Evaluating STT-RAM as an Energy-Efficient Main Memory Alternative

Emre Kültürsay*, Mahmut Kandemir*, Anand Sivasubramaniam*, and Onur Mutlu†
*The Pennsylvania State University and †Carnegie Mellon University

***Abstract*—In this paper, we explore the possibility of using STT-RAM technology to completely replace DRAM in main memory. Our goal is to make STT-RAM performance comparable to DRAM while providing substantial power savings. Towards this goal, we first analyze the performance and energy of STT-RAM, and then identify key optimizations that can be employed to improve its characteristics. Specifically, using partial write and row buffer write bypass, we show that STT-RAM main memory performance and energy can be significantly improved. Our experiments indicate that an optimized, equal capacity STT-RAM main memory can provide performance comparable to DRAM main memory, with an average 60% reduction in main memory energy.**

I. Introduction

The memory wall problem continues to plague the design, implementation, and performance of computer systems. As the increasing degree of on-chip multiprogramming puts more pressure on the memory system, main memory serves a critical role lying between the processing cores and peripheral storage devices that have several orders of magnitude of higher latencies compared to DRAM. Consequently, there is continuing demand for DRAM capacity in order to maintain low page miss rates while serving ever-increasing frequency of requests within acceptable latencies. This has resulted in memory power itself becoming a significant contributor to overall system power. Several studies [2], [7], [11], [13], [22], [24], [32] have shown that main memory now accounts for as much as 30% of overall system power and is a large contributor to operational cost. When memory power becomes such a large concern, one must inevitably start considering alternative technologies that can potentially reduce the total cost of ownership of the system. Towards this goal, solutions that exploit trade-offs between operating and acquisition costs can be employed. Specifically, for memory systems, a technology that has not been considered as a main memory replacement due to its higher acquisition cost can have a significantly better operational cost to reach a lower total cost of ownership.

Recently, there has been a foray into exploring alternate technologies for main memory. While some view flash as a competing "main memory" technology [48], these attempts have not yet been successful. Instead, PCRAM and STT-RAM appear to be the competing alternatives to DRAM. Of these, PCRAM promises substantial density benefits (at least 2-4X over DRAM today), and it has been studied extensively to replace or augment DRAM in building a higher capacity and more scalable main memory system [5], [30], [49], [52], [66], [64]. However, it is both much slower (about 2-4X read, 10-100X write) and much more power hungry (about 2-4X read, 10-50X write), compared to DRAM [30], [50], [55], [61]. In addition, a PCRAM cell wears out with each write, which leads to a limited lifetime. Spin-Transfer Torque RAM (STT-RAM) is another competing technology that has also come under much scrutiny recently [8], [15], [23], [54]. STT-RAM does not necessarily have a density benefit over DRAM. While its read performance (latency and energy) is comparable to that of DRAM, its write performance (latency and energy) is worse (1.25-2X in latency, 5-10X in energy [9], [33]) than that of DRAM. However, STT-RAM has two major advantages over DRAM: non-volatility and decoupled sensing and buffering. When compared to PCRAM, STT-RAM has much better read/write performance and energy characteristics as well as much better write endurance. Yet, STT-RAM technology has so far only been explored as an SRAM substitute for on-chip caches [27], [53], [56], [57], [62], but has not been considered as a candidate for main memory.

In this paper, we ask (and give a positive architectural answer to) the question: *Can STT-RAM be used to completely replace DRAM main memory?* We set out to make STT-RAM main memory performance comparable to DRAM main memory while providing substantial power savings. If we can achieve this goal, then the power savings can allow us to reach much lower total cost of ownership for equal capacity memory, and even enable us to boost main memory capacities and serve a higher number of requests without suffering from a system-wide power increase. Towards this goal, this paper starts with a detailed analysis of the performance and energy consumption of several workloads (both single-threaded and multi-programmed workloads) using various memory technologies. Specifically:

- We give a detailed breakdown of the DRAM power consumption of several applications,
- Similarly, we analyze the STT-RAM power consumption of these applications, assuming STT-RAM is main memory.
- We show that the energy and performance of STT-RAM, without any optimizations, is *not* competitive with DRAM.

An in-depth examination of these results leads to two key observations: (i) actual data to be updated in a memory row constitutes only a small fraction of the row, and (ii) row buffer locality of reads is higher than that of writes. This analysis leads us to two optimizations in STT-RAM operation – *tracking dirty blocks within rows for partial writes*, and *writes bypassing the row buffer*. We find that these improvements substantially improve STT-RAM characteristics as main memory. In particular, we show that:

- An STT-RAM main memory can achieve performance comparable to DRAM main memory, and
- An STT-RAM main memory can bring 60% reduction in average memory subsystem energy over DRAM main memory.

To our knowledge, this is the first paper to characterize and compare DRAM and STT-RAM main memory power and performance, and using these characteristics to propose and evaluate an enhanced STT-RAM-only main memory design. We claim that, although the deployment cost of an optimized

978-1-4673-5779-1/13/$31.00 ©2013 IEEE

STT-RAM main memory is higher than that of an iso-capacity DRAM main memory, the 60% energy improvement of STT-RAM can potentially outweigh this deployment cost and realize a lower total cost of ownership. We hope that the results presented in this paper enable the community to examine STT-RAM as a potential alternative to DRAM.

II. DRAM Organization

In this section, we discuss DRAM, the state-of-the-art main memory technology, along with its basic operations and the peripheral circuitry needed to perform these operations. DRAM operation is described in more detail in [25], [28], [31], [45].

A. How DRAM Fits in the Overall Memory System

In this work, we assume a modern computing system with a state-of-the-art two-level private cache hierarchy. An L2 cache miss is sent to one of multiple on-chip memory controllers. Each memory controller is responsible for a separate memory channel which has one or more memory modules (DIMMs) connected to it. Each DIMM has a number of DRAM chips that are accessed in parallel to have a high bitwidth.

B. Memory Chip Organization

A DRAM chip consists of multiple banks that can be accessed independently in parallel. The only restriction in parallel access of banks is that all banks share the external data/address/command buses, so requests to different banks must be scheduled not to cause any conflicts on these buses. A DRAM bank, shown in Figure 1, comprises an array of storage cells, organized as rows and columns, row/column selection logic, sense amplifiers, and read/write latch and drivers [25]. An address provided to a memory bank consists of two parts: a row address and a column address. The row/column addresses and the memory burst length uniquely identify a particular cache block sized data in the array.

C. DRAM Operations

There are four fundamental DRAM operations that are responsible for a significant fraction of the access latency and the dynamic power consumption in DRAM. These operations, triggered by the memory controller through the memory bus, are: row activate (ACT), precharge (PRE), row buffer read (RD), and row buffer write (WR).

An ACT operation enables access to a row in the memory array and connects this row with the sense amplifiers in the DRAM peripherals. After sensing, coupled inverters in the DRAM sense amplifiers retain the received data, serving as a row buffer (RB). All read/write operations to DRAM must be performed from the row buffer, and therefore, require an ACT operation to be performed before them.

RD and WR operations operate in a block granularity and are served from the row buffer. Which block in the row buffer will be read or written is controlled by the column decoder. A RD operation selects a block in the row buffer and copies its data to the read latch which later transmits the data serially over the DQ pins of the memory chip in a number of bursts. A WR operation receives data from the DQ pins of the memory chip and uses write drivers to overwrite one block in the row buffer. As long as a row is active in the row buffer, the row buffer (i.e., sense amplifiers) remains connected to the row in the memory array. Therefore, a WR operation updates the data stored in the row buffer and the row in the array, simultaneously.

For each memory channel, there is a corresponding memory controller that is responsible for deciding which queued request will be scheduled next for each bank. If the next request that is scheduled by the memory controller for a bank is a part of the currently active row, then the request can be directly served using the row buffer (a.k.a., *a row buffer hit*). When a request to read/write data from/to a row other than the currently active row occurs (a.k.a., *row buffer conflict*), the row in the array and the bitlines must be disconnected (by turning the access transistor off) and the bitlines must be reset to the sensing voltage ($V_{cc}/2$) before the other row can be activated. This operation of preparing the bitlines for activation of another row is called PRE. Since a row buffer conflict involves precharging one row and activating another, its latency is larger than the latency of a row buffer hit.

Figure 2 illustrates the timing of two consecutive read operations received by one DRAM bank, These reads access different rows, namely, rows A and B, in the array. To retrieve row A from DRAM, first, the bitlines are precharged so that the array will be ready for activation. The first activate command starts the sensing process for row A. Towards the end of sensing, a column read command is issued so that the column access can start immediately when row sensing is finished. After the column access latency, a burst of data will be available at the output pins. As the next request in this bank is to a different row (i.e., row B), there is a row buffer conflict and an additional precharge time is needed before proceeding. After this delay, the array is ready for activating row B.

