Rethinking DRAM's page mode with STT-MRAM

Byoungchan Oh, Nilmini Abeyratne, Nam Sung Kim, *Fellow, IEEE,* Jeongseob Ahn, *Member, IEEE,* Ronald G. Dreslinski, *Senior Member, IEEE,* and Trevor Mudge, *Life Fellow, IEEE*

Abstract—STT-MRAM is a promising candidate for drop-in replacement for DRAM-based main memory because of its higher energy efficiency and similar latency to DRAM. However, simply replacing DRAM with STT-MRAM without optimizations severely limits STT-MRAM from exploiting its full potential. STT-MRAM employs costly sense amplifiers that demand an order of magnitude more area and power than DRAM. To manage the high cost, STT-MRAM shares one sense amplifier across multiple bit-lines, exploiting the non-destructive nature of its read operation. This sense amplifier sharing reduces the size of row buffers; as a result, it incurs higher activation energy and lower performance. Other issues arise if STT-MRAM is required to be compatible with DRAM interfaces and policies.

To address these challenges in a cost-effective manner, we propose SMART that, unlike DRAM and conventional STT-MRAM, waits to do bit-line sensing until after receiving a column access command instead of a row activation command. This results in several benefits: larger pages, fewer sense amplifiers, lower activation power, higher bank-level parallelism, shorter latency, fewer address pins, and more efficient repairing of defective columns than conventional STT-MRAM. Our evaluation shows that SMART consumes lower energy while providing higher performance than conventional STT-MRAM and DRAM. Additionally, SMART consumes less area compared to conventional STT-MRAM.

Index Terms—STT-MRAM, DRAM, DDR, Non-Volatile Memory.

1 Introduction

Spin Torque Transfer Magnetic Random Access Memory(STT-MRAM) is an emerging, promising, drop-in replacement for DRAM because its faster speed and has higher endurance than other non-volatile memory (NVM) technologies [1], [2], [3]. However, it has a few disadvantages. One such disadvantage is the current-sensing sense amps that takes up an order of magnitude more space and consumes higher power than voltage-sensing sense amps that DRAM uses. To offset the cost of the large current sense amps, STT-MRAM leverages the non-destructive nature of its read operation to place fewer sense amps than bit-lines: sharing one sense amp with every 16 to 128 bit-lines in each bank [4], [5], [6]. This trick suffers from two limitations.

First, fewer sense amps leads to smaller row buffers (i.e. small page size) since the number of sense amps determines the size of row buffers [7]. Large pages have higher performance because more row-buffer hits are likely when data locality is high, but they also consume more energy when data locality is poor (known as the overfetching problem). Small pages, in contrast, consume less energy when locality is poor, but give lower performance with more row-buffer misses when locality is good. STT-MRAM row buffers are a lot smaller than DRAM row buffers, but still larger than a column access. Therefore, STT-MRAM suffers more row-buffer misses than DRAM without completely eliminating the overfetching problem.

Second, when STT-MRAM attempts to adhere to policies designed for DRAM, it suffers from a column address fragmentation problem [2], [6]. Specifically, DRAM requires 8,192 sense amps to sense all the bit-lines of a 1KB page and latch the data until the next row is opened. If not latched, the data will be lost due to the destructive nature of DRAM's read operation. Therefore, DRAM performs both activation and sensing at the same time with a single row activation (`ACT`) command. Applying the same activation/sensing method to STT-MRAM would require sending both a row address and a partial column address at the same time in order to select a subset of bit-lines to connect to the handful of available sense amps through multiplexers. That is, STT-MRAM with the same capacity as DRAM requires more address pins to send not only a row address but also some part of a column address together with the row command[1]. Besides, such a fragmentation of a column address between row and column commands considerably worsens efficiency and flexibility of column repair mechanisms as a row or column command has only partial column address information [8].

To address these shortcomings, we propose SMART, a new cost-effective STT-MRAM, that implements an activation and sensing policy that builds on the strengths of STT-MRAM . Specifically, exploiting non-destructive reads in STT-MRAM, we propose to provide only 64 sense amps[2] per bank and make a read command instead of a row activation command sense bit-lines. This deceptively simple change, which is not possible for DRAM because of the destructive nature of its read operation, offers the following advantages over conventional designs.

- *B. Oh and N. Abeyratne M. were with University of Michigan, Ann Arbor, MI, 48109. Their current affiliation is Intel Corporation. E-mail: bcoh@umich.edu*
- *N. S. Kim is with University of Illinois at Urbana-Champaign, Champaign, IL, 61820.*
- *J. Ahn is with Ajou University, Suwon, Korea.*
- *R. G. Dreslinski and T. Mudge are with University of Michigan.*

Manuscript received Mar 3, 2021; revised May 23, 2022.

1. DRAM uses the same set of address pins to receive both row and columns addresses in a time multiplexed manner. Since a row address typically needs more bits than a column address, the number of row address bits determines the number of address pins in DRAM.

2. In this paper, we assume ×8 DDR4 DRAM. Therefore, each device must transfer 64 bits to support the burst length of 8 for a single column access command. In addition, the page size in this paper means a page size per chip (*i.e.,*1KB), but not per rank (*i.e.,*8KB).

(1) SMART offers the illusion of providing 16× larger row buffers with 8× fewer sense amps than conventional STT-MRAM. That is, SMART gives you access to a 1KB page with only 64 sense amps per bank, whereas conventional STT-MRAM gives you access to only 64B pages with 512 sense amps per bank. Furthermore, SMART saves latency on subsequent column accesses to an open row. Conventional STT-MRAM repeatedly results in a longer delay ($\texttt{tRC} = 28.5ns$) and higher power to access different columns in the same row which were not selected and sensed by the previous row command. This is because conventional STT-MRAM sees such memory accesses as row-buffer misses and needs a new row command to select and sense different bit-lines. In contrast, SMART recognizes such memory accesses as row-buffer hits and only needs another column access command to select and sense different bit-lines which were already connected to the necessary cells by the previous row activation command.

(2) SMART consumes ∼87% less activation power than conventional STT-MRAM. Specifically, DRAM and conventional STT-MRAM senses 8,192 and 512 bit-lines as part of a row activation, respectively, and sensing power dominates the activation power. In contrast, SMART consumes 11 and 8× less sensing power than DRAM and conventional design because it senses only 64 bit-lines (*i.e.*, granularity of a column access) as part of a column access. Hence, SMART practically eliminates the overfetching problem. Although the sensing power that we remove from the activation power must be added to read power, elimination of overfetch means that only the exact number of sense amps (SA) are activated, thus avoiding unnecessary sensing power. This results in a huge power reduction especially when the page-hit rate is low. Furthermore, whenever a memory write access demands an activation of another row, DRAM and conventional STT-MRAM unnecessarily consumes sensing power because sensing is part of the row activation. However, SMART does not consume any sensing power for such a memory write access, because sensing is not part of a row activation but part of a column read access.

(3) SMART offers shorter latency for memory accesses. Specifically, high sensing power imposes `tRRD` and `tFAW` constraints in DRAM and STT-MRAM and limits the number of row activation commands in a certain time period. As SMART significantly reduces sensing power, it can eliminate these two constraints and handle more row activation commands in a shorter time period than conventional designs (*i.e.*, higher bank level parallelism (BLP)). As previously mentioned, DRAM and conventional STT-MRAM performs sensing as part of row activation while sensing is unnecessary for memory write accesses. Moreover, sensing constitutes a notable fraction of activation time. Therefore, SMART can also reduce latency of memory write accesses especially to different rows compared with conventional STT-MRAM.

(4) SMART needs fewer pins and offer 16× more efficient repairs of defective columns than conventional STT-MRAM. In particular, SMART does not select, connect, or sense specific bit-lines as part of row activation. That is, it needs to receive only a row address for activation like DRAM. Additionally, because SMART receives the full column address with a column command, it can use the same efficient mechanism as DRAM for repairing defective bit-lines. Unlike SMART and DRAM, conventional STT-MRAM only receives partial column address with both row and column commands. The lack of the column address information results in inefficient repair in conventional STT-MRAM.

(5) SMART eliminates the need for sending separate pre-charge commands because SMART can overlap closing an already open row and opening a new row during the row activation. As a result, SMART not only offers 11% shorter latency for memory accesses which are directed to the same bank but incur row-buffer misses, but also consumes less command bus bandwidth than conventional STT-MRAM.

In summary, (1)–(5) reduce energy and improve performance. Our evaluation shows that SMART consumes 16% and 29% lower energy while providing 8% and 3% higher performance than conventional STT-MRAM and DRAM on average, respectively. In addition to these benefits, SMART is 8% smaller than conventional STT-MRAM.

2 Challenges in Architecting STT-MRAM

In this section, we identify challenges and limitations in conventional STT-MRAM and DRAM.

2.1 Large Sense Amps with High Power Consumption

STT-MRAM offers cell read/write speed comparable to DRAM and good endurance ($> 10^{15}$) [9]. A prior study demonstrated 4Gb LPDDR2-compatible STT-MRAM with $9F^2$ cell size and sub-$50ns$ read/write speed [2]. Such characteristics make STT-MRAM a promising alternative to $\sim 7F^2$ DRAM.

However, STT-MRAM poses unique challenges in implementing SAs [10], [11]. First, the SA needs a reference current generator to sense small differences in on/off resistance. This requires STT-MRAM to adopt a complex current SA that is an order of magnitude larger than a voltage SA in DRAM.

