Current and Upcoming Challenges of the Main Memory System (2021)

Konstantin Biriukov
Saint Cloud State University
720 4th Ave S, St Cloud, MN 56301

ABSTRACT

Main memory is acritical component of all computer systems. Memory system must scale in size, cost and performance to maintain overall computer performance growth, but there are a lot of challenges that memory systems have encountered and throughout recent years some (potentially temporary) limitations for memory systems were reached out. This paper will discuss recent challenges in main memory systems and solutions that can be proposed to address those challenges. Problems to be discussed in the paper: memory capacity, energy consumption, periodic refresh, scaling, memory interference, "Rowhammer".

Keywords

DRAM, RAM, DDR, LPDDR, STT-MRAM, ReRam, PCM, Main memory System, Rowhammer

1. INTRODUCTION

There are many challenges and problems that DRAM encountered in recent years but only the most important ones will be discussed in the paper. RAM had become a major bottleneck for the overall growth performance of the computers, but many challenges were accepted by many Research &Development departments of technological companies and computer scientists of research institutes, who delivered fascinating results in attempts to resolve those issues. All information from the articles are compared with the newest features and discussions proposed by different companies and researchers to make sure that the paper covers the latest condition of the Main Memory System challenges.

2. MEMORY CHALLENGES

2.1 Memory Capacity

Core count doubling takes place every 2 years, DRAM (Dynamic Random Access Memory) capacity doubling takes place every 3 years, which means that memory capacity per core expected to drop 30% every 2 years [1]. The need for memory capacity is greatly increasing due to the data-intensive applications and it will continue to increase [2].

Some fields of Computer Science are DRAM-demanding. For example, some sophisticated models of Neural Networks or Advanced Driver Assistance Systems require a lot of memory capacity, and struggling with limited memory bandwidth, which is one of the main bottlenecks for the evolution of the overall hardware performance.

2.2 Energy consumption

Throughout a long period of time memory systems were designed mainly for performance, but energy consumption was not significantly taken in consideration. For example, about 50% energy of IBM servers spent in off-chip memory hierarchy [3, 4].

IBM Research Report has introduced CapMaestro project, a new power management architecture that is used for Cloud and High performance computing data centers. IBM researchers simulated a data center with thousands of servers using publicly published load distribution data, and demonstrated that the CapMaestro architecture is able to increase the number of servers under the existing power infrastructure by 50%. It consists of sophisticated coordinated power controllers to manage power for servers with numerous power supplies. CapMaestro provides a mechanism which reduces overall stranded power and this stranded power is shifted to the servers that are currently throttled in order to gain the performance improvements [4].

Nowadays DRAM power consumption is about 50% of the total system power in average [5]. In response to power consumption problem, many manufacturers have developed DRAM with unique architecture that allows to reduce energy consumption, low-power and low-voltage variants of DRAM (for example: DDR3L, LPDDR3 and LPDDR4). LPDDR is extremely popular and extensively used in mobile devices. The power consumption for this type of memory is much efficient which is critical for mobile devices where autonomy is very important. LPDDR4 DRAM consumes 40% less power than DDR4 DRAM. DDR does not have the additional low-power modes that allows to use the lower supply voltage levels.

Difference between LPDDR generations is quite significant as well. All characteristics of LPDDR3 and LPDDR4 could be found in Mobile DRAM Stack Specification [6].

The main distinguishing features are (Table 1):

1) LPDDR4 power consumption is reduced : LPDDR4 takes 1.17 – 1.06 (with 1.1V as the most typical) whereas LPDDR3 uses 1.7 – 1.14 (with 1.2V as the most typical);

2) LPDDR4 operational speed is increased : LPDDR4 has a two channel die each being 16 Bit;

3) LPDDR4 can dynamically adjust its clock speed and save power when the power saving mode is turned on;

Table 1. Main characteristics of LPDDR3 and LPDDR [6]

Features	LPDDR3	LPDDR4
Typical Voltage (Volt)	1.2	1.1
Prefetch Buffer (Bit)	8	16
Speed (Mbps)	1600 / 1866 / 2133	3200 / 3733 / 4266
Memory density per die (Gb)	1-32	4-32

As DRAM consumes a large amount of the total system power, it is necessary to develop new low-power solutions [7]. Another problem is that it has been always difficult to get accurate power consumption data from DRAM, as:

1) Computer systems do not offer precise control over DRAM commands. Computer systems only give access to high-level operations such as loads and stores.

