Improving System Energy Efficiency with Memory Rank Subsetting

JUNG HO AHN, Seoul National University
NORMAN P. JOUPPI, Hewlett-Packard Labs
CHRISTOS KOZYRAKIS and JACOB LEVERICH, Stanford University
ROBERT S. SCHREIBER, Hewlett-Packard Labs

VLSI process technology scaling has enabled dramatic improvements in the capacity and peak bandwidth of DRAM devices. However, current standard DDR*x* DIMM memory interfaces are not well tailored to achieve high energy efficiency and performance in modern chip-multiprocessor-based computer systems. Their suboptimal performance and energy inefficiency can have a significant impact on system-wide efficiency since much of the system power dissipation is due to memory power. New memory interfaces, better suited for future many-core systems, are needed. In response, there are recent proposals to enhance the energy efficiency of main-memory systems by dividing a memory rank into subsets, and making a subset rather than a whole rank serve a memory request.

We holistically assess the effectiveness of rank subsetting from system-wide performance, energy-efficiency, and reliability perspectives. We identify the impact of rank subsetting on memory power and processor performance analytically, compare two promising rank-subsetting proposals, Multicore DIMM and mini-rank, and verify our analysis by simulating a chip-multiprocessor system using multithreaded and consolidated workloads. We extend the design of Multicore DIMM for high-reliability systems and show that compared with conventional chipkill approaches, rank subsetting can lead to much higher system-level energy efficiency and performance at the cost of additional DRAM devices. This holistic assessment shows that rank subsetting offers compelling alternatives to existing processor-memory interfaces for future DDR systems.

Categories and Subject Descriptors: B.3.2 [**Memory Structures**]: Design Styles—*Primary memory*; B.3.4 [**Memory Structures**]: Reliability, Testing, and Fault-Tolerance—*Error-checking and redundant design*

General Terms: Design, Performance, Reliability

Additional Key Words and Phrases: Memory system, DRAM, rank subsetting, overfetch, multicore DIMM, mini-rank

ACM Reference Format:
Ahn, J., Jouppi, N. P., Kozyrakis, C., Leverich, J., and Schreiber, R. S. 2012. Improving system energy efficiency with memory rank subsetting. ACM Trans. Archit. Code Optim. 9, 1, Article 4 (March 2012), 28 pages.
DOI = 10.1145/2133382.2133386 http://doi.acm.org/10.1145/2133382.2133386

4

1. INTRODUCTION

Performance, energy-efficiency, and reliability are all critical aspects of modern computer systems, and all of them must be considered carefully when a new architectural

---

This article is an extension of Ahn et al. [2009].
J. H. Ahn was supported in part by the Smart IT Convergence System Research Center funded by the Ministry of Education, Science and Technology (MEST) as Global Frontier Project and by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the MEST (2010-0003683).
Author's address: J. Ahn; email: gajh@snu.ac.kr.

DOI 10.1145/2133382.2133386 http://doi.acm.org/10.1145/2133382.2133386

idea is suggested. Transistors become faster and smaller as process technology scales so that high-performance processors with multiple computation cores and main-memory DRAM chips with billions of transistors are commodities [Bohr 2009; Kang et al. 2009]. However, it is challenging to make highly energy-efficient and reliable integrated systems with these small and dense transistors. Leakage and short-circuit power have become comparable to switching power on high-performance circuits [Nose and Sakurai 2000], hurting energy efficiency. Small transistors and narrow wires increase the frequency of manufacturing defects and of hard and soft errors [Mukherjee et al. 2005].

Chip multiprocessors (CMPs) demand high memory throughput and capacity. Their throughput demands are high since many cores are simultaneously requesting memory and on-chip cache capacity is limited [Mutlu and Moscibroda 2008]. Their capacity demands are also high, since the consolidation of workloads on a multicore processor typically implies that their working sets are aggregated. Since global wires, which are used to connect computation cores and storage cells, scale worse than local wires and transistors [Ho et al. 2001], meeting these dual demands of high throughput and high capacity is even more challenging when either energy efficiency or reliability is taken into account, and especially in high-availability systems where both are required. Moreover, power consumption has emerged as a major limiting factor in the design of contemporary datacenters since the cost to power a server can outweigh the cost to purchase the server over its life-cycle [Barroso 2005]. This motivates new tradeoffs in terms of capital and operating expenditures related to computing systems.

Memory accounts for a large and growing fraction of system power [Intel Press 2009; Lim et al. 2008]. There are three drivers: higher power dissipation DRAMs, more DRAMs, and interconnect power. Since 2003, multicore chip TDP (Thermal Design Power), and hence power, are essentially fixed, so the power per core will drop quadratically with feature size; DRAM is not yet under a tight TDP constraint, so its power per cell does not have to drop as fast and power per chip may increase. DRAM is not expected to scale as fast as processor cores, so there is upward pressure on the number of DRAM chips per processor. Finally, memory access involves inter-chip communication, which improves more slowly than storage and local transistor operations in terms of speed and energy consumption.

The energy efficiency of accessing main-memory data is suboptimal on CMPs employing contemporary processor-memory interfaces. Several DRAM chips, which compose a rank in a dual-inline memory module (DIMM), are involved per main-memory access, and the number of bits activated and restored per access could be more than 100 times that of a typical cache line size. As shown in Section 2.1, since memory access streams from multicore and manycore processors have lower correlation in access address (in other words, it looks more random) than those from a single core, conventional techniques of utilizing more data from activated bits by reordering memory access requests [Rixner et al. 2000] become less useful. Therefore, energy to activate data and restore them is largely wasted, causing a problem called memory *overfetch* [Ahn et al. 2008].

Recent proposals [Ahn et al. 2008; Ware and Hampel 2006; Zheng et al. 2008] share the main goal of saving dynamic main-memory access energy by dividing a memory rank into subsets, and making a subset rather than a whole rank serve a memory request. This saves dynamic power by reducing memory overfetch. We refer to this category of proposals as *rank subsetting*. While promising, these studies on rank subsetting are limited. Module threading [Ware and Hampel 2006] focused on microarchitectural details and bus utilization; mini-rank [Zheng et al. 2008] treated memory power and processor power in isolation; Multicore DIMM [Ahn et al. 2008]

did not address memory capacity issues or evaluate DRAM low-power modes. Since a subset of a rank effectively has a narrower datapath than a full rank, data transfer takes longer, which can negate the benefit of dynamic power saving. It is therefore hard to judge if rank subsetting provides enough benefits to computer systems when both processor and main memory are considered and other architectural techniques are taken into account. Moreover, none of these previous studies analyzed or devised solutions for high-reliability systems, which are critical for enterprise computing.