In addition to the four basic DRAM operations explained above, DRAM also has a *refresh* operation due to its volatile nature. Figure 1 also illustrates the DRAM cell storage capacitor and the access transistor. The charge stored in this capacitor slowly leaks through the access transistor and, if not refreshed, gets lost over time. The refresh operation in DRAM is performed at a row granularity by reading one row at a time into the row buffer (i.e., activation) which restores the degraded voltage stored in the cell capacitors. A typical refresh period for today's DRAM memory chips is 64ms. For a detailed description of DRAM refresh, we refer the reader to [36].

III. Spin-Transfer Torque (STT) Technology and STT-RAM Overview

Spin-Transfer Torque technology is one of the front runners among emerging technologies for storage. Its operation is based on magnetic properties of special materials whose magnetic orientation can be controlled and sensed using electrical signals. In this section, we provide basic operating principles of STT-RAM and point out distinctive parts of its cells and peripheral circuitry. A more detailed treatment of STT-RAM can be found in [8], [9], [15], [33], [54].

A. STT-RAM Cell Structure

An STT-RAM cell uses a Magnetic Tunnel Junction (MTJ) to store binary data [8], [15], [54]. An MTJ consists of two

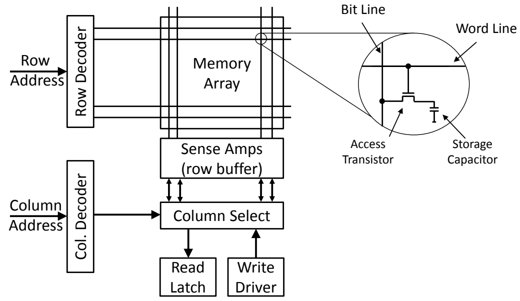


Fig. 1. DRAM bank organization.

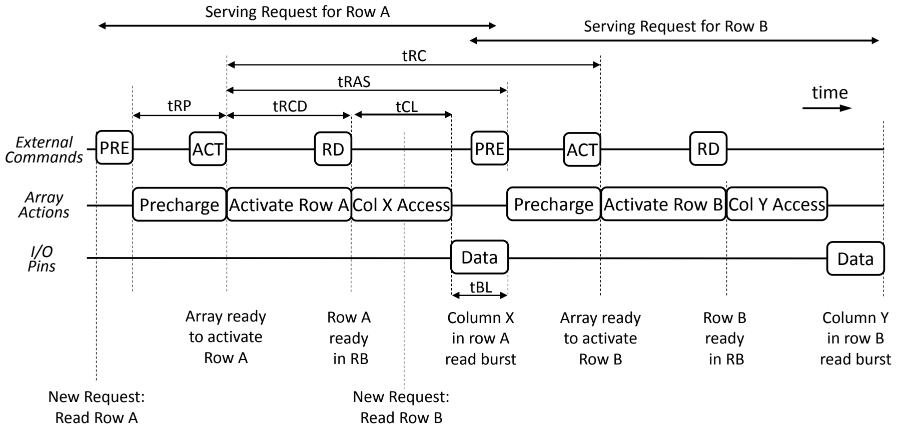
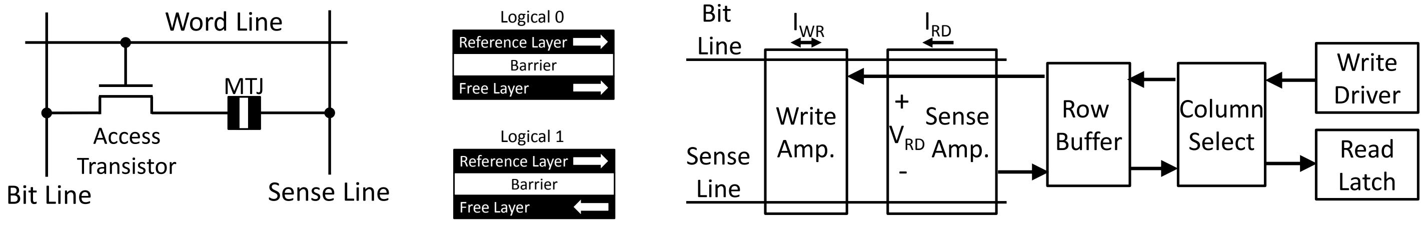


Fig. 2. Timing of two read accesses to distinct rows of a DRAM bank (drawn not to scale) [25], [31].



(a) (b) (c)

Fig. 3. (a) An STT-RAM cell, (b) an STT-RAM MTJ in parallel (top) and anti-parallel (bottom) alignments, and (c) the organization of the sense and write circuitry for STT-RAM bitlines. (Note the existence of separate sense amplifiers and row buffer storage.)

ferromagnetic layers and one tunnel barrier layer. The two ferromagnetic layers are called the reference layer and the free layer. The magnetic direction of the reference layer remains fixed, while the magnetic direction of the free layer can be parallel or anti-parallel, which is used to represent the binary data stored in the cell.

Figure 3(a) shows an STT-RAM cell. Similar to the DRAM cell, the STT-RAM cell also has an access transistor that connects the storage device and the bitline. However, different from DRAM, the other end of the storage device is not connected to ground; instead, it is connected to the sense line.

B. STT-RAM Operation and Peripherals

In an MTJ, data is stored as magnetic orientation of the free layer. This orientation determines the electrical resistance of the device which is used to read the data stored in the cell. As shown in Figure 3(b), when the magnetic field of the free layer and reference layer are parallel (i.e., aligned in the same direction), the MTJ resistance is low, representing a logical 0; and when they are anti-parallel to each other (i.e., aligned in the opposite direction), the MTJ resistance is high, representing a logical 1.

Note that, in DRAM, data is stored as voltage across a capacitor, which is sensed using voltage sensing circuitry. In STT-RAM, it is the resistance of the MTJ that changes based on the stored data. Therefore, different sensing and writing mechanisms must be employed. The sense and write amplifier organization in STT-RAM are shown in Figure 3(c). To read the data stored in a cell (i.e., activation operation), a small voltage is applied between sense and bit lines, and the amount of current flow is sensed. Similarly, writing to an STT-RAM cell is not a voltage-mode operation, but a current-mode operation. Specifically, to write data to an MTJ, a large current must be pushed through the MTJ to change the magnetic orientation of the free layer. Depending on the direction of the current, the free layer becomes parallel or anti-parallel to the fixed layer. The amount of current required for writing into an MTJ is significantly larger than that needed for reading from it. Therefore, large write amplifiers are used.

C. Non-Volatility

The MTJ in an STT-RAM cell is designed such that, even under the highest operating temperature conditions, it takes at least 10 years for thermal disturbances to upset the polarization stored in the junction [1]. Therefore, for all practical purposes, STT-RAM is considered to be non-volatile, i.e., it retains the stored data indefinitely without any power source. Non-volatility also implies that the stored data need not be refreshed periodically; hence, STT-RAM has no refresh power.

D. Non-destructive Reads

In a DRAM with state of the art 1T-1C cells, when the cell access transistors are enabled to perform an array read operation, the charge stored on the cell capacitors is shared with the bitlines (that have already been precharged to $V_{cc}/2$). The sense amplifiers are responsible for sensing and amplifying the differential change in the bitline voltage as a result of this charge sharing. Due to this charge sharing, the data in the cell gets destroyed. Over time, the sense amplifiers restore the data in the cell by pulling the capacitor voltages to the sensed voltage values. This type of array read operation is called a *destructive read*.

In contrast, the array read operation in STT-RAM is *non-destructive*. Reading the data stored in an STT-RAM cell requires a small amount of current to flow through the MTJ which does not disturb the stored data. Therefore, no additional latency to recover the data is required. As soon as the data is sensed by the sense amplifiers, it can be copied to a row buffer that is decoupled from the sense amplifiers. This row buffer organization is different from that of DRAM where the same coupled inverters are used for sensing and buffering. As a result of this decoupled architecture, STT-RAM sense amplifiers and row buffer can operate independently [3]. This leads to a change in the way row buffer operations are handled in STT-RAM. A WR operation is performed solely on the row buffer and does not propagate through the row buffer into the memory array. The contents of the row buffer are flushed to the memory array only when a row buffer conflict occurs. This difference between the DRAM and STT-RAM row buffer structure is analogous to write-through and write-back cache organization. While the DRAM row buffer resembles a write-through cache (for the DRAM array), an STT-RAM row buffer operates like a write-back cache (for the STT-RAM array).