2) Computer systems usually do not have special monitors that track the power consumed by DRAM.

2.3 Periodic refresh

DRAM consumes power when idle and needs a periodic refresh. DRAM consists of millions of capacitors that store charge and they always have leaks. CPU or memory controller has to recharge (refresh) all of the capacitors, and this process, which automatically happens thousands of times per second, slows down the memory [8].

One of the solutions could be the replacement DRAM by static memory (SRAM), which doesn't need constant charge refreshing, as data is stored in logic gates but is more expensive: SRAM circuits require more area on a chip, because a SRAM memory cell requires four to six transistors, compared to a single transistor and a capacitor for DRAM. Some DRAM researchers are looking at getting rid of the capacitor, by storing the charge in the transistor body by using different transistor materials [9].

A group of scientists [10] proposed MicroRefresh, a memory scheme that intends to eliminate the refresh overhead in DRAM caches. Memory can be divided according to the memory location that is relative to the processor. Memory can be located on the component which is installed on the processor itself: on-chip memory. In this case, data is prefetched from memory to SRAM and can be considered ascache for DRAM. Latency rate is usually only a single cycle access time. Off-chip memory is memory "outside" of the processor chip, usually it is DRAM with much higher latency rate. MicroRefresh uses this latency difference between on-chip and off-chip DRAM and controls the balance of system resources usage by eliminating refresh of older DRAM prefetches. It endures any increase in cache misses by using the main memory bandwidth, balancing with latencies of on-chip and off-chip memory. MicroRefresh decreases the refresh energy consumed in the periodic refresh mechanism by 92%. Besides, it increases the overall performance improvements for up to 10% [10].

2.4 Scaling

Manufacturing process for DRAM is more sophisticated than for processor silicon, due to its critical nature. It has to hold data over many logic clock cycles. One of the most obvious way to lower DRAM cost is the size shrinkage (the more chips manufacturers can fit on the wafer, the lower the cost per chip).

DRAM stores charge in a capacitor which has to be large enough for reliable conductivity. Reducing the DRAM cell will make it much less reliable because sensing will be decreased with the size when the specific physical limit is reached, which is the core of "DRAM shrinking problem". Access transistor (WL - word line that controls access) should be also large enough for the low leakage and the high retention time, since the capacitor eventually leaks through it and requires refresh. If this capacitor (CAP) is too small, it becomes more vulnerable to noise and becomes more "leaky" (Figure 1). Shrinking DRAM cell size decreases the capacitor's volume, which reduces the overall effectiveness.

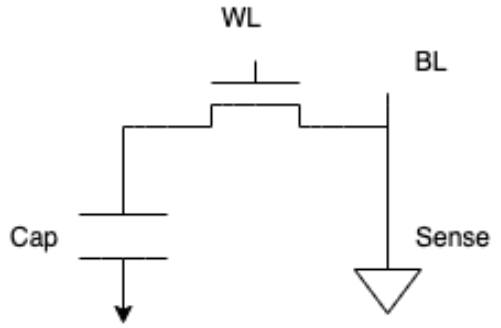

Figure 1. Scheme of charge-based memory

Cap – capacitor, WL – word line, BL – bit line, Sense – sense amplifier. (The figure is adapted from lectures of OnumMultu, Computer Architecture, Carnegie Mellon University).

