An Experimental Characterization of Combined RowHammer and RowPress Read Disturbance in Modern DRAM Chips

Haocong Luo İsmail Emir Yüksel Ataberk Olgun A. Giray Yağlıkçı
Mohammad Sadrosadati Onur Mutlu
ETH Zürich

DRAM read disturbance can break memory isolation, a fundamental property to ensure system robustness (i.e., reliability, security, safety). RowHammer and RowPress are two different DRAM read disturbance phenomena. RowHammer induces bitflips in physically adjacent victim DRAM rows by repeatedly opening and closing an aggressor DRAM row, while RowPress induces bitflips by keeping an aggressor DRAM row open for a long period of time. In this study, we characterize a DRAM access pattern that combines RowHammer and RowPress in 84 real DDR4 DRAM chips from all three major DRAM manufacturers. Our key results show that 1) this combined RowHammer and RowPress pattern takes significantly smaller amount of time (up to 46.1% faster) to induce the first bitflip compared to the state-of-the-art RowPress pattern, and 2) at the minimum aggressor row activation count to induce at least one bitflip, the bits that flip are different across RowHammer, RowPress, and the combined patterns. Based on our results, we provide a key hypothesis that the read disturbance effect caused by RowPress from one of the two aggressor rows in a double-sided pattern is much more significant than the other.

1. Introduction

Memory isolation is a fundamental property for system robustness (i.e., reliability, security, safety). Accesses to one memory location should *not* induce unintended side-effects on other (*unaccessed*) memory locations. Unfortunately, the prevalent main memory technology, dynamic random access memory (DRAM) [1], is vulnerable to *read disturbance* (i.e., accessing a DRAM cell disturbs the integrity of data in physically adjacent but unaccessed DRAM cells) that can violate memory isolation.

RowHammer [2–11] and RowPress [12] are two read disturbance phenomenon identified and demonstrated in commodity DRAM chips. RowHammer causes *bitflips* in a DRAM row (victim row) by repeatedly opening and closing a physically adjacent DRAM row (aggressor row). RowPress causes bitflips in the victim row by keeping the aggressor row open for a long period of time (i.e., having a longer aggressor row on time, t_{AggON}). Prior works [12, 13] show that RowHammer and RowPress have *different* underlying read-disturb mechanisms, and cause bitflips with *different* directionalities [12, 13].

Read disturbance is a critical vulnerability because attackers can leverage bitflips to perform privilege escalation and leak data [3, 4, 8, 14–66]. For robust (i.e., secure, safe, and reliable) operation of computing systems, it is critical to develop a comprehensive understanding of DRAM read disturbance.

In this paper, **our goal** is to experimentally characterize the bitflips caused by a DRAM access pattern that combines both RowHammer and RowPress. As Fig. 1 shows, this involves repeated activations of two aggressor rows (i.e., R0 and R2), where one aggressor row (R2) is open for only the minimal amount of time specified by the DRAM standard (i.e., RowHammer, $t_{AggON} = t_{RAS}$), while the other aggressor row (R0) is open for a longer period of time (i.e., RowPress, $t_{AggON} > t_{RAS}$).

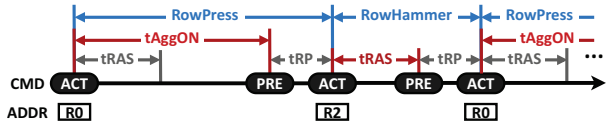

Figure 1: The combined RowHammer and RowPress pattern.

We characterize this combined pattern on 84 commodity DDR4 DRAM chips (from 14 DRAM modules) from all three major DRAM manufacturers and spans different die densities and die revisions. Our key characterization results demonstrate that 1) the combined pattern induces bitflips faster (up to 46.1%) than conventional RowPress patterns and with much fewer (up to 46.9%) aggressor row activations than conventional RowHammer patterns, and 2) induces different bitflips as t_{AggON} increases compared to conventional RowPress and RowHammer patterns. Based on our experimental results, we hypothesize that the read disturbance effect caused by RowPress from one of the two aggressor rows in a double-sided pattern is much more significant than the other.

We make the following contributions in this paper:

- To our knowledge, this is the first work to experimentally characterize the bitflips from a combined RowHammer and RowPress access pattern in real DRAM chips.
- We demonstrate the key differences of the bitflips caused by the combined RowHammer and RowPress pattern and conventional RowPress and RowHammer patterns.
- We provide insights into and hypotheses about the low-level failure mechanisms of RowHammer and RowPress.

2. Background

2.1. DRAM Organization

Fig. 2 illustrates the hierarchical organization of DRAM-based main memory. A *memory controller* communicates with one or more *memory ranks* over a *memory channel*. A rank consists of multiple DRAM *chips* that operates in lock-step. Inside a DRAM chip, there are multiple DRAM *banks* ❶ that can be accessed independently. In a bank, multiple DRAM *cells* ❸ are organized into a 2D array. A DRAM cell stores one bit of information in the form of electrical charge in the *capacitor*, connected to a *bitline* through an *access transistor* controlled by a *wordline*.

