange color indicates that an aggressor row is active/precharged.

6.1 Impact of Aggressor Row's On-Time

Fig. 7 and Fig. 8 show the RowHammer bit flips per row (*BER*) and HC_{first} distributions using box plots[5] and letter-value plots,[6] respectively, across all DRAM chips, as we vary t_{AggOn} from 34.5 ns (t_{RAS}) to 154.5 ns.

Obsv. 8. *As the aggressor row stays active longer (i.e., t_{AggOn} increases), more DRAM cells experience RowHammer bit flips and they experience RowHammer bit flips at lower hammer counts.*

We observe that increasing t_{AggOn} from 34.5 ns to 154.5 ns *significantly* 1) increases *BER* by 10.2×, 3.1×, 4.4×, and 9.6× on average and 2) decreases HC_{first} by 40.0%, 28.3%, 32.7%, and 37.3% on average, in DRAM chips from Mfrs. A, B, C and D, respectively.

Obsv. 9. *RowHammer vulnerability consistently worsens as t_{AggOn} increases in DRAM chips from all four manufacturers.*

To see how RowHammer vulnerability changes as t_{AggOn} increases, we examine the coefficient of variation (*CV*)[7] values of the *BER* and HC_{first} distributions (not shown in the figures). We find that *CV* decreases by around 15% and 10% for *BER* and HC_{first}, respectively, across all four manufacturers, as t_{AggOn} increases from

[5] In a box plot [141], the box shows the lower and upper quartile of the data (i.e., the box spans the 25$^{\text{th}}$ to the 75$^{\text{th}}$ percentile of the data). The line in the box represents the median. The bottom and top whiskers each represent an additional 1.5× the *interquartile range* (IQR, the range between the bottom and the top of the box) beyond the lower and upper quartile, respectively.

[6] In a letter-value plot [40], the widest box shows the lower and upper quartile of the data. The line in the box represents the median. The narrower box extended from the bottom of the widest box shows the lower octile (12.5$^{\text{th}}$ percentile) and the lower quartile of the data, and the narrower box extended from the top of the widest box shows the upper octile and the upper quartile of the data, etc.. Boxes are plotted until all remaining data are outliers. Outliers are defined as the 0.7% extreme values in the dataset, and are plotted as fliers in the plot.

[7] $CV = standard\ deviation/average$ [24].

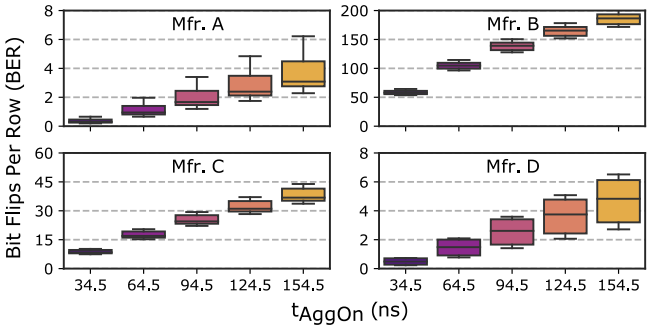

Fig. 7: Distribution of the average number of bit flips per victim row across chips as aggressor row on-time (t_{AggOn}) increases.

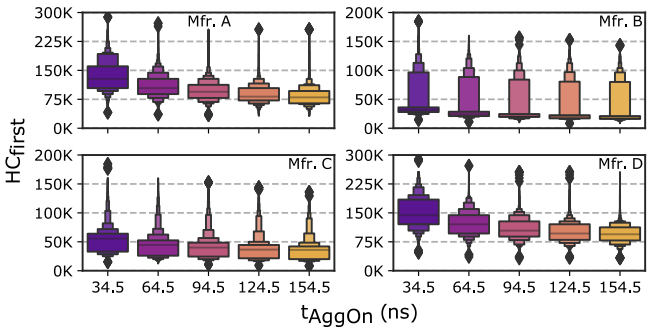

Fig. 8: Distribution of per-row HC_{first} across chips as aggressor row on-time (t_{AggOn}) increases.

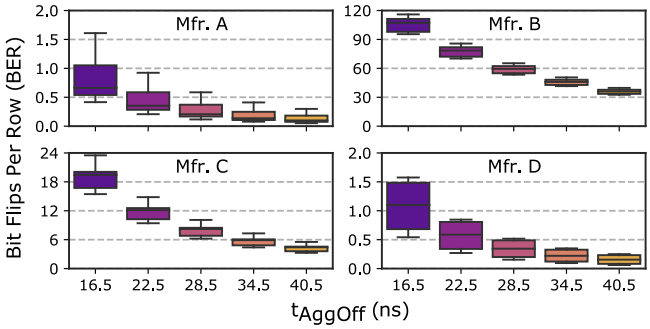

Fig. 9: Distribution of the average number of bit flips per victim row across chips as aggressor row off-time (t_{AggOff}) increases.

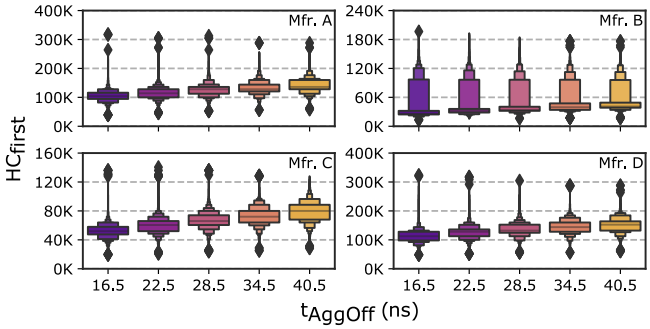

Fig. 10: Distribution of per-row HC_{first} across chips as aggressor row off-time (t_{AggOff}) increases.

34.5 ns to 154.5 ns. This indicates that increasing the aggressor row active time consistently worsens RowHammer vulnerability across the DRAM chips we test.

We conclude from Obsvs. 8 and 9 that increasing t_{AggOn} makes victim DRAM cells much more vulnerable to a RowHammer attack. We exploit these observations in §8.

Takeaway 3. *As an aggressor row stays active longer, victim DRAM cells become more vulnerable to RowHammer.*

6.2 Impact of Aggressor Row's Off-Time

Figs. 9 and 10 show the *BER* and HC_{first} distributions, respectively, as we vary t_{AggOff} from 16.5 ns (t_{RP}) to 40.5 ns.[8]

Obsv. 10. *As the bank stays precharged longer (i.e., t_{AggOff} increases), fewer DRAM cells experience RowHammer bit flips and they experience RowHammer bit flips at higher hammer counts.*

We observe that increasing t_{AggOff} from 16.5 ns to 40.5 ns *significantly* 1) decreases *BER* by 6.3×, 2.9×, 4.9×, and 5.0× on average, and 2) increases HC_{first} by 33.8%, 24.7%, 50.1%, and 33.7% on average, in DRAM chips from Mfrs. A, B, C, and D, respectively.

Obsv. 11. *RowHammer vulnerability consistently reduces as t_{AggOff} increases in DRAM chips from all four manufacturers.*

We observe that the *CV* of HC_{first} (not shown in the figures) does not increase for any manufacturer as we increase t_{AggOff}. Hence, the level of reduction in RowHammer vulnerability is similar across different rows' *most vulnerable cells*. In contrast, the *CV* of *BER* increases by 18% on average for all four manufacturers, indicating that the level of reduction in RowHammer vulnerability is different across different rows.

We conclude from Obsvs. 10 and 11 that increasing t_{AggOff} makes it harder for a RowHammer attack to be successful. We exploit this to improve RowHammer defense mechanisms in §8.2.

Takeaway 4. *RowHammer vulnerability of victim cells decreases when the bank is precharged for a longer time.*

6.3 Circuit-level Justification

Prior work explains two circuit- and device-level mechanisms, causing RowHammer bit flips: 1) electron injection into the victim cell [145, 157], and 2) wordline-to-wordline cross-talk noise between aggressor and victim rows that occurs when the aggressor row is being activated [123, 145]. We hypothesize that increasing the aggressor row's active time (t_{AggOn}) has a larger impact on exacerbating electron injection to the victim cell, compared to the reduction in cross-talk noise due to lower activation frequency. Thus, RowHammer vulnerability worsens when t_{AggOn} increases, as our Obsvs. 8 and 9 show.

On the other hand, increasing a bank's precharged time (t_{AggOff}) decreases RowHammer vulnerability (Obsvs. 10 and 11) because longer t_{AggOff} reduces the effect of cross-talk noise without affecting electron injection (since t_{AggOn} is unchanged). We leave the detailed device-level analysis and explanation of our observations to future works.

[8] Statistical configurations of the box and letter-value plots in Figs. 9 and 10 are identical to those in Figs. 7 and 8, respectively.