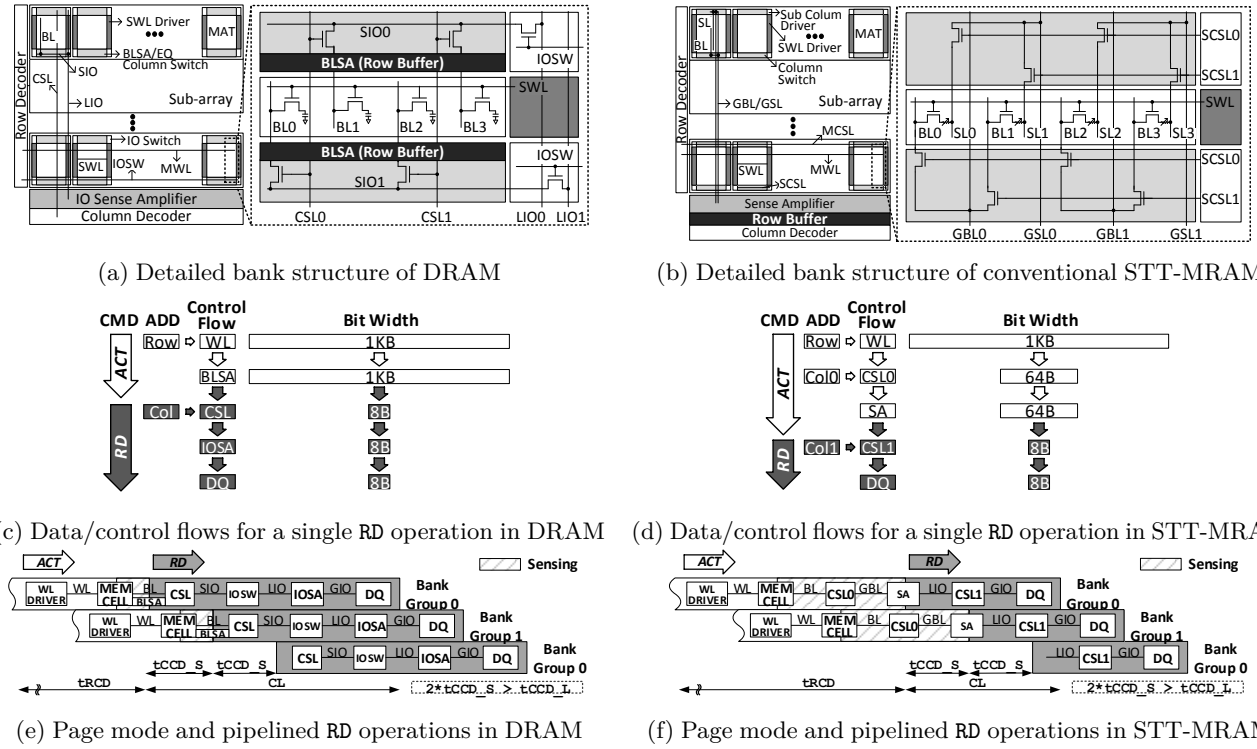
Second, sensing in STT-MRAM consumes high power. DRAM SAs simply charge/discharge bit-lines (BLs) once per sensing, whereas STT-MRAM SAs need to continuously flow current to BLs until they reach a level sufficient for sensing. To eliminate the power consumed by the continuous current flow, data is copied from the SAs to separate buffers and turned off immediately after sensing BLs [10], [12].

2.2 Page Mode Operation

Page mode is an operation mode of DRAM where accessing one row allows accessing multiple columns. In other words, the page mode avoids repeatedly opening the same row to access a different column. Such an operating mode has not changed in modern DRAM [13]. Workloads with high spatial locality can improve performance and save energy using page mode. In addition, since memory access is divided into row and column accesses, row and column addresses arrive at different times. This allows row and column addresses to share pins and save on pin cost. In general, off-chip pin counts are critical because of system design limitations [14]. Furthermore, page mode saves area because its bulk operation in the row access allows the sharing of decoders, drivers, and other circuits across a large number of memory cells [7].

There are several drawbacks to page mode. The most well-known is the overfetching problem under poor spatial locality [15]. In some modern workloads and multi-core systems, the degree of the spatial locality tends to be low [16]. In

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2022.3207131



(a) Detailed bank structure of DRAM

(b) Detailed bank structure of conventional STT-MRAM

(c) Data/control flows for a single RD operation in DRAM

(d) Data/control flows for a single RD operation in STT-MRAM

(e) Page mode and pipelined RD operations in DRAM

(f) Page mode and pipelined RD operations in STT-MRAM

Fig. 1: Bank architecture and operation of DRAM (left) and conventional STT-MRAM (right).

addition, the latency is variable depending on page hit and miss. There was an attempt in the memory industry to introduce reduced latency DRAM (RLDRAM) that had SRAM-like non-page mode [17] and no overfetching problems. Despite its superior performance, the cost-sensitive main memory market has not widely adopted RLDRAM. The memory structure to support non-page mode results in huge area overhead and only enables low-density DRAM (*e.g.*, ∼1Gb). The motivation of this paper is to eliminate the inefficiency of the page mode while preserving its benefits.

2.3 Limitations with Shared Sense Amps

Fig 1a describes the DRAM bank structure. DRAM has bit-line sense amps (BLSAs) both above and below a sub-array because charge sharing limits the BL length [2], [18]. DRAM couples each BL with a dedicated SA in every sub-array due to the destructive nature of its read operation. That is, all 8,192 cells (1KB) connected to a word-line (WL) are sensed and restored by 8,192 SAs through BLs during activation.

Traditional I/O interconnect of DRAM has a hierarchy. The wire from a memory cell to a SA is called the BL and the wire from a SA to DQ is called IO. During a column access, 64 BLs and BLSAs in a sub-array are selected by column selection lines (CSLs) and are connected to segmented I/O (SIO) lines through column switches (*i.e.*, multiplexers). Then, the SIO lines are connected to Local I/O (LIO) lines running vertically through I/O switches (IOSWs). Lastly, larger IO sense amplifiers (IOSA) are placed near the column decoder of each bank to assist data transfer through the LIO lines because LIO lines are too long to be driven by small BLSAs. Fig. 1c summarizes the flow from activation to read data out. In page mode, when an ACT is received, DRAM senses cell states of a row and stores them into a row buffer[3]. This allows DRAM to precharge/activate a row once per multiple column accesses, as depicted in Fig. 1e.

Fig. 1b describes a conventional STT-MRAM bank structure. STT-MRAM decouples SAs from BLs with multiplexers and shares one SA with 16 to 128 BLs [2], [4], [6]. This approach takes advantage of the non-destructive nature of STT-MRAM read operation to manage the cost of SAs. Fig. 1d describes serving a read request in STT-MRAM following the same page mode as DRAM but sharing one SA with 16 BLs to reduce the cost. An ACT command first asserts a word-line (WL) connected to 8,192 cells (1KB). Since there are only 512 SAs (*i.e.*, 64B row buffer), a column selection signal (CSL0) uses part of the column address (Col0) to select one BL out of 16 BLs. The column selection signal (MCSL-SCSL) in an STT-MRAM bank corresponds to the WL selection signal (MWL-SWL). The remaining part of the column address (Col1) provided in the column access command (RD or WR) selects global bit-lines (GBLs). The page mode in the traditional STT-MRAM is described in Fig. 1f. Such STT-MRAM, however, suffers from the following limitations.

Limitation 1: Longer latency. Unlike DRAM, where each sub-array has BLSAs, STT-MRAM does not require BLSAs, placing only column multiplexers above and below a sub-array with one set of SAs at the bottom of a bank [2]. The column multiplexers connect one BL in a group of BLs to a SA through a GBL, which makes the sensing path of STT-MRAM longer than that of DRAM. The sensing path of STT-MRAM equals the height of a bank, while in DRAM it is the height of a sub-array. The longer sensing path increases tRCD—minimum time from ACT to RD or WR —while decreasing CL—time between the

3. SAs shared by two adjacent sub-arrays serve as a row buffer.

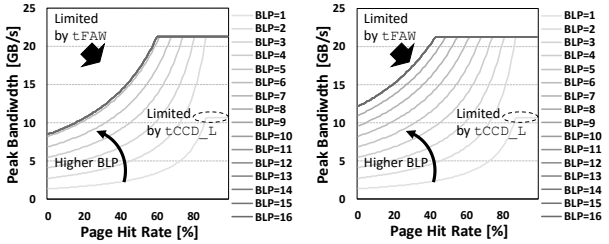
moment at which a RD command is sent and the moment at which its first data is out (D_{OUT}), as shown in Fig. 1f [6].

Limitation 2: Smaller pages. Consider STT-MRAM sharing one SA with N BLs where N is 16∼128 in prior work [2], [4], [6]. When STT-MRAM has 8,192 cells connected to a WL like DRAM, the page size of such STT-MRAM becomes $1/N$ of that of DRAM. Page-size-sensitivity analyses in our background research indicated that streaming benchmarks with sequential accesses can see IPC degradation as high as 33% for a page size of 1/16 of 1KB. Although the small pages reduce power, large pages generally perform better in many cases [15]. With fewer SAs than BLs (*i.e.*, cells in a row), STT-MRAM encounters the following three cases: (1) an access to another row (*i.e.*, row miss); (2) an access to the same row with BLs selected and sensed by previous ACT (*i.e.*, row hit); or (3) an access to the same row but BLs which are not selected and sensed by previous ACT yet (*i.e.*, row hit but row buffer miss). The third case needs to be handled like a row miss because connecting appropriate BLs to SAs and sensing them are coupled with ACT in STT-MRAM sharing one SA with many BLs. That is, some of a physical column address becomes part of a logical row address since they are needed before sensing appropriate BLs.

Consequently, selecting different BLs which were not selected by the previous ACT always demands another ACT. Because new activation can be performed only after deselecting the old BLs, the third case (row hit but row buffer miss), also generates a new precharge (PRE) prior to the ACT [2], [12], [19].

Limitation 3: Lower repair efficiency. Splitting a column address between row activation and column access commands, *referred to as column address fragmentation in this paper*, creates two important challenges in managing chip yield and compatibility with existing DDR interfaces. Typically, the minimum repair granularity is a row or a column. Any redundant WL can replace any defective WL in the same bank and any redundant BL can replace any faulty BL in the same mat. This technique is known as any-to-any replacement [20]. In an STT-MRAM architecture that shares one SA with multiple BLs, however, neither ACT nor RD/WR have the full column address information. STT-MRAM uses the partial column address from ACT to select one BL in each of the 64 BL groups in each mat. Therefore, if the partial column address were to replace a defective BL in a BL group, it needs to replace every BL selected by the same partial column address. Consequently, STT-MRAM needs at least one redundant BL for every BL group (*i.e.*, 64 redundant BLs per mat). On the other hand, STT-MRAM uses the partial column address from RD or WR to select 64 BL groups from 512 BL groups (*i.e.*, the number of SAs). Hence, if it were to use the partial column address to replace a defective BL in a BL group, it needs to replace every BL in the BL group. That is, STT-MRAM needs at least one redundant BL group in each mat, (*i.e.*, 16 BLs per group). Both cases negatively affect chip yield because they limit the flexibility of replacement and increase the minimum repair granularity.

Limitation 4: Higher pin cost. The number of address pins in the page mode are equal to the number of row address bits because there are typically more rows than columns (*e.g.*, 64K rows and 1K columns in ×8-8Gb DRAM). In the shared SA structure, however, more address pins are needed to send both the row address and a partial column address (to assist BL selection) at once. For example, 4 more address pins are required when $N = 16$. Therefore, the shared SA architecture is not compatible with conventional DDR interfaces. To solve this problem, comboAS [6] and [2] propose waiting to start row activation until after RD or WR sends a column address; but this always increases CL by the row activation time. LPDDR2-NVM [19] proposes another command, PREACT to deliver a part of the row address before sending ACT so that LPDDR2-NVM can compose a complete row address after receiving the ACT command. However, PREACT needs to know the row address for the next ACT at precharging time. Otherwise, an extra PREACT should be issued before ACT to compose the complete row address. This extra step increases row activation time while consuming more command bus bandwidth.