DRAM cells are accessed at *row* ❷ granularity. When activated, the wordline of a row is driven high, enabling all the

2833-292X/24/$31.00 ©2024 IEEE
DOI 10.1109/DSN-S60304.2024.00013

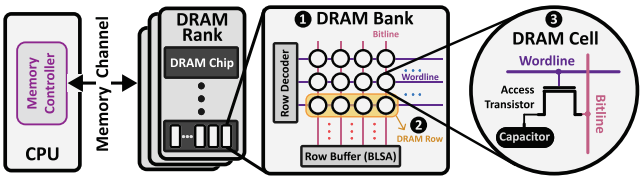

Figure 2: Hierarchical organization of modern DRAM. Reproduced from [7, 12].

access transistors of the DRAM cells in the row that connect the capacitors to their respective bitlines. The bitlines are connected to the row buffer, which is used to read from or write to the DRAM cells.

2.2. Key DRAM Operation and Timing

DRAM Access. To access DRAM, the memory controller first sends an Activate (ACT) command to a DRAM bank, which *opens* a DRAM row (i.e., places the row's contents into the row buffer). Second, the memory controller sends Read/Write commands (RD/WR) to the DRAM cells in the opened row. Third, to access another row in the same bank, the memory controller sends a Precharge (PRE) command that *closes* the currently open row. The DRAM row must remain open (closed) for at least t_{RAS} (t_{RP}) amount of time.

DRAM Refresh. DRAM needs to be *periodically refreshed* to ensure data integrity because the capacitors in DRAM cells lose charge over time [67, 68]. Therefore, the memory controller periodically (every t_{REFI}) sends refresh (REF) commands to DRAM to restore lost charge. The JEDEC DDR4 standard [69] specifies that $t_{REFI} = 7.8\mu s$ and each DRAM row must be refreshed once every $t_{REFW} = 64ms$ under normal operating conditions.

2.3. DRAM Read Disturbance

DRAM read disturbance is the phenomenon that accessing a DRAM row (i.e., the aggressor row) disturbs the charge stored in the DRAM cells in physically adjacent (*unaccessed*) DRAM rows (i.e., victim rows), causing *bitflips*.

RowHammer. RowHammer causes bitflips in victim rows through many (e.g., tens of thousands of) repeated openings and closings (activation & precharge) of the aggressor row [2–11]. In a RowHammer access pattern, each aggressor row is opened (closed) for the minimum amount of time specified by the DRAM standard (i.e., aggressor row on time $t_{AggON} = t_{RAS}$, and aggressor row off time $t_{AggOFF} = t_{RP}$).

RowPress. RowPress causes bitflips in victim rows by keeping the aggressor row open for a long period of time (i.e., $t_{AggON} > t_{RAS}$) [12]. Compared to RowHammer bitflips, as t_{AggON} increases, RowPress bitflips require (much) *fewer* aggressor row activations to induce and have an *opposite* direction compared to RowHammer [12].

3. Experimental Methodology

3.1. DRAM Characterization Infrastructure

We develop an FPGA-based commodity DRAM chip characterization infrastructure building on DRAM Bender [70, 71] and SoftMC [72, 73]. The infrastructure enables 1) fine-grained control over the DRAM commands and timings, and 2) stable temperature control[1] of the tested DRAM chips tested with heater pads controlled by a PID-based temperature controller [74].

We avoid potential interference to directly observe and analyze the bitflips from the circuit-level following a similar methodology used in prior works [5–7, 10–12, 52]. First, we do *not* send periodic REF commands to the DRAM under test to 1) keep the timings of our experiments precise, and 2) not trigger any on-die RowHammer mitigation mechanisms (e.g., target-row-refresh, TRR [46, 52]). Second, we make sure the runtime of each iteration of our characterization experiment does not exceed $60ms$ (strictly smaller than $t_{REFW} = 64ms$) to avoid any retention failure bitflips. Third, we do not implement rank-level ECC in our infrastructure and make sure that the DRAM chips we test do *not* have on-die ECC.

3.2. Commodity DDR4 DRAM Chips Tested

Table 1 describes the 84 (14) DRAM chips (modules) we test from all three major DRAM manufacturers (Mfr. S, H, and M). For each manufacturer, we test a variety of DRAM die densities and revisions. To account for row address remapping inside DRAM, we reverse-engineer the physical layout of the DRAM rows, following prior works' methodology [5–7, 10–12, 52].

Table 1: DDR4 DRAM Chips Tested.

Mfr.	#DIMMs	#Chips	Density	Die Rev.	Org.	Date
Mfr. S (Samsung)	1	8	8Gb	C	x8	N/A
	3	24	8Gb	D	x8	2110
	1	8	16Gb	A	x8	2212
Mfr. H (Hynix)	2	8	8Gb	D	x8	Mar. 21
	2	8	16Gb	C	x8	2136
Mfr. M (Micron)	1	4	4Gb	F	x16	N/A
	2	16	8Gb	B	x8	N/A
	1	4	16Gb	B	x16	2126
	1	4	16Gb	E	x16	2046

3.3. Combined RowHammer and RowPress Pattern

Fig. 3 shows command sequences and timings of the DRAM access patterns we characterize in this paper. Fig. 3.a shows the conventional single-sided RowPress pattern involving only one aggressor row (R0) that is open for t_{AggON} amount of time per activation. If $t_{AggON} = t_{RAS}$, then this pattern is identical to the conventional single-sided RowHammer pattern. Fig. 3.b shows the conventional double-sided RowPress pattern involving alternating activations to two aggressor rows (R0 and R2). Both R0 and R2 are open for t_{AggON} per activation. When $t_{AggON} = t_{RAS}$, this pattern is identical to the conventional double-sided RowHammer pattern. Fig. 3.c shows the combined RowHammer and RowPress access pattern (that is not explored in prior work). This pattern involves alternating activations to two aggressor rows (R0 and R2), but R0 is open for t_{AggON} ($> t_{RAS}$) amount of time and R2 is *always* open for t_{RAS}, the minimal amount of row open time allowed by the JEDEC standard.