7 SPATIAL VARIATION ANALYSIS

We provide the first rigorous spatial variation analysis of RowHammer across DRAM rows, subarrays, and columns. Prior work [69, 70, 105–107] analyzes RowHammer vulnerability at the DRAM bank granularity across many DRAM modules without providing analysis of the variation of this vulnerability across rows, subarrays, and columns. We provide this analysis and show that it is useful for improving both attacks and defense mechanisms. Our experiments in this section are conducted at 75 °C.

7.1 Variation Across DRAM Rows

Fig. 11 shows the distribution of HC_{first} values across all vulnerable DRAM rows among the rows we test (§4.2). For each row, we plot the minimum HC_{first} value observed across 5 repetitions of the test. Each subplot shows DRAM modules from a different manufacturer, and each curve corresponds to a different DRAM module. The x-axis shows all the tested rows, sorted by decreasing HC_{first} and marked with percentiles ranging from P1 to P99.

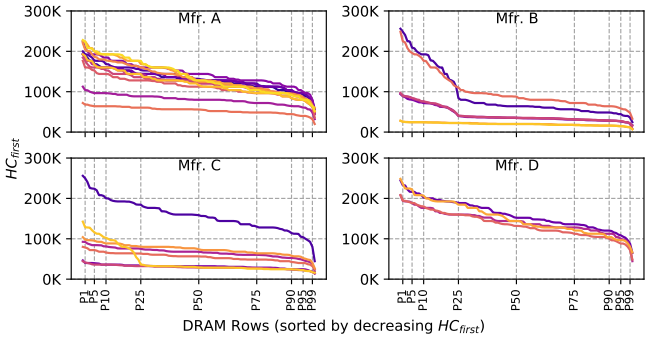

Fig. 11: Distribution of HC_{first} across vulnerable DRAM rows. Each curve represents a different tested DRAM module.

Obsv. 12. *A small fraction of DRAM rows are significantly more vulnerable to RowHammer than the vast majority of the rows.*

HC_{first} varies significantly across rows. We observe that 99%, 95%, and 90% of tested rows exhibit HC_{first} values that are at least 1.6×, 2.0×, and 2.2× greater than the most vulnerable row's HC_{first}, on average across all four manufacturers. For example, the lowest HC_{first} across all tested rows in a DRAM module from Mfr. B is 33K, while 99%, 95%, and 90% of the rows in the same module exhibit HC_{first} values equal to or greater than 48.5K, 60.5K, and 64K, respectively. Therefore, we conclude that a small fraction of DRAM rows are significantly more vulnerable to RowHammer than the vast majority of the rows.

The large variation in HC_{first} across DRAM rows can enable future improvements in low-cost RowHammer defenses (§8.2).

7.2 Variation Across Columns

Fig. 12 shows the distribution of the number of RowHammer bit flips across columns in eight representative DRAM chips from each of all four manufacturers. For each DRAM chip (y-axis), we count the bit flips in each column (x-axis) across all 24K tested rows. The color-scale next to each subplot shows the bit flip count: a brighter color indicates more bit flips.

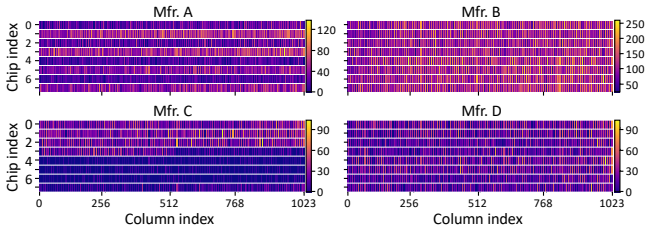

Fig. 12: RowHammer bit flip distribution across columns in representative DRAM chips from four different manufacturers.

Obsv. 13. *Certain columns are significantly more vulnerable to RowHammer than other columns.*

All chips show significant variation in *BER* across columns. For example, the difference between the maximum and the minimum bit flip counts per column is larger than 100 in modules from all four manufacturers. Except for the module from Mfr. B, where every column shows at least 6 bit flips, all the other tested modules have a considerable fraction of columns where *no* bit flip occurs (27.80%/31.10%/9.96% in Mfr. A/C/D), along with a very small fraction of columns with more than 100 bit flips (0.59%/0.01%/0.61% in Mfr. A/C/D). Therefore, we conclude that certain columns are significantly more vulnerable to RowHammer than other columns.

To better understand this column-to-column variation, we study how RowHammer vulnerability varies between columns *within* a single DRAM chip and *across* different DRAM chips. Understanding this variation can provide insights into the impact of circuit design on a column's RowHammer vulnerability, which is important for understanding and overcoming RowHammer. A smaller variation in a column's RowHammer vulnerability across chips indicates a stronger influence of design-induced variation [66, 76], while a larger variation across chips that implement the same design indicates a stronger influence of manufacturing process variation [16, 19, 67, 68, 77, 83, 84, 112]. To differentiate between these two sources of variation in our experiments, we cluster every column in a given DRAM module based on two metrics. The first metric is the column's *relative RowHammer vulnerability*, defined as the column's *BER*, normalized to the maximum *BER* across all columns in the same module. The second metric is *the RowHammer vulnerability variation* at a column address. We quantify the variation using the coefficient of variation (*CV*) of the relative RowHammer vulnerability in columns with the same column address from different DRAM chips. Fig. 13 shows a two-dimensional histogram with the *relative RowHammer vulnerability* (y-axis) and *Rowhammer vulnerability variation* (x-axis) uniformly quantized into 11 buckets each (i.e., 121 total buckets across each subplot).[9] Each bucket is illustrated as a rectangle containing a percentage value, which shows the percent of all columns that fall within the bucket. Empty buckets are omitted for clarity.

Obsv. 14. *Both design and manufacturing processes may affect a DRAM column's RowHammer vulnerability.*

We find that 50.9% and 16.6%[10] of all vulnerable columns in DRAM modules from Mfrs. B and C have CV=0.0, which indicates

[9] We plot the x-axis as saturated at 1.0 because a $CV > 1$ means that the standard deviation is larger than the average, i.e., the variation is very large across chips.
[10] These numbers represent the population of columns whose CV across chips is zero, i.e., sum of all annotated percentage values where CV=0.

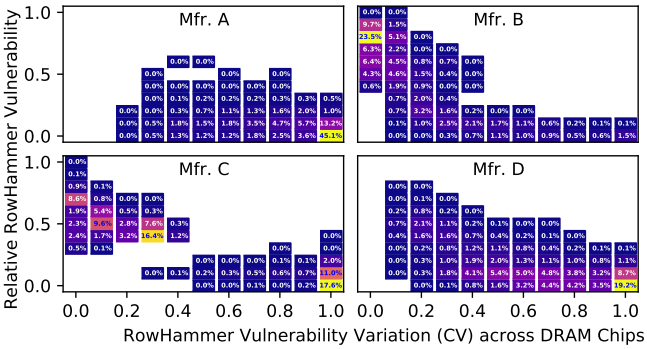

Fig. 13: Population of DRAM columns, clustered by relative RowHammer vulnerability.

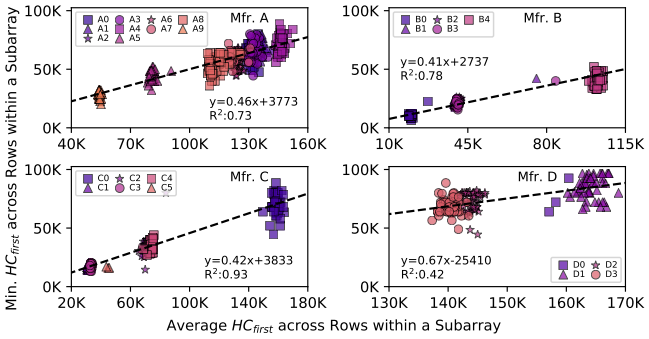

Fig. 14: HC_{first} variation across subarrays. Each subarray is represented by the average (x-axis) and the minimum (y-axis) HC_{first} across the rows within the subarray.

that each of these columns exhibit the same level of RowHammer vulnerability consistently across *all* DRAM chips in a module. This consistency across chips implies that *systematic variation* is present, induced by a chip's design [16, 18, 66, 76–78, 135, 144]. In contrast, 59.8%, 30.6%, and 29.1% of vulnerable columns in DRAM modules from Mfrs. A, C, and D show a very large variation across chips (CV=1.0). This large variation across chips suggests that *manufacturing process* variation is *also* a significant factor in determining a given DRAM column's RowHammer vulnerability.

We conclude from Obsvs. 12–14 that there is significant variation in RowHammer vulnerability across DRAM rows, columns, and chips. These observations are useful for 1) crafting attacks that target vulnerable locations (see §8.1) or 2) improving defense mechanisms and error correction schemes that exploit the heterogeneity of vulnerability across DRAM rows and columns (see §8.2).

