This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LCA.2014.2332177, IEEE Computer Architecture Letters

Architectural Support for Mitigating Row Hammering in DRAM Memories

Dae-Hyun Kim, Prashant J. Nair, and Moinuddin K. Qureshi, Georgia Institute of Technology
{*dhkim,pnair6,moin*}*@ece.gatech.edu*

Abstract—DRAM scaling has been the prime driver of increasing capacity of main memory systems. Unfortunately, lower technology nodes worsen the cell reliability as it increases the coupling between adjacent DRAM cells, thereby exacerbating different failure modes. This paper investigates the reliability problem due to *Row Hammering*, whereby frequent activations of a given row can cause data loss for its neighboring rows. As DRAM scales to lower technology nodes, the threshold for the number of row activations that causes data loss for the neighboring rows reduces, making Row Hammering a challenging problem for future DRAM chips. To overcome Row Hammering, we propose two architectural solutions: First, *Counter-Based Row Activation (CRA)*, which uses a counter with each row to count the number of row activations. If the count exceeds the row hammering threshold, a dummy activation is sent to neighboring rows proactively to refresh the data. Second, *Probabilistic Row Activation (PRA)*, which obviates storage overhead of tracking and simply allows the memory controller to proactively issue dummy activations to neighboring rows with a small probability for all memory access. Our evaluations show that these solutions are effective at mitigating Row hammering while causing negligible performance loss ($<$ 1%).

1 Introduction

Dynamic random access memory (DRAM) is used as main memory in computer systems. Each DRAM cell stores data by placing charge on a capacitor. To increase capacity, DRAM has continued to scale towards high-density chips with smaller feature sizes. Unfortunately, scaling DRAM to smaller nodes exacerbates several failure modes and creates new failure modes.

Below the feature size of 100nm, DRAM cell transistors suffer from short channel effect (SCE), which lowers threshold voltage, increases leakage, and reduces the retention time of DRAM cells. To overcome SCE and maintain the retention time, DRAM vendors now exploit three-dimensional (3D) cell transistors [1], [2], [3]. However, such 3D cell transistors severely suffer from the activation of adjacent rows [3] potentially causing data errors to neighboring rows. Furthermore, DRAM has scaled down to a smaller feature size and transitioned from $6F^2$ to $4F^2$, which reduces the distance between transistors and increases coupling from neighboring transistors and their word lines. Coupling from neighboring rows lowers threshold voltage and increases sub-threshold leakage current. Higher sub-threshold current accelerates charge leakage from DRAM storage nodes and reduces the retention time of the cell. This phenomenon of increasing leakage in cells of adjacent rows (victim rows) by frequent activations on a given row is called *Row Hammering*. Row Hammering is a problem not only for current DRAM devices, but as technology shrinks, this becomes an even more serious problem. This paper is geared towards tolerating Row Hammering in high-density DRAM memories.

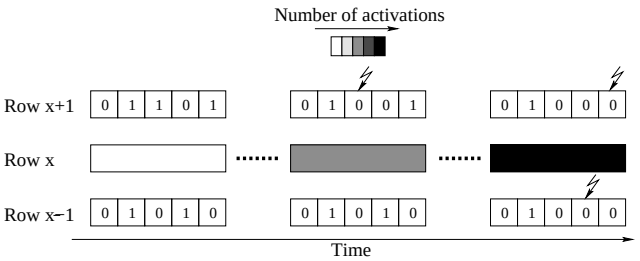

Fig. 1: Impact of row hammering on neighboring DRAM cells as the number of activations on target row X increases

Row hammering causes data loss when one row in the memory receives a large number of activations, and the rows neighboring this row do not have an activation (otherwise these rows would get a precharge and restore the data back to its original state). Figure 1 captures the problem of Row Hammering, for a given row X, where the neighboring rows are labeled $X-1$ and $X+1$. If X is accessed frequently, and $X-1$ and $X+1$ are not accessed, then the contents of these neighboring rows can get lost due to Row Hammering. The threshold for the number of activations within a refresh cycle required to cause data loss due to Row Hammering is called the *Row-Hammering Threshold (RH_{th})*. With each technology generation, this threshold reduces, making the Row Hammering problem much severe for future nanoscale DRAMs. We show that for future DRAMs this threshold could be in the range of several tens of thousands of row activations for a given row, a threshold that can be easily reached by current workloads. Furthermore, it is quite easy to write malicious (or memory stress) programs that can trivially cross this threshold. To maintain data integrity, future DRAMs must mitigate Row Hammering for both typical workloads as well as for worst-case (or malicious) workloads.