In this article, we holistically assess the effectiveness of rank subsetting on the performance, energy-efficiency, and reliability of a whole system, not just of individual components. We first model memory system power analytically including the degree of rank subsetting, memory capacity, system performance, and reliability. Then we compare the microarchitecture of two promising rank-subsetting proposals, Multicore DIMM and mini-rank, and estimate the system-wide implications of the two proposals. We validate these analyses by simulating a chip-multiprocessor system using multithreaded and consolidated workloads. We also develop a novel solution which extends the Multicore DIMM design for high-reliability systems, and show that compared with conventional chipkill [Dell 1997] approaches it can lead to much higher system-level energy efficiency and performance at the expense of additional DRAM capacity. Throughout our evaluation, we consistently use the *system energy-delay product* metric to judge the efficacy of rank subsetting in the context of a whole system.

Our key findings and contributions regarding rank subsetting in modern and future processor-memory interfaces are as follows.

— From model analyses and simulations, we show that 2 to 4 rank subsets is a sweet spot in terms of main-memory power and system energy-delay product. The best configuration depends on application characteristics and memory system capacity.
— Our study shows that subsetting memory ranks and exploiting DRAM low-power modes are largely complementary, since the former technique is applied to saving dynamic energy on memory accesses while the latter one is effective when DRAM chips mostly stay idle. They can be synergistic as well, which is especially apparent on high-reliability systems since both access and static power of main memory take a large portion of total system power.
— We extensively compare two recently proposed memory module architectures. This includes the first results for mini-rank DIMMs which take processor power into consideration.
— Finally, we demonstrate that rank subsetting affords a new category of trade-off in the design of high-reliability memory systems. Traditional chipkill solutions achieve energy-*inefficient* chip-level error recovery at no capacity or component cost relative to conventional ECC, whereas rank subsetting enables energy-efficient reliability in exchange for reduced capacity-efficiency.

In a typical high-throughput server configuration with 4 ranks per memory controller and 4 memory controllers in a system, dividing a memory rank into four subsets and applying DRAM low-power modes provides 22.3% saving in memory dynamic power, 62.0% in memory static power, and 17.8% improvement in system energy-delay product with largely unchanged IPC (instructions per cycle) on tested applications. In high-availability systems that work correctly even with a failed chip in every memory rank, and when DRAM low-power modes are used whenever it is profitable, Multicore chipkill DIMM uses 42.8% less dynamic memory power, and provides a 12.0% better system energy-delay product compared with a conventional chipkill system of the same data capacity while needing 22.2% more DRAMs.

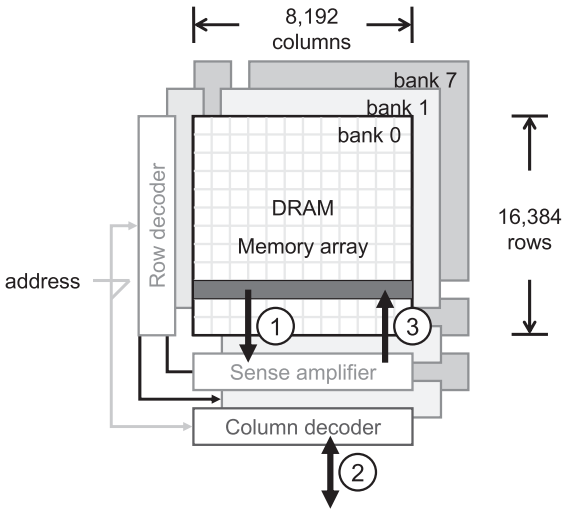


Fig. 1. A canonical representation of a DRAM chip with 8 banks. The movement of data during a typical memory transaction consists of an ACTIVATE command (1), READ or WRITE commands (2) and PRECHARGE command (3).

2. ENERGY EFFICIENT AND RELIABLE MEMORY MODULES

In this section we first review modern processor-memory interfaces and the concept of rank subsetting, which has been recently proposed to improve the energy efficiency of main-memory accesses. System-level impacts of rank subsetting are analyzed on performance, energy-efficiency, and reliability, which are all tightly coupled. Then we qualitatively compare two promising rank-subsetting proposals called Multicore DIMM and mini-rank, and extend the Multicore DIMM design for high-reliability systems.

2.1 Background

2.1.1 Organizations of Modern DRAM Devices and Memory Channels. Main-memory systems are built from DRAM modules, which in turn are composed of DRAM chips [Jacob et al. 2007]. DRAM is used to store main-memory data since its storage density (smaller than $10F^2$ per cell, where F is the minimum feature size of a fabrication process) is much higher than that of SRAM (bigger than $100F^2$ per cell) [Thoziyoor et al. 2008b] and its random access time (lower than 100ns) is much lower than that of NAND flash (in the order of us) and rotating (not MRAM) magnetic (in the order of ms) storage. Modern DRAM chips contain billions of bit cells, organized as a number of two-dimensional banks. The data output, command, and address signals of a DRAM chip are all shared by its banks. Figure 1 shows a canonical representation of a DRAM bank, numbered to indicate the salient phases of a typical interaction with a bank. First, a row-level command called ACTIVATE (1) is issued to the chip, instructing a specified bank to latch all of the bits in a given row into sense amplifiers and restore them to the original bit cells since the readout from the row is destructive. Subsequently, one or more column-level commands (READ or WRITE) may follow (2), initiating data transfers. The number of bits transferred per READ or WRITE command is the product of the data path width of the chip (typically 4, 8, or 16 bits per chip, termed $\times 4$, $\times 8$, and $\times 16$) and the burst length (8 in DDR3 [JEDEC 2007]). Once a sequence of read or write operations is over, a row-level command called PRECHARGE (3) is sent to the bank in order to precharge the bitlines in preparation

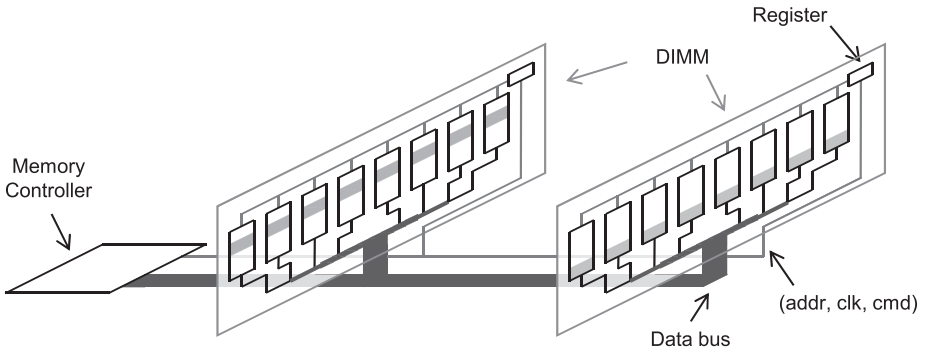


Fig. 2. A conventional memory channel where a memory controller and two memory modules are connected through a shared bus. Each memory module is composed of one or more ranks, each rank with 8 or 16 DRAM chips.

for the next ACTIVATE command. When two consecutive memory accesses to the same bank of the same DRAM rank select different rows, a DRAM *row-buffer conflict* [Zhang et al. 2000] occurs requiring PRECHARGE and ACTIVATE commands before the second row can be accessed. Were the accesses to go to the same row (as is common in single-threaded workloads), these row-level commands could be omitted.