Takeaway 5. *RowHammer vulnerability significantly varies across DRAM rows and columns due to both design-induced and manufacturing-process-induced variation.*

7.3 Variation Across Subarrays

We analyze the RowHammer vulnerability of individual subarrays across DRAM chips. Since subarray boundaries are not publicly available, we conservatively assume a subarray size of 512 rows as reported in prior work [17, 66, 72, 76, 144].[11]

Fig. 14 shows the variation of HC_{first} characteristics in a DRAM bank across subarrays both 1) in a DRAM module and 2) across modules from the same manufacturer. Each color-marker pair represents a different DRAM module. We represent the HC_{first} of a subarray in terms of 1) the average (x-axis) and 2) the minimum (y-axis) of HC_{first} across the subarray's rows. For each manufacturer, we annotate a dashed line that fits to the data via linear regression with the specified R^2-score [150].

Obsv. 15. *The most vulnerable DRAM row in a subarray is significantly more vulnerable than the other rows in the subarray.*

We make two observations from Fig. 14. First, the average HC_{first} across all rows in a subarray is on the order of 2× the most vulnerable row's HC_{first}, i.e., the minimum HC_{first}. Therefore, the most vulnerable row in a subarray is *significantly* more vulnerable than the other rows in the same subarray. Second, this relation between the minimum and average HC_{first} values is similar across subarrays from different modules from the same manufacturer, and thus can be modeled using a linear regression. For example, the minimum HC_{first} value in a subarray from Mfr. C can be estimated using a well-fitting linear model with a R^2-score of 0.93. This observation is important because it indicates an underlying relationship between the average and minimum HC_{first} values across subarrays. For example, although subarrays in module C0 have significantly larger HC_{first} values than subarrays from module C3, a the linear model accurately expresses the relationship between both subarray's minimum and average HC_{first} values. Therefore, given a module from Mfr. C, the data shows that it may be possible to predict the minimum (worst-case) HC_{first} values of another module's subarrays, given the average HC_{first} values of those subarrays.

We conclude from these two observations that 1) the most vulnerable DRAM row in a subarray is significantly more vulnerable than the other rows in the subarray and 2) the worst-case HC_{first} in a subarray can be predicted based on the average HC_{first} values and the linear models we provide.

To analyze and quantify the similarity between the RowHammer vulnerability of different subarrays, we statistically compare each subarray against all other subarrays from the same manufacturer. To compare two given subarrays, we first compare their HC_{first} distributions using Bhattacharyya distance (BD) [11], which is used to measure the similarity of two statistical distributions. Second, for each pair of subarrays (S_A and S_B), we normalize BD to the BD between the first subarray S_A and itself: $BD_{norm} = BD(S_A, S_B)/BD(S_A, S_A)$. Therefore, BD_{norm} is 1.0 if two distributions are identical, while BD_{norm} value gets farther from 1.0 as the variation across two distributions increases. Fig. 15 shows the cumulative distribution of BD_{norm} values for subarray pairs from 1) the same DRAM module and 2) different DRAM modules. We annotate P5, P95, and the central P90 of the total population (y-axis) to show the range of BD_{norm} values in common-case.

Obsv. 16. *HC_{first} distributions of subarrays within a DRAM module exhibit significantly more similarity to each other than HC_{first} distributions of subarrays from different modules.*

We observe that, when both S_A and S_B are from the same DRAM module (orange curves), the central 90th percentile (i.e., between 5%

[11] We verify this for some of our chips by performing 1) single-sided RowHammer attack tests [69, 70] that induce bit flips in both rows adjacent to the aggressor row if the aggressor row is *not* at the edge of a subarray and 2) RowClone tests [27, 103, 128] that can successfully copy data only between two rows within the same subarray.

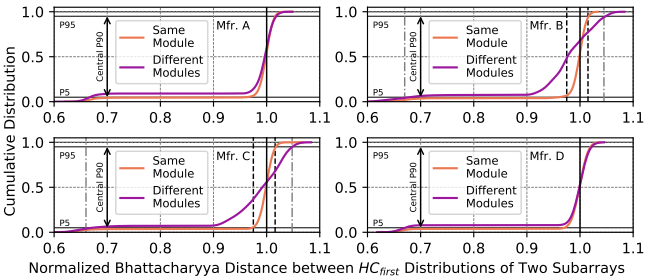

Fig. 15: Cumulative distribution of normalized Bhattacharyya distance values between HC_{first} distributions of different subarrays from 1) the same DRAM module and 2) different DRAM modules

and 95% of the population, as marked in Fig. 15) of all subarray pairs exhibit BD_{norm} values close to 1.0 (e.g., $BD_{norm} = 0.975$ at the 5th percentile for Mfr. C), which means that their HC_{first} distributions are very similar. In contrast, BD_{norm} values from different modules (purple curves) show a significantly wider distribution, especially for Mfrs. B and C (e.g., $BD_{norm} = 0.66$ at the 5th percentile for Mfr. C). From this analysis, we conclude that the HC_{first} distribution within a subarray can be representative of other subarrays from the same DRAM module (e.g., Mfrs. B and C), while the HC_{first} distribution within a subarray is often not representative of that of other subarrays for different DRAM modules.

Obsvs. 15 and 16 can be useful for improving DRAM profiling techniques and RowHammer defense mechanisms (§8.2).

Takeaway 6. *HC_{first} distribution in a subarray 1) contains a diverse set of values and 2) is similar to other subarrays in the same DRAM module.*

7.4 Circuit-level Justification

We observe that RowHammer vulnerability significantly varies across DRAM rows, columns, and chips, while different subarrays in the same chip exhibit similar vulnerability characteristics.

Variation across rows, columns, and chips. We hypothesize that two distinct factors cause the variation in RowHammer vulnerability that we observe across rows, columns, and chips.

First, *manufacturing process variation* causes differences in cell size and bitline/wordline impedance values, which introduces variation in cell reliability characteristics within and across DRAM chips [16, 19, 67, 68, 77, 83, 84, 103, 104, 112]. We hypothesize that similar imperfections in the manufacturing process (e.g., variation in cell-to-cell and cell-to-wordline spacings) cause RowHammer vulnerability to vary between cells in different DRAM chips.

Second, *design-induced variation* causes cell access latency characteristics to vary deterministically based on a cell's physical location in the memory chip (e.g., its proximity to I/O circuitry) [66, 76]. In particular, prior work [76] shows that columns closer to wordline drivers (which are typically distributed along a row) can be accessed faster. Similarly, we hypothesize that columns that are closer to repeating analog circuit elements (e.g., wordline drivers, voltage boosters) more sensitive to RowHammer disturbance than columns that are farther away from such elements.

Similarity across subarrays. Prior works [66, 76] demonstrate similar DRAM access latency characteristics across different subarrays. This is because a cell's access latency is dominated by its physical distance from the peripheral structures (e.g., local sense amplifiers and wordline drivers) within the subarray [16, 18, 66, 76–78, 135, 144], causing corresponding cells in *different* subarrays to exhibit *similar* access latency characteristics. We hypothesize that different subarrays in a DRAM chip exhibit similar RowHammer vulnerability characteristics for a similar reason. We leave further analysis and validation of these hypotheses for future work.

8 IMPLICATIONS

The observations we make in §5-§7 can be leveraged for both 1) crafting more effective RowHammer attacks and 2) developing more effective and more efficient RowHammer defenses.

8.1 Potential Attack Improvements

Our new observations and characterization data can help improve the success probability of a RowHammer attack. We propose three attack improvements based on our analyses of temperature (§5), aggressor row active time (§6), and spatial variation (§7).

Improvement 1. Obsvs. 1–3 can be used to craft more effective RowHammer attacks where the attacker can control or monitor the DRAM temperature. Obsvs. 1–3 show that a DRAM cell is more vulnerable to RowHammer within a specific temperature range. An attacker that can monitor the DRAM temperature (e.g., a malicious employee in a datacenter or an attacker who performs a remote RowHammer attack [82, 139] on a physically accessible IoT device) can increase the chance of a bit flip in two ways. First, the attacker can force the sensitive data to be stored in the DRAM cells that are more vulnerable at the current operating temperature, using known techniques [32, 118]. Second, the attacker can heat up or cool down the chip to a temperature level at which the cells that store sensitive data become more vulnerable to RowHammer. As a result, the attacker can significantly reduce the hammer count, and consequently, the attack time, necessary to cause a bit flip, thereby reducing the probability of being detected. For example, without our observations, an attacker might choose an aggressor row based on an *uninformed* decision with respect to temperature characteristics. In such a case, the chosen row could require a hammer count larger than 100K (Fig. 11). However, by leveraging our Obsvs. 1–3, an attacker can make a more *informed* decision and choose a row whose HC_{first} reduces by 50% (Fig. 5) at the temperature level the attack is designed to take place.