(a) 2KB page size (b) 1KB page size

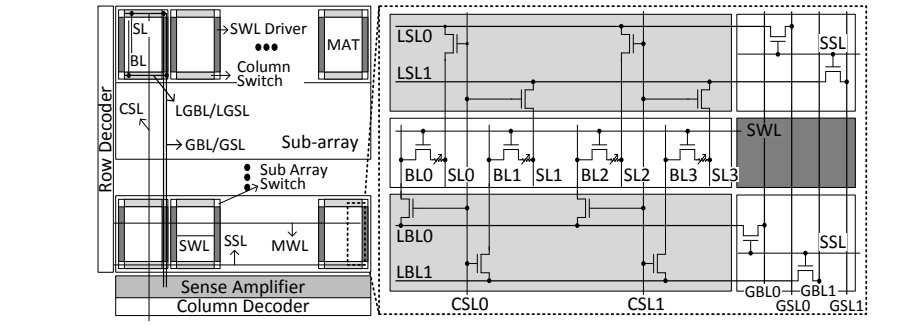
Fig. 2: Memory bandwidth change according to page-hit rate and bank-level parallelism of DDR4 DRAM [13].

Limitation 5: Limited bank-level parallelism. In the conventional page mode, activating a page of memory cells (*e.g.* 8,192) and as many SAs draws a large amount of current. As a result, the internal supply voltage drops and requires time to recover from the voltage drop of the power delivery network. This recovery time is enforced by tRRD (RAS to RAS delay) and tFAW (four activation window). Both timing restrictions limit bank interleaving and can negatively affect the performance when page-hit rate is low (random memory access). As shown in Fig. 2, high memory bandwidth can be achieved by high page-hit rate, high bank-level parallelism (BLP), or both. Ideally, bank interleaving hides long page-miss latency and achieves high bandwidth for page misses. The number of required banks to hide the latency increases as page-hit rate lowers (higher BLP in Fig. 2). However, since tFAW limits the number of activated banks to four during its time window, not enough banks to hide the latency can be activated and the insufficient bank interleaving leads to bandwidth drop. The bandwidth loss is proportional to the page size because larger page size has higher activation power and longer tFAW. Although the number of activated SAs is decreased by $1/N$ in the shared SA structure, each STT-MRAM SA consumes higher power than DRAM's SA. Therefore, a similar timing constraint to tFAW would have to be imposed to the shared SA structure.

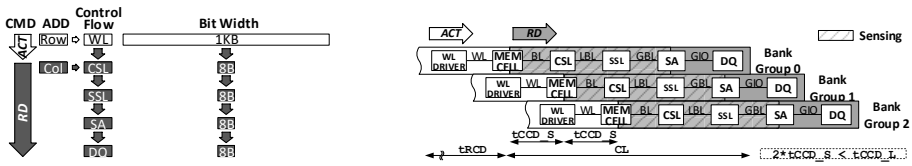
3 SMART Architecture

Due to destructive reads in DRAM, the SAs have to sense every cell and write back in the row after ACT asserts the WL. This requires the number of SAs to be equal to the number of cells in a row. Therefore, the number of cells in a row determines both the page size and the activation power in DRAM. This makes it impossible for DRAM to offer both the high performance of

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2022.3207131



(a) Bank architecture



(b) Data and control flow (c) Page mode operation and pipelined RD commands

Fig. 3: SMART bank architecture and data/control flow.

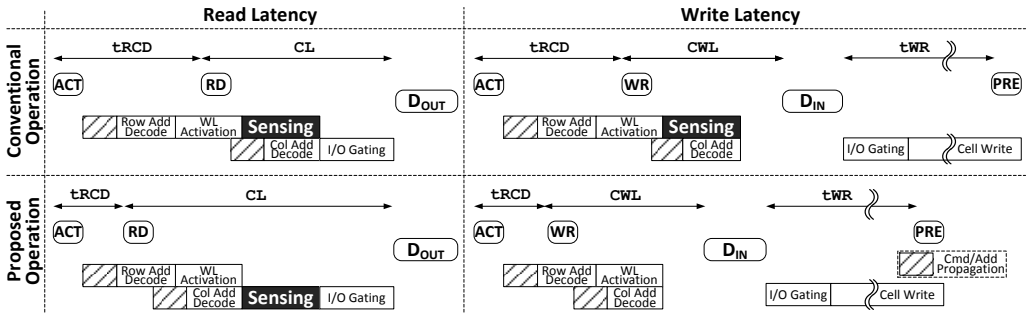


Fig. 4: Change of read and write accesses after making sensing operation triggered by a RD command.

large pages and the low power of small pages at the same time. In contrast, the non-destructive reads in STT-MRAM mean that SAs do not need to sense cell states after ACT asserts the WL. Exploiting such a property, we re-define the page mode operation to sense BLs following the RD command instead of the ACT command. We preserve the original intent of the page mode concept in the sense that whole pages are accessed without repeated activation. In the remainder of this section, we first present the details of the SMART architecture and then discuss the five key benefits of SMART over conventional STT-MRAM and/or DRAM.

3.1 Re-architecting STT-MRAM

In SMART, the ACT command ends after asserting the WL. That is, ACT does not sense any BL. The subsequent RD command will sense only the 64 BLs specified by the column address. This order of operations allows SMART to provide any page size with only 64 SAs per bank along with other significant benefits which will be discussed in Sec. 3.2. To efficiently support such ACT and RD commands for SMART, we propose the bank architecture depicted in Fig. 3a.

The data wiring in SMART is similar to DRAM, but different from the conventional STT-MRAM. The SIO, LIO and IOSW in DRAM correspond to Local BL (LBL), GBL, and sub-array selection line (SSL), respectively, in SMART. However, SMART does not need 8,192 SAs to implement a 1KB page like DRAM and it does not limit itself to implementing only $1/N$ of a page like the conventional shared SA structure.

This SMART bank architecture does not increase latency of memory read accesses, because the total amount of time for performing ACT and RD remains the same as conventional STT-MRAM. As shown in Fig. 4(left), compared with conventional STT-MRAM, SMART simply reduces the amount of time for ACT (tRCD) while increasing the amount of time for RD (CL). On the other hand, as shown in Fig. 4(right), SMART can decrease latency of memory write accesses. Specifically, compared with the coventional STT-MRAM, ACT-WR does not consume any time for sensing BLs without affecting CWL, time between the moment at which a WR command is sent and the moment at which its first data (D_{IN}) is placed. Although SMART does not directly reduce the sensing time, removing the sensing operation from tRCD improves the overall latency.

In DRAM, the SIO-LIO lines are electrically isolated from the GIO lines, as shown in Fig. 1a. This allows DRAM to internally pipeline more than two consecutive RD commands which can be issued at every tCCD_S interval (CAS to CAS delay for different banks accesses, typically ~3ns) without waiting for long CL (typically ~14ns) when combined with bank interleaving operation, as shown in Fig. 1e. Similarly, SMART can allow multiple consecutive RD commands with tCCD_S

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2022.3207131

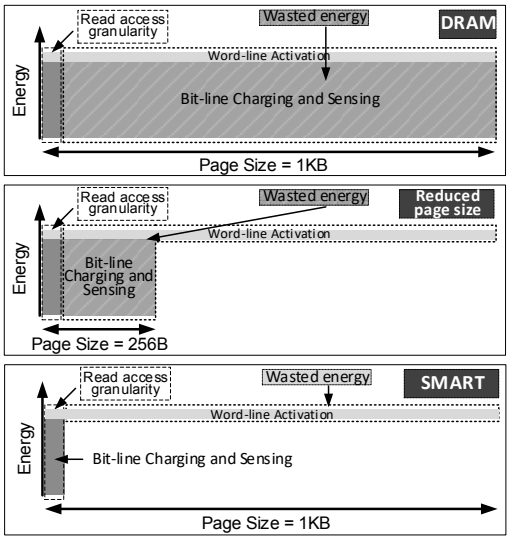
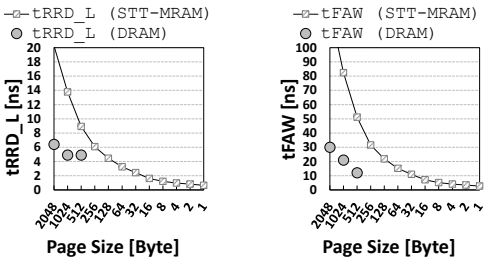


Fig. 5: Relation between page size and ACT energy.



(a) tRRD_L (b) tFAW

Fig. 6: tRRD and tFAW for various page sizes. tRRD_L and tFAW for DRAM are obtained from a DDR4 datasheet [13].

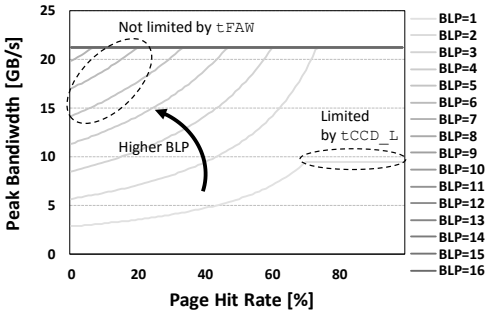


Fig. 7: Improved bandwidth of STT-MRAM due to the relaxed tFAW and reduced tRC.

interval by interleaving bank groups as shown in Fig. 3c. Thus, such long sensing time and CL in SMART can be hidden and consecutive RD commands can be served the same as DRAM.

In a prior work, the number of LBL and GBL lines, column multiplexers, and SAs were doubled to hide the long sensing time for consecutive RD commands [21]. However, unlike the prior work, this work enables multiple consecutive RD commands by exploiting DDR4's bank group structure without introducing dual sensing paths that increase the bank area.

3.2 Benefits

With the bank architecture presented in Sec. 3.1, SMART provides the following notable benefits over conventional STT-MRAM and improves performance while saving energy.