DRAM technology scaling is ending, International Technology Roadmap for Semiconductors (ITRS) hasstated that DRAM will not scale easily below 35 - 40 nm (2013), but in 2015 scaling reached 22 nm and in 2020 it reached 10 nm [11] (Table 2, Table 3). For example, throughout many years 5 nm transistors for CPU were a "solid physical limitation", but the size was shrinking rapidly every year and in 2020 this limitation was reached. Now 3 nm and 2 nm process are expected to be seen in 2022 and 2023 correspondingly on the global market, but 10 nm size for DRAM is not expected to be overcome in the nearest future, which means that DRAM "loses the race" with processors not only by capacity doubling but for shrinking the size as well, which potentially can have a significant impact in performance growth of the computers overall [12]. DRAM process sizes shrank significantly in recent years, until 2016 (Table 2).

Table 2. Generations of DRAM transistors. (Data is derived from [11]).

Year	Transistor density	Class
2008	49 nm – 40 nm	4x (40 nm-class)
2010	39 nm – 30 nm	3x (30 nm-class)
2011	29 nm – 20 nm	2x (20 nm-class)
2016	19 nm – 10 nm	1x (10 nm-class)

In 2020 vendors are still shipping at the 1xnm node level. There are three sub-levels in the industry nowadays (Table 3).

Table 3. DRAM subclasses in 1x class (Data is derived from [11]).

Name of the sublclass	Transistor density	Generation
1 xnm	19 nm – 17 nm	1
1 ynm	16 nm – 14 nm	2
1 znm	13 nm – 11 nm	2

In Research &Development vendors have three more scaled generations of DRAM on the roadmap, and all of these generations are still at the 1xnm level: 1anm (Gen 4) 1bnm (Gen 5) 1cnm (Gen 6). 1anm (Gen 4) manufacturing is

expected to start in late 2021. DRAM researchers expect that decreasing DRAM cell size beyond the limits met at the 1x nanometer processes will not happen earlier than in 2025 and probably breakthrough in materials will be the trigger of the next significant DRAM shrinking.

One of the potential solutions could be the reduction in volume that can be compensated by increasing the capacitor depth or height (instead of length and width). Another potential solution of scaling is the DRAM replacement by the next-generation nonvolatile types of memories, such as:

1) STT-MRAM (Spin-Transfer TorqueMagnetoresistive RAM). It enables higher densities, low power consumption and reduced cost. STT-MRAM has the potential to become a leading storage technology since it the memory with one of the highest performance (can challenge DRAM and SRAM) that can scale well below 10 nm [13].

2) ReRam (Resistive random-access memory) - works by changing the resistance across a dielectric solid-state material, often referred to as a memristor. ReRAM is cheaper, and also faster, consumer less power and heat [14].

3) PCM (phase-change memory). Phase change memory uses a special alloys, including Germanium Antimony Tellurium (GST), which have innovative characteristics that enable the non-volatile storage. The alloy can be altered with heat to have two different states or "phases" (crystalline and amorphous) which is how data is stored. PCM provides faster write cycles, faster access time, lower power consumption.

Most of those types of RAMs were on Research &Development stage for a long period of time, but in the recent years manufacturing process has already started and presumably,there will be much more news about these types of RAMs in the next few years. In 2019-2020 Everspin Technologies and Avalanche Technology manufactured STT-MRAM with 12 nm processes. In 2020, Sony announced that it is accelerating ReRam development with an aim to commercialize it in 2021. PCM is already actively manufactured and used by many technological companies. PCM market is expected to reach 46.52 Billion USD by 2026.

2.5 Memory Interference

When more cores that are accessing the main memory are added, they interfere with each other during the access. This uncontrolled interference leads to many problems: quality of service, predictability, performance issues and affects both the overall system performance and each application's performance [2]. Application-unaware design of memory controllers sometimes leads to unpredictable interference of co-running applications in the memory system. Such uncontrolled interference can lead to denial of service to some applications, low system performance and slowdowns.