Manuscript submitted: 15-Apr-2014. Manuscript accepted: 17-May-2014. Final manuscript received: 20-May-2014

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LCA.2014.2332177, IEEE Computer Architecture Letters

2 Background and Motivation

2.1 Origins of Row-Hammering Phenomenon

Row hammering originates from two effects:

2.1.1 Word line to Word line (WL-WL) Coupling

DRAM voltage does not scale down proportional to feature size. At a smaller feature size, the ratio of coupling noise to stored signal voltage on a non-accessed DRAM cell of neighboring rows increases because of WL-WL coupling [4], [5]. Coupling noise (crosstalk) between word lines increases the sub-threshold leakage current of cell transistors on adjacent rows [4].

2.1.2 Passing-Gate Effect

Although 3-D transistors mitigate SCE, they are susceptible to coupling from adjacent gates and affect a victim gate [2]. A gate close to the victim gate using the same active area is referred to as "active adjacent gate". A gate close to the victim gate that do not use the same active area is called as "passing gate". Activating any active adjacent gate or passing gate changes the electric field around the victim gate, which lowers the threshold voltage and increases leakage current of victim cell transistors.

2.2 Analysis of Row Hammering Threshold

The leakage current of a cell transistor (I_{leak}) increases at lower technology nodes. DRAM vendors keep a guard-band for the retention time (t_{ret-GB}) that is β times refresh rate (t_{ret-th}) as a safety margin to conform to the JEDEC refresh standard. At this guard-band, let the leakage current be $I_{leak-GB}$. Let t_{ret-RH} be the time during which a cell on a victim row suffers from row hammering and $I_{leak-RH}$ be the increased leakage current by α times under row hammering. At sub-nanometer nodes, $I_{leak-RH}$ is represented by (2.1):

$$I_{leak-RH} = \alpha \cdot I_{leak-GB}. \tag{2.1}$$

Alternatively, I_{leak} can be represented by

$$I_{leak} = \frac{Q}{t} = \frac{C \cdot V}{t} \Rightarrow C \cdot V = I_{leak} \cdot t \tag{2.2}$$

in which C_s is a cell capacitance of a DRAM cell and V is the capacitor driving voltage. Using (2.1) and (2.2),

$$\begin{aligned} & I_{leak-GB} \cdot t_{ret-GB} = C_s \cdot V \\ & = I_{leak-GB} \cdot (t_{ret-th} - t_{ret-RH}) + I_{leak-RH} \cdot t_{ret-RH} \\ & = I_{leak-GB} \cdot (t_{ret-th} - t_{ret-RH}) + \alpha \cdot I_{leak-GB} \cdot t_{ret-RH} \\ & \Rightarrow \boxed{t_{ret-GB} = t_{ret-th} + (\alpha - 1) \cdot t_{ret-RH}} \end{aligned} \tag{2.3}$$

Expressing t_{ret-GB} in terms of t_{ret-th}

$$t_{ret-GB} = \beta \cdot t_{ret-th} \tag{2.4}$$

From (2.3) and (2.4), Row-Hammering Threshold (RH_{th}) is given by

$$\boxed{RH_{th} = \frac{\beta - 1}{\alpha - 1} \times M_{max}} \tag{2.5}$$

where M_{max} is total possible number of activations in a refresh rate. At t_{ret-th} of 64ms, $M_{max} \approx 1.3$ million.

$$RH_{th}@64ms = \frac{\beta - 1}{\alpha - 1} \times 1.3M \tag{2.6}$$

The main component of I_{leak} is the sub-threshold leakage current I_{sub} from row hammering ($I_{leak-RH}$) [4], [5], [2].

$$I_{leak} \approx I_{sub} \propto e^{q\Delta V_{th-sub}/nkT} \tag{2.7}$$

in which ΔV_{th-sub} is variation in sub-threshold voltage and n is body-effect coefficient. ΔV_{th-sub} = 50mV to 70mV in 50nm DRAM with SRCAT [2] and body-effect coefficient (n) = 1.1 $\sim$ 1.4 [6]. So α ranges from 4 to 11.7. For example, for α=11, β=2 $\Rightarrow$ RH_{th} of 130K, which can be expected of current generation DRAM modules. However, α is related to fabrication process and is increasing as DRAM scales down. Thus, future nanoscale DRAM can be expected to have RH_{th} in the range of few tens of thousands. To address the problem for future technology nodes, we will assume $RH_{th} = 32K$ in our study.