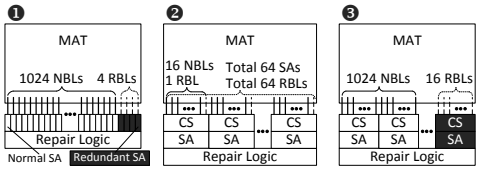
Benefit 1: Larger pages and fewer SAs. SMART's re-defined page mode can give the illusion of larger pages with fewer SAs than the conventional STT-MRAM. Specifically, ACT only asserts the WL to connect 8,192 cells to 8,192 BLs. It is a subsequent RD command that selects the appropriate 64 BLs and senses them. Then, SMART needs only 64 SAs per bank for 1KB pages whereas conventional STT-MRAM implements 512 SAs per bank for 64B pages. Because conventional STT-MRAM has 16× fewer SAs than BLs, it repeatedly consumes a long tRC, which is the time interval between two ACT commands to the same bank for accessing different columns in the same row that were not selected and sensed by the previous ACT command. SMART, on the other hand, does not consume tRC for consecutive column accesses because it only needs another RD command to connect and sense these columns. This significantly reduces the latency of sequential memory accesses. Note that the SAs in conventional STT-MRAM consume ∼9% of STT-MRAM space based on an adapted version of DRAMSpec [22]. That is, SMART can reduce the space and activation power of the SAs to 12.5% of conventional STT-MRAM.

Benefit 2: Lower activation power with fewer SAs. ACT energy is a large portion of total energy in DRAM. Fig. 5 illustrates the relationship between page size and ACT energy in DRAM, STT-MRAM and SMART. In a DRAM, WL activation for a 1KB page connects 8,192 cells to 8,192 BLs. Although we consider higher WL voltage (V_{PP}=1.8V) than the BL voltage (V_{DD}=1.1V), charging/discharging the BLs dominates the ACT energy because of the large number of BLs. Since a RD command accesses only ∼0.8% (= 64/8,192) of cells in a row, sensing all 8,192 BLs for only a few RD accesses, referred to as the overfetching problem, wastes a significant amount of energy as shown in Fig. 5(top). To reduce ACT energy, DRAM architectures with smaller pages and fine-grained activation have been proposed in a prior study [15]. In conventional STT-MRAM, ACT senses only $1/N$ BLs and consumes $1/N$ of BL charging and sensing energy, as depicted in Fig. 5(middle). Nonetheless, it still wastes a considerable amount of energy since an STT-MRAM SA consumes higher power than a DRAM SA. In addition, high ACT power limits BLP as discussed in Sec. 2.3 (Limitation 5). In contrast, SMART reduces ACT energy while providing larger pages with fine-grained activation at a smaller cost than STT-MRAM.

Benefit 3: Shorter latency with lower activation power. As discussed in Limitation 5 (Sec 2.3), high ACT power affects not only total memory energy but also overall memory performance. For the poor data locality (high page-miss rate), the long miss latency can be hidden and high bandwidth can be achieved by interleaving multiple banks. In fact, DRAM has enough banks to hide the miss latency, but bank interleaving is limited by tRRD and tFAW. In addition, because these constraints stem from the SA operation and its power consumption, they are imposed on not only DRAM but also other memory technologies having SAs and the page mode operation. For example, LPDDR2-NVM, which was devised for non-volatile main memories, also defines tRRD and tFAW [19].

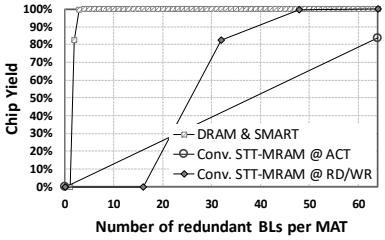
We plot tRRD and tFAW for various page sizes in Fig. 6, where the maximum activation current values are determined

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2022.3207131



	❶ DRAM & SMART	❷ Conv. STT-MAM Repair @ *ACT*	❸ Conv. STT-MAM Repair @ *RD/WR*
Repair granularity	1 BL	64 BLs	16 BLs
Repair flexibility	Within a MAT	Within a BL group	Within a MAT

(a) Example of various column repair schemes



(b) Chip yield according to repair schemes

Fig. 8: Comparison of the three repair schemes.

by a method of prior work [23]. `tRRD` and `tFAW` of conventional STT-MRAM for 64B pages are $\sim 3.2ns$ and $\sim 15ns$, respectively. Since a current SA consumes far more power than a voltage SA, STT-MRAM needs longer `tRRD` than DRAM for the same page size. In contrast, SMART activates 8× fewer SAs and consumes less power than conventional STT-MRAM. Note that SMART `ACT` does not consume any sensing power and the recovery time for activating 64 SAs during `RD` is short compared to `tCCD_S` and `tCCD_L` (CAS to CAS delay for the same bank accesses). Hence, SMART can practically eliminate these two constraints, and achieve high bandwidth under low page-hit rate as shown in Fig 7. Handling multiple `ACT` commands within a short time duration significantly reduces the latency of memory accesses especially when applications exhibit poor locality where subsequent accesses would be mapped to different banks rather than to the same page in the same bank.

Lastly, SMART also requires less time and less power than conventional STT-MRAM for memory write accesses. This is because an `ACT` command of SMART does not consume time and power for sensing which is unnecessary for write accesses, whereas conventional STT-MRAM still does.

Benefit 4: Fewer pins and more efficient repair. Unlike conventional STT-MRAM, SMART does not demand a partial column address with `ACT`, and uses the same number of address pins as DRAM. That is, SMART does not suffer from column address fragmentation discussed in Sec. 2.3. Column address fragmentation splits a physical column address into `ACT` and `RD/WR` and makes repairing a mat less efficient and flexible for conventional STT-MRAM. This can significantly increase the cost of repairing mats or decrease the yield of STT-MRAM chips. Fig. 8a describes the column repair schemes for DRAM and conventional STT-MRAM. In DRAM (❶), we can replace any 1,024 BLs with any 4 redundant BLs in a mat, and we repair a BL with a `RD` or `WR` command comprising a complete column address [26]. As SMART also exposes a complete physical column address to a `RD` or `WR` command, it can adopt the same column repair scheme and achieves the same yield under the same bit error rate (BER) as DRAM. In conventional

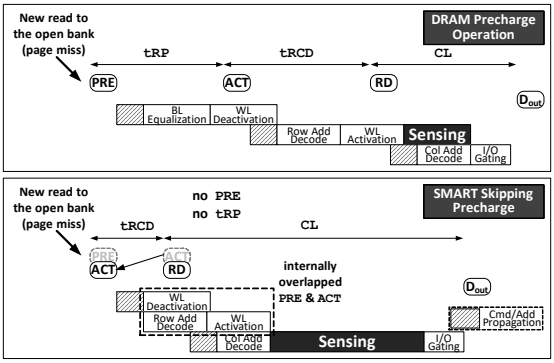


Fig. 9: Difference of command scheduling on row misses.

STT-MRAM, however, neither a `ACT` command nor a `RD/WR` command controls a complete physical column address. This makes repairing BLs far less efficient and flexible than DRAM or SMART.

Consider STT-MRAM with 1,024 BLs per mat and N (= 16) BLs per SA, (*i.e.*, $1{,}024/N = 64$ BL groups). If we are to repair a BL with `ACT` (❷), we need at least one redundant BL for every N BLs (= 64 redundant BLs per mat) as `ACT` can select only one BL in each BL group, but cannot differentiate BL groups due to the incomplete column address. On the other hand, if we are to repair a BL with `RD` or `WR` (❸), we need to replace the entire BL group including a defective BL with a redundant BL group (= 16 redundant BLs) because `RD` or `WR` can select only BL groups and BL selection in a BL group is already done with `ACT`.

We analyze the chip yield for various numbers of redundant BLs in Fig. 8b where we assumed that capacity of a chip is 8Gb, the number of mats is 16,384 in a chip where each mat has 512 WLs and 1024 BLs, N is 16 for conventional STT-MRAM, memory defects follow a `Poisson` distribution [26], the target BER after row repair is 10^{-7} and both DRAM/SMART and conventional STT-MRAM use the same row repair scheme. This shows that conventional STT-MRAM has a lower chip yield than DRAM or SMART for the same number of redundant BLs. In other words, to accomplish the same chip yield (99%) under the same BER (10^{-7}), conventional STT-MRAM requires 10.7× more redundant BLs than DRAM and SMART.

Benefit 5: Eliminating precharge commands. To activate another row in the same bank (a row-buffer miss), DRAM needs a `PRE` before an `ACT`. Specifically, `PRE` has two phases: (1) deactivating an asserted WL and (2) initializing BLs (BL equalization in DRAM) before `ACT` senses BLs. (2) can destroy cell states if it is performed before the WL is completely deactivated. Therefore, (1) should complete fully before (2) is started, and the total time for (1) and (2) (`tRP`) increases latency of memory accesses when row-buffer misses occur, as shown in Fig. 9 (top). Furthermore, when STT-MRAM follows the same page mode as DRAM, it experiences more row-buffer misses with smaller pages and pays this penalty more frequently than DRAM.

Unlike DRAM, however, STT-MRAM does not need to serialize (1) and (2) because of the non-destructive nature of its cells. Moreover, STT-MRAM immediately transfers the cell states sensed by SAs to registers serving as a row buffer (Sec. 2.1). This allows STT-MRAM to initialize the BLs and SAs immediately after sensing BLs as part of `ACT` and reduce

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2022.3207131

TABLE 1: Comparison with previous studies (Conv-Delay and Conv-Pin are the conventional STT-MRAM designs)

	DRAM	LPDDR2-NVM [19], [24]	Conv-Delay [2], [6]	Conv-Pin [12], [25]	This work (SMART)
Page size	Large (1KB)	Small (32B)	Small (64B)	Small (64B)	Large (1KB)
Sensing operation	During `ACT`	During `ACT`	During `ACT`	During `ACT`	During `RD`
Activation energy	High	Medium	Medium	Medium	Extremely low
Bank-level parallelism	Limited by `tRRD`/`tFAW`	Same as DRAM	Same as DRAM	Same as DRAM	No limitation
Pin-compatible with DDR	Yes	Yes (3-phase addressing)	Yes (delayed `ACT`)	No	Yes
Repair flexibility	High	Low	Low	Low	High
`PRE` technique	SALP [18]: Can violate `tRP` to access different sub-array	No	EarlyPA [6]: Internally perform `PRE` after the sensing and set `tRP`=1	No	No `PRE` command and no `tRP`
Need software/OS support	No	Yes, for write operation	No	No	No

Highlighted green: good features , highlighted yellow: good, but limited

`tRP`. Note that STT-MRAM initializes its BLs by discharging them to V_{SS}. The driving strength of charging BLs is limited to prevent unexpected changes of cell states (read disturbance) [5], but that of discharging BL is not limited. Hence, the amount of time for (2) can be much shorter than DRAM and it is already included in `tRAS`, which is the minimum time duration between `ACT` and `PRE` instead of `tRP` [2], [6], [10]. This reduces `tRP` of STT-MRAM by the amount of time for (2). However, it does not reduce or hide the amount of time for (1), still demanding a separate `PRE` command. In contrast, SMART can overlap the time for (1) ($\sim 3.7ns$) with the time to decode a row address and compare the address with addresses in a row repair table [26] during the early phase of `ACT` for the next row ($\sim 4.2ns$)[4]. This allows SMART to completely remove `PRE` and not consume any `tRP` before `ACT`, further reducing the latency to handle row-buffer misses as shown in Fig. 9 (bottom). Note that the conventional STT-MRAM could have skipped precharge because immediate BL initialization after sensing and the non-destructive sensing operation are common features. However, prior work [6] only reduced `tRP` without the overlap applied in SMART.