There are different approaches to mitigate interference at the different components of memory system. One of the solution was proposed by Mutlu et al. [2]. The resources could be aware of the interference (“smartresources”) or not (“dumb resources”). To make resources “smart” it is necessary to 1) allocate shared cache capacity to applications in order to make it aware of their cache utility, and 2) modify the cache replacement and insertion policies, make them aware of the data reuse and memory access behavior of co-running applications. Interference is detected by means of monitoring their access characteristics and allocate resources.

Subramanian et al. [15] proposed to separate applications into two groups only: one group contains interference-causing applications, and the other group contains vulnerable-to-interference applications. Then, they prioritized the vulnerable-to-interference group over the interference-causing group. This scheme reduces the hardware complexity and critical path latency of the memory scheduler. It also improves system performance.

The “dumb resources” approach does not modify the resources to make them application-aware but allows to control the resources and their allocation at different points in the system to mitigate performance degradation. Kayiran et al. [16] proposed to throttle the thread-level parallelism of the GPU to mitigate memory contention-related slowdowns in heterogeneous architectures consisting of both CPUs and GPUs.

Another proposed approach is Fairness via Source Throttling (FST), which throttles applications at the source (processor core) to regulate the number of requests that are sent to the shared caches and main memory from the processor core. Also one of the approaches is to map applications to cores (by modifying the application scheduler in the operating system) to make application aware about memory access characteristics [2].

There are many more resolutions of memory interference problem. Most of them requires changes in the architecture of the computer system which is usually responsible for the memory interference.

2.6 Rowhammer

2.6.1. The core of the problem

“Rowhammer” problem is a critical issue affecting modern DRAM chips that allows attackers to get kernel privileges on a targeted system by repeatedly accessing memory cells and induce bit flips. Different hardware-based techniques exist to prevent the Rowhammer effect from occurring, including required support in some processors and types of DRAM memory modules. Rowhammer involves the execution of a program over and over on a "row" of transistors in a computer's memory chip. The idea is to access that row constantly, until it leaks some electricity into the adjacent row(s). That leakage can cause a bit in the target row to "flip" from one position to another, from one row to another, slightly altering the data stored in memory. After the leakage, the skilled Rowhammer attacker can start to exploit these tiny data changes to gain more system access [17].

It’s called “Rowhammer” because the core idea of issue is to attack a row of the bits in the memory (Figure 2a). This row is “hammered”, because it’s being constantly attacked with intention to trigger the leakage (Figure 2b).

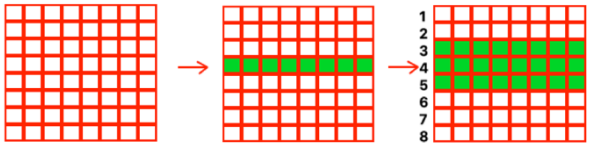

Figure 2. Memory rows (a); “hammered” row (b); leakage to the adjacent rows / bit flips (c).

When retrieving the data from the row, the information can leak through to the other adjacent rows (Figure 2c). It is happening because when retrieving the data constantly, the charge is drained faster in a particular row, the charge from

the adjacent rows can be depleted and some of the bits can flip around. For example, memory in the attacked row is used by a browser (Figure 2c, row #4), but adjacent rows (Figure 2c, row#3 and row#5) are used by a part of operating system that pertains to permissions. Since those bits were flipped (some bits of rows #3 and #5 leaked to the row #4, Figure 3c), now this information can be accessible in the row that attacker constantly trying to access, the row #4. It suffices to change one bit to get a permission and take over a program.

2.6.2. The Rowhammer appearance

The problem appears in 2010 because the capacity of memory started to grow, "rows" started to be denser, and some key physical limitations were reached which led to more noise and easier ways of charge movement. 85% of DRAM that were manufactured after 2011 vulnerable to Rowhammer problem. "Rowhammer" attack affects many DDR3 and DDR4 SDRAM modules and can alter data stored on those types of modules [18]. Increasingly sophisticated Rowhammer exploits allow an attacker that can execute code on a vulnerable system to escalate privileges and compromise browsers. In all these attacks, the common assumption is that attackers first need to obtain code execution on the victim machine to be able to exploit Rowhammer either by having unprivileged code execution on the victim machine or by attracting the user to a website that employs a malicious JavaScript application which starts to "hammer" the memory row in the operating processes [19].