Improvement 2. Obsv. 3 can be used to enable a new RowHammer attack variant as a temperature-dependent trigger of the main attack (which could be a RowHammer attack, or some other security attack). Obsv. 3 demonstrates that some DRAM cells are vulnerable to RowHammer in a very narrow temperature range. To implement a temperature-dependent trigger using a RowHammer bit flip, an attacker can place the victim data in a row that contains a cell that flips at the target temperature, which allows the attacker to determine whether or not the target temperature is reached to trigger the main attack. This could be useful for an attacker in two scenarios: 1) to trigger the attack only when a precise temperature is reached (e.g., triggering an attack against an IoT device in the field when the device is heated or cooled), and 2) to identify abnormal operating conditions (e.g., triggering the attack during peak hours by using cells whose vulnerable temperature ranges are above the

common DRAM chip temperature). For example, to detect that the temperature of a DRAM chip is precisely 60 °C (above 60 °C) an attacker can use the cells with a vulnerable temperature range of 60 °C–60 °C (all ranges with lower limit equal or higher than 60 °C), which are 0.3%/0.3%/0.3%/0.2% (90.7%/86.3%/91.4%/91.7%) of all vulnerable cells in Mfrs. A/B/C/D (Fig. 3).

Improvement 3. Obsv. 8 shows that keeping an aggressor row active for a longer time results in more bit flips and lower HC_{first} values, which can be used to craft more powerful RowHammer attacks. For example, an attacker can increase the aggressor row active time by issuing more READ commands to the aggressor row, which can potentially 1) increase the number of bit flips for a given hammer count, or 2) defeat already-deployed RowHammer defenses [4, 23, 80, 108, 132, 134, 155, 156, 162] by inducing bit flips at a smaller hammer count than the HC_{first} value used for configuring a defense mechanism. For example, issuing 10 to 15 READ commands per aggressor row activation can increase the aggressor row active time by about 5×, increasing *BER* by 3.2×–10.2× or causing bits to flip at a hammer count that is 36% smaller than the HC_{first} value that may be used to configure a defense mechanism that does not consider our Observation 8.

8.2 Potential Defense Improvements

Our characterization data can potentially be used in five ways to improve RowHammer defense methods.

Improvement 1. Obsv. 12 shows that there is a large spatial variation in HC_{first} across rows. A system designer can leverage this observation to make existing RowHammer defense mechanisms more effective and efficient. A limitation of these mechanisms is that they are configured for the smallest *(worst-case)* HC_{first} across all rows in a DRAM bank even though an overwhelming majority of rows exhibit significantly larger HC_{first} values. This is an important limitation because, when configured for a smaller HC_{first} value, the performance, energy, and area overheads of many RowHammer defense mechanisms significantly increase [69, 108, 156]. To overcome this limitation, a system designer can configure a RowHammer defense mechanism to use different HC_{first} values for different DRAM rows. For example, BlockHammer's [156] and Graphene's [108] area costs can reach approximately 0.6% and 0.5% of a high-end processor's die area [156]. However, based on our Obsv. 12, 95% of DRAM rows exhibit an HC_{first} value greater than 2× the worst-case HC_{first}. Therefore, both BlockHammer and Graphene can be configured with the worst-case HC_{first} for only 5% of the rows and with 2× HC_{first} for the 95% of the rows, drastically reducing their area costs down to 0.4% and 0.1% of the processor die area, translating to 33% and 80% area cost reduction, respectively.[12] Similarly, the most area-efficient defense mechanism PARA [70] incurs 28% slowdown on average for benign workloads when configured for an HC_{first} of 1K [69]. This large performance overhead can be halved [69] for 95% of the rows by simply using lower probability thresholds for less vulnerable rows. We leave the comprehensive evaluation of such improvements to future work.

Improvement 2. Obsvs. 15 and 16 on *spatial variation* of HC_{first} across subarrays can be leveraged to reduce the time required to profile a given DRAM module's RowHammer vulnerability characteristics. This is an important challenge because profiling a DRAM module's RowHammer characteristics requires analyzing several environmental conditions and attack properties (e.g., data pattern, access pattern, and temperature), requiring time-consuming tests that lead to long profiling times [20, 26, 69, 70, 75, 106, 107, 109, 159]. According to our Obsvs. 15 and 16, characterizing a *small subset* of subarrays can provide approximate yet reliable profiling data for an *entire* DRAM chip. For example, assuming that a DRAM bank contains 128 subarrays, profiling eight randomly-chosen subarrays reduces RowHammer characterization time by at least an order of magnitude. This low-cost approximate profiling can be useful in two cases. First, finding the HC_{first} of a DRAM row requires performing a RowHammer test with varying hammer counts. Profiling the HC_{first} value for a few subarrays can be used to limit the HC_{first} search space for the rows in the rest of the subarrays based on our Obsv. 16. Second, one can profile a few subarrays within a DRAM module and use our linear regression models (Obsv. 16) to estimate the DRAM module's RowHammer vulnerability for systems whose reliability and security are not as critical (e.g., accelerators and systems running error-resilient workloads) [74, 87, 101, 102, 140].

Improvement 3. Obsvs. 1 and 3 show a vulnerable DRAM cell experiences bit flips at a particular temperature range. To improve a DRAM chip's reliability, the system might incorporate a mechanism to temporarily or permanently retire DRAM rows (e.g., via software page offlining [89] or hardware DRAM row remapping [15, 161]) that are vulnerable to RowHammer within a particular operating temperature range. To adapt to changes in temperature, the row retirement mechanism might dynamically adjust the rows that are retired, potentially leveraging previously-proposed techniques (e.g., Rowclone [128], LISA [18], NoM [121], FIGARO [146]) to efficiently move data between these rows.

Improvement 4. Obsv. 4 demonstrates that overall *BER* significantly increases with temperature across modules from three of the four manufacturers. To reduce the success probability of a RowHammer attack, a system designer can improve the cooling infrastructure for systems that use such DRAM modules. Doing so can reduce the number of RowHammer bit flips in a DRAM row. For example, when temperature drops from 90 °C to 50 °C, *BER* reduces by 25% on average across DRAM modules from Mfr. A. (see Fig. 4).

Improvement 5. Obsv. 8 shows that keeping an aggressor row active for a longer time increases the probability of RowHammer bit flips. Therefore, RowHammer defenses should take aggressor row active time into account. Unfortunately, monitoring the active time of all potential aggressor rows throughout an entire refresh window is not feasible for emerging lightweight on-DRAM-die RowHammer defense mechanisms [9, 23, 50, 53, 155], because such monitoring would require substantial storage and logic to track all potential aggressor rows' active times. To address this issue, the memory controller can be modified to limit or reduce the active times of all rows by changes to memory request scheduling algorithms and/or row buffer policies (e.g., via mechanisms similar to [31, 43, 58, 71, 93, 98, 99, 122, 136, 137, 156]). In this way, a RowHammer defense mechanism or the memory controller can inherently keep under control an aggressor row's active time. This is an example of a

[12] Our preliminary evaluation estimates BlockHammer's [156] and Graphene's [108] area costs for 2×HC_{first}, following the methodology described in BlockHammer [156].

system-DRAM cooperative scheme, similar to those recommended by prior work [69, 70, 94, 95, 104].

Improvement 6. Obsvs. 13 and 14 show that RowHammer vulnerability exhibits significant design-induced variation across columns within a chip and manufacturing process-induced variation across chips in a DRAM module. To make error correction codes (ECC) more effective and efficient at correcting RowHammer bit flips, a system designer can 1) design ECC schemes optimized for non-uniform bit error probability distributions across columns and 2) modify the chipkill ECC mechanism [22, 55, 85] to reduce a system's dependency on the most vulnerable DRAM chip, as proposed in a concurrent work, revisiting ECC for RowHammer [116].

9 RELATED WORK

This is the first work that rigorously and experimentally analyzes how RowHammer vulnerability changes with three fundamental properties: 1) DRAM chip temperature, 2) aggressor row active time, and 3) victim DRAM cell's physical location.

We divide prior work on RowHammer into four categories: 1) attacks, 2) defenses, 3) characterization of real DRAM chips, and 4) circuit-level simulation-based studies. Two works [95, 97] provide an overview of the RowHammer literature, and project the effect of increased RowHammer vulnerability in future DRAM chips and DRAM-based memory systems.