3.3 Discussion

We summarize the key differences among DRAM, LPDDR2-NVM, two conventional STT-MRAM designs and SMART in Table 1. Both conventional STT-MRAM designs, which are conventional STT-MRAM with pin incompatibility (Conv-Pin) and conventional STT-MRAM with delayed activation (Conv-Delay), have the shared SA structure. Conv-Pin emulates the designs in [12], [25]. Although [12] is a PCM design, its bank structure and memory operation apply to most non-volatile memories. However, they have no consideration for the expanded address pins from the shared SA structure and they are not compatible with JEDEC DDR. Conv-Delay internally delays the activation instead of increasing the pin count [2], [6].

SMART does not increase the latency of serving a single read request or consecutive read requests issued at the `tCCD_S` interval. But it still increases CL, which may increase overall read latency of serving multiple read requests issued at longer intervals than `tCCD_S`. Our evaluation shows that the performance degradation caused by the increased `CL` is outweighed by the performance increase of larger pages, higher BLP, and lower row-buffer miss latency.

Since a row buffer holds data only for a previously accessed column, SMART cannot compare-before-write when a `WR` command is sent to a different column of the activated row. Comparing data before writing reduces write energy and improves the endurance by not overwriting the same data [2], [27]. However, the high endurance of STT-MRAM cells ($> 10^{15}$) guarantees practically unlimited write operations (Sec. 2.1). Moreover, cell write energy is not a major component in overall write energy. Therefore, such a technique has limited impact on giga-bit scale STT-MRAM (Sec. 4.3).

Unlike conventional shared SA designs [12], [25] that use extra address pins, SMART does not change the physical interface (DDR PHY). Moreover, changes to access timing (*e.g.*,`tRCD` and `CL`) do not require modification in memory controllers because programmable parameters are a built-in feature to support various DRAM's speed grades [13]. Although issuing the next `ACT` without `PRE` is prohibited in the traditional open-page policy, we expect minimal modification in the scheduler can enable skipping precharge for SMART.

Off-chip error correction code (ECC), where extra chips are added to store parity data on memory modules, is not dependent on memory cells; hence, the same ECC can be adopted in SMART. On-chip ECC proposed in LPDDR4 [28], can also be adopted because SMART keeps the same access granularity, which determines the code word for ECC, as DRAM. However, placing parity columns and their SAs for on-chip ECC would require more area in STT-MRAM than in DRAM because of bigger memory cells and SAs.

Partition-level parallelism (PALP) exploits parallelism in a bank and mitigates the bank conflicts in NVMs [29]. Partitions are the equivalent notion to subarrays in this paper and PALP and Subarray-level parallelism (SALP) share the same basic idea in terms of allowing multiple activated pages in a bank [18].

Since a bank is already partitioned, it is expected that SMART can adopt SALP/PALP with minimal modifications (*e.g.* adding more pins to deliver partition addresses, allowing multiple activation in a bank and introducing new timing constraints for the same bank accesses) and can take the advantages of SALP/PALP (*e.g.*, less bank conflict).

Lastly, as this work focuses on re-architecting STT-MRAM for higher performance and lower energy, we do not discuss challenges related to its cells, such as thermal stability, write endurance, and read disturbance in detail [30]. Nonetheless, prior studies have demonstrated small (sub-20nm) STT-MRAM cells that can offer fast switching time (sub-10ns) under low write current (sub-10uA), high write endurance ($> 10^{15}$), thermal stability and read disturbance [31].

4. We leverage the $\sim 0.5ns$ difference, but some overlap between deactivating the previous WL and activating the new WL is acceptable because such overlap does not destroy the cell states.

4 Device Modeling

To evaluate DRAM, STT-MRAM and SMART, we use DRAMSpec [22], a detailed timing, power, and area exploration tool originally developed for DRAM. We adapt it for STT-MRAM for our evaluation. For the baseline DRAM evaluation, we consider a ×8 8Gb DRAM device. Then, we adapt some parameters of DRAMSpec to obtain similar results for a baseline DRAM in 30 nm technology. Keeping the same chip floor-plan as DRAM, we adapt the bank architecture and interconnect models to model STT-MRAM and SMART with parameters extracted from NVSim [32] and prior work [2], [33] and extrapolate them to a 30nm technology.

4.1 Area Model

We assume a $7.2F^2$ DRAM cell (2F×3.6F = WL×BL) provided by the DRAMSpec's $20nm$ technology model and a $10.9F^2$ (3F×3.6F) STT-MRAM cell. For conventional STT-MRAM, we take the shared SA ratio of 16:1 ($N = 16$) which is from an industry STT-MRAM chip [2]. Following the JEDEC standard, SMART has the same number of rows and columns as DRAM. However, SMART can implement any page size with the number of SAs equal to the number of bits per column access.

Similar to baseline [13], both conventional STT-MRAM and SMART are modeled as an 16-bank (4 bank groups and 4 banks per bank group), 8Gb device. Each bank consists of 1024 mats comprised of 512 WLs and 1024 BLs. Because SMART has the floor-plan and bank structure similar to DRAM, SMART can employ similar approaches to increase the capacity, such as increasing the number of banks and rows. We also assume 12 redundant WLs and BLs per mat for DRAM and SMART whereas we suppose 32 redundant BLs per mat for conventional STT-MRAM (Sec. 3.2). Prior work [10] demonstrated the layout area of various SAs for STT-MRAM, but it designed the SAs with a logic technology. Thus, we convert design rules to those of a memory technology based on the ITRS roadmap to re-estimate the area [34].

TABLE 2: Area comparison

	DRAM	Conventional STT-MRAM	SMART
Cell size	$7.2F^2$	$10.9F^2$	$10.9F^2$
SA size	$1{,}213F^2$	$27{,}111F^2$	$27{,}111F^2$
# of SA per bank	524,288	512	64
Bank area (μm^2)	1,258×5,250	1,274×6,141	1,263×5,735
Chip area (mm^2)	114.79	137.42 (20% ↑)	126.49 (10% ↑)

Table 2 summarizes the analyzed area of key memory components. The total height of the BLSA blocks in DRAM is ∼20% of the total chip height in a $20nm$ 8Gb DRAM device [35]. Therefore, the number and size of SAs greatly affect the total chip size. Albeit SMART uses 1.5× and 21.4× larger cells and SAs than DRAM, it uses 8,192× fewer SAs. This is because a bank has 128 sub-arrays and a pair of two sub-arrays shares 8,192 SA in DRAM. SMART is only 10% larger than DRAM compared to conventional STT-MRAM that is 20% larger. Moreover, the dual sensing paths which is one area overhead in the prior work [21] has been eliminated, this brings further area saving in this work. Lastly, the chip area of SMART is less sensitive to memory cell size than that of the conventional designs because of fewer redundant and reference BLs.

4.2 Timing Model

TABLE 3: Timing and latency comparison

	DRAM	Conventional STT-MRAM	SMART
`tRCD` (clock cycle)	19	29 (1)	14
`tRAS` (clock cycle)	43	30 (31)	15
`tWR` (clock cycle)	20	31 (31)	31
`tRP` (clock cycle)	19	8 (8)	8
`tRTP` (clock cycle)	10	1 (30)	15
`tRRD_S` (clock cycle)	4	3 (3)	1
`tRRD_L` (clock cycle)	8	6 (6)	1
`tFAW` (clock cycle)	28	21 (21)	4
`tCCD_S` (clock cycle)	4	4 (4)	4
`tCCD_L` (clock cycle)	7	8 (8)	9
`CL` (clock cycle)	19	14 (43)	29
Latency for single `RD`	38	43 (44)	43
Latency for five `RD`s (different banks)	66	64 (65)	47
Latency for two `RD`s (row miss)	100	81 (82)	73

The numbers in ( ) are for the delaying `ACT` like comboAS [2], [6].

Table. 3 summarizes the timing parameters and read access latency of DRAM, conventional STT-MRAM and SMART based on the memory clock frequency of 1333MHz (DDR4-2666). Thanks to the bank group structure, all three devices have `tCCD_S` that determines peak bandwidth. `tCCD_S` is equal to Burst Length (BL)/2=4 clock cycles. In contrast, `tCCD_L` is different in all three devices. In the conventional STT-MRAM, `tCCD_L` is determined by max{SIO/LIO time, GIO time}, but `tCCD_L` becomes GIO time, since cell data already passed GBL during the activation, which corresponds to SIO/LIO during the activation. On the other hand, max{sensing time, GIO time} determines `tCCD_L` in SMART and sensing time is longer than GIO time.

DRAM gives the shortest latency for a single read because its sensing speed is faster than that of STT-MRAM. However, the overall latency of multiple reads is determined not only by the sensing speed but also by other timing parameters such as `tRRD`, `tFAW`, `tRAS` and `tRP`, especially when different banks and rows need to be accessed. Thus, SMART shows shorter latency than DRAM and conventional STT-MRAM under memory accesses as shown in Table. 3 (bottom).

4.3 Energy Model

TABLE 4: Energy and current comparison

	DRAM	Conventional STT-MRAM	SMART
`ACT` (nJ)	0.54	0.39	0.05
Single `RD` (nJ)	0.15	0.14	0.16
Single `WR` (nJ)	0.14	0.18	0.18
Energy for 8B read (nJ)	0.69	0.47	0.21
Energy for 1KB read (nJ)	19.61	23.77	20.15
Energy for 8B write (nJ)	0.69	0.51	0.23
Energy for 1KB write (nJ)	18.70	28.28	22.76
`IDD0` (mA)	51	47	38
`IPP0` (mA)	3	3	3
`IDD2P` (mA)	25	28	27
`IDD2N` (mA)	35	38	37
`IDD3N` (mA)	46	38	37
`IPP3N` (mA)	3	3	43
`IDD4R` (mA)	146	141	150
`IDD4W` (mA)	142	152	151
`IDD5` (mA)	61	-	-
`IPP5` (mA)	5	-	-

In conventional STT-MRAM, activating fewer SAs reduces the energy consumption of a single `ACT` command. SMART, however, completely removes sensing energy from `ACT` and

thus it gives much smaller ACT energy than conventional STT-MRAM, as shown in Table 4. Instead, SMART includes the sensing energy in RD. Note that although the RD energy in SMART is higher than DRAM and conventional STT-MRAM due to the sensing energy, SMART does not waste the sensing energy for unaccessed cells. The conventional STT-MRAM has the least RD energy because of the reduced read path as described in Sec. 2.3.