The bandwidth and capacity demands from a modern microprocessor keep increasing, since the processor has more cores, the cache size per core does not increase much, and emerging applications such as in-memory databases need even higher bandwidth, more capacity, or both from main memory. A single DRAM chip therefore cannot satisfy the latency, bandwidth, and capacity demands of a microprocessor as a main memory. As a result, several (typically 8 or 16) DRAM chips compose an access unit called a *rank*. All DRAM chips in a rank operate in unison, that is, receiving the same control (address and command) signals and transferring data in parallel. One or more ranks are packaged on a printed circuit board and called a memory module. Dual in-line memory modules (DIMMs) are widely used in contemporary computer systems that have 64-bit data buses. Memory modules are connected to a memory controller, which feeds control signals to ranks and exchanges data through a shared bus to form a memory channel.

Figure 2 shows a conventional memory channel which contains two DIMMs attached to a memory controller through a bus. In a memory channel, commands generated from the memory controller may be issued to one rank while transferring data to or from another rank in a pipelined fashion. A module with several ranks is logically similar to having several modules, each with only one rank. Command, clock, and address signals from a memory controller are broadcast to all DRAM chips in all ranks on a memory channel. As a result, all of the DRAM chips within a rank act like a single DRAM chip with a wider data path and longer rows. Chip select signals are used to mask unintended recipients from commands. Memory controllers have historically been placed outside of microprocessors (in a chip called the northbridge), but are more recently integrated [Bohr 2009; Keltcher et al. 2003].

2.1.2 Trends in Modern DRAM Devices and the Overfetch Problem. As the throughput of microprocessors has increased, significant effort has been invested in improving the data path bandwidth of DRAM chips. This is primarily achieved by boosting the signaling rate of the data bus and by internally coarsening the granularity of access in a DRAM chip (essentially, enlarging the number of bits that are fetched from rows in parallel). Despite the increase in the capacity and bandwidth of DRAM chips, the latency

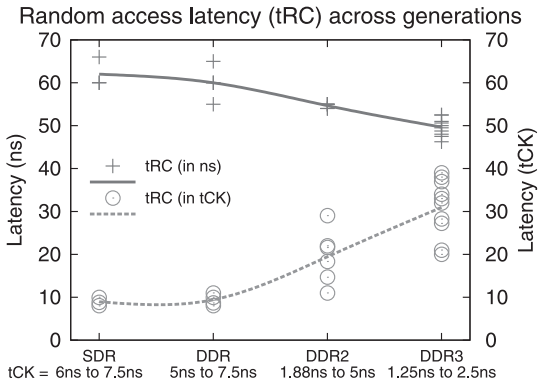


Fig. 3. Random access latencies of chips from several speed bins of successive DRAM generations are shown. While random access latency has fallen slightly in absolute terms from SDRAM to DDR3, it has increased significantly in terms of clock cycles (tCK).

of a random access (tRC) composed of a sequence of PRECHARGE, ACTIVATE, and READ/WRITE in a chip has remained relatively constant in terms of absolute time, and has increased in terms of bus cycles, as seen in Figure 3. As data transfer rates improve, the maximum number of ranks attached to a bus decreases and control signals are registered per rank, due in both cases to signal integrity issues. To enhance energy efficiency, memory controllers may utilize the low-power states of DRAMs when the requests from the processors are infrequent [Hur and Lin 2008].

An error correcting code (ECC) is often employed to cope with hard and soft errors on data storage and communication paths. Single bit error correction and double bit error detection (SECDED) is the most common scheme, but some computer systems need higher levels of error correction. The most well known technique to correct multibit errors is chipkill [Dell 1997], which protects against single memory-chip failure.

While the organizational philosophy of DRAM chips has changed little over the past several technology generations, their capacity and communication throughput have followed Moore's law [Jacob et al. 2007]. As the capacity of DRAM chips has grown over successive generations, the number of rows and the number of banks in a chip have each increased to keep pace. The length of a row (termed the *page size*) of modern DRAM chips is between 4K and 16K bits. Assuming that the page size is 8K bits and a rank consists of 8 DRAM chips, 64K bits are activated by an ACTIVATE command to the rank. Considering that the unit of most accesses to DRAM ranks is a cache line and the size of a cache line is typically 1K bits or below, only few bits are used by column accesses relative to the page size following a row activation. This phenomenon is called overfetch.

This memory overfetch can significantly lower the energy efficiency of accessing main-memory data stored in modern DDR*x* DRAM chips. Figure 4 shows a power breakdown of a Micron 2Gb DDR3 DRAM chip [Micron Technology Inc. 2006]. The labels $\times 8$ and $\times 16$ indicate the width of a data path and row/col means the ratio between row-level command pairs (ACTIVATE/PRECHARGE) and column-level commands (READ or WRITE). Row-level commands not only take a long time, but also consume a lot of power. When row/col = 1, ACTIVATE and PRECHARGE take more than half the DRAM power. Thus, it is desirable to decrease the row/col ratio. In modern DRAM chips, refresh power is much smaller than other components; refresh power consumption is mainly noticeable on large memory capacity configurations such as those in Section 4.3.