IV. Experimental Setup

A. Simulation Framework and Target System

We used an in-house instruction-trace-based cycle-level multicore simulator with a front end similar to Pin [38]. This simulator can model the memory subsystem of a multicore system in detail. It models the execution in an out-of-order core including the instruction window, and on the memory side, it enforces channel, rank, bank, and bus conflicts, thereby capturing all the bandwidth limitations and modeling memory performance characteristics accurately. The memory model in the simulator is verified using DRAMSim [60] and its parameters are set to DDR3 memory timing parameters [41]. Table I shows our major processor and memory parameters.

Our simulator also employs a resource utilization model similar to [6] which counts the occurrences of various memory activities to estimate the overall memory system energy consumption. For DRAM and STT-RAM, we used CACTI [44] and modified it to model STT-RAM cells and peripherals and provide us with accurate estimates for dynamic energy cost of individual DRAM and STT-RAM operations. We also calibrated our DRAM model using the Micron power calculator [42]. The normalized energy values for all memory operations modeled in this work (DRAM and STT-RAM) are given in Table II. These energy values correspond to the basic per-bit hardware events in the evaluated DRAM and STT-RAM main memories. In DRAM, the array read/write energy components involve charging/discharging the bitlines and the cell capacitances through the access transistors and the precharge component measures the energy cost of driving the bitlines to $V_{cc}/2$. In STT-RAM, the read component includes the energy of driving a small amount of current into the cell and sensing the voltage difference, whereas the write component drives a much larger current into the cells using the write drivers. The total energy for each of the high level memory commands (i.e., precharge, activate, refresh, read/write) are obtained using (i) the number of bits involved in each event, and (ii) which combination of events are triggered with each command. The granularity of individual operations is common to both types of memories: row activation, precharge, and refresh (DRAM-only) commands operate on 4KB data, and read and write commands operate on 64B data. However, since DRAM and STT-RAM differ in the way array writes are performed, we provide a breakdown of total DRAM and STT-RAM energy into different components.

For DRAM, we provide a breakdown of total energy into the following three components:

1) *ACT+PRE:* A row can be activated only after a precharge which prepares bitlines for activation. Therefore, we merge the energy of activation and precharge and report them together.
2) *RD+WR:* We report the total energy for DRAM read and write energy. Note that in DRAM, read and write operations both access the row buffer, but write operation also charges/discharges the bitlines and DRAM cells.
3) *REF:* We also report refresh energy, which is the biggest component in DRAM background energy.

For STT-RAM, we provide a breakdown of total energy into the following three components:

1) *ACT+PRE:* Similar to DRAM, we provide the total energy for row activation and bitline precharge together.
2) *RB:* STT-RAM read and write operations are both done on the row buffer. Unlike DRAM, STT-RAM write operation does not involve bitlines or memory cells.
3) *WB:* In STT-RAM, an array write-back must be performed when a row buffer conflict occurs. This component involves the excess energy needed for changing the magnetic orientation of STT-RAM MTJs.[1]

B. Workloads

We used a representative subset of applications from the SPEC CPU2006 suite [21] in our evaluations (14 total, 8 integer, 6 floating point). We compiled the benchmarks using *gcc* with the `-O3` optimization level. Table III shows the application characteristics, including the number of pages accessed by the application, the pressure they put on the memory system in terms of level-2 cache misses per kilo-instructions (MPKI), level-2 cache write-backs per kilo-instructions (WBPKI) and their memory row buffer hit rates. We executed target benchmarks for 5 billion cycles, which corresponds to a real execution time of 2 seconds at 2.5 GHz.

We also present results for multiprogrammed workloads on multicore systems. Each of our multiprogrammed workloads

[1] The write-bypass optimization discussed in Section VI changes the default STT-RAM write behavior. When write-bypass is used, STT-RAM writes do not use the row buffer, and directly access the memory array. Hence, in write-bypass STT-RAM, write operations do not impact the *RB* energy, but only increase the *WB* energy.

Parameter	Value
Processor	128 entry instruction window, 3 instr. per cycle per core, 1 can be a memory op.
L1 Caches	32 KB per core, 4-way set associative, 64B block size, 2 cycle access latency
L2 Caches	512 KB per core, 16-way set associative, 64B block size, 12 cycle access latency
Memory Parameters	1 channel, 1GB, 8 banks, 4KB row buffer, 8-chips per DIMM, 64-bit wide channel
Memory Latency	75 and 125 cycles for row buffer hit and conflict, 10ns (25 cycles) extra STT-RAM write latency
Memory Scheduling	Queuing model with FR-FCFS memory scheduling policy [68]

TABLE I. Major processor and memory system parameters.

Operation	Norm. Energy
DRAM Array Read/Write	1.19
DRAM Precharge	0.39
STT-RAM Array Read	1.08
STT-RAM Array Write	2.83
Row Buffer Access	1.00

TABLE II. Per-bit energy consumption of memory operations normalized to the energy of accessing the row buffer for DRAM [41], [42] and STT-RAM memories [44], [57].

No	Application	Type	Pages	L2 MPKI/WBPKI	RB Hit
1	cactusADM	FP	185K	6.8 / 2.1	64%
2	calculix	FP	63K	3.8 / 0.9	88%
3	gamess	FP	29K	3.7 / 0.3	91%
4	gobmk	INT	28K	4.0 / 0.7	80%
5	gromacs	FP	27K	3.7 / 0.7	82%
6	hmmer	INT	24K	3.3 / 1.2	89%
7	lbm	FP	156K	25.2 / 9.5	87%
8	libquantum	INT	52K	1.2 / 0.4	57%
9	mcf	INT	260K	25.1 / 7.0	58%
10	omnetpp	INT	36K	8.6 / 0.4	64%
11	perlbench	INT	60K	3.6 / 0.6	80%
12	sjeng	INT	64K	4.5 / 1.8	76%
13	tonto	FP	39K	2.7 / 0.3	91%
14	xalancbmk	INT	39K	13.9 / 0.8	81%

TABLE III. Evaluated applications and their characteristics.

Mix	Applications	L2 MPKI/WBPKI
0	1,2,3,4	17.9 / 5.2
1	5,6,7,8	32.1 / 16.9
2	9,10,11,12	27.0 / 7.7
3	13,14,1,2	22.7 / 5.2
4	3,4,5,6	14.9 / 4.0
5	7,8,9,10	60.9 / 26.2
6	11,12,13,14	18.2 / 3.2
7	1,3,5,7	30.2 / 13.1
8	9,11,13,2	22.0 / 7.0
9	4,6,8,10	23.5 / 7.3
10	12,14,1,9	35.9 / 12.0

TABLE IV. Evaluated multiprogrammed workload mixes and their L2 MPKI and L2 WBPKI values.

has 4 applications from our set of SPEC applications, as shown in Table IV. All multiprogrammed workloads have a one-to-one mapping between the running applications and cores.

V. Evaluation of DRAM-based and STT-RAM-based Main Memories

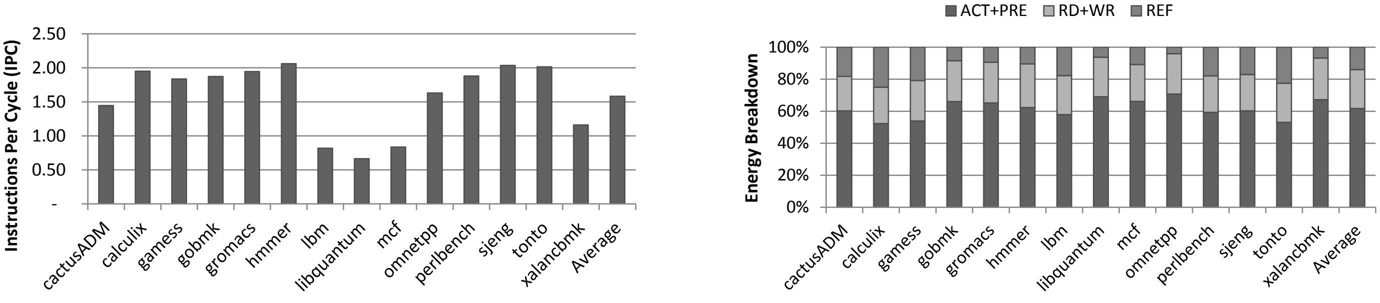
In this section, we first experiment with a baseline DRAM main memory. We then evaluate an STT-RAM-only memory and a DRAM/STT-RAM hybrid memory, and compare them with the baseline DRAM. The STT-RAM-based memories presented in this section are obtained from direct replacement of DRAM cells and peripherals with STT-RAM cells and peripherals, and they do *not* include any STT-RAM-specific architectural optimizations (We identify and evaluate STT-RAM optimizations in Sections VI and VII).