STT-MRAM consumes more WR energy than DRAM because of the higher write current per cell. However, considering the energy consumption to transfer data across the chip through long interconnects, the impact of the cell write energy on the total write energy is limited. Table 4 shows the dynamic energy consumption of a single memory device for a single column (8B) request and a single page (1KB) request. If all columns in a page are accessed, STT-MRAM consumes more energy because of its inherent higher sensing and cell write energies. However, a single read/write access in STT-MRAM consumes less energy than DRAM because of the low ACT energy consumption. Comparing the two STT-MRAM's energies, SMART is more energy-efficient in all cases. This is because there is no wasted energy for the single column access and there are fewer ACT commands for the the single page access. In particular, because SMART does not include sensing energy in write requests, the gap between the two STT-MRAM designs in a 1KB write is larger than that in a 1KB read. In addition, we expect that the increased sensing energy by the longer sensing path in larger capacity is not directly reflected in total energy consumption of SMART because sensing energy is not a dominant component in SMART.

IDD0 is the average activation current and determined by the total activation current and the minimum row cycle (tRCmin = tRAS + tRP). Since SMART has shorter tRCmin as well as lower total activation current, IDD0 difference can be relatively smaller than the activation energy difference. IDD2P is the power-down current and it is close to the sum of total transistor leakage. Thus, IDD2P is usually proportional to the total transistor width under the same technology. For simplicity, we scale IDD2P of STT-MRAM linearly with chip size. IDD2N and IDD3N are background current under precharge- and active-standby, respectively. The difference between IDD2P and IDD2N mainly results from the address/clock buffer current components. Thus, we assume the same increment for STT-MRAM's IDD2N. However, IDD3N stems from DRAM's unique leakage component. When a row in a bank is activated, 16,384 BLs (total BLs of two sub-arrays) are fully charged or discharged, whereas all BLs are precharged to $V_{DD}/2$ level when the row is deactivated. If the BL length is 512, then 8,388,608 cells are connected to the 16,384 BLs. The increased voltage difference between BL to a cell transistor increases leakage current, which is mostly gate induced drain leakage (GIDL) and junction leakage current [36]. Because of this large number, small leakage current changes cause huge increases in IDD3N. However, the BL/SL condition of STT-MRAM is different from that of DRAM during active-standby, because BLs/SLs are always discharged except during sensing and writing. Therefore, we assume IDD3N is the same as IDD2N in STT-MRAM. Lastly, IDD5 is the auto refresh current and we assume no IDD5 in STT-MRAM due to nonvolatile characteristic of its cells.

TABLE 5: Default system configuration

Component	Specification
Processor	Single and quad core
Last Level Cache	2MB-8 way (single), 4MB-16 way (quad)
Memory Controller	Priority-based scheduling: Same command (1st) - Different bank (2nd) - Open page (3rd) - Age (4th) Open-page policy, Ro:Co-hi:Ba:Bg:Co-lo mapping
Memory Device	8GB with 8Gb x8-DDR4-2666

5 Evaluation

5.1 Evaluation Methodology

We evaluate SMART using MARSSx86 [37], which is a full-system simulator, and DRAMSim2 [38] which is a cycle-accurate memory simulator integrated into MARSSx86 to simulate the memory system and measure memory latency and power. The DRAMSim2 source code was modified to deactivate refresh and to skip *PRE* commands for STT-MRAM. The system configuration is shown in Table 5. Power-down mode is enabled to minimize standby power when there are no pending requests in the memory controller. For baseline DRAM, an 8GB memory channel (single rank) is composed using eight 16-bank 8Gb x8 DDR4 having 1KB page size [13]. We compare to two conventional STT-MRAM designs with the shared SA structure explained in Sec. 3.3: Conv-Pin and Conv-Delay.

TABLE 6: Workloads for multi-core simulations

Workload	Application list
mix1	imagick, sssp, stream_add, mcf
mix2	leela, deepsjeng, omnetpp, stream_copy
mix3	sssp, bfs, stream_scale, lbm
mix4	bfs, stream_add, mcf, lbm
mix5	bfs, mcf, stream_triad, lbm
mix6	sssp, stream_scale, stream_triad, stream_copy
mix7	mcf, stream_triad, lbm, stream_copy

We employ three benchmark suites: SPEC CPU 2017 [39], STREAM [40] and GAP [41]. Several SPECrate®2017 integer and floating point workloads are selected in SPEC CPU 2017 and a Kronecker graph with 2^{20} vertices (Graph500 specifications) is used for the workloads in GAP benchmark. For multi-core simulations, multi-program workloads are composed using SPEC CPU 2017, STREAM and GAP, as shown in Table 6. Misses-per-kilo-instructions (MPKI) increases from mix1 to mix7. For all workloads, one billion instructions are simulated in their region of interest (ROI).

5.2 Improvement in Performance

In SMART, we strive to implement large pages (1KB) with low cost. Memory performance is mainly affected by row buffer hit rate and BLP discussed in prior sections. Both DRAM and SMART can implement 1KB pages, while the conventional STT-MRAM can implement only 64B pages. The row buffer hit rates are shown in Fig. 10. Because our default address mapping scheme (Ro:Co-hi:Ba:Bg:Co-lo) assigns LSB-side bits for column selection to exploit data locality, Fig. 10 exhibits good row hit in some workloads. Details about the mapping scheme will be discussed in Sec 5.4. In multi-program workloads, because workloads of different

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2022.3207131

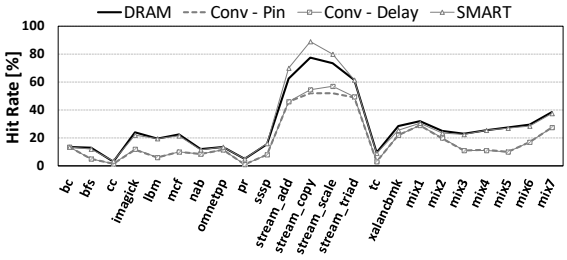


Fig. 10: Row hit rate in various workloads.

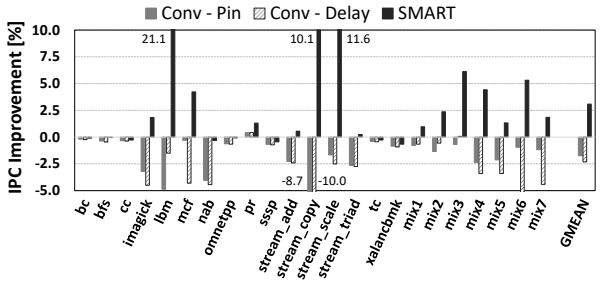


Fig. 12: Performance improvement compared to DRAM.

MPKI are mixed, high MPKI workloads (*e.g.,* STREAM and lbm) dominate overall memory accesses. SMART has a row hit rate within 1% of DRAM's, because page size, rather than column latency, mainly determines row hit rate under the same scheduling and memory address mapping scheme. However, both conventional designs have substantially lower hit rates, especially in STREAM workloads. Although all designs show low hit rate in a few workloads, the designs with the larger pages still perform better.

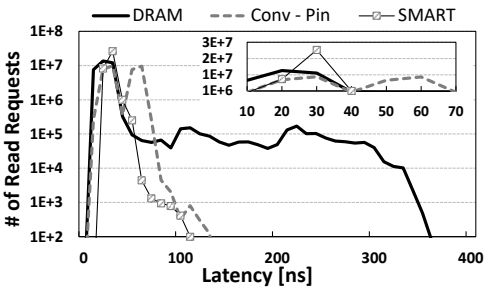


Fig. 11: Read latency profile of stream_triad workload.

Fig. 11 shows the overall read latency distribution of the three devices under the stream_triad workload. DRAM has a long tail latency because of refresh, but STT-MRAM has a short tail. In the Conv-Pin, we clearly observe two peaks in its distribution corresponding to read latency under row hits and misses, respectively. In SMART, there is no clear second peak because row misses are serviced faster and their read latency is partially overlapped with read latency under row hits. Although SMART has no data in 10∼19 ns range due to its long `CL`, it mostly falls within 30∼39 ns, implying that it is neither too quick nor too slow. DRAM has most of its latency within 20∼29 ns, but its long tail negatively affects the overall latency. The average read latency of DRAM, Conv-Pin, and SMART is 35.4, 45.4 and 31.1 ns, respectively.

Fig. 12 shows system IPC improvement normalized to DRAM. Compared to DRAM, Conv-Pin and Conv-Delay perform worse by 1.7% and 2.3%, respectively. For some memory intensive workloads IPC degrades more than 30%. Although conventional STT-MRAM designs do not have refresh overhead, the the smaller page size and not much improved latency yields the IPC degradation. A substantial drop in row hit rate over DRAM (*e.g.*, lbm, mix5 and STREAMs) significantly degrades their IPC. For workloads with similar row hit rates (*e.g.*, bc and pr), IPC is comparable due to lack of refresh and restoration operations and the reduced precharge time. For non-memory-intensive workloads, regardless of hit rate, the IPC difference is negligible. Between the two conventional designs, Conv-Delay shows worse overall performance than Conv-Pin because of its long `CL`.

SMART, on the other hand, improves IPC on average by 3.1% over DRAM. Because SMART has the same page size as DRAM, it achieves high row hit rates for applications having sequential memory accesses. In addition, applications having random memory accesses (low row hit rate – *e.g.* mcf and pr) see better row-miss latency and BLP. As a result, MPKI is correlated to the IPC difference. In memory intensive workloads with (MPKI >15) (*e.g.*, lbm and STREAMs), there is up to a 21% IPC improvement over DRAM. Non-memory intensive workloads with (MPKI < 1) (*e.g.*, bc, bfs and cc), show marginal IPC improvement.

5.3 Improvement in Energy

Compared to DRAM, SMART reduces energy because of three reasons. First, `ACT` energy is extremely low because of sensing, which was the main energy contributor, was moved to `RD`. Second, cell leakage current is eliminated while the bank is activated (i.e., low `IDD3N`). Last, STT-MRAM cells are non-volatile and have no refresh. While both conventional designs enjoy similar advantages over DRAM, they saved `ACT` energy by reducing the page size and impacted the row hits. In addition, the substantial sensing energy waste in `ACT` still remains in both conventional designs.