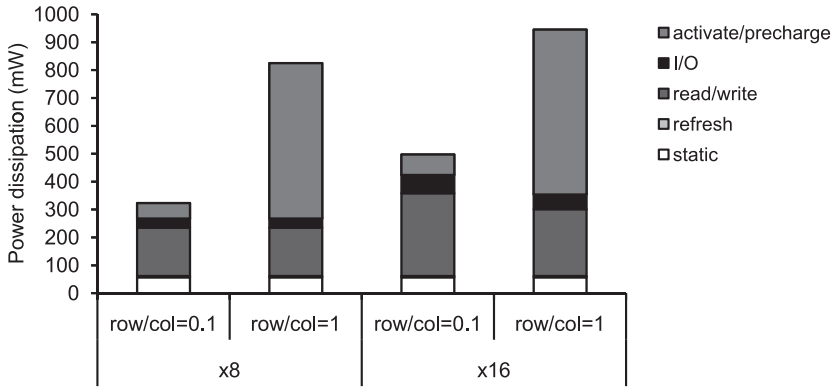


Fig. 4. DRAM power breakdown of a Micron 2Gb DDR3-1333 SDRAM chip Micron Technology Inc. [2006]. row/col means a ratio between row-level commands (ACTIVATE/PRECHARGE) and column-level commands (READ/WRITE). The DRAM is assumed to output read data for 50% of clock cycles and input data for 50% of clock cycles except the case of $\times 16$ and row/col = 1, where it inputs and outputs data for 40% of clock cycles each to satisfy the minimum delay between ACTIVATE commands to different banks.

2.1.3 Alleviating Overfetch by Memory Access Scheduling and Rank Subsetting. Out-of-order execution with nonblocking caches, simultaneous multithreading, chip multiprocessing, and direct-memory accesses [Hennessy and Patterson 2011] are now common in microprocessors. Such processors support multiple outstanding memory requests. A naïve memory controller serves these requests first-come-first-serve (FCFS). More advanced memory controllers adopt memory access scheduling schemes to reorder and group these requests to lower the DRAM row-buffer conflicts and to minimize the performance degradation due to various timing constraints on DRAM accesses. Reducing row-buffer conflicts helps with overfetch, too. There have been multiple memory access scheduling proposals [Mutlu and Moscibroda 2008; Nesbit et al. 2006; Rixner et al. 2000] to exploit these characteristics to achieve higher performance.

New processor-memory interfaces such as module threading [Ware and Hampel 2006], mini-rank [Zheng et al. 2008], and Multicore DIMM [Ahn et al. 2008] address the overfetch issue. These proposals alleviate the overfetch problem by dividing the DRAM chips within a rank into multiple subsets and making a subset (not a whole rank) serve a memory access. We call this technique *rank subsetting*. Rank subsetting requires minimal changes (a few additional address lines if existing address lines are not enough) to the existing processor-memory interface, since conventional DRAM chips can be used without modification. Memory controllers treat each subset as a separate rank with longer data transfer time, so that the modifications to the memory controllers are minor. Narrowing each data channel and providing more channels has effects similar to rank subsetting but requires substantial changes to memory module standards, more memory controllers, and more control signals from microprocessors to provide the same main-memory throughput. These proposals primarily save DRAM access energy, but have additional costs and benefits. Rank subsetting increases DRAM access latency and changes the effective bandwidth of memory channels. Since it changes the number of DRAM chips involved in a memory access, traditional reliability solutions such as chipkill must be revisited as well. So the effectiveness of rank subsetting must be assessed in the context of the performance, energy efficiency, and reliability of a whole system, not just of individual components.

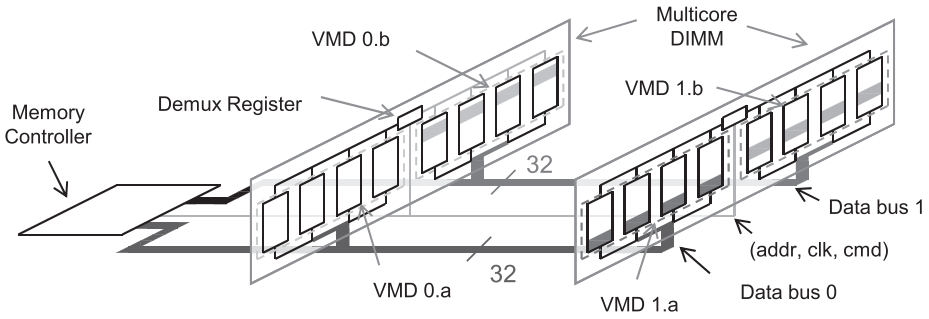


Fig. 5. A memory channel with two Multicore DIMMs (MCDIMMs) with each divided into two subsets called virtual memory devices (VMDs). Each MCDIMM has a demux register instead of a normal register routing control signals to each VMD.

2.2 Implications of Rank Subsetting

Rank subsetting alleviates the overfetch problem by dividing each rank into smaller subsets of chips and sending memory commands only to a subset. Figure 5 shows an exemplary Multicore DIMM memory channel with two memory ranks, one per DIMM. Each rank is divided into two rank subsets called virtual memory devices (VMDs). Physically the same data bus as in a conventional memory channel is used ($\times$64 in a DIMM memory channel), but it is divided into two logical data buses, each occupying half of the physical bus ($\times$32 per logical data bus). A demux register is placed on each rank, which routes (demultiplexes) control signals to the proper rank subset to provide independent operations.

The primary goal of rank subsetting is to improve the energy efficiency of memory systems by saving DRAM access energy, which is important since main-memory DRAM power can reach or surpass the processor power in high memory capacity or reliable systems as shown in Section 4. In order to understand how much energy can be saved by rank subsetting, we first identify the sources of DRAM power consumption. DRAM power can be categorized into two parts, static power and dynamic power. Static power is independent of activity, and mainly comprised of power consumed from peripheral circuits (like DLL and I/O buffers), leaky transistors, and refresh operations. Dynamic power can further be categorized to two parts since DRAM access is a two step process. First, bitlines in a bank of DRAM chip are precharged, and data in a row of the bank is delivered to the bitlines and latched (activated) to sense amplifiers by row-level commands. This consumes activate-precharge power. Then, a part of the row is read or updated by column-level commands. This consumes read-write power. Dynamic power consumption is proportional to the rate of each operation. However since a row can be read or written multiple times once it is activated, the rates of activate-precharge and read-write operations can be different.