A. DRAM Main Memory

To establish a baseline, we first analyze the performance and energy of a DRAM main memory using a single core system with the memory organization described in Section IV. Figure 4(a) gives the IPC (instructions per cycle) values for individual benchmarks, which range from 0.66 (*libquantum*) to 2.06 (*hmmer*). The distribution of memory energy into its components over the entire execution of the target applications is given in Figure 4(b). While activation/precharge (ACT+PRE, 62%) and read/write (RD+WR, 24%) are the largest consumers of energy, refresh (REF) accounts for another 14% of the total energy consumption, on average. In these experiments, we used a 1GB DRAM main memory that is large enough to accommodate all pages any of our target applications. However, to be fair to applications that do not utilize all this memory capacity from a refresh energy perspective, we assumed that a selective refresh mechanism [47] is adopted, which refreshes only the memory rows that contain pages allocated to the target application. To be able to compare different memory technologies and organizations, all energy values that will be presented in the rest of this paper are *normalized* to the total energy values obtained with this baseline DRAM main memory.

B. STT-RAM Main Memory

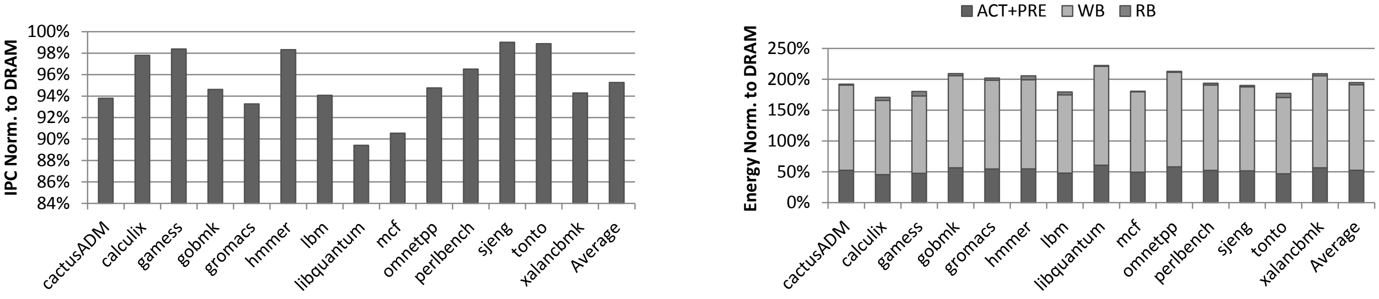
Now that we have a baseline, we start evaluating STT-RAM as main memory. Based on the characteristics of DRAM and STT-RAM given in Sections II, III, and IV, we expect an equal-capacity STT-RAM to perform worse in performance due to its high write latency and worse in energy due to its high write energy. Figures 5(a) and 5(b) present the IPC and energy of STT-RAM normalized to those of the baseline DRAM. While



(a) IPC

(b) Energy

Fig. 4. IPC and energy distribution of individual applications with DRAM memory.



(a) IPC

(b) Energy

Fig. 5. IPC and energy distribution of applications with STT-RAM memory (values normalized to DRAM).

the performance degradation of each application depends on its memory latency sensitivity, an average performance degradation of 5% is experienced when main memory is composed entirely of STT-RAM. On the other hand, the impact of using STT-RAM as main memory on memory energy consumption is very large, averaging a 96% increase over the baseline DRAM memory system. The most important factor in such a dramatic increase in memory energy consumption is write-back (WB) from the row buffer to the memory array which includes the STT-RAM cell write operation. The reduction in activation energy and the complete elimination of the refresh energy cannot compensate for this large increase in write energy. Note that the row buffer energy in STT-RAM is very low as STT-RAM row buffer read/write operations are isolated from the bitlines (unlike DRAM), and therefore, drive much smaller capacitances. Also note that the activation and write-back energies in STT-RAM are much larger than the row buffer energy as the former are performed on 4KB data whereas the latter operates at a 64B granularity. Clearly, in this unoptimized form, its high energy consumption hinders STT-RAM from being an attractive alternative to DRAM.

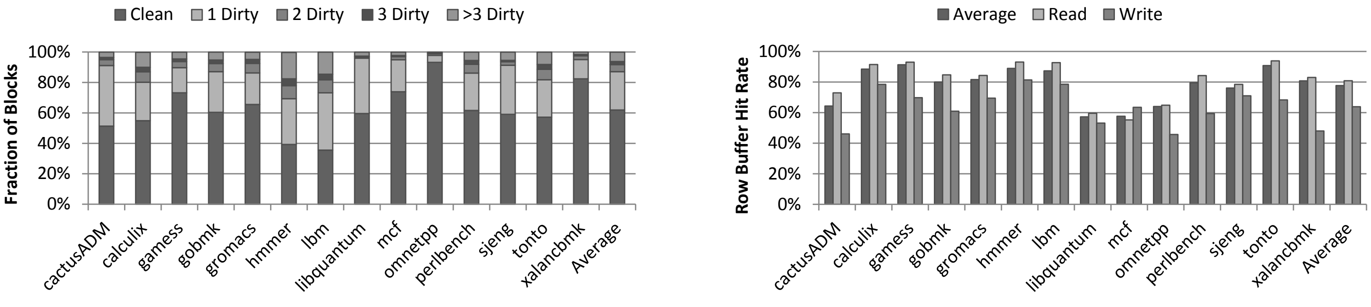
C. STT-RAM Main Memory with a DRAM Cache

Due to high STT-RAM write latency and energy, one can consider using a row-granularity DRAM cache before STT-RAM main memory in the memory hierarchy to reduce these overheads. Such an organization has been used with PCRAM main memory [40], [49], [64] for a similar purpose. The rationale behind using a DRAM cache is that, the most recently used pages in a system will reside in the DRAM-cache and accesses to these pages will be faster and less energy consuming. We experimented with various DRAM cache capacities and obtained the best results using DRAM cache with a capacity that is 10% of the STT-RAM main memory capacity. The average performance of this hybrid memory is only 1% worse than DRAM, which is about 5% better than our baseline STT-RAM. From an energy perspective, this hybrid memory reduced average memory energy by 61% vs. pure (unoptimized) STT-RAM. However, it still had 35% higher energy than the DRAM baseline as it can only partially eliminate STT-RAM writes. Therefore, using DRAM cache was not able to improve memory system power to a level that is better than our DRAM baseline.

D. Summary of the Baseline Evaluation

In conclusion, the two STT-RAM main memory organizations, namely, the standalone STT-RAM and the STT-RAM with DRAM-cache, analyzed in this section are observed to bring neither performance nor energy benefits over the DRAM-only option.[2] However, these STT-RAM memories do *not* really make use of any potential STT-RAM-specific architectural optimizations and are obtained by simply replacing DRAM cells and peripherals with STT-RAM equivalents. Instead of such a technology-agnostic approach, in the next section, we investigate the main problems with energy and performance of this baseline pure STT-RAM, and show some optimizations that lead to significant improvements in STT-RAM based main memory performance and energy, enabling it to surpass DRAM based main memory.

[2] We use equal capacity DRAM and STT-RAM main memories in our evaluation. In this work, we do not evaluate memory capacity or density, although this could be a potential issue with STT-RAM technology [1].



(a) Row buffer dirtiness at row conflict time

(b) Row buffer hit rate for reads and writes

Fig. 6. Characterization of evaluated benchmarks: row buffer dirtiness and row buffer hit rate.

VI. Optimizing STT-RAM

In this section, we present optimizations that can be used to improve STT-RAM to enable its adoption as a main memory technology. Evaluation of these optimizations will be given later in Section VII.

A. *Selective and Partial Write Operations*

A very important advantage of STT-RAM that has not been exploited in the STT-RAM main memory described in Section V is the decoupled structure of its sense amplifiers and row buffers. In DRAM, sense amplifiers constitute the row buffer and any operation on the row buffer (e.g., writing to it) also affects the DRAM cells. However, STT-RAM operations can be done only on the row buffer without involving the sense amplifiers and memory cells (as also observed in [39]). Specifically, in STT-RAM, when a row buffer conflict occurs, a selective write-back can be performed, where the row buffer is not written back to the array if it is clean. If the row buffer is dirty, the row *must* be written back as the array contains stale data and the most recent version of the data is in the row buffer. This *selective write* optimization can improve performance by expediting row buffer conflicts and can improve energy by eliminating the cost of redundant array writes.