3.4. Real DRAM Chip Characterization Methodology

For each DRAM module, we evaluate the test patterns on 3K DRAM rows in an arbitrarily chosen DRAM bank (1K rows at the beginning, middle, and end of the bank, respectively). We use a checkerboard data pattern that initializes the aggressor row(s) with 0xAA and the victim row(s) with 0x55. For each

[1]The maximum variation in temperature readings we observe over 24 hours is $\pm 0.2^{\circ}$C from the target temperature.

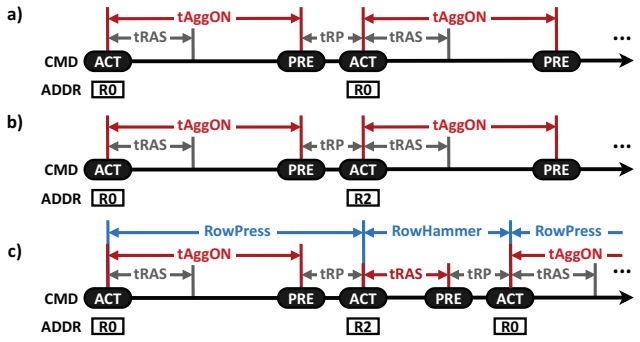

Figure 3: Comparison of a) the conventional single-sided RowPress (RowHammer, when $t_{AggON} = t_{RAS}$) pattern, b) the conventional double-sided RowPress (RowHammer, when $t_{AggON} = t_{RAS}$) pattern, and c) the combined RowHammer and RowPress pattern.

pattern, we sweep t_{AggON} from the minimum value of 36ns (i.e., $t_{AggON} = t_{RAS}$) up to 300μs. We repeat each experiment to measure the minimum number of total aggressor row activations to cause at least one bitflip (AC_{min}) three times. We conduct all our characterization at 50°C.

4. Major Characterization Results

Fig. 4 shows how time to first bitflip (y-axis, first row of plots) and AC_{min} (y-axis, second row of plots) of the combined RowHammer and RowPress pattern (solid blue lines) and the conventional double-sided RowPress (RowHammer) pattern (dashed orange lines) changes as t_{AggON} (x-axis) increases for DRAM modules from Mfr. S, H, and M, respectively, at 50°C. Each data point shows the average time to first bitflip or AC_{min} at a given t_{AggON} value across all tested DRAM dies for each manufacturer. The error band represents the standard deviation. We highlight t_{AggON} = 36ns (= t_{RAS}) as dashed dark red lines on the x-axis because both the combined pattern and the conventional double-sided RowPress pattern are identical to the conventional double-sided RowHammer pattern when t_{AggON} = 36ns (= t_{RAS}). We highlight t_{AggON} = 7.8 µs (= t_{REFI}) and 70.2 µs (= 9×t_{REFI}) as dashed dark red lines on the x-axis because these are the potential upper bounds of t_{AggON} as specified by the JEDEC standard [69]. We make three major observations from Fig. 4.

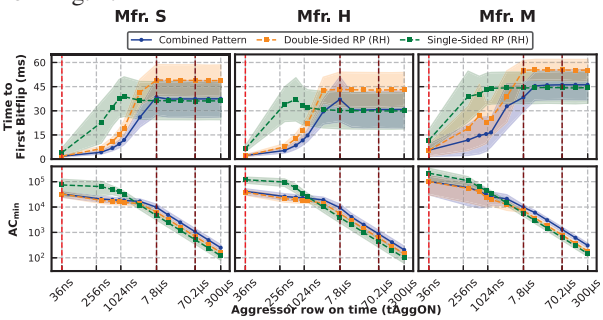

Figure 4: Time to first bitflip (first row of plots) and AC_{min} (first row of plots) of the combined RowHammer and RowPress pattern (blue solid line) and the conventional single- and double-sided RowPress (RowHammer) patterns (green and orange dashed lines).

Observation 1. As t_{AggON} initially starts to increase, the combined RowHammer and RowPress pattern takes much less time to induce the first bitflip compared to both the conventional single- and double-sided RowPress patterns.

For example, when $t_{AggON} = 636ns$, it takes the combined pattern only 6.8 ms, 8.5 ms, 14.6 ms on average to induce the first bitflip in the victim row, for Mfr. S, H, M, respectively. This is 37.6%, 33.6%, 46.1% faster compared to the conventional double-sided RowPress pattern (which takes 10.9 ms, 12.8 ms, 27.1 ms for Mfr. S, H, M, respectively, to induce the first bitflip). Compared to the single-sided RowPress pattern (which takes 32.2 ms, 37.1 ms, 40.4 ms for Mfr. S, H, M, respectively, to induce the first bitflip), the combined pattern is 78.9%, 77.1%, 63.9% faster.

Takeaway 1. Read disturbance bitflips can be induced in a smaller amount of time by combining RowPress and RowHammer compared to using solely RowPress or RowHammer.