We can model the total power consumed in a memory channel as follows. When D is the number of DRAM chips per subset, S is the number of subsets per rank, and R is the number of ranks per channel,

$$\text{Total main} - \text{memory power} = D \cdot S \cdot R \cdot SP + E_{RW} \cdot BW_{RW} + D \cdot E_{AP} \cdot f_{AP}, \quad (1)$$

where SP is the static power of a DRAM chip, E_{RW} is the energy needed to read or write a bit [1], BW_{RW} is the read-write bandwidth per memory channel (measured, not

[1]The major portion of read or write power consumption is from wires transferring control and data signals through chip-to-chip I/O and DRAM chip global interconnects, which is similar for both read and write operations. We therefore assume that both operations consume the same power as a first order approximation.

peak), E_{AP} is the energy to activate and precharge a row in a DRAM chip, and f_{AP} is the frequency of the activate-precharge operation pairs in the memory channel. The first term of Equation (1) is the static power portion of the total memory power, the second is the read-write power, and the third is the activate-precharge power which, due to overfetch, grows linearly with D. Assuming that misses from last-level caches are the dominant portion of memory access requests, $BW_{RW} = f_{CM} \cdot CL$ where f_{CM} is the frequency of cache misses and CL is the line size of the last-level caches. If we analyze f_{AP} further,

$$f_{AP} = \frac{f_{AP}}{f_{CM}} \cdot f_{CM} = \frac{f_{AP}}{f_{CM}} \cdot \frac{BW_{RW}}{CL} = \beta \cdot \frac{BW_{RW}}{CL}, \tag{2}$$

showing that the dynamic power of main memory is proportional to the read-write bandwidth per memory channel and β, the ratio of the number of rows being activated to the number of memory requests to the memory channel. β indicates the frequency of row-buffer conflicts.

Rank subsetting lowers D while keeping $D \cdot S$ constant. Equations (1) and (2) indicate that this mainly decreases activate-precharge power, which can be more than half of DRAM power [Micron Technology Inc. 2007] as shown in Figure 4. (We show below that SP and BW_{RW} are affected by rank subsetting as well.) Saving activate-precharge power is more significant if β is higher. The frequency of row-buffer conflicts depends on various factors including memory access patterns of applications and the number of applications running on computation cores, memory address interleaving across memory controllers, memory access scheduling policies in memory controllers, and the number of DRAM banks in memory systems. Section 4.1 further evaluates the dependency of row-buffer conflicts on the access patterns of running applications and the number of DRAM banks in an example chip-multithreaded (CMT) processor. Activate-precharge power can also be lowered by increasing the cache line size (CL). However CL is around 64bytes while the size of a row in a rank is typically 8 or 16Kbytes, which is orders of magnitude smaller. Also larger cache line sizes can increase miss rate and harm application performance [Gee et al. 1993; Woo et al. 1995].

Both dynamic power terms in (1) are proportional to the bandwidth of memory channels BW_{RW}, which is in turn proportional to instructions per cycle (IPC). Rank subsetting increases the access latency of memory requests. However, modern throughput-oriented CMPs such as the Sun UltraSparc T2 [Johnson and Nawathe 2007] can amortize or tolerate this latency with features like simultaneous multithreading or speculative out-of-order execution. They allow memory controllers to reorder and pipeline memory requests, to lower the correlation between memory latency and bandwidth.

Effective bandwidth depends on multiple factors such as load-balancing across memory channels and DRAM banks, memory access patterns, and inherent timing constraints of DRAM chips. In DDR3 [Micron Technology Inc. 2006], for example, 7.5ns is the minimum time from the end of a previous write transaction to the issuance of a following read command (`tWTR`). Recent DRAM chips are limited in the rate at which they can activate different banks in a device (`tRR` and `tFAW`). This limitation can lower the effective throughput since the row-buffer conflict ratio is high on modern CMPs. This throughput degradation can be alleviated when R or S increases. As R increases, there are more banks a memory controller can utilize and there is no timing constraint on activating rows in different ranks. Rank subsetting also effectively increases the number of independent DRAM banks on a module, since each subset can be accessing different banks. As S increases, cache line transfer time per rank subset increases reaching or surpassing the minimal inter-activate time, which effectively renders it

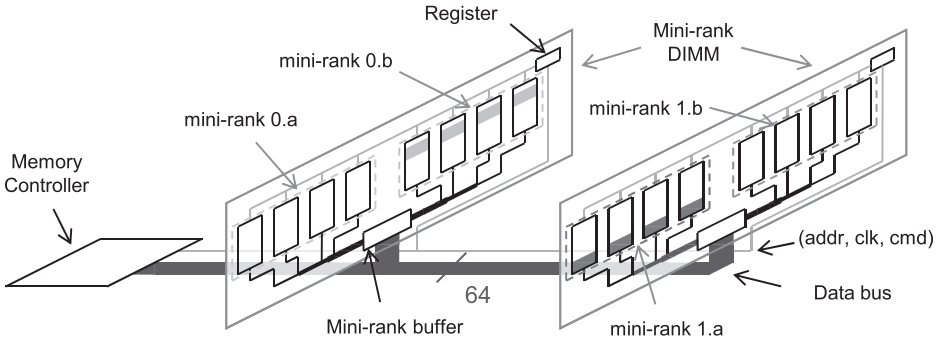


Fig. 6. A memory channel with two mini-rank DIMMs with each divided into two subsets called mini-ranks. Each mini-rank DIMM has 2 mini-ranks, a register buffering control signals, and a Mini-Rank Buffer (MRB) buffering data from all mini-ranks.

irrelevant. Increasing S reduces the effect of other timing constraints as well, such as switches between read and write operations and bus ownership changes. So the impact of rank subsetting (varying S) on system performance is determined by the interplay of these different factors, and also depends on the characteristics of the applications running on the system.

Furthermore, the number of ranks in a memory channel can significantly affect the memory system power. When R is large or the memory channel is not utilized frequently (BW_{RW} is small), static power dominates. The memory controller can then utilize low-power DRAM modes to decrease the static power (SP). This typically lowers BW_{RW} as well since it takes time for DRAM chips to enter or exit low-power modes, and commands to utilize low-power modes compete with other commands to DRAM.

It can be seen that rank subsetting changes both the energy-efficiency and performance of computer systems. So we need a metric to measure the effectiveness of rank subsetting at the system level, combining both performance and energy-efficiency instead of presenting memory system power and processor performance separately. We pick system energy-delay product (EDP), which is a product of the execution time of a workload running on the system and the energy consumed by computation cores and memory systems during execution.

2.3 Multicore DIMM and Mini-Rank

Besides the Multicore DIMM, the mini-rank DIMM [Zheng et al. 2008] is another module that implements rank subsetting, where each subset is called a mini-rank. The mini-rank DIMM places a device called the *Mini-Rank Buffer* on the data-path of the DRAM chips, which buffers data to and from the memory controller as shown in Figure 6. All mini-ranks on a memory channel share a single data bus, and all DRAM chips receive control signals (which are then gated by chip select signals). On the contrary, the demux register in a Multicore DIMM replaces the register normally found on a registered DIMM (Figure 5) and forwards control signal to individual DRAM chips. The data paths of DRAM chips in a Multicore DIMM are directly connected to the memory controller via data buses. Note that the Module threading scheme [Ware and Hampel 2006] does not have either the Mini-Rank Buffer or the demux register. It is much closer to the Multicore DIMM since divided data buses are connected directly to the memory controller and the functionality of the demux register is integrated to the controller. Due to these similarities, only Multicore DIMM and mini-rank are compared qualitatively or quantitatively hereafter.