The amount of performance and energy improvement selective writes can bring depends on how frequently the row buffer is clean at the time of a row buffer conflict. To quantify this frequency, we count the *number of dirty blocks* in the row buffer whenever there is a row buffer conflict and build up a histogram. Figure 6(a) presents this histogram data as a stacked bar chart for our benchmarks. It shows that the row buffer is completely clean in more than 60% of the row buffer conflicts. Therefore, the selective write scheme can be expected to bring significant benefits. In order to implement selective write, we need to keep one dirty bit for the row buffer in the memory controller. At the time of a row buffer conflict, the memory controller uses this bit as the indicator for whether the row buffer contents are clean, and if it is so, the controller skips the write-back of the active row into the STT-RAM array. Note that the benefits of this optimization is twofolds: it eliminates the write-back energy of clean row buffers and it reduces the access latency of row buffer conflicts when the row buffer is clean.

Figure 6(a) also shows that in another 32% of row buffer conflicts, the row buffer contains only 1, 2, or 3 dirty blocks. In fact, we have more than 3 dirty blocks in only 6% of the conflicts. Motivated by this mostly-clean nature of the row buffer, we can keep one dirty bit per 64B block (64 bit overhead per 4KB row buffer for a system with 64B sized cache blocks). Using dirty bits at a cache block granularity, when a row buffer conflict occurs, we can perform a *partial write* of only the dirty blocks into the STT-RAM array. Note that partial write is a pure energy optimization and has no effect on the performance of STT-RAM on top of selective write.

B. *Row Buffer Write Bypass*

The *selective write* and *partial write* explained above are optimizations that target the high write energy of STT-RAM, reducing the number of write-back operations and their cost. Another way of improving STT-RAM is to improve the row buffer hit rate of applications. To explore what optimizations can be performed in this direction, we analyze the row buffer hit rate of read and write operations separately. Figure 6(b) shows that, on average, reads have an 81% hit rate and writes have a 64% hit rate in the row buffer (data obtained using the baseline policies). The reason for such a behavior is that memory read requests arise due to read misses which directly propagate from the processor all the way up to off-chip memory, and hence, have a better locality. In contrast, memory write requests are due to evictions from the last level cache, in which case their locality is degraded by factors such as distribution of addresses to different cache sets and the approximate implementation of the LRU replacement policy. Since memory read operations are observed to have higher row buffer hit rates when compared to memory writes, one can consider memory writes as operations that access rows with less locality. Further, while a memory write is just an eviction from the last level cache that can typically be delayed significantly without stalling the processor, a memory read is a demand operation to access data that the processor needs immediately.

Based on these observations and considerations, we propose an optimization where memory reads and writes are handled differently with respect to the row buffer. Specifically, memory write operations *bypass* the row buffer and are directly sent to the memory array while memory reads are still served from the row buffer. The STT-RAM peripherals shown in Section III can easily support this optimization. The only needed change is to use externally provided data instead of the row buffer contents while writing into STT-RAM (i.e., write driver must directly feed the write amplifiers in Figure 3(c)). As a result of this scheme, we expect writes *not to* evict rows with high locality from the row buffer and reads to have a higher chance of hitting in the row buffers. Note that, similar to the FR-FCFS policy, we still prioritize row buffer hits over misses (including writes), but in our case, row buffer hits only occur when serving read requests. A write request to the currently active row is deprioritized over other read requests to prevent any coherence issues between the memory array and the row buffer. With this write bypass scheme, the row buffer contents are always guaranteed to be clean and are never required to be written back to the memory array.

To illustrate this optimization, consider a sequence of memory operations $\{R_1, R_1, R_1, W_2, W_2, R_1, R_1, R_1\}$ to a memory bank, where R/W denotes the type of operation (Read or Write) and the subscript denotes the index of the accessed row. For this sequence, the row buffer Hit/Conflict sequence is $\{C, H, H, C, H, C, H, H\}$ which has a row buffer hit rate of $5/8 = 62\%$. Note that the sequence has more reads than writes, and reads have a higher row buffer hit rate than writes (66% vs. 50%). Further, two of the three row buffer conflicts occur only because the row buffer contents were evicted to serve the writes. The total time to serve this sequence is $5 \times t_h + 3 \times t_c$, where t_h and t_c are hit and conflict latencies, respectively. Our write bypass optimization converts this sequence into $\{C, H, H, B, B, H, H, H\}$, where the two writes bypass (B) the row buffer and one of the conflicts is converted into a hit. Then, the total service time becomes $5 \times t_h + t_c + 2 \times t_b$, where t_b stands for write bypass time. Considering the higher latency of a conflict than bypass, this new sequence has a shorter total service time. It should be noted that while this row buffer write bypass optimization can improve performance by improving the row buffer hit rate (and, in turn, the latency) of reads, it can also hurt performance applications with high row buffer write hit rates as it converts row buffer write hits into slower row buffer bypass writes.

VII. Evaluation of Optimized STT-RAM

In this section, we evaluate the impact of the optimizations presented in Section VI from both energy and performance perspectives.

A. Selective and Partial Write Operations

The energy improvement obtained with selective writes can be observed by comparing Figure 7 with the STT-RAM baseline given in Figure 5(b). By adding only one dirty bit per row buffer, average energy consumption of the main memory drops from 196% to 108% of baseline DRAM-only memory system. This energy reduction is solely due to the elimination of the write-back of clean rows in the row buffer, as a result of which the write-back energy is more than halved. Such a reduction in write-back energy is in agreement with our expectation in Section VI, which was based on the observation that the row buffer is clean for 60% of the row buffer conflicts.

The benefit of using the partial write scheme can be observed by looking at Figure 8 and comparing it with Figure 7. Average energy consumption is reduced down to only 59% of DRAM, which is an additional 49% improvement on top of selective write. Clearly, this reduction is associated with the very low amounts of dirtiness in the row buffer identified in the previous section, illustrated with Figure 6(a).

B. Row Buffer Write Bypass

Figure 9 shows the energy breakdown when the write bypass optimization is added on top of partial write. When compared to the results in Figure 8, we can see that write bypass reduces the total energy consumption to only 42% of that of DRAM. This means that, the energy benefits brought by write bypass alone is 17% of DRAM, beyond selective and partial write operations.

The effect of the optimizations on STT-RAM performance is shown in Figure 10. It is interesting to observe that, average IPC of the target applications improves to a level where the STT-RAM based system performs 1% better than the DRAM based system. When compared to the unoptimized STT-RAM that had 5% degradation in performance, this actually means that the selective write and write bypass optimizations improved instruction throughput by approximately 6%. This improvement is due to improved average row-conflict service times, reduced latency of write operations, and a 5% increase in average row buffer hit rates for reads. While applications with high write intensity but low row buffer locality (such as *mcf*) benefit from reduced write latency, applications with a higher read/write row buffer locality imbalance (such as *cactusADM*) benefit from increased row buffer hit rate.

Although they are not directly applicable and require significant modifications to DRAM peripherals, we also evaluated the energy impact of performing the partial write and write bypass optimizations in a DRAM-based memory system. In this optimized DRAM, similar to the optimized STT-RAM explained in Section VI, all memory read operations use the row buffer and can cause row buffer conflicts, whereas memory write operations are performed directly on the memory array and are at cache line granularity. The results of these optimizations on DRAM are shown in Figure 11. The overall energy consumption is 70% of baseline DRAM, which is still approximately 30% more than that of optimized STT-RAM. The main reason why our optimizations are not very effective in DRAM is that, even if a row is accessed only for reading, its data must be slowly restored to the memory array due to destructive DRAM reads. Further, all rows in DRAM must always be refreshed, adding a non-negligible overhead of 14%, which does not exist in STT-RAM.

C. Evaluation in Multicore Systems

When multiple applications or threads are executed in parallel on a multicore system, there is higher pressure on memory which increases the importance of faster memory service. Further, data accesses originating from different cores can interleave in any fashion which can cause the memory row buffer locality to degrade [4], [43], [46], [65]. To evaluate the impact of multiprogramming on memory performance of applications, we analyzed the row buffer hit rate of each application (i) when it is executed alone on the processor, and (ii) when it is running together with three other applications. Figure 12 shows the changes in row buffer hit rates of different applications in each multiprogrammed workload mix. On average, when 4 applications are executed together, the row buffer hit rate drops by 8%, due to interleaved accesses to the row buffers from

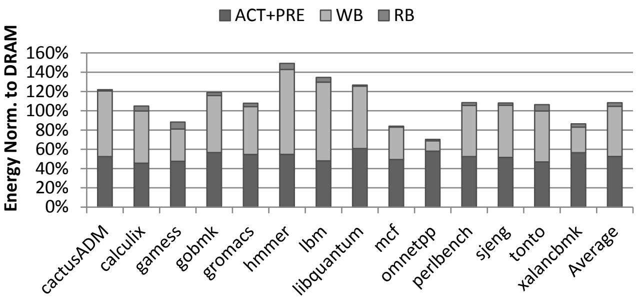


Fig. 7. Energy of STT-RAM with selective write (i.e., one dirty bit per row buffer).