We hypothesize that the reason for Observations 1 is that in a double-sided RowPress pattern, the read disturbance effect caused by RowPress from one of the two aggressor rows is much more significant compared to the other such that reducing the t_{AggON} of this other aggressor row does *not* significantly change AC_{min}.

Hypothesis 1. As t_{AggON} initially starts to increase, the read disturbance effect caused by RowPress from one of the two aggressor rows in the double-sided pattern is much more significant than the other.

Observation 2. As t_{AggON} initially starts to increase, the combined pattern needs slightly more aggressor row activations to induce at least one bitflip than the conventional double-sided RowPress pattern.

When $t_{AggON} = 636ns$, compared to $t_{AggON} = 36ns$ (i.e., RowHammer), the AC_{min} of the combined pattern reduces by 40.5%, 42.0%, 46.9% on average for Mfr. S, H, M, respectively. This is 7.5%, 8.0%, and 7.4% less AC_{min} reduction for Mfr. S, H, M, respectively, compared to the conventional double-sided RowPress pattern (48.0%, 50.0%, 54.3%).

Observation 3. As t_{AggON} continues to increase, the combined pattern takes a similar amount of time to induce the first bitflip as the conventional single-sided RowPress pattern.

When $t_{AggON} = 70.2\mu s$, the combined pattern takes on average 37.4ms, 30.8ms, 46.1ms to induce the first bitflip for Mfr. S, H, and M, respectively. This is 3.9%, 3.0%, 4.1% slower than the conventional single-sided RowPress pattern, which takes 36.0ms, 29.9ms, 44.3ms to induce the first bitflip.

We hypothesize that the reason for Observation 3 is that as t_{AggON} becomes large, the read disturbance effect from RowPress is dominant compared to RowHammer due to the significantly reduced number of aggressor row activations, causing the combined RowHammer and RowPress pattern to behave very similarly to the conventional single-sided RowPress pattern.

Hypothesis 2. For large t_{AggON} values, the read disturbance effect from RowPress is dominant compared to RowHammer in the combined RowHammer and RowPress pattern.

Fig. 5 shows the fraction of 1-to-0 bitflips of all the bitflips we observe from the combined RowHammer and RowPress pattern. We make the following observation from Fig. 5.

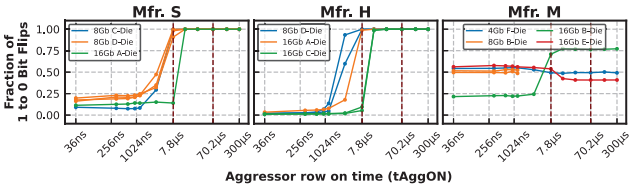

Figure 5: The fraction of 1 to 0 bitflips due to the combined RowHammer and RowPress pattern.

Observation 4. As t_{AggON} increases, the directionality of bitflips caused by the combined RowHammer and RowPress pattern changes.

We observe that for all DRAM dies tested from Mfr. S and H, as t_{AggON} initially starts to increase, the majority of the bitflips from the combined pattern are 0-to-1 bitflips. As t_{AggON} continues to increase, the fraction of 1-to-0 bitflips significantly increases. For sufficiently large t_{AggON} values, almost 100% of the bitflips are 1-to-0. Such a change in the directionality of bitflips as t_{AggON} increases is the same observation as in the original RowPress paper [12].[2] This observation also supports our Hypothesis 2 that the RowPress effect is dominant in the combined RowHammer and RowPress pattern.

Fig. 6 shows the overlap (y-axis) between the bitflips from the combined RowHammer and RowPress pattern and the conventional single- (first row of plots) and double-sided (second row of plots) RowPress (RowHammer) pattern as t_{AggON} (x-axis) increases. We define such overlap as the number of unique bitflips that are observed in both the combined pattern and the conventional RowPress (RowHammer) patterns divided by the total number of unique bitflips observed in the conventional pattern. We make two observations from the figure.

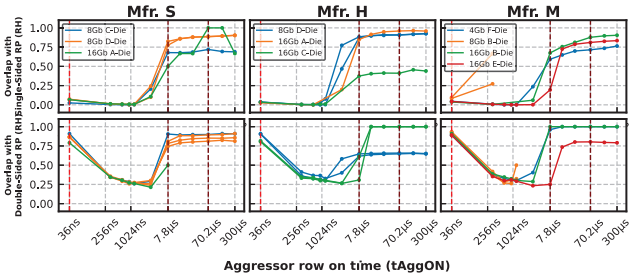

Figure 6: The overlap ratio between the bitflips from the combined RowHammer and RowPress pattern and the conventional single-sided (top row of plots) and double-sided (bottom row of plots) RowPress (RowHammer) pattern.

Observation 5. The overlap between the bitflips from the combined pattern and conventional single-sided RowPress pattern increases as t_{AggON} increases.

[2]For Mfr. M, we observe an opposite trend where the fraction of 1-to-0 bitflips *decreases* as t_{AggON} increases for all but the 16Gb B-Dies. We hypothesize that this is a result of a different true- and anti-cell layout in Mfr. M's DRAM design compared to the other two manufacturers. Such an observation on DRAM dies from Mfr. M is similar to that in the original RowPress paper [12].

Observation 6. The overlap between the bitflips from the combined pattern and conventional double-sided RowPress pattern first decreases as t_{AggON} initially starts to increase, and then increases as t_{AggON} continues to increase.