In October 2016, a group of researchers published DRAMMER, an Android application that uses row hammer, together with other methods, to reliably gain root access on several popular smartphones, so LPDDR is vulnerable to Rowhammer as well [20].

2.6.3. Rowhammer test made by Google allows testing user memory for the DRAM "Rowhammer" problem[21].

There is the general test to find out if the current DRAM is vulnerable to Rowhammer. Google use a probabilistic approach for picking memory locations: a program repeatedly picks random pairs of addresses. If a machine has 8 banks of DRAM, there should be a 1/8 chance that the two chosen addresses corresponds to the same bank. The test reserves a block of memory and repeatedly picks more than 2 random addresses within the chosen block, hammers them, and checks if the bit flip takes place in this block. If the program detects a bit flip, Rowhammer problem exists in this type of memory. If it never detects a bit flip, the program will never be terminated.

2.6.4.Methods for successful mitigation, correction, prevention of the Rowhammer problem [17, 22, 23].

1) More frequent memory refreshing. It is possible to make more refreshes with intervals less than the default 64 ms. To avoid unnecessary refreshes, it is also possible to build a counter to track the quantity of accesses for each memory row and to proactively refresh rows-neighbors that are accessed more frequently. There is the obvious overhead, higher power consumption and processing overhead.

2) Target row refresh (TRR), is the hardware built-in feature that prevents the Rowhammer effect without negatively impacting performance or power consumption. It was firstly introduced in the LPDDR4 mobile memory standard published by JEDEC [24]. Besides, some companies have started to embed TRR in their DDR4 products, despite the fact it is not a part of the DDR4 memory standard. But it is important to remember that there is a large amount of legacy DDR that were produced and actively used since 2011.

3) Using of ECC RAM (Error Correction Code). ECC RAM is able to check whether the positions of bits are the same or it was changed by flipping. In the case when the positions are different, it means that bit flip took place, but ECC RAM memorized the previous positions and is able to correct the positions of bits. It doesn't work properly in some complexed cases like leaking from several rows at the same time.

4) Monitoring every memory access which is extremely expensive overhead.

There are many more methods that are less successful or have a larger overhead. Besides, all solutions are only partial and not covering all Rowhammer cases. The complete and standardized solution to evade this attack is only expected to be found, but the solutions #2 and #3 are relatively solid.

3. CONCLUSION

There are a lot of problems and challenges that memory systems encountered in the recent years. There are many more challenges and problems, but the author focused on the main ones. In the next 5-10 years a lot of changes are expected in the Memory Systems and this work has highlighted some of the major problems that probably will be addressedeven more in the future. The author have discussed the states of those challenges for the last 5 years, summarizing potential solutions as well as expectations for the following years.

Short summary of some main problems and potential solutions that were summarized in the paper:

For *energy consumptionproblem* – new power management architecture (CapMaestro) and new types of LDDR (low-power) memories. For *memory refreshing problem* the solution is quite similar: DRAM replacement - by static memory (SRAM) which doesn't require refreshing, and new scheme for eliminating the refresh overhead (MicroRefresh). *Scaling problem* - it's difficult to overcome 10 nm barrier, so replacement by the next-generation nonvolatile memories is expected. Besides, some DRAM researchers consider the solution to get rid of the capacitor, by storing the charge in the transistor body by using different transistor materials. *Memory interference* – new application development architecture which is aware of the memory interference and some changes in operating systems. For example, by modifying the application scheduler or by making intentional processor core throttling to regulate requests. There is no final solution for *Rowhammer problem*, but there are some more or less successful ones: more often memory refreshing, monitoring all memory accesses, manufacturing new types of memory that supported target row refresh technique or using of ECC RAM.