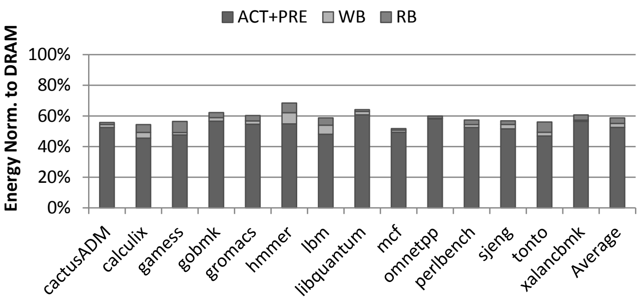


Fig. 8. Energy of STT-RAM with partial write (i.e., one dirty bit per 64B in row buffer).

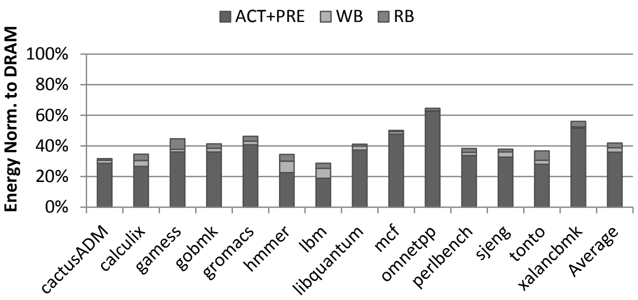


Fig. 9. Energy of STT-RAM with partial write and write bypass.

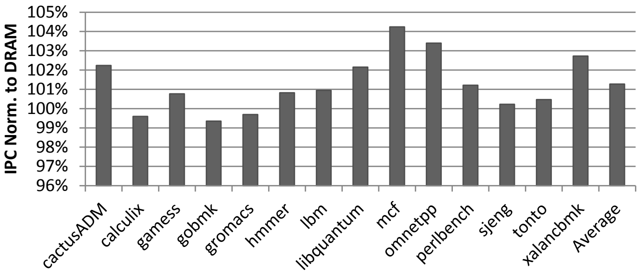


Fig. 10. Performance of STT-RAM with partial write and write bypass.

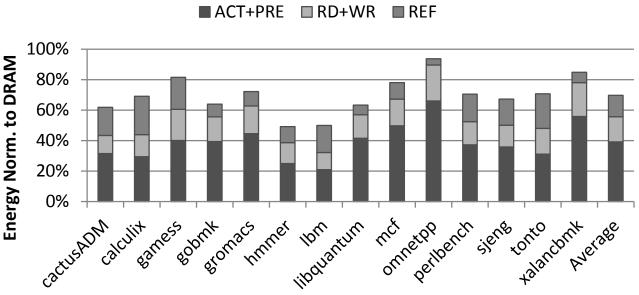


Fig. 11. Energy results of applying partial write and write bypass to DRAM (requires significant DRAM modifications).

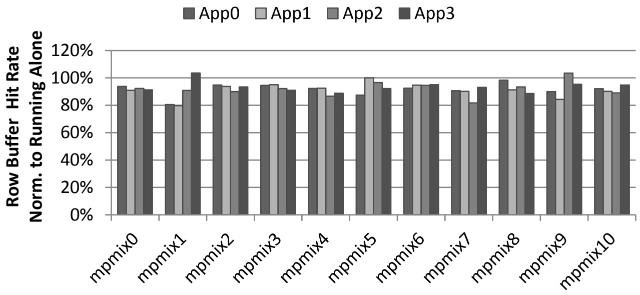


Fig. 12. Effect of multiprogramming on row buffer hit rate.

co-runner applications. Note that the drop in the hit rate with the FR-FCFS scheduling policy [68] is not as high as what it would be with the FCFS policy as it prioritizes row buffer hits over row buffer conflicts, thereby partially reducing the effect of interleaving of memory requests at the memory controller queues on row buffer hit rates.

Figures 13 and 14 show the energy consumption of the baseline DRAM and STT-RAM memories when multiprogrammed workloads are executed. This unoptimized STT-RAM has an average 2X energy consumption over pure DRAM, which is an even higher ratio than our single core results (due to the reduced row buffer hit rates). After adding the optimizations described in Section VI, STT-RAM energy again drops significantly and becomes only 37% of the DRAM energy, on average. The energy distribution of our multiprogrammed workloads with optimized STT-RAM is shown in Figure 15. Looking at this figure, one can observe that write-back and row buffer access energies are almost completely eliminated and the only significant source of energy remaining is the activation energy.

We also evaluated the impact of our optimizations on the performance of multiprogrammed workloads. Weighted speedup values for our workloads are given in Figure 16, *normalized* to the weighted speedup of the same workloads running on baseline DRAM main memory. This time, our optimized STT-RAM performs 6% worse than the DRAM baseline, mainly due to the reduced availability of STT-RAM banks when compared to DRAM. When multiple applications running together share the memory system, service time of individual requests become more important and the extra latency of STT-RAM writes keep the memory banks busy for longer durations. The result of this increased service latency manifests itself as a degradation in performance. Note that the workload mixes that degrade the most (*mpmix* 1, 2, 5, 7, and 10) are the ones with the highest total L2 cache write-back MPKI values in Table IV.

In addition to our multiprogrammed workloads, we also

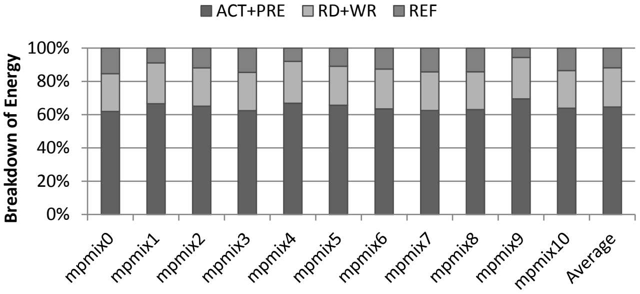


Fig. 13. Energy distribution of DRAM with multiprogrammed workloads.

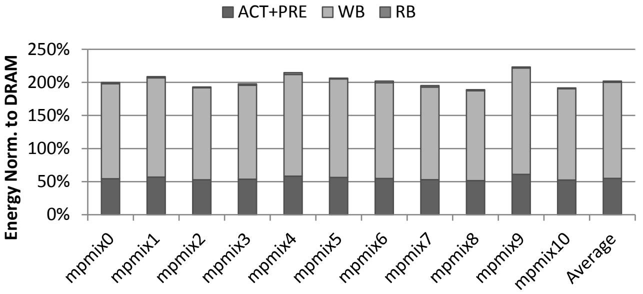


Fig. 14. Energy distribution of baseline STT-RAM with multiprogrammed workloads.

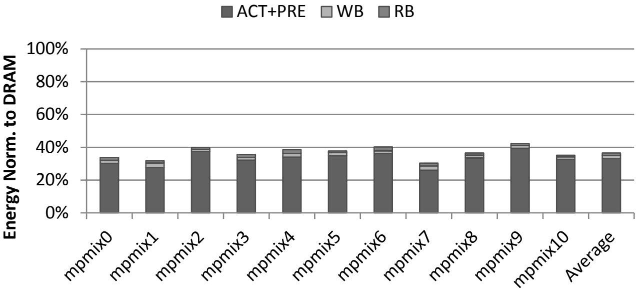


Fig. 15. Energy distribution of optimized STT-RAM using partial write and row bypass with multiprogrammed workloads.

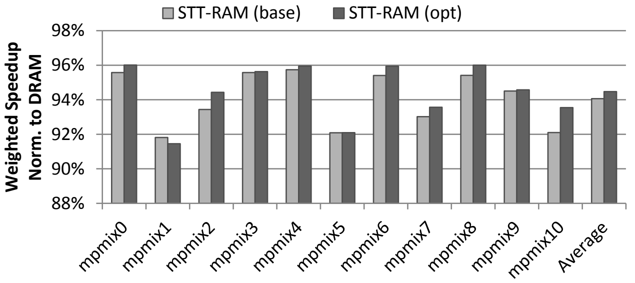


Fig. 16. Weighted speedup of optimized STT-RAM (normalized to DRAM) with multiprogrammed workloads.

experiment with SPECjbb2005 which is a multi-threaded Java server benchmark. We configured this application to have four threads pinned to the four cores of our system. Our results indicate that the proposed optimizations enable optimized STT-RAM to achieve 3% better throughput and 79% better energy than the baseline DRAM main memory.