When t_{AggON} initially starts to increase, the overlap between the bitflips from the combined RowHammer and RowPress pattern and the conventional single-sided RowPress pattern remains very small, but the overlap between the bitflips from the combined pattern and the conventional double-sided RowPress pattern significantly decreases. As t_{AggON} continues to increase beyond a certain level (e.g., $> 7.8\,\mu s$), both the overlap between the combined RowHammer and RowPress pattern and the conventional single- (double-) sided RowPress patterns significantly increases to more than 75%.

Takeaway 2. The combined RowHammer and RowPress pattern induces different bitflips compared to the conventional single- and double-sided RowPress patterns.

5. Related Works

To our knowledge, this is the first work to experimentally demonstrate and characterize read disturbance caused by a combined RowHammer and RowPress access pattern. Existing works on experimental characterization of DRAM read disturbance test either only RowHammer patterns [2, 5–7] or separate RowHammer and RowPress patterns [10–13, 75] patterns. Prior works on device-level mechanisms of RowHammer [76–82] and RowPress [83] do not investigate combining RowHammer and RowPress.

6. Conclusion

In this paper, we experimentally demonstrate and characterize, for the first time, the bitflips caused by a DRAM access pattern that combines RowHammer and RowPress. Our characterization results show that the combined access pattern 1) induces bitflips faster compared to conventional single- and double-sided RowPress patterns, and 2) induces different bitflips compared to single- and double-sided RowPress (RowHammer) patterns.

We plan to investigate deeper into the combined RowHammer and RowPress pattern by 1) performing more comprehensive and rigorous characterization and analysis of the bitflips by testing more DRAM chips with more data patterns and temperatures, 2) look into the device-level mechanisms of RowHammer and RowPress to verify our hypotheses, and 3) understand the architectural implications by analyzing and evaluating how existing mitigation mechanisms need to be changed.

We hope the results and insights from this paper lead to more comprehensive and fundamental understanding of DRAM read disturbance and further research in building more robust DRAM-based memory systems.

Acknowledgments

We thank the anonymous reviewers of DSN Disrupt 2024 for their encouraging feedback. We thank the SAFARI Research Group members for providing a stimulating intellectual environment. We acknowledge the generous gifts from our industrial partners, including Google, Huawei, Intel, and Microsoft. This work is supported in part by the Microsoft-Swiss Joint Research Center and a Google Security & Privacy Research Award.

References

[1] R. H. Dennard, "Field-Effect Transistor Memory," 1968, U.S. Patent 3,387,286.

[2] Y. Kim, R. Daly, J. Kim, C. Fallin, J. H. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," in *ISCA*, 2014.

[3] O. Mutlu, "The RowHammer Problem and Other Issues We May Face as Memory Becomes Denser," in *DATE*, 2017.

[4] O. Mutlu and J. S. Kim, "RowHammer: A Retrospective," *TCAD*, 2019.

[5] J. S. Kim, M. Patel, A. G. Yağlıkçı, H. Hassan, R. Azizi, L. Orosa, and O. Mutlu, "Revisiting RowHammer: An Experimental Analysis of Modern Devices and Mitigation Techniques," in *ISCA*, 2020.

[6] L. Orosa, A. G. Yağlıkçı, H. Luo, A. Olgun, J. Park, H. Hassan, M. Patel, J. S. Kim, and O. Mutlu, "A Deeper Look into RowHammer's Sensitivities: Experimental Analysis of Real DRAM Chips and Implications on Future Attacks and Defenses," in *MICRO*, 2021.

[7] A. G. Yağlıkçı, H. Luo, G. F. Oliveira, A. Olgun, M. Patel, J. Park, H. Hassan, J. S. Kim, L. Orosa, and O. Mutlu, "Understanding RowHammer Under Reduced Wordline Voltage: An Experimental Study Using Real DRAM Devices," in *DSN*, 2022.

[8] O. Mutlu, A. Olgun, and A. G. Yağlıkcı, "Fundamentally Understanding and Solving RowHammer," in *ASP-DAC*, 2023.

[9] O. Mutlu, "Retrospective: Flipping bits in memory without accessing them: An experimental study of dram disturbance errors," in *ISCA@50 25-Year Retrospective: 1996-2020*, J. F. Martínez and L. K. John, Eds. ACM SIGARCH and IEEE TCCA, 2023.

[10] A. Olgun, M. Osserian, A. G. Yağlıkçı, Y. Can Tugrul, H. Luo, S. Rhyner, B. Salami, J. Gomez-Luna, and O. Mutlu, "An Experimental Analysis of RowHammer in HBM2 DRAM Chips," in *DSN Disrupt*, 2023.

[11] A. Olgun, M. Osserian, A. G. Yağlıkçı, Y. Can Tugrul, H. Luo, S. Rhyner, B. Salami, J. Gomez-Luna, and O. Mutlu, "Read Disturbance in High Bandwidth Memory: A Detailed Experimental Study on HBM2 DRAM Chips," in *DSN*, 2024.

[12] H. Luo, A. Olgun, A. G. Yağlıkçı, Y. C. Tuğrul, S. Rhyner, M. B. Cavlak, J. Lindegger, M. Sadrosadati, and O. Mutlu, "RowPress: Amplifying Read Disturbance in Modern DRAM Chips," in *ISCA*, 2023.