TABLE 1: System Configuration (default of USIMM)

Number of cores	Two: 4-wide, 3.2GHz
Processor ROB size	160
Cache line size	64Byte
Last Level Cache	512KB per core
Memory bus speed	800MHz
DDR3 Memory channels	2, each 8GB DIMM

2.3 Architectural Impact of Row Hammering

Typical Workloads: Current workloads can have an activation patterns that target a few DRAM rows frequently. We study the possibility of row hammering using USIMM [7] and uses workloads from the memory scheduling championship [8]. We evaluate a system with eight 8Gb chips [9]. Table 1 shows the configuration for our system.

Figure 2 shows the maximum number of activations (activation peak) of a row at the refresh rate of 64ms for a few PARSEC, SPEC, BIOBENCH and COMMERCIAL benchmarks. Figure 2 shows that workloads have an activation peak of several thousand activations between refreshes. For example, MT-Fluid has 400K activations for a single row within 64ms. As technology scales these activation peaks can easily surpass the Row-Hammering Threshold.

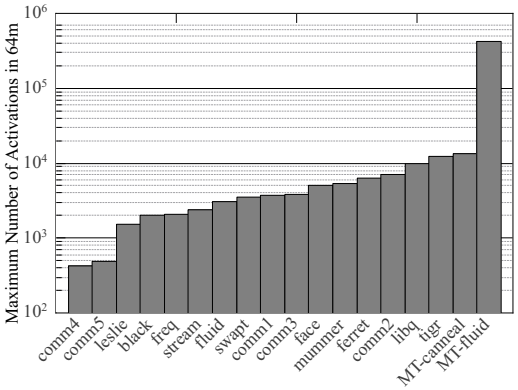

Fig. 2: Maximum number of activations for a given row within a time period of one refresh interval (64ms).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LCA.2014.2332177, IEEE Computer Architecture Letters

Malicious Programs: Malicious programs that frequently activate a given row can easily cause data loss due to Row Hammering. Unfortunately, such malicious programs are quite easy to write (they are similar to the attack kernels in [10], which can be written in about ten lines of C code). Thus, for future DRAMs, Row Hammering not only poses a reliability issue, it also present a security issue whereby a malicious program can intentionally cause data loss for a co-running program. We present two hardware solutions to mitigate Row Hammering in DRAM memories.

3 Row-Hammering-Aware System

Mitigation of Row Hammering can be done by sending a proactive activation to the victim rows before the target row crosses the row-hammering threshold. Such a proactive activation acts as a refresh command for the victim rows, and refreshes the contents of these rows, thereby preventing data loss due to the activity of the neighboring rows. We propose two schemes, *Counter-Based Row Activation (CRA)* and *Probabilistic Row Activation (PRA)*. CRA scheme tracks activations for each row and provides guaranteed mitigation, whereas PRA is a probabilistic scheme that avoids storage overhead and yet provides highly robust mitigation.

3.1 Counter-Based Row Activation

The CRA scheme maintains a row activation counter (RAC) for every row to keep track of the number of activations to each row. These activation counters are incremented when the row is activated and cleared when mitigation is performed. As soon as the number of activations of a target row is equal to the row-hammering threshold, the victim rows associated with the given row gets activated. Such proactive dummy activations of victim rows refresh their data and prevent data corruption caused by row hammering. To cope with future memory systems with lower row-hammering threshold, this work employs two-byte (16 bits) long activation counter per row that can count up to 64K activations per refresh cycle.

For a 8GB memory system with one million rows, the total size of the counters for all rows will be 2MB. However, it is impractical to devote multi megabyte of on-chip SRAM storage for storing the counter of CRA. Instead, we propose a CRA implementation that stores the counters in a reserved area in the DRAM (0.0375% of main memory reserved for the counters). To mitigate performance penalty of counter accesses we employ a dedicated counter-cache on chip. A memory controller checks the counter-cache for activation counters and caches them from the reserved area. The reserved area is only accessed on a counter-cache miss for activation counters. Every access to the reserved area brings a cache line with activation counters for 32 contiguous rows, which ensures high locality in the counter-cache. In steady state, rows with frequent accesses and high locality will have their counters cached. A memory controller increments the counter of row activated and clears the counter after a row is refreshed or on mitigation. Figure 3 shows the sequence of events for CRA.