Finally, we also evaluated the combination of *optimized* STT-RAM and DRAM cache, similar to the hybrid memory system employed in [40], [49], [64]. Similar to the unoptimized STT-RAM only case, the performance of this hybrid structure was on par with the DRAM-only memory. However, this time, its energy was close to DRAM-only memory energy, which is significantly worse than our optimized STT-RAM memory. This is because the optimizations we evaluate are closely tied to the STT-RAM decoupled sensing-buffering architecture which does not exist in the DRAM cache. Hence, employing a DRAM cache does not bring energy benefits to our optimized STT-RAM memory.

D. Sensitivity Analysis and Comparison to PCRAM

When designing an STT-RAM main memory, a parameter to consider is the duration of the write pulse. Prior work experimented with aggressive STT-RAM pulse durations as low as 2-3ns [27], [56], [63]. Shorter write pulses require higher write current densities, which not only can lead to potential thermal issues (that may further be exacerbated by locality), but also require larger write amplifiers. To identify the effect of STT-RAM write pulse duration on our results, we performed a sensitivity analysis. Using write pulse durations of 8ns, 6ns, and 3ns (while keeping the total energy for writes fixed), the 6% performance degradation with multiprogrammed workloads (obtained with a write pulse of 10ns in our described experiments) reduced to 3%, 1%, and less than 1%, respectively. While the performance degradation got reduced, our memory system energy improvements still remained at around 60%, indicating that our proposed solutions scale well with technology. The reason for the consistent energy savings is that the total energy of optimized STT-RAM is dominated by activation energy which is independent of the write pulse duration.

In our experiments, we assumed a 512KB private (per-core) last level cache configuration. Using larger L2 cache capacities can improve cache hit rates and reduce the number of off-chip accesses. This can, in turn, skew the off-chip access distribution to be more write-dominated. However, please note that, as we identify in this paper, STT-RAM writes can be more efficient than DRAM writes, simply because less bits are actually written using selective/partial write with the help of dirty-bits.

We also analyzed whether a PCRAM main memory with partial write and write bypass schemes can achieve similar energy efficiency improvements. We evaluated a PCRAM with read/write energy 2X/10X of DRAM, and two read/write latency values: (i) 2X/3X of DRAM and (ii) 1X/2X of DRAM. Our evaluation (on single-core systems) showed that PCRAM also benefits from the discussed optimizations and improves the energy efficiency of the memory subsystem. However, these energy savings (6-18%) are less than that provided by STT-RAM mainly due to higher PCRAM read/write energy for the two configurations. The high read and write latency also brings a significant performance penalty. We saw an average of 17% and 7% performance degradation on our benchmarks when executed on the two PCRAM configurations, respectively. This performance degradation impacts the whole system and can negate the energy benefits obtained from the optimized PCRAM memory subsystem. However, it should be noted that the applications used in our experimental evaluation already fit into the main memory for the baseline DRAM technology. The density benefit of PCRAM can still be beneficial in other applications that do not fit into DRAM main memory [30], [49]. For this type of workloads, using PCRAM can reduce the number of page faults (i.e., the number of disk accesses) and bring significant performance and energy efficiency improvements.

Overall, based on our results, we conclude that an optimized STT-RAM memory achieves much better energy characteristics than both DRAM-only and DRAM-cache based memories. As a result, we believe that STT-RAM could reduce or eliminate the need for potentially more costly hybrid memory solutions and can be a promising alternative to DRAM as the main memory technology, at least for many applications.

VIII. Related Work

Several alternatives to mainstream DRAM have emerged recently. Phase change memory (PCRAM) has been proposed as a promising alternative to commodity DRAM [10], [12], [59], and there has been significant effort in making optimizations to PCRAM-based main memory to improve its latency, energy, and endurance. One direction is to build DRAM/PCRAM hybrid memory systems and delegate management of the hybrid memory to system software. Page allocation [51], page migration [14], [64], and block migration [40] are three approaches to software-exposed hybrid memory. In the purely architectural optimization direction, Lee et al. [30] evaluate row buffer reorganizations to improve latency and energy of PCRAM memories, and Qureshi et al. [49] propose a hybrid memory system with an off-chip DRAM-cache which can have the latency benefits of DRAM and the capacity benefits of PCRAM. Another proposal that is orthogonal to PCRAM but in parallel with the DRAM caching idea is made in [26] which showed that caching hot pages in on-chip DRAM caches can improve performance. From the endurance angle, both [30] and [49] used partial PCRAM writes. In this work, we consider STT-RAM which is a memory technology that has latency/energy characteristics much closer to DRAM with a much better write endurance than PCRAM. Therefore, we perform architectural optimizations not for lifetime improvement, but for energy optimization. As opposed to PCRAM-based optimizations whose goal is to have existing PCRAM energy consumption approach DRAM, our results show that an optimized STT-RAM can achieve *much better* energy-efficiency than DRAM and does not need an additional DRAM-cache.

Several prior works have proposed the use of STT technology [29], [35] to reduce energy consumption of the processor and caches [17], [20], [27], [53], [56], [57], [62], [67]. While [20] proposes to migrate most of the functionality of a modern microprocessor from CMOS to STT-RAM technology, others [27], [56], [53], [57], [62], [67] proposed use of STT-RAM on chip for cache or embedded DRAM replacement. In contrast to these prior studies, we propose and evaluate STT-RAM as an alternative to DRAM main memory.

Researchers have tried to reduce refresh energy in DRAM [19], [34], [36], [37], [58]. Emma et al. [18] propose to use error correction codes to increase refresh intervals. [34] studies a range of line-level refresh policies. Venkatesan et al. [58] propose a retention-aware page placement algorithm that reduces DRAM refresh frequency. Liu et al. [36] propose RAIDR, which skips unnecessary refreshes to DRAM rows that can retain data longer than the minimum refresh interval. Ghosh and Lee [19] avoid refreshes to recently accessed rows. In [37], non-critical data is refreshed at lower rates. These works reduce only the refresh component of DRAM energy, which is eliminated by replacing volatile DRAM with non-volatile STT-RAM.

Other approaches, such as dynamic voltage and frequency scaling [11], [13] and power-down of banks/ranks [16] have also been examined within the context of DRAM. These approaches are fundamentally applicable to any type of memory technology, including STT-RAM.

IX. Concluding Remarks

This paper tries to answer the question: can STT-RAM be used as a technology replacement for DRAM at the main memory level? We first compare baseline DRAM with an unoptimized STT-RAM and show that, without any optimizations, STT-RAM consumes significantly more energy than DRAM. By analyzing the sources of high energy consumption in STT-RAM, we identify the importance of *partial write* and *write bypass* operations in improving STT-RAM energy. Our results on both single threaded and multiprogrammed SPEC CPU2006 workloads show that an optimized STT-RAM main memory achieves performance comparable to DRAM, while reducing the main memory energy by 60%. We hope this study serves as an inspiration and motivation for future in-depth study and optimization of the STT-RAM technology and architecture as an alternative to DRAM main memory.

Acknowledgments

We acknowledge the support of our industrial sponsors, AMD, HP Labs, IBM, Intel, Oracle, and Samsung. This work was partially supported by grants from the National Science Foundation (CAREER Award CCF-0953246, 1147397, 1147388, 1152479, 1017882, 0963839, 0811670, 1205618, 1213052), Intel Corporation URO Memory Hierarchy Program, and Microsoft Corporation.

References

[1] International technology roadmap for semiconductors. In *ITRS*, 2012.

[2] N. Aggarwal et al. Power-Efficient DRAM Speculation. In *HPCA*, 2008.

[3] T. W. Andre et al. A 4-Mb 0.18-.m 1T1MTJ Toggle MRAM With Balanced Three Input Sensing Scheme and Locally Mirrored Unidirectional Write Drivers. *JSCC*, 40(1), 2005.

[4] R. Ausavarungnirun et al. Staged Memory Scheduling: Achieving High Performance and Scalability in Heterogeneous Systems. In *ISCA*, 2012.

[5] R. Bheda et al. Energy Efficient Phase Change Memory based Main Memory for Future High Performance Systems. In *IGCC*, 2011.

[6] D. Brooks et al. Wattch: A Framework for Architectural-level Power Analysis and Optimizations. In *ISCA*, 2000.

[7] J. Carter and K. Rajamani. Designing Energy-Efficient Servers and Data Centers. *Computer*, 43(7), July 2010.

[8] E. Chen et al. Advances and Future Prospects of Spin-Transfer Torque Random Access Memory. *Magnetics, IEEE Transactions on*, 46(6), June 2010.

[9] S. Chung et al. Fully Integrated 54nm STT-RAM with the Smallest Bit Cell Dimension for High Density Memory Application. In *IEDM*, 2010.