[13] H. Nam, S. Baek, M. Wi, M. Kim, J. Park, C. Song, N. Kim, and J. Ahn, "X-ray: Discovering DRAM Internal Structure and Error Characteristics by Issuing Memory Commands," in *CAL*, 2023.

[14] D. Gruss, C. Maurice, and S. Mangard, "Rowhammer.js: A Remote Software-Induced Fault Attack in JavaScript," *arXiv:1507.06955 [cs.CR]*, 2015.

[15] A. P. Fournaris, L. Pocero Fraile, and O. Koufopavlou, "Exploiting Hardware Vulnerabilities to Attack Embedded System Devices: A Survey of Potent Microarchitectural Attacks," *Electronics*, 2017.

[16] D. Poddebniak, J. Somorovsky, S. Schinzel, M. Lochter, and P. Rösler, "Attacking Deterministic Signature Schemes using Fault Attacks," in *EuroS&P*, 2018.

[17] A. Tatar, R. K. Konoth, E. Athanasopoulos, C. Giuffrida, H. Bos, and K. Razavi, "Throwhammer: Rowhammer Attacks Over the Network and Defenses," in *USENIX ATC*, 2018.

[18] S. Carre, M. Desjardins, A. Facon, and S. Guilley, "OpenSSL Bellcore's Protection Helps Fault Attack," in *DSD*, 2018.

[19] A. Barenghi, L. Breveglieri, N. Izzo, and G. Pelosi, "Software-Only Reverse Engineering of Physical DRAM Mappings for Rowhammer Attacks," in *IVSW*, 2018.

[20] Z. Zhang, Z. Zhan, D. Balasubramanian, X. Koutsoukos, and G. Karsai, "Triggering Rowhammer Hardware Faults on ARM: A Revisit," in *ASHES*, 2018.

[21] S. Bhattacharya and D. Mukhopadhyay, "Advanced Fault Attacks in Software: Exploiting the Rowhammer Bug," *Fault Tolerant Architectures for Cryptography and Hardware Security*, 2018.

[22] M. Seaborn and T. Dullien, "Exploiting the DRAM Rowhammer Bug to Gain Kernel Privileges," http://googleprojectzero.blogspot.com.tr/2015/03/exploiting-dram-rowhammer-bug-to-gain.html, 2015.

[23] SAFARI Research Group, "RowHammer — GitHub Repository," https://github.com/CMU-SAFARI/rowhammer, 2021.

[24] M. Seaborn and T. Dullien, "Exploiting the DRAM Rowhammer Bug to Gain Kernel Privileges," *Black Hat*, 2015.

[25] V. van der Veen, Y. Fratantonio, M. Lindorfer, D. Gruss, C. Maurice, G. Vigna, H. Bos, K. Razavi, and C. Giuffrida, "Drammer: Deterministic Rowhammer Attacks on Mobile Platforms," in *CCS*, 2016.

[26] D. Gruss, C. Maurice, and S. Mangard, "Rowhammer.js: A Remote Software-Induced Fault Attack in Javascript," in *DIMVA*, 2016.

[27] K. Razavi, B. Gras, E. Bosman, B. Preneel, C. Giuffrida, and H. Bos, "Flip Feng Shui: Hammering a Needle in the Software Stack," in *USENIX Security*, 2016.

[28] P. Pessl, D. Gruss, C. Maurice, M. Schwarz, and S. Mangard, "DRAMA: Exploiting DRAM Addressing for Cross-CPU Attacks," in *USENIX Security*, 2016.

[29] Y. Xiao, X. Zhang, Y. Zhang, and R. Teodorescu, "One Bit Flips, One Cloud Flops: Cross-VM Row Hammer Attacks and Privilege Escalation," in *USENIX Security*, 2016.

[30] E. Bosman, K. Razavi, H. Bos, and C. Giuffrida, "Dedup Est Machina: Memory Deduplication as an Advanced Exploitation Vector," in *S&P*, 2016.

[31] S. Bhattacharya and D. Mukhopadhyay, "Curious Case of Rowhammer: Flipping Secret Exponent Bits Using Timing Analysis," in *CHES*, 2016.

[32] W. Burleson, O. Mutlu, and M. Tiwari, "Invited: Who is the Major Threat to Tomorrow's Security? You, the Hardware Designer," in *DAC*, 2016.

[33] R. Qiao and M. Seaborn, "A New Approach for RowHammer Attacks," in *HOST*, 2016.

[34] F. Brasser, L. Davi, D. Gens, C. Liebchen, and A.-R. Sadeghi, "Can't Touch This: Software-Only Mitigation Against Rowhammer Attacks Targeting Kernel Memory," in *USENIX Security*, 2017.

[35] Y. Jang, J. Lee, S. Lee, and T. Kim, "SGX-Bomb: Locking Down the Processor via Rowhammer Attack," in *SOSP*, 2017.

[36] M. T. Aga, Z. B. Aweke, and T. Austin, "When Good Protections Go Bad: Exploiting Anti-DoS Measures to Accelerate Rowhammer Attacks," in *HOST*, 2017.

[37] A. Tatar, C. Giuffrida, H. Bos, and K. Razavi, "Defeating Software Mitigations Against Rowhammer: A Surgical Precision Hammer," in *RAID*, 2018.