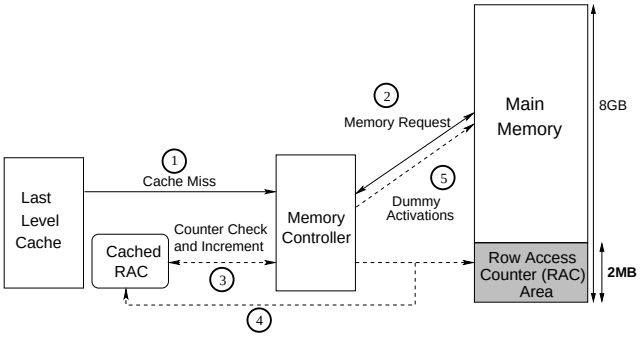

Fig. 3: Sequence of memory operations with CRA

Figure 4 shows the performance impact of CRA on execution time, as the size of the on-chip counter cache is varied. Even though we used a 32K threshold for this study, the performance degradation stems mainly from the memory accesses for the counters. With a counter cache of 128KB, CRA scheme has less than 0.5% performance degradation, while ensuring that the victim rows get refreshed before the target row reaches the threshold.

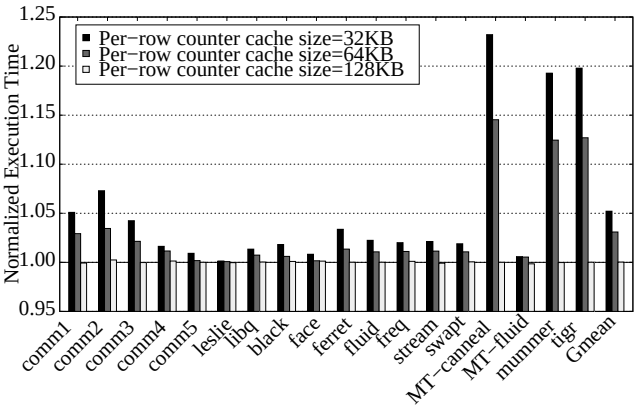

Fig. 4: Impact on execution time from CRA (Row Hammering threshold of 32K)

3.2 Probabilistic Row Activation

The PRA scheme avoids the storage overhead of the CRA scheme by obviating any tracking structures. Instead, it performs a row activation of the neighboring rows with a small probability every time a given row is accessed. For example, if the probability of activation is set to 0.1%, then for each row activation, the memory controller consults a random number generator to find if the proactive activations must be issued. If so, the memory controller proactively inserts activations for the two neighboring rows for the row being accessed. The key insight for PRA is that hammered rows will have frequent activations and hence are highly likely to get selected for probabilistic mitigation.

3.2.1 Analysis

Let us consider a system that performs dummy activations of neighboring rows on each access with a probability N. The probability of a target row not being activated after M activations in total within a refresh rate and resulting in a potential system failure is given by (3.8):

$$P_{error} = (1-N)^M = (1-N)^{\frac{1}{N}MN}. \qquad (3.8)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LCA.2014.2332177, IEEE Computer Architecture Letters

If we perform these pro-active dummy activations with a very small probability, then such activations have a negligible effect on performance.

Since $\lim_{N\to 0}(1-N)^{\frac{1}{N}} = e^{-1}$ for very small values of N, we deduce (3.9) from (3.8):

$$P_{error} = e^{-M \cdot N}. \tag{3.9}$$

From (3.9), the probability that the system will have no errors is given by (3.10):

$$P_{no-error} = 1 - e^{-M \cdot N}. \tag{3.10}$$

Let the system during its lifetime have K such instances. The probability of having no failures in the entire lifetime of the system is given by (3.11):

$$P_{no-failure} = (1 - e^{-M \cdot N})^k. \tag{3.11}$$

Subsequently, the probability of at least having one failure during its total runtime is given by (3.12):

$$P_{failure} = 1 - (1 - e^{-M \cdot N})^k. \tag{3.12}$$

In the worst case, if N = 0.1%, M=32K (Row Hammering threshold) and for runtime of 10 years, K≈25 billion; then (3.13) and (3.14) shows that the probability of data loss with PRA would be 1 in ten million, over a period of 10 years.

$$P_{failure} = 1 - (1 - e^{-32K \times 0.001})^{25 \times 10^9}. \tag{3.13}$$

Since $(1-n)^x \approx 1 - nx$ for small nx, (3.13) degenerates to (3.14):

$$P_{failure} = \frac{25 \times 10^9}{e^{32}} \approx 10^{-7}. \tag{3.14}$$

The failure probability could be made as low as 10^{-120}, with a N=1%. Thus, even though PRA does not require any storage structures, it can still provide very robust protection against row hammering, even at very small Row Hammering threshold.