[10] J. Coburn et al. NV-Heaps: Making Persistent Objects Fast and Safe with Next-Generation, Non-Volatile Memories. In *ASPLOS*, 2011.

[11] H. David et al. Memory Power Management via Dynamic Voltage/Frequency Scaling. In *ICAC*, 2011.

[12] A. De et al. Understanding the Impact of Emerging Non-Volatile Memories on High-Performance, IO-Intensive Computing. In *SC*, 2010.

[13] Q. Deng et al. MemScale: Active Low-power Modes for Main Memory. In *ASPLOS*, 2011.

[14] G. Dhiman et al. PDRAM: A Hybrid PRAM and DRAM Main Memory System. In *DAC*, 2009.

[15] Z. Diao et al. Spin-transfer Torque Switching in Magnetic Tunnel Junctions and Spin-transfer Torque Random Access Memory. *Journal of Physics: Condensed Matter*, 19, 2007.

[16] B. Diniz et al. Limiting the Power Consumption of Main Memory. In *ISCA*, 2007.

[17] X. Dong et al. Circuit and microarchitecture evaluation of 3D stacking magnetic RAM (MRAM) as a universal memory replacement. In *DAC*, 2008.

[18] P. Emma et al. Rethinking Refresh: Increasing Availability and Reducing Power in DRAM for Cache Applications. *Micro, IEEE*, 28, 2008.

[19] M. Ghosh and H.-H. S. Lee. Smart Refresh: An Enhanced Memory Controller Design for Reducing Energy in Conventional and 3D Die-Stacked DRAMs. In *MICRO*, 2007.

[20] X. Guo et al. Resistive Computation: Avoiding the Power Wall with Low-leakage, STT-MRAM based Computing. In *ISCA*, 2010.

[21] J. L. Henning. SPEC CPU2006 benchmark descriptions. *SIGARCH Comput. Archit. News*, 34(4):1–17, 2006.

[22] U. Hoelzle and L. A. Barroso. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*. 2009.

[23] M. Hosomi et al. A Novel Nonvolatile Memory with Spin Torque Transfer Magnetization Switching: spin-RAM. In *IEDM*, 2005.

[24] I. Hur and C. Lin. A Comprehensive Approach to DRAM Power Management. In *HPCA*, 2008.

[25] B. Jacob et al. *Memory Systems: Cache, DRAM, Disk*. Morgan Kaufmann, 2007.

[26] X. Jiang et al. CHOP: Adaptive Filter-based DRAM Caching for CMP Server Platforms. In *HPCA*, 2010.

[27] A. Jog et al. Cache Revive: Architecting Volatile STT-RAM Caches for Enhanced Performance in CMPs. In *DAC*, 2012.

[28] Y. Kim et al. A Case for Exploiting Subarray-level Parallelism (SALP) in DRAM. In *ISCA*, 2012.

[29] T. Kishi et al. Lower-current and fast switching of a perpendicular TMR for high speed and high density spin-transfer-torque MRAM. In *IEDM*, 2008.

[30] B. C. Lee et al. Architecting Phase Change Memory as a Scalable DRAM Alternative. In *ISCA*, 2009.

[31] D. Lee et al. Tiered-Latency DRAM: A Low Latency and Low Cost DRAM Architecture. In *HPCA*, 2013.

[32] C. Lefurgy et al. Energy management for commercial servers. *Computer*, 36(12), December 2003.

[33] H. Li et al. Performance, Power, and Reliability Tradeoffs of STT-RAM Cell Subject to Architecture-Level Requirement. *IEEE Transactions on Magnetics*, 47(10), October 2011.

[34] X. Liang et al. Process Variation Tolerant 3T1D-Based Cache Architectures. In *MICRO*, 2007.

[35] C. J. Lin et al. 45nm low power CMOS logic compatible embedded STT MRAM utilizing a reverse-connection 1T/1MTJ cell. In *IEDM*, 2009.

[36] J. Liu et al. RAIDR: Retention-Aware Intelligent DRAM Refresh. In *ISCA*, 2012.

[37] S. Liu et al. Flikker: saving DRAM refresh-power through critical data partitioning. In *ASPLOS*, 2011.

[38] C.-K. Luk et al. Pin: Building Customized Program Analysis Tools with Dynamic Instrumentation. In *PLDI*, 2005.

[39] J. Meza et al. A Case for Small Row Buffers in Non-volatile Main Memories. In *ICCD*, 2012.

[40] J. Meza et al. Enabling Efficient and Scalable Hybrid Memories Using Fine-Granularity DRAM Cache Management. *IEEE Computer Architecture Letters*, 11(2):61–64, 2012.

[41] Micron. 1Gb DDR3 SDRAM Component: MT41J256M4.

[42] Micron. Micron DDR3 power calculator.

[43] S. P. Muralidhara et al. Reducing Memory Interference in Multicore Systems via Application-aware Memory Channel Partitioning. In *MICRO*, 2011.

[44] N. Muralimanohar et al. Cacti 6.0: A Tool to Understand Large Caches. Technical report, University of Utah and Hewlett Packard Laboratories, 2007.

[45] O. Mutlu and T. Moscibroda. Stall-Time Fair Memory Access Scheduling for Chip Multiprocessors. In *MICRO*, 2007.

[46] O. Mutlu and T. Moscibroda. Parallelism-Aware Batch Scheduling: Enhancing both Performance and Fairness of Shared DRAM Systems. In *ISCA*, 2008.

[47] T. Ohsawa et al. Optimizing the DRAM Refresh Count for Merged DRAM/Logic LSIs. In *ISLPED*, 1998.

[48] C. Park et al. A Low-cost Memory Architecture with NAND XIP for Mobile Embedded Systems. In *CODES+ISSS*, 2003.

[49] M. K. Qureshi et al. Scalable High Performance Main Memory System using Phase-change Memory Technology. In *ISCA*, 2009.

[50] M. K. Qureshi et al. *Phase Change Memory: From Devices to Systems*. Synthesis Lectures on Computer Architecture. Morgan and Claypool Publishers, 2011.

[51] L. E. Ramos et al. Page Placement in Hybrid Memory Systems. In *ICS*, 2011.

[52] S. Raoux et al. Phase-change Random Access Memory: A Scalable Technology. *IBM Journal of Research and Development*, 52, july 2008.

[53] M. Rasquinha et al. An Energy Efficient Cache Design using Spin Torque Transfer RAM. In *ISLPED*, 2010.

[54] A. Raychowdhury et al. Design Space and Scalability Exploration of 1T-1STT MTJ Memory Arrays in the Presence of Variability and Disturbances. In *IEDM*, December 2009.

[55] G. Servalli. A 45nm Generation Phase Change Memory Technology. In *IEDM*, 2009.

[56] C. Smullen et al. Relaxing Non-Volatility for Fast and Energy-Efficient STT-RAM Caches. In *HPCA*, 2011.

[57] G. Sun et al. A Novel Architecture of the 3D stacked MRAM L2 Cache for CMPs. In *HPCA*, 2009.

[58] R. Venkatesan et al. Retention-aware Placement in DRAM (RAPID): Software Methods for Quasi-non-volatile DRAM. In *HPCA*, 2006.

[59] H. Volos et al. Mnemosyne: Lightweight Persistent Memory. In *ASPLOS*, 2011.

[60] D. Wang et al. DRAMsim: A Memory System Simulator. *SIGARCH Comput. Archit. News*, 33, November 2005.

[61] H. Wong et al. Phase Change Memory. *Proceedings of the IEEE*, 2010.

[62] X. Wu et al. Hybrid Cache Architecture with Disparate Memory Technologies. In *ISCA*, 2009.

[63] C. X. Xu et al. Device-architecture Co-optimization of STT-RAM based Memory for Low Power Embedded Systems. In *ICCAD*, 2011.

[64] H. Yoon et al. Row Buffer Locality Aware Caching Policies for Hybrid Memories. In *ICCD*, 2012.

[65] G. L. Yuan et al. Complexity Effective Memory Access Scheduling for Many-core Accelerator Architectures. In *MICRO*, 2009.

[66] W. Zhang and T. Li. Exploring Phase Change Memory and 3D Die-Stacking for Power/Thermal Friendly, Fast and Durable Memory Architectures. In *PACT*, 2009.

[67] P. Zhou et al. Energy Reduction for STT-RAM using Early Write Termination. In *ICCAD*, 2009.

[68] W. K. Zuravleff and T. Robinson. Controller for a Synchronous DRAM that Maximizes Throughput by Allowing Memory Requests and Commands to be Issued Out of Order. *U.S. Patent No: 5,630,096*, 1997.