[38] D. Gruss, M. Lipp, M. Schwarz, D. Genkin, J. Juffinger, S. O'Connell, W. Schoechl, and Y. Yarom, "Another Flip in the Wall of Rowhammer Defenses," in *S&P*, 2018.

[39] M. Lipp, M. T. Aga, M. Schwarz, D. Gruss, C. Maurice, L. Raab, and L. Lamster, "Nethammer: Inducing Rowhammer Faults Through Network Requests," arXiv:1805.04956 [cs.CR], 2018.

[40] V. van der Veen, M. Lindorfer, Y. Fratantonio, H. P. Pillai, G. Vigna, C. Kruegel, H. Bos, and K. Razavi, "GuardION: Practical Mitigation of DMA-Based Rowhammer Attacks on ARM," in *DIMVA*, 2018.

[41] P. Frigo, C. Giuffrida, H. Bos, and K. Razavi, "Grand Pwning Unit: Accelerating Microarchitectural Attacks with the GPU," in *S&P*, 2018.

[42] L. Cojocar, K. Razavi, C. Giuffrida, and H. Bos, "Exploiting Correcting Codes: On the Effectiveness of ECC Memory Against Rowhammer Attacks," in *S&P*, 2019.

[43] S. Ji, Y. Ko, S. Oh, and J. Kim, "Pinpoint Rowhammer: Suppressing Unwanted Bit Flips on Rowhammer Attacks," in *ASIACCS*, 2019.

[44] S. Hong, P. Frigo, Y. Kaya, C. Giuffrida, and T. Dumitraş, "Terminal Brain Damage: Exposing the Graceless Degradation in Deep Neural Networks Under Hardware Fault Attacks," in *USENIX Security*, 2019.

[45] A. Kwong, D. Genkin, D. Gruss, and Y. Yarom, "RAMBleed: Reading Bits in Memory Without Accessing Them," in *S&P*, 2020.

[46] P. Frigo, E. Vannacci, H. Hassan, V. van der Veen, O. Mutlu, C. Giuffrida, H. Bos, and K. Razavi, "TRRespass: Exploiting the Many Sides of Target Row Refresh," in *S&P*, 2020.

[47] L. Cojocar, J. Kim, M. Patel, L. Tsai, S. Saroiu, A. Wolman, and O. Mutlu, "Are We Susceptible to Rowhammer? An End-to-End Methodology for Cloud Providers," in *S&P*, 2020.

[48] Z. Weissman, T. Tiemann, D. Moghimi, E. Custodio, T. Eisenbarth, and B. Sunar, "JackHammer: Efficient Rowhammer on Heterogeneous FPGA–CPU Platforms," arXiv:1912.11523 [cs.CR], 2020.

[49] Z. Zhang, Y. Cheng, D. Liu, S. Nepal, Z. Wang, and Y. Yarom, "PThammer: Cross-User-Kernel-Boundary Rowhammer through Implicit Accesses," in *MICRO*, 2020.

[50] F. Yao, A. S. Rakin, and D. Fan, "DeepHammer: Depleting the Intelligence of Deep Neural Networks Through Targeted Chain of Bit Flips," in *USENIX Security*, 2020.

[51] F. de Ridder, P. Frigo, E. Vannacci, H. Bos, C. Giuffrida, and K. Razavi, "SMASH: Synchronized Many-Sided Rowhammer Attacks from JavaScript," in *USENIX Security*, 2021.

[52] H. Hassan, Y. C. Tugrul, J. S. Kim, V. v. d. Veen, K. Razavi, and O. Mutlu, "Uncovering in-DRAM RowHammer Protection Mechanisms: A New Methodology, Custom RowHammer Patterns, and Implications," in *MICRO*, 2021.

[53] P. Jattke, V. van der Veen, P. Frigo, S. Gunter, and K. Razavi, "Blacksmith: Scalable Rowhammering in the Frequency Domain," in *SP*, 2022.

[54] M. C. Tol, S. Islam, B. Sunar, and Z. Zhang, "Toward Realistic Backdoor Injection Attacks on DNNs using RowHammer," arXiv:2110.07683v2 [cs.LG], 2022.

[55] A. Kogler, J. Juffinger, S. Qazi, Y. Kim, M. Lipp, N. Boichat, E. Shiu, M. Nissler, and D. Gruss, "Half-Double: Hammering From the Next Row Over," in *USENIX Security*, 2022.

[56] L. Orosa, U. Rührmair, A. G. Yaglikci, H. Luo, A. Olgun, P. Jattke, M. Patel, J. Kim, K. Razavi, and O. Mutlu, "SpyHammer: Using RowHammer to Remotely Spy on Temperature," arXiv:2210.04084, 2022.

[57] Z. Zhang, W. He, Y. Cheng, W. Wang, Y. Gao, D. Liu, K. Li, S. Nepal, A. Fu, and Y. Zou, "Implicit Hammer: Cross-Privilege-Boundary Rowhammer through Implicit Accesses," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[58] L. Liu, Y. Guo, Y. Cheng, Y. Zhang, and J. Yang, "Generating Robust DNN with Resistance to Bit-Flip based Adversarial Weight Attack," *IEEE Transactions on Computers*, 2022.

[59] Y. Cohen, K. S. Tharayil, A. Haenel, D. Genkin, A. D. Keromytis, Y. Oren, and Y. Yarom, "HammerScope: Observing DRAM Power Consumption Using Rowhammer," in *CCS*, 2022.