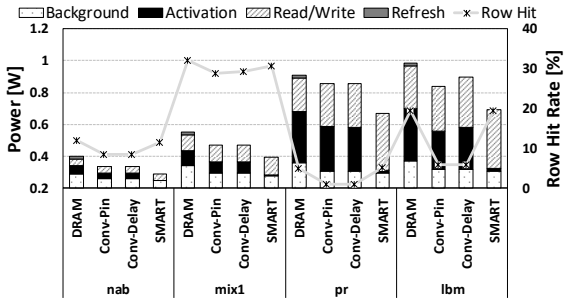


Fig. 13: Breakdown of average memory power.

Fig. 13 shows the comparison of power consumption by different memory designs. This power is broken down in terms of background, activation, read/write, and refresh power. DRAM consumes more `ACT` power than `RD/WR` power. The average refresh power is 2∼4% of the total average memory power and its share increases in less memory-intensive workloads (*e.g.*, nab). Although `ACT` power in the conventional STT-MRAM is less than in DRAM, the difference decreases when

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2022.3207131

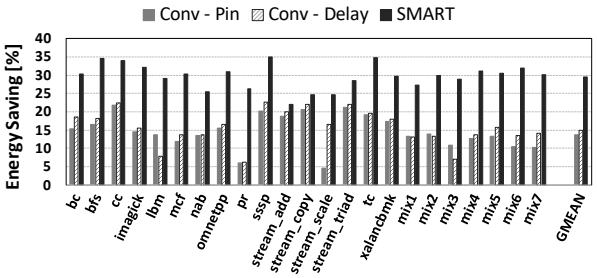
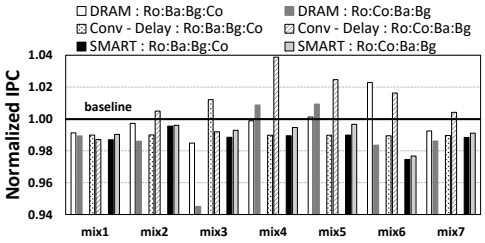
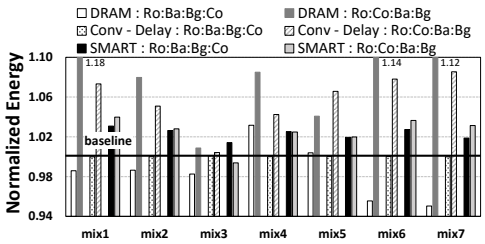


Fig. 14: Energy savings over DRAM.



(a) Normalized IPC



(b) Normalized energy

Fig. 15: Normalized IPC and energy with various address mapping schemes.

the conventional designs have low row hit rates (*e.g.*, pr, lbm). In these workloads, because memory has fewer chances to enter power-down mode due to row misses, the background power becomes higher than that of DRAM in spite of the lower `IDD3N`. `RD`/`WR` power is also higher than DRAM because of higher cell write current in the conventional STT-MRAM. In contrast, in SMART, the `ACT` power does not dominate total memory power. Although its `RD`/`WR` power is the highest among the four devices, the total dynamic power, which is the sum of `ACT` and `RD`/`WR` power, is the lowest. In addition, because SMART stays in power-down mode longer than other devices due to higher performance and no refresh, background power stays low.

Fig. 14 shows the energy savings over DRAM. On average, the Conv-Pin and Conv-Delay save 13.8% and 15.1% of energy and SMART saves 29.4%. Due to small page size, energy savings of the conventional STT-MRAM are more sensitive to the row hit rate difference. For example, the energy saving in the workloads exhibiting high page-hit rate (*e.g.*, STREAMs and mix1∼7) is higher than in the workloads having low page-hit rate (*e.g.*, lbm, mcf and pr). Whereas SMART shows less sensitive energy saving regardless of the workloads. Because SMART dramatically reduces `ACT` energy (good for random access) while the large page size of SMART yields less `ACT` commands (good for sequential access).

5.4 Address Mapping Sensitivity

The baseline address mapping scheme (Ro:Co-hi:Ba:Bg:Co-lo) applied in Fig. 12 and 14 maps the memory address to row, upper column, bank, bank group and lower column selection bits with MSB to LSB order. For example, the address change in MSB-side results in the different row (page) accesses whereas the change in LSB-side yields the same page access. In addition, the middle bits are used for both the same page and different bank accesses. In the all mapping schemes applied in the experiment, the bank and bank group selection bits are XORed with a part of row selection bits. Thus, some row address changes can make bank/bank group address change together and the bank conflict can be mitigated [42]. Overall, the baseline mapping scheme is well balanced for both high page-hit and BLP. However, since this mapping is not optimized neither sequential nor random accesses, other mapping schemes could perform better for highly biased workloads to the sequential or random accesses.

Fig. 15 shows the IPC and energy consumption changes according to the address mappings. Each result is normalized to its baseline mapping result. The Ro:Ba:Bg:Co scheme exploits the special locality by mapping near distance address to the same page. In contrast, the Ro:Co:Ba:Bg scheme is intended to utilize BLP by placing bank/bank group selection bits at the bottom. Depending on the workloads, the best performing mapping scheme varies. Overall, DRAM performs better with the Ro:Ba:Bg:Co scheme than with the Ro:Co:Ba:Bg scheme because the Ro:Co:Ba:Bg scheme tends to activate more banks, but BLP is limited by `tFAW` in DRAM. On the other hand, the conventional STT-MRAM shows better performance with the Ro:Co:Ba:Bg because of the small page size and relaxed `tFAW`. However, SMART is less sensitive to the mapping schemes. This is because that SMART implements the large pages while improving BLP at the same time.

Unlike the performance, the energy consumption shows the clear trend where the Ro:Co:Ba:Bg scheme has higher energy consumption than the Ro:Ba:Bg:Co scheme in most workloads (on average, 9.4% and 5.5% higher in DRAM and the conventional STT-MRAM, respectively). Since the Ro:Co:Ba:Bg scheme generates more `ACT` commands and `ACT` energy is the main contributor to DRAM and the conventional STT-MRAM's total energy, the higher energy consumption is observed even in the higher performing workloads (*e.g.*, mix4 and mix5). However, SMART has negligible energy difference by the mapping schemes considering the performance difference. Since SMART eliminates the wasted energy in `ACT`, the mapping schemes and the number of `ACT` commands do not significantly affect the total energy consumption.

6 Related Work

Asifuzzaman *et al.* analyzes the impact of a slow STT-MRAM main memory on high performance computing (HPC) [1]. They add 20% slowdown, estimated from industry, to the main memory over DRAM and evaluate it on the HPC applications. Their evaluation results yield that 20% slower main memory has negligible impact on overall system performance due to the limited role of the main memory on the system and out-of-order pipelining. Kultursay *et al.* evaluate STT-MRAM as a main memory with optimizations such as partial write and write bypass [43]. They show comparable performance with DRAM

and a 60% reduction in memory dynamic energy. Wang *et al.* investigate the design challenges of shared SAs such as small pages and pin compatibility [6]. With memory-architectural study, they propose three optimizations, comboAS, DynLat, and EarlyPA. Although these studies solve the compatibility problem and compensate for the reduced performance, the root cause of small pages and low chip yield remains unsolved.

LPDDR2-SX was designed for DRAM and its counterpart LPDDR2-NVM was designed for non-volatile devices with long write latency and large read/write circuits such as PCM [19], [44]. LPDDR2-NVM introduces a new command (`PREACT`) to deliver column selection rather than using additional pins. Although it can be a good candidate for STT-MRAM, its inherent performance is worse than LPDDR2-SX because of its three-phase addressing and software managed indirect write. In the prior work, 4Gb STT-MRAM has been demonstrated with LPDDR2-SX but not LPDDR2-NVM [2].

SALP-1 [18], EarlyPA [6] and LL-PCM [45] are similar to our skipping precharge in terms of alleviating `PRE` overhead (`tRP`). Unlike our technique, however, SALP-1 can only overlap `PRE` and next `ACT` when their sub-arrays are different. In EarlyPA and LL-PCM, the precharge operation is automatically performed immediately after the sensing operation. Although EarlyPA can efficiently hide the precharging time when the following command is `RD`, WL must be reactivated when the following command is `WR`. In contrast, our technique can hide `PRE` latency for both `RD` and `WR`.

7 Conclusion

We proposed SMART, a new cost-effective STT-MRAM architecture. We showed that by performing the sensing operation after a `RD` command instead of an `ACT` command, several benefits result. They include: larger pages, fewer sense amps, lower activation power, higher bank-level parallelism, shorter latency, fewer address pins, and more efficient column repair over conventional design. These benefits not only reduce energy but also improve performance compared to both DRAM and conventional STT-MRAM. In addition to the improvements in energy consumption and performance, SMART saves area in comparison to conventional STT-MRAM.

Acknowledgments

The material is based in part on research sponsored by Air Force Research Lab-oratory (AFRL) and Defense Advanced Research Projects Agency (DARPA) under agreement number FA8650-18-2-7864 and National Science Foundation (NSF) under CNS 1705047.

References

[1] K. Asifuzzaman, M. Pavlovic, M. Radulovic, D. Zaragoza, O. Kwon, K.-C. Ryoo *et al.*, "Performance Impact of a Slower Main Memory: A Case Study of STT-MRAM in HPC," in *International Symposium on Memory Systems*, 2016.

[2] K. Rho, K. Tsuchida, D. Kim, Y. Shirai, J. Bae, T. Inaba *et al.*, "23.5 A 4Gb LPDDR2 STT-MRAM with compact 9F2 1T1MTJ cell and hierarchical bitline architecture," in *International Solid-State Circuits Conference*, 2017.

[3] N. Rizzo, D. Houssameddine, R. Janesky, J chand Whig, F. Mancoff, M. Schneider, M. DeHerrera *et al.*, "A fully functional 64 Mb DDR3 ST-MRAM built on 90 nm CMOS technology," *IEEE Transactions on Magnetics*, 2013.

[4] C. Kim, K. Kwon, C. Park, S. Jang, and J. Choi, "7.4 A covalent-bonded cross-coupled current-mode sense amplifier for STT-MRAM with 1t1mtj common source-line structure array," in *International Solid-State Circuits Conference*, 2015.

[5] K. Tsuchida, T. Inaba, K. Fujita, Y. Ueda, T. Shimizu, Y. Asao *et al.*, "A 64Mb MRAM with clamped-reference and adequate-reference schemes," in *International Solid-State Circuits Conference*, 2010.

[6] J. Wang, X. Dong, and Y. Xie, "Enabling high-performance LPDDRx-compatible MRAM," in *International Symposium on Low Power Electronics and Design*, 2014.