4. REFERENCES

[1] LimK., Jichuan C., Mudge T., Ranganathan P., Reinhardt S., Wenisch T. 2009. Disaggregated Memory for Expansion and Sharing in Blade Servers. ISCA '09 2009 Computer Science.

[2] Mutlu, O., Meza, J., Subramanian, L. 2015. The Main Memory System: Challenges and Opportunities.

Communications of the Korean Institute of Information Scientists and Engineers, 33, 16-41.

[3] Lefurgy C., Rajamani K., Rawson F., Felter W., Kistler M. Tom K. 2003. "Energy Management for Commercial Servers", IEEE Computer, pp. 39-48, December, 2003.

[4] Li Y., Lefurgy C., Rajamani K., Allen-Ware M., Silva G., Heimsoth D., Ghose S., Mutlu O. 2018. CapMaestro: Exploiting Power Redundancy, Data Center-Wide Priorities, and Stranded Power for Boosting Data Center Performance, IBM Research Report, RC25680, March 28, 2018.

[5] Micron. 2017. Technical Note Calculating Memory Power for DDR4 SDRAM.

[6] Samsung (2016). Mobile DRAM Stack Specification.

[7] Ghose S., Yaliki A., Gupta R, et al. 2018.What Your DRAM Power Models Are Not Telling You: Lessons from a Detailed Experimental Study. SIGMETRICS 2018.

[8] Tanenbaum A. 2005. Structured Computer Organization (5th. ed.). Prentice Hall PTR,USA.

[9] Cheng W., Shen P., Li.X. 2019. Retention-Aware DRAM Auto-Refresh Scheme for Energyand Performance Efficiency Micromachines 2019, 10(9), 590.

[10] Gulur N.,Govindarajan, R., Mahesh M. 2016. MicroRefresh: Minimizing Refresh Overhead in DRAM Caches.MEMSYS '16: Proceedings of the Second InternationalSymposium on Memory Systems,pages 350–361.

[11] Mellor C. 2020, April 13. Why DRAM is stuck in a 10nm trap.Blocksandfiles.

[12] LaPedus M. 2016, February 18. 1xnm DRAM Challenges. Semi-engineering.

[13] MRAM-Info.2018-2020.

[14] RRAM-Info. 2018-2020.

[15] Subramanian L.,Usui H., Chang K., Mutlu O. 2016.DASH: Deadline-aware high-performance memory scheduler for heterogeneous systems with hardware accelerators. ACM Transactions on Architecture and Code Optimization (TACO).

[16] KayiranO. et al. 2019. Quantifying Data Locality in Dynamic Parallelism in GPUs. ACM SIGMETRICS Performance Evaluation Review 47(1):25-26.

[17] Tatar A., Krishnan R., Athanasopoulos E., Giuffrida C., Bos H., and Razavi K. 2018. Throwhammer: Rowhammer Attacks over the Network and Defenses. USENIX ATC ’18. July 11–13.

[18] Kim Y. et al. 2014. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. 2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA), Minneapolis, MN, 2014, pp. 361-372.

[19] Seaborn M.2015, March 9. Exploiting the DRAM rowhammer bug to gain kernel privileges.Blog: Google Project Zero.

[20] Vusec. 2018.Drammer: Flip Feng Shui Goes Mobile.

[21] Google. 2015, August 10. Program for testing for the DRAM "rowhammer" problem. Github.

[22] Restifo M., Bernardi P., De Luca S., Sansonetti A. 2017. On-line software-based self-test for ECC of embedded RAM memories," 2017 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), Cambridge, 2017, pp. 1-6.

[23] Newman L.H. 2018, November 21. An Ingenious Data Hack Is More Dangerous Than Anyone Feared.Wired.

[24] JEDEC. 2019, February 19. Updates Standard for Low Power Memory Devices: LPDDR5.