[60] M. Zheng, Q. Lou, and L. Jiang, "TrojViT: Trojan Insertion in Vision Transformers," arXiv:2208.13049, 2022.

[61] M. Fahr Jr, H. Kippen, A. Kwong, T. Dang, J. Lichtinger, D. Dachman-Soled, D. Genkin, A. Nelson, R. Perlner, A. Yerukhimovich *et al.*, "When Frodo Flips: End-to-End Key Recovery on FrodoKEM via Rowhammer," *CCS*, 2022.

[62] Y. Tobah, A. Kwong, I. Kang, D. Genkin, and K. G. Shin, "SpecHammer: Combining Spectre and Rowhammer for New Speculative Attacks," in *SP*, 2022.

[63] A. S. Rakin, M. H. I. Chowdhuryy, F. Yao, and D. Fan, "DeepSteal: Advanced Model Extractions Leveraging Efficient Weight Stealing in Memories," in *SP*, 2022.

[64] K. Mus, Y. Doröz, M. C. Tol, K. Rahman, and B. Sunar, "Jolt: Recovering tls signing keys via rowhammer faults," in *S&P*, 2023.

[65] M. C. Tol, S. Islam, A. J. Adiletta, B. Sunar, and Z. Zhang, "Don't knock! rowhammer at the backdoor of dnn models," in *S&P*, 2023.

[66] P. Jattke, M. Wipfli, F. Solt, M. Marazzi, M. Bölcskei, and K. Razavi, "ZenHammer: Rowhammer Attacks on AMD Zen-based Platforms," in *USENIX Security*, 2024.

[67] J. Liu, B. Jaiyen, R. Veras, and O. Mutlu, "RAIDR: Retention-Aware Intelligent DRAM Refresh," in *ISCA*, 2012.

[68] J. Liu, B. Jaiyen, Y. Kim, C. Wilkerson, O. Mutlu, J. Liu, B. Jaiyen, Y. Kim, C. Wilkerson, and O. Mutlu, "An Experimental Study of Data Retention Behavior in Modern DRAM Devices," in *ISCA*, 2013.

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

[70] A. Olgun, H. Hassan, A. G. Yaglikci, Y. C. Tugrul, L. Orosa, H. Luo, M. Patel, O. Ergin, and O. Mutlu, "DRAM Bender: An Extensible and Versatile FPGA-based Infrastructure to Easily Test State-of-the-art DRAM Chips," *IEEE TCAD*, 2023.

[71] SAFARI Research Group, "DRAM Bender — GitHub Repository," https://github.com/CMU-SAFARI/DRAM-Bender.

[72] H. Hassan, N. Vijaykumar, S. Khan, S. Ghose, K. Chang, G. Pekhimenko, D. Lee,

O. Ergin, and O. Mutlu, "SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies," in *HPCA*, 2017.
[73] SAFARI Research Group, "SoftMC — GitHub Repository," https://github.com/CMU-SAFARI/softmc, 2021.
[74] Maxwell, "FT20X," https://www.maxwell-fa.com/upload/files/base/8/m/311.pdf.
[75] A. G. Yağlıkçı, G. F. Oliveira, Y. Can Tugrul, I. E. Yuksel, A. Olgun, H. Luo, and O. Mutlu, "Spatial Variation-Aware Read Disturbance Defenses: Experimental Analysis of Real DRAM Chips and Implications on Future Solutions," in *HPCA*, 2024.
[76] C. Yang, C. K. Wei, Y. J. Chang, T. C. Wu, H. P. Chen, and C. S. Lai, "Suppression of RowHammer Effect by Doping Profile Modification in Saddle-Fin Array Devices for Sub-30-nm DRAM Technology," *TDMR*, 2016.
[77] K. Park, C. Lim, D. Yun, and S. Baeg, "Experiments and Root Cause Analysis for Active-Precharge Hammering Fault in DDR3 SDRAM under 3xnm Technology," *Microelectronics Reliability*, 2016.
[78] S.-W. Ryu, K. Min, J. Shin, H. Kwon, D. Nam, T. Oh, T.-S. Jang, M. Yoo, Y. Kim, and S. Hong, "Overcoming the Reliability Limitation in the Ultimately Scaled DRAM using Silicon Migration Technique by Hydrogen Annealing," in *IEDM*, 2017.
[79] T. Yang and X.-W. Lin, "Trap-Assisted DRAM Row Hammer Effect," *EDL*, 2019.
[80] A. J. Walker, S. Lee, and D. Beery, "On DRAM RowHammer and the Physics on Insecurity," *IEEE TED*, 2021.
[81] L. Zhou, J. Li, Z. Qiao, P. Ren, Z. Sun, J. Wang, B. Wu, Z. Ji, R. Wang, K. Cao, and R. Huang, "Double-sided row hammer effect in sub-20 nm dram: Physical mechanism, key features and mitigation," in *IRPS*, 2023.
[82] J. Li, L. Zhou, S. Ye, Z. Qiao, and Z. Ji, "Understanding the competitive interaction in leakage mechanisms for effective row hammer mitigation in sub-20 nm dram," in *IEEE Electron Device Letters*, 2024.
[83] L. Zhou, J. Li, P. Ren, S. Ye, D. Wang, Z. Qiao, and Z. Ji, "Understanding the physical mechanism of rowpress at the device-level in sub-20 nm dram," in *IRPS*, 2024.