RowHammer Attacks and Defenses. Many works [1, 10, 13, 20, 21, 25, 26, 32, 33, 37, 41, 47, 54, 75, 82, 95, 97, 114, 115, 118, 125, 127, 138, 142, 143, 148, 151, 160, 164] exploit the RowHammer vulnerability to induce bit flips in main memory, as §2.3 explains. These works activate two (double-sided attack [69, 70, 127]) or more (many-sided attack [26]) aggressor rows, *as rapidly as possible*, aiming to maximize the number of RowHammer-induced bit flips. However, these works do *not* consider RowHammer's sensitivities to temperature, aggressor row active time, and spatial variation. Similarly, existing RowHammer defense mechanisms [2–7, 14, 23, 30, 35, 50–53, 57, 64, 70, 73, 80, 108, 132, 134, 143, 155, 156, 162] are not designed to account for these three properties. The new observations and insights we provide can be used to improve both RowHammer attacks and defenses, as §8 describes. We leave a full exploration of such attacks and defenses to future work, as our goal is to develop a fundamental understanding of RowHammer properties as opposed to developing new attacks and defenses.

Characterization of Real DRAM Chips. Two major works extensively characterize the RowHammer vulnerability using real DRAM chips [69, 70]. The original RowHammer work [70], published in 2014, 1) investigates the vulnerability of 129 commodity DDR3 DRAM modules to various RowHammer attack models, 2) demonstrates for the first time that RowHammer is a real problem for commodity DRAM chips, 3) characterizes RowHammer's sensitivity to refresh rate and activation rate in terms of *BER*, HC_{first}, and the physical distance between aggressor and victim rows, and 4) examines various potential solutions and proposes a new low-cost mitigation mechanism. The second work [69], published in 2020, conducts comprehensive scaling experiments on a wide range of 1580 DDR3, DDR4, and LPDDR4 commodity DRAM chips from different DRAM generations and technology nodes, clearly demonstrating that RowHammer has become an even more serious problem over DRAM generations. Even though these two works rigorously characterize various aspects of the RowHammer vulnerability in real DRAM chips, they do not analyze the effects of temperature, aggressor row active time, and victim DRAM cell's physical location on the RowHammer vulnerability. Our work complements and furthers the analyses of these two papers [69, 70] by 1) rigorously analyzing how these three properties affect the RowHammer vulnerability, and 2) providing new insights into crafting more effective and efficient RowHammer attacks and defenses.

Three other works [105–107] present *preliminary* experimental data from only three [105, 107] or five [106] DDR3 DRAM chips to build models that explain how the RowHammer vulnerability of DRAM cells varies with the three properties we analyze. Unfortunately, the experimental data provided by these works is not rigorous and conclusive enough due to 1) their extremely small sample set of DRAM cells, rows, and chips and 2) the lack of analysis of system-level implications. Our work, in contrast, 1) *rigorously* analyzes the effects of all three properties by testing a significantly larger set of 272 DRAM chips, and 2) provides insights into resulting RowHammer attack and defense improvements.

Simulation-based Studies. Prior works [28, 56, 119, 123, 145, 157–159] attempt to explain the error mechanisms that cause RowHammer bit flips through circuit-level simulations of capacitative-coupling and charge-trapping mechanisms, without testing real DRAM chips. These works, some of which we discuss in §5.3 and §6.3, are orthogonal to our experimental study.

10 CONCLUSION

This work provides the first study that experimentally analyzes the impact of DRAM chip temperature, aggressor row active time, and victim DRAM cell's physical location on RowHammer vulnerability, through extensive characterization of real DRAM chips. We rigorously characterize 248 DDR4 and 24 DDR3 modern DRAM chips from four major DRAM manufacturers using a carefully designed methodology and metrics, providing 16 key observations and 6 key takeaways. We highlight three major observations: 1) a DRAM cell experiences RowHammer bit flips at a bounded temperature range, 2) a DRAM row is more vulnerable to RowHammer when the aggressor row stays active for longer, and 3) a small fraction of DRAM rows are significantly more vulnerable to RowHammer than the other rows within a DRAM module. We describe and analyze how our insights can be used to improve both RowHammer attacks and defenses. We hope that the novel experimental results and insights of our study will inspire and aid future work to develop effective and efficient solutions to the RowHammer problem.

ACKNOWLEDGMENTS

We thank the anonymous reviewers of MICRO 2021 for feedback. We thank the SAFARI Research Group members for valuable feedback and the stimulating intellectual environment they provide. We acknowledge the generous gifts provided by our industrial partners: Google, Huawei, Intel, Microsoft, and VMware.

REFERENCES

[1] Misiker Tadesse Aga, Zelalem Birhanu Aweke, and Todd Austin. When Good Protections Go Bad: Exploiting Anti-DoS Measures to Accelerate Rowhammer Attacks. In *HOST*. 2017.

[2] Barbara Aichinger. DDR Memory Errors Caused by Row Hammer. In *HPEC*. 2015.

[3] Apple Inc. About the Security Content of Mac EFI Security Update 2015-001. https://support.apple.com/en-us/HT204934. June 2015.

[4] Zelalem Birhanu Aweke, Salessawi Ferede Yitbarek, Rui Qiao, Reetuparna Das, Matthew Hicks, Yossi Oren, et al. ANVIL: Software-Based Protection Against Next-Generation Rowhammer Attacks. In *ASPLOS*. 2016.

[5] Kuljit Bains, John Halbert, Christopher Mozak, Theodore Schoenborn, and Zvika Greenfield. Row Hammer Refresh Command. U.S. Patent 9,117,544. 2015.

[6] Kuljit S Bains and John B Halbert. Distributed Row Hammer Tracking. U.S. Patent 9,299,400. 2016.

[7] Kuljit S Bains and John B Halbert. Row Hammer Monitoring Based on Stored Row Hammer Threshold Value. U.S. Patent 9,384,821. 2016.

[8] Alessandro Barenghi, Luca Breveglieri, Niccolò Izzo, and Gerardo Pelosi. Software-Only Reverse Engineering of Physical DRAM Mappings for Rowhammer Attacks. In *IVSW*. 2018.

[9] Tanj Bennett, Stefan Saroiu, Alec Wolman, and Lucian Cojocar. Panopticon: A Complete In-DRAM Rowhammer Mitigation. In *Workshop on DRAM Security (DRAMSec)*. 2021.

[10] Sarani Bhattacharya and Debdeep Mukhopadhyay. Curious Case of Rowhammer: Flipping Secret Exponent Bits Using Timing Analysis. In *CHES*. 2016.

[11] Anil Bhattacharyya. On a Measure of Divergence between Two Statistical Populations Defined by Their Probability Distributions. *Bull. Calcutta Math. Soc.* 1943.

[12] Raj Chandra Bose and Dwijendra K Ray-Chaudhuri. On a Class of Error Correcting Binary Group Codes. *Information and control*. 1960.

[13] Erik Bosman, Kaveh Razavi, Herbert Bos, and Cristiano Giuffrida. Dedup Est Machina: Memory Deduplication as An Advanced Exploitation Vector. In *S&P*. 2016.

[14] Ferdinand Brasser, Lucas Davi, David Gens, Christopher Liebchen, and Ahmad-Reza Sadeghi. Can't Touch This: Software-Only Mitigation Against Rowhammer Attacks Targeting Kernel Memory. In *USENIX Security*. 2017.

[15] John Carter, Wilson Hsieh, Leigh Stoller, Mark Swanson, Lixin Zhang, Erik Brunvand, et al. Impulse: Building a Smarter Memory Controller. In *HPCA*. 1999.

[16] Kevin K Chang, Abhijith Kashyap, Hasan Hassan, Saugata Ghose, Kevin Hsieh, Donghyuk Lee, et al. Understanding Latency Variation in Modern DRAM Chips: Experimental Characterization, Analysis, and Optimization. In *SIGMETRICS*. 2016.

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

[18] Kevin K Chang, Prashant J Nair, Donghyuk Lee, Saugata Ghose, Moinuddin K Qureshi, and Onur Mutlu. Low-Cost Inter-Linked Subarrays (LISA): Enabling Fast Inter-Subarray Data Movement in DRAM. In *HPCA*. 2016.

[19] Kevin K Chang, A Giray Yağlıkçı, Saugata Ghose, Aditya Agrawal, Niladrish Chatterjee, Abhijith Kashyap, et al. Understanding Reduced-Voltage Operation in Modern DRAM Devices: Experimental Characterization, Analysis, and Mechanisms. In *SIGMETRICS*. 2017.

[20] Lucian Cojocar, Jeremie Kim, Minesh Patel, Lillian Tsai, Stefan Saroiu, Alec Wolman, et al. Are We Susceptible to Rowhammer? An End-to-End Methodology for Cloud Providers. In *S&P*. 2020.

[21] Lucian Cojocar, Kaveh Razavi, Cristiano Giuffrida, and Herbert Bos. Exploiting Correcting Codes: On the Effectiveness of ECC Memory Against Rowhammer Attacks. In *S&P*. 2019.