While Multicore DIMM and mini-rank are conceptually similar, choices made in their microarchitectural design cause noticeable differences in performance and energy efficiency of the memory interfaces.

— Because the Mini-Rank Buffer holds and retransmits data, more I/O power is needed compared to the demux register, which holds and forwards control signals.
— The demux register can translate a column level command to multiple commands. This saves control signal bandwidth and I/O power when a DRAM chip does not support longer bursts, whereas a memory controller serving conventional DIMMs or mini-rank DIMMs must send multiple column commands to transfer a cache line.
— Since all data transfers to/from all mini-ranks in a DRAM rank go through the Mini-Rank Buffer, a data transfer is more likely affected by other data transfers. For example, when the MCDIMM memory controller in Figure 5 sends a sequence of data to VMD 1.a while it receives data from VMD 1.b, the reads and writes happen concurrently. With the mini-rank DIMM (Figure 6), the memory controller confronts timing constraints due to a bus ownership change [Jacob et al. 2007] when sending data to mini-rank 1.a and receiving from mini-rank 1.b. Since data transfers happen during shorter periods of time through a mini-rank's wider bus compared to an MCDIMM, this sort of timing constraint (which is absolute in time) proportionally penalizes mini-rank DIMMs worse and increases the complexity of of memory controllers supporting mini-rank DIMMs.
— Mini-rank DIMMs can achieve higher data transfer rates when a memory channel has multiple ranks and memory accesses are not evenly distributed across subsets. For example on the MCDIMM configuration in Figure 5, if all the memory requests are concentrated to VMD 0.b and 1.b, the data transfer rate cannot be more than the half of the peak bandwidth. However on the mini-rank DIMMs (Figure 6), full bandwidth can be achieved even though all the requests are to mini-rank 0.b and 1.b since they share a single wide data bus.

2.4 Adding Reliability to Multicore DIMMs

2.4.1 Rank Subsets with Single-Bit Error Correction and Double-Bit Error Detection. Soft and hard errors occur frequently on modern DRAM devices [Schroeder et al. 2009]. Memory systems adopt error correcting codes (ECCs [Peterson and Weldon 1972]) that add redundant or parity symbols (typically bits) to data symbols in order to recover from these errors. Since data stored in DRAM devices are accessed at a block granularity such as an OS page or a cache line, linear block codes that are processed on a block-by-block basis are popular. An n-symbol codeword of a linear block code is composed of k data symbols and $(n - k)$ parity symbols that are encoded and decoded together. The redundancy of a code is described in terms of its code rate, k/n, which is a ratio of data symbols to overall symbols. The minimal Hamming distance of two codewords is the number of locations where the corresponding symbols are different and determines the strength (the number of detectable or correctable errors) of the code. Codes with lower code rates or higher redundancy have higher Hamming distance so that they can correct or detect more errors. The most popular codes used in memory systems enable single-bit error correction and double-bit error detection (SECDED), in which 8 parity and 64 data bits occur in a codeword of 72 bits, so the code rate is 8/9. It is a Hamming code [Peterson and Weldon 1972] with the minimal Hamming distance of 4, which is needed to correct one symbol error and detect two symbol errors. Since it has 64 data bits, it is used on a rank and one $\times 8$ DRAM chip or two $\times 4$ chips are added to the rank to provide the parity bits.

There are two ways to support SECDED-level reliability on Multicore DIMMs. One way is to add a DRAM chip per subset to supply parity bits. When a rank is divided into S rank subsets, each subset has a data path width of $64/S$. 64 data bits of the 72-bit SECDED codeword are hence from S bursts of the rank subset, so 8 parity bits can be provided by a DRAM chip with a data path width of $8/S$. When a rank has more than 2 subsets, the data path width needed for parity is lower than 4. Since DRAM chips with this low data path with are not popular, the code rate (aka coding efficiency) of supporting SECDED-level reliability is lower on Multicore DIMMs compared to conventional DIMMs. One exception is the case of 2 subsets having 9 $\times$4 DRAM chips each. The other way is to use a DRAM device with a wider data path, such as $\times$9, instead of $\times$8. Though not popular, some DRAM vendors provide such configurations (RLDRAM [Micron Technology Inc. 2008] and RDRAM [Rambus 1999]).

2.4.2 Rank Subsets with Single-Chip Error Correction and Double-Chip Error Detection. SECDED schemes protect against single-bit errors, such as DRAM bit cell faults and wire stuck-at faults at DRAM ranks or rank subsets. High-availability systems often demand the stronger reliability of single-chip error correct, double-chip error detect (SCCDCD) schemes, sometimes called *chipkill*. These schemes can correct the failure of an entire DRAM chip, and detect the failure of two DRAM chips. There are two common practices for implementing SCCDCD reliability in memory systems: interleaving SECDED-quality codewords across multiple ranks [Dell 1997], as implemented by IBM's xSeries, or employing stronger error correcting codes [AMD 2007], as found in the AMD Opteron.

The first scheme observes that codewords can be interleaved across ranks such that no more than 1 bit of a codeword comes from a single DRAM chip. For example, if a rank supporting SECDED-level reliability consists of 18 $\times$4 DRAM chips, we combine 4 of these into a single conceptual rank consisting of 72 chips. In this case, four SECDED codewords of 72-bits interleaved across 288-bits will suffice to recover from whole chip failures, since each bit from a DRAM chip belongs to a different codeword. While conceptually simple, this solution suffers from the fact that all DRAM chips in the conceptual rank (72 chips in the example above) must be activated on each access exacerbating the memory overfetch problem and leading to poor dynamic energy efficiency. This type of SCCDCD solution is also impractical with $\times$8 and $\times$16 chips, since they would require 8 and 16 ranks per transaction, respectively. It should be noted that DRAMs with long minimum burst lengths (like DDR3, which has a minimum burst length of 8) render SCCDCD schemes that interleave transactions across several ranks unreasonable, since they grow the memory transaction length beyond that of a typical cache line. For example, were the IBM chipkill solution to be implemented with $\times$4 DRAM chips with burst-length 8, each memory transaction would cover four 512 bit cache lines.