[7] B. Jacob, S. Ng, and D. Wang, *Memory systems: cache, DRAM, disk*. Morgan Kaufmann, 2010.

[8] D.-G. Kim and K.-T. Park, "Semiconductor memory device with three-dimensional array and repair method thereof," Oct. 4 2011, uS Patent 8,031,544.

[9] J. Kan, C. Park, C. Ching, J. Ahn, L. Xue, R. Wang *et al.*, "Systematic validation of 2x nm diameter perpendicular MTJ arrays and MgO barrier for sub-10 nm embedded STT-MRAM with practically unlimited endurance," in *International Electron Devices Meeting*, 2016.

[10] B. Song, T. Na, J. Kim, J. P. Kim, S. H. Kang, and S.-O. Jung, "Latch offset cancellation sense amplifier for deep submicrometer STT-RAM," *IEEE Transactions on Circuits and Systems I*, 2015.

[11] T. Na, J. Kim, J. P. Kim, S. H. Kang, and S.-O. Jung, "An offset-canceling triple-stage sensing circuit for deep submicrometer STT-RAM," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2014.

[12] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable dram alternative," in *International Symposium on Computer Architecture*, 2009.

[13] Micron, "8Gb DDR4, MT40A1G8," 2015.

[14] C. Shiah, C. Chang, R. Crisp, C. Lin, C. Pan, C. Chuang *et al.*, "A 4.8 gb/s 256mb (x16) reduced-pin-count dram and controller architecture (rpca) to reduce form-factor & cost for iot/wearable/tcon/video/ai-edge systems," in *Symposium on VLSI Circuits*. IEEE, 2019, pp. C116–C117.

[15] T. Zhang, K. Chen, C. Xu, G. Sun, T. Wang, and Y. Xie, "Half-DRAM: a High-bandwidth and Low-power DRAM Architecture from the Rethinking of Fine-grained Activation," in *International Symposium on Computer Architecture*, 2014.

[16] A. N. Udipi, N. Muralimanohar, N. Chatterjee, R. Balasubramonian, A. Davis, and N. P. Jouppi, "Rethinking DRAM design and organization for energy-constrained multi-cores," in *International Symposium on Computer Architecture*, 2010.

[17] Micron, "RLDRAM3, MT44K64M18," 2015.

[18] Y. Kim, V. Seshadri, D. Lee, J. Liu, and O. Mutlu, "A case for exploiting subarray-level parallelism (SALP) in DRAM," in *International Symposium on Computer Architecture*, 2012.

[19] JEDEC, "Low Power Double Data Rate 2 (LPDDR2)," http://www.jedec.org/sites/default/files/docs/JESD209-2B.pdf, 2009.

[20] J.-H. Yoo, C. H. Kim, K. C. Lee, K.-H. Kyung, S.-M. Yoo, J. H. Lee *et al.*, "A 32-bank 1 Gb DRAM with 1 GB/s bandwidth," in *International Solid-State Circuits Conference*, 1996.

[21] B. Oh, N. Abeyratne, N. S. Kim, R. G. Dreslinski, and T. Mudge, "Smart: Stt-mram architecture for smart activation and sensing," in *Proceedings of the International Symposium on Memory Systems*, 2019, pp. 316–330.

[22] O. Naji, C. Weis, M. Jung, N. Wehn, and A. Hansson, "A high-level DRAM timing, power and area exploration tool," in *Embedded Computer Systems: Architectures, Modeling, and Simulation*. IEEE, 2015.

[23] D. U. Lee, K. S. Lee, Y. Lee, K. W. Kim, J. H. Kang, J. Lee *et al.*, "Design considerations of HBM stacked DRAM and the memory architecture extension," in *Custom Integrated Circuits Conference*, 2015.

[24] Y. Choi, I. Song, M.-H. Park, H. Chung, S. Chang, B. Cho *et al.*, "A 20nm 1.8 V 8Gb PRAM with 40MB/s program bandwidth," in *International Solid-State Circuits Conference*, 2012.

[25] J. Meza, J. Li, and O. Mutlu, "Evaluating row buffer locality in future non-volatile main memories," 2012.

[26] M. Horiguchi and K. Itoh, *Nanoscale memory repair*. Springer Science & Business Media, 2011.

[27] B.-C. Oh, J.-H. Bae, K. Fujita, and Y. Shirai, "Electronic device including semiconductor memory and operation method thereof," 2014, uS Patent 6,442,585.

[28] SK hynix, "Evolutionary Migration from LPDDR3 to LPDDR4," https://www.jedec.org/sites/default/files/Minho_SK%20hynix_CES_14_new.pdf, 2014.

[29] S. Song, A. Das, O. Mutlu, and N. Kandasamy, "Enabling and exploiting partition-level parallelism (palp) in phase change memories," *ACM Transactions on Embedded Computing Systems*, vol. 18, no. 5s, pp. 1–25, 2019.

[30] L. Thomas, G. Jan, J. Zhu, H. Liu, Y.-J. Lee, S. Le *et al.*, "Perpendicular spin transfer torque magnetic random access memories with high spin torque efficiency and thermal stability for embedded applications," *Journal of Applied Physics*, 2014.

[31] D. Saida, S. Kashiwada, M. Yakabe, T. Daibou, M. Fukumoto, S. Miwa *et al.*, "1x- to 2x-nm perpendicular MTJ Switching at Sub-3-ns Pulses Below 100uA for High-Performance Embedded STT-MRAM for Sub-20-nm CMOS," *IEEE Transactions on Electron Devices*, vol. 64, no. 2, pp. 427–431, 2017.

[32] X. Dong, C. Xu, Y. Xie, and N. P. Jouppi, "NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2012.

[33] S.-W. Chung, T. Kishi, J. Park, M. Yoshikawa, K. Park, T. Nagase *et al.*, "4Gbit density STT-MRAM using perpendicular MTJ realized with compact cell structure," in *International Electron Devices Meeting*, 2016.

[34] ITRS, http://www.itrs2.net/2013-itrs.html, 2013.

[35] J. Park, D.-H. Shin, Y.-H. Cho, and K.-W. Kwon, "Inverted bit-line sense amplifier with offset-cancellation capability," *Electronics Letters*, 2016.

[36] B. Oh, N. Abeyratne, J. Ahn, R. G. Dreslinski, and T. Mudge, "Enhancing DRAM Self-Refresh for Idle Power Reduction," in *International Symposium on Low Power Electronics and Design*, 2016.

[37] A. Patel, F. Afram, S. Chen, and K. Ghose, "MARSS: a full system simulator for multicore x86 CPUs," in *Design Automation Conference*, 2011.

[38] P. Rosenfeld, E. Cooper-Balis, and B. Jacob, "DRAMSim2: A cycle accurate memory system simulator," *IEEE Computer Architecture Letters*, 2011.

[39] Standard Performance Evaluation Corporation, "SPEC CPU® 2017," https://www.spec.org/cpu2017, 2017.

[40] J. D. McCalpin, "A survey of memory bandwidth and machine balance in current high performance computers," *IEEE TCCA Newsletter*, 1995.

[41] S. Beamer, K. Asanović, and D. Patterson, "The gap benchmark suite," *arXiv preprint arXiv:1508.03619*, 2015.

[42] M. Ghasempour, A. Jaleel, J. D. Garside, and M. Luján, "Dream: Dynamic re-arrangement of address mapping to improve the performance of drams," in *Proceedings of the Second International Symposium on Memory Systems*, 2016, pp. 362–373.

[43] E. Kültürsay, M. Kandemir, A. Sivasubramaniam, and O. Mutlu, "Evaluating STT-RAM as an energy-efficient main memory alternative," in *Performance Analysis of Systems and Software*, 2013.

[44] Z. Li, R. Zhou, and T. Li, "Exploring high-performance and energy proportional interface for phase change memory systems," in *International Symposium on High Performance Computer Architecture*, 2013.

[45] N. S. Kim, C. Song, W. Y. Cho, J. Huang, and M. Jung, "Ll-pcm: Low-latency phase change memory architecture," in *Proceedings of the 56th Annual Design Automation Conference*, 2019, pp. 1–6.

Byoungchan Oh He received the Ph.D in Electrical and Computer Engineering from University of Michigan, Ann Arbor, in 2014. He is a memory architect at Intel Corporation. His research interest includes memory technology, memory controller policy, error correction code, data compression, address translation algorithm and modeling/simulator development.

Nilmini Abeyratne She received the Ph.D in Computer Science and Engineering from University of Michigan, Ann Arbor, in 2013. She is a performance architect at Intel Corporation. Her particular area of research is computer architecture as it applies to high performance computing, supercomputers, memory, and interconnects.

Nam Sum Kim He is the W.J. 'Jerry' Sanders III – Advanced Micro Devices, Inc. Endowed Chair Professor at the University of Illinois, Urbana-Champaign and a fellow of both ACM and IEEE. From 2018 to 2020, he took a leave of absence and as a Sr. Vice President at a major memory manufacturing company he led the development of next-generation DRAM products, including the industry's first HBM-PIM that will play a significant role in shaping the future computing landscape. He has published more than 200 refereed articles to highly-selective conferences and journals in the field of digital circuit, processor architecture, and computer-aided design. He is a recipient of many awards including ACM/IEEE Most Influential ISCA Paper Award in 2017 and SIGMICRO 2021 Test of Time Awards in 2021. He is a hall of fame member of all three major computer architecture conferences, HPCA , MICRO, and ISCA.

Jeongseob Ahn He received the BS degree in Computer Science and Engineering from Dongguk University, in 2009, and the MS and PhD degrees in Computer Science from the Korea Advanced Institute of Science Technology, in 2011 and 2015, respectively. He is currently an associate professor with the Department of Software and Computer Engineering, Ajou University. His research interests include building high performance computer systems.

Ronald G. Dreslinski He is an assistant professor in the Computer Science and Engineering Department at the University of Michigan. His research interests include near-threshold computing (NTC), architectural simulator development, and high-radix on-chip interconnects. He received a PhD in computer science and engineering from the University of Michigan. He is the winner of the ISSCC 2011 student design contest and is the recipient of the Young Computer Architect Award from the IEEE Computer Society's Technical Committee on Computer Architecture.

Trevor Mudge He received the Ph.D. in Computer Science from the University of Illinois, Urbana. He is now a faculty at the University of Michigan, Ann Arbor, where he is the Bredt Family Professor of Computer Science and Engineering. He is author of numerous papers on computer architecture, programming languages, VLSI design, and computer vision. He has also chaired 51 theses in these areas. In 2014 he received the ACM/IEEE CS Eckert-Mauchly Award and the University of Illinois Distinguished Alumni Award. He is a Life Fellow of the IEEE, a member of the ACM, the IET, and the British Computer Society.