[22] Timothy J Dell. A White Paper on the Benefits of Chipkill-Correct ECC for PC Server Main Memory. *IBM Microelectronics Division*. 1997.

[23] Fabrice Devaux and Renaud Ayrignac. Method and Circuit for Protecting a DRAM Memory Device from the Row Hammer Effect. 10,885,966. 2021.

[24] Brian Everitt. *DRAM Circuit Design: Fundamental and High-Speed Topics*. Cambridge University Press. 1998.

[25] Pietro Frigo, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi. Grand Pwning Unit: Accelerating Microarchitectural Attacks with the GPU. In *S&P*. 2018.

[26] Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, et al. TRRespass: Exploiting the Many Sides of Target Row Refresh. In *S&P*. 2020.

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

[28] SK Gautam, SK Manhas, Arvind Kumar, Mahendra Pakala, and Ellie Yieh. Row Hammering Mitigation Using Metal Nanowire in Saddle Fin DRAM. *IEEE TED*. 2019.

[29] Saugata Ghose, A. Giray Yaglikci, Raghav Gupta, Donghyuk Lee, Kais Kudrolli, William Liu, et al. What Your DRAM Power Models Are Not Telling You: Lessons from a Detailed Experimental Study. In *SIGMETRICS*. 2018.

[30] H. Gomez, A. Amaya, and E. Roa. DRAM Row-Hammer Attack Reduction Using Dummy Cells. In *NORCAS*. 2016.

[31] Sven Goossens, Benny Akesson, and Kees Goossens. Conservative Open-Page Policy for Mixed Time-Criticality Memory Controllers. In *DATE*. 2013.

[32] Daniel Gruss, Moritz Lipp, Michael Schwarz, Daniel Genkin, Jonas Juffinger, Sioli O'Connell, et al. Another Flip in the Wall of Rowhammer Defenses. In *S&P*. 2018.

[33] Daniel Gruss, Clémentine Maurice, and Stefan Mangard. Rowhammer.js: A Remote Software-Induced Fault Attack in Javascript. arXiv:1507.06955 [cs.CR]. 2016.

[34] Richard W Hamming. Error Detecting and Error Correcting Codes. *The Bell system technical journal*. 1950.

[35] H. Hassan, M. Patel, J. S. Kim, A. G. Yağlıkçı, N. Vijaykumar, N. Mansouri Ghiasi, et al. CROW: A Low-Cost Substrate for Improving DRAM Performance, Energy Efficiency, and Reliability. In *ISCA*. 2019.

[36] Hasan Hassan, Gennady Pekhimenko, Nandita Vijaykumar, Vivek Seshadri, Donghyuk Lee, Oguz Ergin, et al. ChargeCache: Reducing DRAM Latency by Exploiting Row Access Locality. In *HPCA*. 2016.

[37] 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.

[38] Hasan Hassan, Nandita Vijaykumar, Samira Khan, Saugata Ghose, Kevin Chang, Gennady Pekhimenko, et al. SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies. In *HPCA*. 2017.

[39] Alexis Hocquenghem. Codes Correcteurs d'Erreurs. *Chiffers*. 1959.

[40] Heike Hofmann, Hadley Wickham, and Karen Kafadar. Letter-Value Plots: Boxplots for Large Data. *Journal of Computational and Graphical Statistics* 26, 3. 2017. https://doi.org/10.1080/10618600.2017.1305277 arXiv:https://doi.org/10.1080/10618600.2017.1305277

[41] Sanghyun Hong, Pietro Frigo, Yiğitcan Kaya, Cristiano Giuffrida, and Tudor Dumitraş. Terminal Brain Damage: Exposing the Graceless Degradation in Deep Neural Networks Under Hardware Fault Attacks. In *USENIX Security*. 2019.

[42] Masashi Horiguchi. Redundancy Techniques for High-Density DRAMs. In *ISIS*. 1997.

[43] Dandan Huan, Zusong Li, Weiwu Hu, and Zhiyong Liu. Processor Directed Dynamic Page Policy. In *ACSAC*. 2006.

[44] Micron Technology Inc. 2017. *ECC Brings Reliability and Power Efficiency to Mobile Devices*. Technical Report. Micron Technology Inc.

[45] Engin Ipek, Onur Mutlu, José F Martínez, and Rich Caruana. Self-Optimizing Memory Controllers: A Reinforcement Learning Approach. In *ISCA*. 2008.

[46] Kiyoo Itoh. *VLSI Memory Chip Design*. Springer. 2001.

[47] Yeongjin Jang, Jaehyuk Lee, Sangho Lee, and Taesoo Kim. SGX-Bomb: Locking Down the Processor via Rowhammer Attack. In *SOSP*. 2017.

[48] JEDEC. *EIA/JESD51-1: Integrated Circuits Thermal Measurement Method - Electrical Test Method (Single Semiconductor Device)*. 1995.

[49] JEDEC. *JESD79-3: DDR3 SDRAM Standard*. 2012.

[50] JEDEC. *JESD209-5A: LPDDR5 SDRAM Standard*. 2020.

[51] JEDEC. *JESD235C: High Bandwidth Memory (HBM) DRAM*. 2020.

[52] JEDEC. *JESD79-4C: DDR4 SDRAM Standard*. 2020.

[53] JEDEC. *JESD79-5: DDR5 SDRAM Standard*. 2020.

[54] Sangwoo Ji, Youngjoo Ko, Saeyoung Oh, and Jong Kim. Pinpoint Rowhammer: Suppressing Unwanted Bit Flips on Rowhammer Attacks. In *ASIACCS*. 2019.

[55] Xun Jian and Rakesh Kumar. Adaptive Reliability Chipkill Correct (ARCC). In *HPCA*. 2013.

[56] Yichen Jiang, Huifeng Zhu, Dean Sullivan, Xiaolong Guo, Xuan Zhang, and Yier Jin. Quantifying RowHammer Vulnerability for DRAM Security. In *DAC*. 2021.

[57] Ingab Kang, Eojin Lee, and Jung Ho Ahn. CAT-TWO: Counter-Based Adaptive Tree, Time Window Optimized for DRAM Row-Hammer Prevention. *IEEE Access*. 2020.

[58] Dimitris Kaseridis, Jeffrey Stuecheli, and Lizy Kurian John. Minimalist Open-Page: A DRAM Page-Mode Scheduling Policy for the Many-Core Era. In *MICRO*. 2011.

[59] B. Keeth and R.J. Baker. *DRAM Circuit Design: A Tutorial*. Wiley. 2001.

[60] 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.

[61] Samira Khan, Donghyuk Lee, and Onur Mutlu. PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM. In *DSN*. 2016.

[62] 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.

[63] 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.

[64] Dae-Hyun Kim, Prashant J Nair, and Moinuddin K Qureshi. Architectural Support for Mitigating Row Hammering in DRAM Memories. *CAL*. 2014.

[65] Jungrae Kim, Michael Sullivan, Sangkug Lym, and Mattan Erez. All-Inclusive ECC: Thorough End-to-End Protection for Reliable Computer memory. In *ISCA*. 2016.