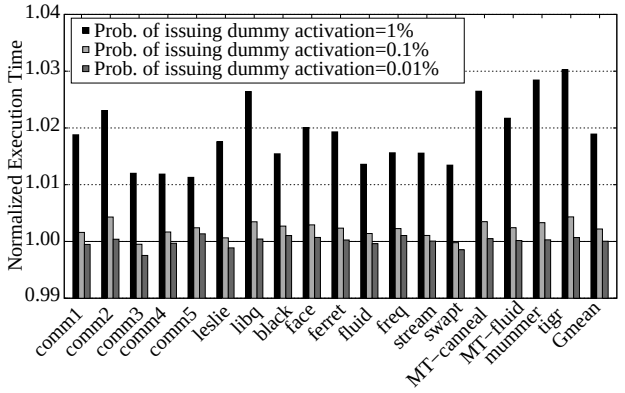

Fig. 5: Impact on execution time from PRA

3.2.2 Results

Figure 5 shows the impact of PRA on execution time, as the probability of dummy activation is increased from 1% to 0.01%. Higher probability of issuing dummy activation (N) degrades the performance of most workloads. For example, probability of activation of 1% results in 2% increase of activations. On an average, this increases the execution time by around 2%. As the probability of issuing dummy activation is reduced to 0.1%, the performance degradation is negligible ($< 0.2\%$ on average). Thus, PRA avoids both storage and performance overhead of CRA, and still provides robust mitigation to Row Hammering.

4 Conclusions

Frequent activations to a row can influence neighboring DRAM cells and cause data corruption due to Row Hammering. To mitigate Row Hammering, we propose two architectural solutions: Counter-Based Row Activation (CRA) and Probabilistic Row Activation (PRA).

We expect Row Hammering to become even more severe for future memory chips. In fact, an upcoming parallel work [11] experimentally shows that Row Hammering is indeed prevalent in modern DRAM chips, and their measured threshold of 128K is consistent with our 130K based on our theoretical model.

Technology scaling accelerates Row Hammering and makes DRAM vulnerable to other sources of errors. We show that architectural solutions can help mitigate such errors efficiently and thus help with DRAM scaling.

Acknowledgments

This work was supported in part by C-FAR (one of the six SRC STARnet Centers, sponsored by MARCO and DARPA) and NSF grant CCF-1319587. Daehyun Kim is supported by a Samsung Fellowship. We thank Yoongu Kim for sharing (on May 2nd) a preprint of their ISCA paper.

References

[1] J. Y. Kim *et al.*, "S-rcat(sphere-shaped-recess-channel-array transistor) technology for 70nm dram feature size and beyond," in *VLSI*, 2005.

[2] M. S. Y. et al., "Saddle-fin Cell Transistors with Oxide Etch Rate Control by Using Tilted Ion Implantation (TIS-Fin) for Sub-50-nm DRAMs," in *Journal of the Korean Physical Society*, vol. 56, 2010.

[3] S. Hong, "Memory technology trend and future challenges," in *IEDM*, 2010.

[4] M. Redeker, B. F. Cockburn, and D. G. Elliott, "An investigation into crosstalk noise in dram structure," in *IEEE International Workshop on Memory Technology, Design and Testing*, 2002.

[5] D.-S. Min and D. W. Langer, "Multiple Twisted Dataline Techniques for Multigigabit DRAM's," in *JSSC*, vol. 34, 1999.

[6] Y. Taur and T. H. Ning, *Fundamentals of Modern VLSI Devices*. New York, NY, USA: Cambridge University Press, 2nd ed., 2009.

[7] N. Chatterjee *et al.*, "Usimm:the utah simulated memory module," 2012.

[8] *2012 Memory scheduling championship (msc)*. http://www.cs.utah.edu/ rajeev/jwac12/.

[9] *MT41J512M4:8Gb QuadDie DDR3 SDRAM Rev. A 03/11, Micron, 2010.*

[10] M. K. Qureshi *et al.*, "Enhancing lifetime and security of pcm-based main memory with start-gap wear leveling," in *MICRO-42*, 2009.

[11] Y. Kim *et al.*, "Flipping bits in memory without accessing them: An experimental study of dram disturbance errors," to appear in ISCA, 2014.