The other approach observes that DRAM chip failures manifest as a burst of bit-errors in codewords with a burst-length the same as or multiples of the width of the chip's data path (e.g. failure of a $\times$4 chip is a burst-error of length 4). In practice, nonbinary cyclic error-correcting codes like Reed-Solomon (RS) codes can be used to support the SCCDCD feature [Xambo-Descamps 2003]. While a bit is a symbol in the SECDED Hamming code, multiple bits constitute a symbol in the RS codes, and a natural symbol size for the chipkill protection is the data path width of a DRAM chip so that errors from a chip can be regarded as a single symbol error, which is correctable. Using m-bit symbols, the maximum codeword size is $2^m - 1$ symbols. An RS code is a maximum distance separable code that has the minimum Hamming distance $k + 1$ with k parity symbols, so that at least 3 parity symbols (so the minimum Hamming distance 4) in addition to the data symbols are needed to detect two symbol errors and

Table I.

This table compares a baseline system without parity, conventional chipkill solutions [AMD 2007; Dell 1997], and several configurations of SCCDCD MCDIMMs (reliability-enhanced MCDIMM). Min TX is the minimum memory transaction size assuming DDR3's burst length of 8. Chips/TX is the number of DRAM chips activated per memory transaction. The IBM chipkill solution can correct certain types of multichip errors while the Opteron and MCDIMM solutions provide protection against equivalent categories of single-chip errors.

	No parity	**IBM**	**Opteron**	MCDIMM $S=2$			MCDIMM $S=4$			MCDIMM $S=8$	
Chip width	×4	×4	×4	×4	×8	×16	×4	×8	×16	×4	×8
Min TX (bytes)	64	256	128	32			16			8	
Chips/TX	16	72	36	11	7	5	7	5	4	5	4
Code Rate	1.0	0.89	0.89	0.73	0.58	0.4	0.58	0.4	0.25	0.4	0.25

correct one symbol error. An RS code with 4-bit symbols can't be used for standard DIMMs since its maximum code size is $((2^4 - 1) \times 4 =)60$ bits, which is lower than the 72-bit data path of a standard DIMM. Hence a RS code with 8-bit symbols should be used instead, with all the bits from a given DRAM chip's output contributing to the same symbol. When ×4 chips are used, data from 2 bursts of each chip compose an 8-bit symbol. Note that the AMD Opteron uses 2 ranks of ×4 chips to construct a 144-bit cyclic code [AMD 2007]. The Opteron solution, with 4 parity chips, comes close to this theoretic lower bound. However, this result also means that the Opteron solution is only practical with ×4 chips, since 2 ranks of ×8 chips would only provide 2 parity chips. Both the IBM and Opteron solutions achieve a high code rate of 8/9. Compared to the IBM chipkill solution that needs 72 DRAM chips, the Opteron solution needs half of them but still activates 36 DRAM chips per transaction, leading to poor energy efficiency.

The approach of using multibit symbols can be applied to Multicore DIMMs to support SCCDCD-level reliability by adding 3 chips per rank subset to achieve the minimum Hamming distance of 4. In the case of 2 subsets ($S = 2$) with ×4 chips, this equates to 11 chips per subset (8 for data and 3 for parity). Compared to the conventional chipkill solutions described earlier, this worsens the code rate from 8/9 to 8/11. But only 11 DRAM chips are activated per transaction, instead of 36 (Opteron) or 72 (IBM), leading to substantial access energy savings. The number of chips per transaction for several configurations of MCDIMMs is compared with that of conventional chipkill solutions in Table I. While the Opteron and IBM chipkill solutions are only practical for ×4 chips, it is feasible to implement SCCDCD MCDIMMs across several potential configurations ranging from 2 to 8 rank subsets and using either ×4, ×8, or ×16 chips. Each configuration represents a compromise between code rate and energy efficiency. For example, when comparing SCCDCD with 2 subsets of ×4 and ×8 chips, the ×4 configuration activates 11 chips per transaction while the ×8 configuration only activates 7, leading to improved energy efficiency. On the other hand, the ×4 configuration has the code rate of 73%, while the ×8 configuration has the code rate of 58%. Another advantage of using ×4 DRAM chips for Multicore DIMMs is that a RS code with 4-bit symbols can be used, lowering the complexity of encoding and decoding parts of the codewords [Sarwate and Shanbhag 2001].

2.4.3 Implications and Limitations of Adding Reliability to Multicore DIMMs. We can apply Equations (1) and (2) to model the power consumption of a Multicore DIMM memory channel augmented with SCCDCD reliability. Here D is the number of DRAM chips activated to provide both data and parity bits, and R is the number of ranks that operate independently within a channel. For example, $R = 1$ on a memory channel with 2

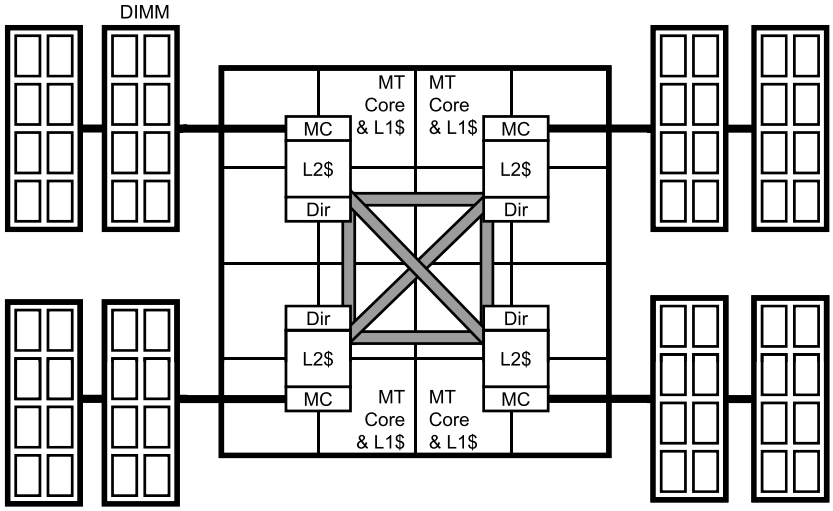


Fig. 7. System architecture assumed in this article. A processor consists of 16 in-order cores, 4 threads per core, 16 L1I and L1D caches, 4 L2 caches, 4 memory channels (MCs), and 4 directories (Dirs). From one rank to four ranks (i.e., two dual-rank DIMMs) are connected per memory channel.

ranks employing the chipkill protection scheme in the AMD Opteron since all 36 chips in both ranks operate in unison. When the total memory capacity excluding parity bits of a system stays constant, implementing SCCDCD-level reliability increases D, decreases R, and increases E_{RW} due to parity overhead. Dynamic power plays a bigger role in the memory system power due to these changes.