[66] 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.
[67] 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.
[68] 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.
[69] Jeremie S. Kim, Minesh Patel, Abdullah Giray Yağlıkçı, Hasan Hassan, Roknoddin Azizi, Lois Orosa, et al. Revisiting RowHammer: An Experimental Analysis of Modern Devices and Mitigation Techniques. In *ISCA*. 2020.
[70] Y. Kim, R. Daly, J. Kim, C. Fallin, J. H. Lee, D. Lee, et al. Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors. In *ISCA*. 2014.
[71] Yoongu Kim, Dongsu Han, Onur Mutlu, and Mor Harchol-Balter. ATLAS: A Scalable and High-Performance Scheduling Algorithm for Multiple Memory Controllers. In *HPCA*. 2010.
[72] Yoongu Kim, Vivek Seshadri, Donghyuk Lee, Jamie Liu, Onur Mutlu, Yoongu Kim, et al. A Case for Exploiting Subarray-Level Parallelism (SALP) in DRAM. In *ISCA*. 2012.
[73] Radhesh Krishnan Konoth, Marco Oliverio, Andrei Tatar, Dennis Andriesse, Herbert Bos, Cristiano Giuffrida, et al. ZebRAM: Comprehensive and Compatible Software Protection Against Rowhammer Attacks. In *OSDI*. 2018.
[74] Skanda Koppula, Lois Orosa, A Giray Yağlıkçı, Roknoddin Azizi, Taha Shahroodi, Konstantinos Kanellopoulos, et al. EDEN: Enabling Energy-Efficient, High-Performance Deep Neural Network Inference Using Approximate DRAM. In *MICRO*. 2019.
[75] Andrew Kwong, Daniel Genkin, Daniel Gruss, and Yuval Yarom. RAMBleed: Reading Bits in Memory Without Accessing Them. In *S&P*. 2020.
[76] Donghyuk Lee, Samira Khan, Lavanya Subramanian, Saugata Ghose, Rachata Ausavarungnirun, Gennady Pekhimenko, et al. Design-Induced Latency Variation in Modern DRAM Chips: Characterization, Analysis, and Latency Reduction Mechanisms. In *SIGMETRICS*. 2017.
[77] Donghyuk Lee, Yoongu Kim, Gennady Pekhimenko, Samira Khan, Vivek Seshadri, Kevin Chang, et al. Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case. In *HPCA*. 2015.
[78] Donghyuk Lee, Yoongu Kim, Vivek Seshadri, Jamie Liu, Lavanya Subramanian, and Onur Mutlu. Tiered-Latency DRAM: A Low Latency and Low Cost DRAM Architecture. In *HPCA*. 2013.
[79] Donghyuk Lee, Lavanya Subramanian, Rachata Ausavarungnirun, Jongmoo Choi, and Onur Mutlu. Decoupled Direct Memory Access: Isolating CPU and IO Traffic by Leveraging a Dual-Data-Port DRAM. In *PACT*. 2015.
[80] Eojin Lee, Ingab Kang, Sukhan Lee, G Edward Suh, and Jung Ho Ahn. TWiCe: Preventing Row-Hammering by Exploiting Time Window Counters. In *ISCA*. 2019.
[81] J. Lee. Green Memory Solution. Investor's Forum. 2014.
[82] Moritz Lipp, Misiker Tadesse Aga, Michael Schwarz, Daniel Gruss, Clémentine Maurice, Lukas Raab, et al. Nethammer: Inducing Rowhammer Faults Through Network Requests. arXiv:1805.04956 [cs.CR]. 2018.
[83] Jamie Liu, Ben Jaiyen, Yoongu Kim, Chris Wilkerson, Onur Mutlu, J Liu, et al. An Experimental Study of Data Retention Behavior in Modern DRAM Devices. In *ISCA*. 2013.
[84] Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu. RAIDR: Retention-Aware Intelligent DRAM Refresh. In *ISCA*. 2012.
[85] David Locklear. Chipkill Correct Memory Architecture. *Dell Enterprise Systems Group, Technology Brief*. 2000.
[86] Haocong Luo, Taha Shahroodi, Hasan Hassan, Minesh Patel, Abdullah Giray Yaglikci, Lois Orosa, et al. CLR-DRAM: A Low-Cost DRAM Architecture Enabling Dynamic Capacity-Latency Trade-Off. In *ISCA*. 2020.
[87] Yixin Luo, Sriram Govindan, Bikash Sharma, Mark Santaniello, Justin Meza, Aman Kansal, et al. Characterizing Application Memory Error Vulnerability to Optimize Datacenter Cost via Heterogeneous-Reliability Memory. In *DSN*. 2014.
[88] Maxwell. FT20X User Manual. https://www.maxwell-fa.com/upload/files/base/8/m/311.pdf.
[89] 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.
[90] Micron. DDR4 SDRAM Datasheet. In *Micron*. 380. 2016.
[91] Micron Technology. TN-00-08: Thermal Applications. 2002.
[92] Micron Technology. SDRAM, 4Gb: x4, x8, x16 DDR4 SDRAM Features. 2014.
[93] Thomas Moscibroda and Onur Mutlu. Memory Performance Attacks: Denial of Memory Service in Multi-Core Systems. In *USENIX Security*. 2007.
[94] Onur Mutlu. Memory Scaling: A Systems Architecture Perspective. In *IMW*. 2013.
[95] Onur Mutlu. The RowHammer Problem and Other Issues We May Face as Memory Becomes Denser. In *DATE*. 2017.
[96] Onur Mutlu. RowHammer. https://people.inf.ethz.ch/omutlu/pub/onur-Rowhammer-TopPicksinHardwareEmbeddedSecurity-November-8-2018.pdf. Top Picks in Hardware and Embedded Security. 2018.
[97] Onur Mutlu and Jeremie S Kim. RowHammer: A Retrospective. *TCAD*. 2019.
[98] Onur Mutlu and Thomas Moscibroda. Stall-Time Fair Memory Access Scheduling for Chip Multiprocessors. In *MICRO*. 2007.
[99] Onur Mutlu and Thomas Moscibroda. Parallelism-Aware Batch Scheduling: Enhancing Both Performance and Fairness of Shared DRAM Systems. In *ISCA*. 2008.
[100] Prashant J Nair, Vilas Sridharan, and Moinuddin K Qureshi. XED: Exposing on-die error detection information for strong memory reliability. In *ISCA*. 2016.
[101] Duy-Thanh Nguyen, Nhut-Minh Ho, and Ik-Joon Chang. St-DRC: Stretchable DRAM Refresh Controller with No Parity-overhead Error Correction Scheme for Energy-efficient DNNs. In *DAC*. 2019.
[102] Duy Thanh Nguyen, Hyun Kim, Hyuk-Jae Lee, and Ik-Joon Chang. An Approximate Memory Architecture for a Reduction of Refresh Power Consumption in Deep Learning Applications. In *ISCAS*. 2018.
[103] Ataberk Olgun, Minesh Patel, A Giray Yağlıkçı, Haocong Luo, Jeremie S Kim, Nisa Bostancı, et al. QUAC-TRNG: High-Throughput True Random Number Generation Using Quadruple Row Activation in Commodity DRAM Chips. In *ISCA*. 2021.
[104] Lois Orosa, Yaohua Wang, Mohammad Sadrosadati, Jeremie S Kim, Minesh Patel, Ivan Puddu, et al. CODIC: A Low-Cost Substrate for Enabling Custom In-DRAM Functionalities and Optimizations. In *ISCA*. 2021.
[105] Kyungbae Park, Sanghyeon Baeg, ShiJie Wen, and Richard Wong. Active-Precharge Hammering on a Row-Induced Failure in DDR3 SDRAMs Under 3x nm Technology. In *IIRW*. 2014.
[106] 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.
[107] Kyungbae Park, Donghyuk Yun, and Sanghyeon Baeg. Statistical Distributions of Row-Hammering Induced Failures in DDR3 Components. *Microelectronics Reliability*. 2016.
[108] Yeonhong Park, Woosuk Kwon, Eojin Lee, Tae Jun Ham, Jung Ho Ahn, and Jae W Lee. Graphene: Strong yet Lightweight Row Hammer Protection. In *MICRO*. 2020.
[109] Minesh Patel, Geraldo Francisco de Oliveira Jr., and Onur Mutlu. HARP: Practically and Effectively Identifying Uncorrectable Errors in Main Memory Chips That Use On-Die ECC. In *MICRO*. 2021.
[110] 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 (Best Paper). In *MICRO*. 2020.
[111] 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.
[112] 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.
[113] PCI Special Interest Group. PCI Express Base Specification Revision 1.0. 2003.
[114] Peter Pessl, Daniel Gruss, Clémentine Maurice, Michael Schwarz, and Stefan Mangard. DRAMA: Exploiting DRAM Addressing for Cross-CPU Attacks. In *USENIX Security*. 2016.
[115] Rui Qiao and Mark Seaborn. A New Approach for RowHammer Attacks. In *HOST*. 2016.
[116] Moinuddin Qureshi. Rethinking ECC in the Era of Row-Hammer. *DRAMSec*. 2021.
[117] M.K. Qureshi, Dae-Hyun Kim, S. Khan, P.J. Nair, and O. Mutlu. AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems. In *DSN*. 2015.
[118] Kaveh Razavi, Ben Gras, Erik Bosman, Bart Preneel, Cristiano Giuffrida, and Herbert Bos. Flip Feng Shui: Hammering a Needle in the Software Stack. In *USENIX Security*. 2016.
[119] Michael Redeker, Bruce F Cockburn, and Duncan G Elliott. An Investigation into Crosstalk Noise in DRAM Structures. In *MTDT*. 2002.
[120] Irving S Reed and Gustave Solomon. Polynomial Codes over Certain Finite Fields. *SIAM*. 1960.
[121] Seyyed Hossein SeyyedAghaei Rezaei, Mehdi Modarressi, Rachata Ausavarungnirun, Mohammad Sadrosadati, Onur Mutlu, and Masoud Daneshtalab. NoM: Network-on-Memory for Inter-Bank Data Transfer in Highly-Banked Memories. *CAL*. 2020.
[122] Scott Rixner, William J. Dally, Ujval J. Kapasi, Peter Mattson, and John D. Owens. Memory Access Scheduling. In *ISCA*. 2000.
[123] Seong-Wan Ryu, Kyungkyu Min, Jungho Shin, Heimi Kwon, Donghoon Nam, Taekyung Oh, et al. Overcoming the Reliability Limitation in the Ultimately Scaled DRAM using Silicon Migration Technique by Hydrogen Annealing. In *IEDM*. 2017.
[124] SAFARI Research Group. A Deeper Look into RowHammer's Sensitivities — GitHub Repository. https://github.com/CMU-SAFARI/deeperrowhammer.

[125] SAFARI Research Group. RowHammer — GitHub Repository. https://github.com/CMU-SAFARI/rowhammer.
[126] SAFARI Research Group. SoftMC — GitHub Repository. https://github.com/CMU-SAFARI/softmc.
[127] Mark Seaborn and Thomas Dullien. Exploiting the DRAM Rowhammer Bug to Gain Kernel Privileges. *Black Hat*. 2015.
[128] Vivek Seshadri, Yoongu Kim, Chris Fallin, Donghyuk Lee, Rachata Ausavarungnirun, Gennady Pekhimenko, et al. RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization. In *MICRO*. 2013.
[129] Vivek Seshadri, Donghyuk Lee, Thomas Mullins, Hasan Hassan, Amirali Boroumand, Jeremie Kim, et al. Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology. In *MICRO*. 2017.
[130] Vivek Seshadri, Thomas Mullins, Amirali Boroumand, Onur Mutlu, Phillip B Gibbons, Michael A Kozuch, et al. Gather-Scatter DRAM: In-DRAM Address Translation to Improve the Spatial Locality of Non-Unit Strided Accesses. In *MICRO*. 2015.
[131] Vivek Seshadri and Onur Mutlu. In-DRAM Bulk Bitwise Execution Engine. *arXiv:1905.09822*. 2019.
[132] S. M. Seyedzadeh, A. K. Jones, and R. Melhem. Mitigating Wordline Crosstalk Using Adaptive Trees of Counters. In *ISCA*. 2018.
[133] 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.
[134] Mungyu Son, Hyunsun Park, Junwhan Ahn, and Sungjoo Yoo. Making DRAM Stronger Against Row Hammering. In *DAC*. 2017.
[135] Young Hoon Son, O Seongil, Yuhwan Ro, Jae W Lee, and Jung Ho Ahn. Reducing Memory Access Latency with Asymmetric DRAM Bank Organizations. In *ISCA*. 2013.
[136] Lavanya Subramanian, Donghyuk Lee, Vivek Seshadri, Harsha Rastogi, and Onur Mutlu. The Blacklisting Memory Scheduler: Achieving High Performance and Fairness at Low Cost. In *ICCD*. 2014.
[137] Lavanya Subramanian, Donghyuk Lee, Vivek Seshadri, Harsha Rastogi, and Onur Mutlu. BLISS: Balancing Performance, Fairness and Complexity in Memory Access Scheduling. *TPDS*. 2016.
[138] Andrei Tatar, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi. Defeating Software Mitigations Against Rowhammer: A Surgical Precision Hammer. In *RAID*. 2018.
[139] Andrei Tatar, Radhesh Krishnan Konoth, Elias Athanasopoulos, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi. Throwhammer: Rowhammer Attacks Over the Network and Defenses. In *USENIX ATC*. 2018.
[140] Fengbin Tu, Weiwei Wu, Shouyi Yin, Leibo Liu, and Shaojun Wei. RANA: Towards Efficient Neural Acceleration with Refresh-Optimized Embedded DRAM. In *ISCA*. 2018.
[141] John Tukey. *Exploratory Data Analysis*. Addison–Wesley. 1977.
[142] Victor van der Veen, Yanick Fratantonio, Martina Lindorfer, Daniel Gruss, Clementine Maurice, Giovanni Vigna, et al. Drammer: Deterministic Rowhammer Attacks on Mobile Platforms. In *CCS*. 2016.
[143] Victor van der Veen, Martina Lindorfer, Yanick Fratantonio, Harikrishnan Padmanabha Pillai, Giovanni Vigna, Christopher Kruegel, et al. GuardION: Practical Mitigation of DMA-Based Rowhammer Attacks on ARM. In *DIMVA*. 2018.
[144] Thomas Vogelsang. Understanding the Energy Consumption of Dynamic Random Access Memories. In *MICRO*. 2010.
[145] Andrew J. Walker, Sungkwon Lee, and Dafna Beery. On DRAM RowHammer and the Physics on Insecurity. *IEEE TED*. 2021.
[146] Yaohua Wang, Lois Orosa, Xiangjun Peng, Yang Guo, Saugata Ghose, Minesh Patel, et al. FIGARO: Improving System Performance via Fine-Grained In-DRAM Data Relocation and Caching. In *MICRO*. 2020.
[147] Yaohua Wang, Arash Tavakkol, Lois Orosa, Saugata Ghose, Nika Mansouri Ghiasi, Minesh Patel, et al. Reducing DRAM Latency via Charge-Level-Aware Look-Ahead Partial Restoration. In *MICRO*. 2018.
[148] Zane Weissman, Thore Tiemann, Daniel Moghimi, Evan Custodio, Thomas Eisenbarth, and Berk Sunar. JackHammer: Efficient Rowhammer on Heterogeneous FPGA–CPU Platforms. arXiv:1912.11523 [cs.CR]. 2020.
[149] Wikipedia. RS-485. https://en.wikipedia.org/wiki/RS-485.
[150] Sewall Wright. Correlation and Causation. *Agricultural Research*. 1921.
[151] Yuan Xiao, Xiaokuan Zhang, Yinqian Zhang, and Radu Teodorescu. One Bit Flips, One Cloud Flops: Cross-VM Row Hammer Attacks and Privilege Escalation. In *USENIX Security*. 2016.
[152] Xilinx. ML605 Reference Design User Guide. https://www.xilinx.com/content/dam/xilinx/support/documentation/boards_and_kits/ug535.pdf.
[153] Xilinx. UltraScale Architecture-Based FPGAs Memory IP v1.4. https://www.xilinx.com/support/documentation/ip_documentation/ultrascale_memory_ip/v1_4/pg150-ultrascale-memory-ip.pdf.
[154] Xilinx. Xilinx Alveo U200 FPGA Board. https://www.xilinx.com/products/boards-and-kits/alveo/u200.html.
[155] A. Giray Yağlıkçı, Jeremie S. Kim, Fabrice Devaux, and Onur Mutlu. Security Analysis of the Silver Bullet Technique for RowHammer Prevention. arXiv:cs.CR/2106.07084 2021.
[156] A Giray Yağlıkçı, Minesh Patel, Jeremie S. Kim, Roknoddin Azizibarzoki, Ataberk Olgun, Lois Orosa, et al. BlockHammer: Preventing RowHammer at Low Cost by Blacklisting Rapidly-Accessed DRAM Rows. In *HPCA*. 2021.
[157] Chia Yang, Chen Kang Wei, Yu Jing Chang, Tieh Chiang Wu, Hsiu Pin Chen, and Chao Sung Lai. Suppression of RowHammer Effect by Doping Profile Modification in Saddle-Fin Array Devices for Sub-30-nm DRAM Technology. *TDMR*. 2016.
[158] Chia-Ming Yang, Chen-Kang Wei, Hsiu-Pin Chen, Jian-Shing Luo, Yu Jing Chang, Tieh-Chiang Wu, et al. Scanning Spreading Resistance Microscopy for Doping Profile in Saddle-Fin Devices. *IEEE Transactions on Nanotechnology*. 2017.
[159] Thomas Yang and Xi-Wei Lin. Trap-Assisted DRAM Row Hammer Effect. *EDL*. 2019.
[160] Fan Yao, Adnan Siraj Rakin, and Deliang Fan. DeepHammer: Depleting the Intelligence of Deep Neural Networks Through Targeted Chain of Bit Flips. In *USENIX Security*. 2020.
[161] Leonid Yavits, Lois Orosa, Suyash Mahar, João Dinis Ferreira, Mattan Erez, Ran Ginosar, et al. WoLFRaM: Enhancing Wear-Leveling and Fault Tolerance in Resistive Memories Using Programmable Address Decoders. In *ICCD*. 2020.
[162] Jung Min You and Joon-Sung Yang. MRLoc: Mitigating Row-Hammering Based on Memory Locality. In *DAC*. 2019.
[163] Tao Zhang, Ke Chen, Cong Xu, Guangyu Sun, Tao Wang, and Yuan Xie. Half-DRAM: A High-Bandwidth and Low-Power DRAM Architecture from the Rethinking of Fine-Grained Activation. In *ISCA*. 2014.
[164] Zhi Zhang, Yueqiang Cheng, Dongxi Liu, Surya Nepal, Zhi Wang, and Yuval Yarom. PThammer: Cross-User-Kernel-Boundary Rowhammer through Implicit Accesses. In *MICRO*. 2020.