Practical implementations of the high-reliability techniques discussed in this section need to take into account issues of form-factor, module compatibility, and memory channel wire-count into consideration. For instance, while it is easy to imagine a single physical module specification that could be shared between modules with 2, 4, or 8 rank subsets, SCCDCD protection presumes a distinct datapath width and part count between different numbers of rank subsets. First, we observe that module slots already have pins dedicated to ECC even though many systems use DIMMs that do not have ECC chips. Similarly, we expect that module standards for DIMMs which incorporate rank-subsetting would include some number of pins for ECC chips. Second, like the Opteron SCCDCD solution, the implementation of SCCDCD MCDIMMs need not match the theoretic upper bound for code rate. For example, on an SCCDCD MCDIMM with 2 rank subsets and using $\times 4$ or $\times 8$ chips, SCCDCD reliability could be achieved by spreading codewords across both rank subsets. While counterproductive towards the goal of minimizing overfetch, this solution is still far more energy efficient than the conventional SCCDCD alternatives, and could reduce transfer times.

3. EXPERIMENTAL SETUP

To evaluate the impact of rank subsetting in the context of performance, energy-efficiency, and reliability, we model a Niagara-like system [Kongetira et al. 2005] (Figure 7) with multiple cores and memory channels. A processor has 16 in-order cores and 4 threads per core, for a total of 64 concurrent threads. The cores run at 2GHz and process up to 1 instruction and 1 memory access per cycle. Each core has its own separate L1 instruction and data cache, while there is an L2 cache shared by each cluster of 4 cores. L2 caches are not shared between clusters. All caches have 64B

Table II.

Power and performance parameters of the memory hierarchy used in this study. On DRAM chips, dynamic read energy includes precharge and activation energy assuming random access sequences. All caches use 64-byte blocks.

	Cache			Directory	DRAM chip	
	L1 I	L1 D	L2		×4	×8
Capacity	16KB×16	32KB×16	1MB×4	×4	4Gb	8Gb
Associativity	4	4	8	32	N/A	N/A
Access time	1 cycle	2 cycles	4 cycles	4 cycles	93 cycles	95 cycles
Cycle time	1 cycle	1 cycle	2 cycles	2 cycles	55 cycles	59 cycles
Dynamic read energy	0091nJ	0.095nJ	0.183nJ	0.021nJ	1.32nJ	1.52nJ
Static power	4.8mW	8.9mW	185mW	86.5mW	75.6mW	104.8mW

cache lines. There is a crossbar between the 4 L2 caches and 4 memory controllers. A hierarchical MESI protocol is used for cache coherency and a reverse directory, similar to what was implemented in the Niagara processor, is associated with each memory controller. A memory controller is connected to from 1 to 4 ranks and there are either 1, 2, 4, or 8 rank subsets (VMDs or mini-ranks) per memory rank (S). When $S = 1$, cache line transfer time in a data bus is 4ns or 8 cycles. As S doubles, cache line transfer time doubles as well. The controller has 32-entry scheduling buffers (or windows) and employs the Parallelism-Aware Batch Scheduling algorithm [Mutlu and Moscibroda 2008], where the requests in the current batch or group have the highest priority, requests that can be served by single column-level commands are prioritized within a batch, and an older request is served earlier than a younger one when both have the same priority. An XOR-based [Frailong et al. 1985] interleaving scheme is used to map memory addresses across memory controllers, ranks, and rank subsets pseudorandomly in rank-subset page granularity. (A rank-subset page is the product of the number of DRAM devices per subset and the number of columns in a DRAM bank.)

A 32nm process technology based on ITRS projections is assumed for both processor and memory chips. We use CACTI 5.3 [Thoziyoor et al. 2008a] to compute access time, cycle time, dynamic energy, and static power of uncore parts such as caches, directories, and DRAM chips, as summarized in Table II. To compute the power of processor cores, we use McPAT [Li et al. 2009], whose technology modeling is built on CACTI 5.3's technology model. DDR3 DRAM chips are assumed for the main memory, with a prefetch size of 8, a row size of 16,384 bits, 8 banks per chip, and 2Gbps data pin bandwidth. The product of prefetch size and data-path size is the smallest possible number of bits accessed when a row is read or written. The energy dissipated by a data bus of a memory channel is calculated as the DC power of the output driver and termination resistance specified in the DDR3 standard [Micron Technology Inc. 2007] after scaling the operating voltage from 1.5V to 1.0V in order to account for a 32nm process. As more ranks are attached per memory channel, more energy is dissipated per data transaction. In a memory channel with mini-rank DIMMs, a data bus is divided into two parts by Mini-Rank Buffers. The part between the memory controller and the Mini-Rank Buffers is the same as a normal data bus, while the part between each Mini-Rank Buffer and the DRAM chips in each mini-rank DIMM is the same as a data bus in a memory channel with a single rank. This is reflected during memory I/O power calculation. The energy consumption of the address and command bus is calculated by computing the capacitance of driver, wire, and load [Ghosh and Lee 2007]. A memory controller puts a DRAM in a low-power state (similar to the precharge power-down mode in DDR3) when all banks in it are at the precharge state. We assume that

Table III. SPLASH-2 Datasets and SPEC 2006 Application Mixes Are Listed with L2 Misses per Instruction

SPLASH-2

Application	Dataset	L2 miss per instruction
Barnes	16K particles	0.0003
Cholesky	tk17.O	0.0030
FFT	1024K points	0.0051
FMM	16K particles	0.0006
LU	512×512 matrix	0.0004
Ocean	258×258 grids	0.0073
Radiosity	room	0.0003
Radix	8M integers	0.0186
Raytrace	car	0.0017
Volrend	head	0.0007
Water-Sp	512 molecules	0.0001

SPEC CPU2006

Set	Applications	L2 miss per instruction
CFP		
high	433.milc, 450.soplex, 459.GemsFDTD, 470.lbm	0.0219
med	410.bwaves, 434.zeusmp, 437.leslie3d, 481.wrf	0.0099
low	436.cactusADM, 447.dealII, 454.calculix, 482.sphinx3	0.0073
CINT		
high	429.mcf, 462.libquantum, 471.omnetpp, 473.astar	0.0189
med	403.gcc, 445.gobmk, 464.h264ref, 483.xalancbmk	0.0046
low	400.perlbench, 401.bzip2, 456.hmmer, 458.sjeng	0.0037

a DRAM chip in a low-power state consumes 20% of normal static power and needs two cycles to enter and exit the state [Micron Technology Inc. 2006].

We developed a multicore simulation infrastructure in which a timing simulator and a functional simulator are decoupled in a way similar to GEMS [Martin et al. 2005; Shen and Lipasti 2005]. A user-level thread library [Pan et al. 2005], which was developed as a Pin [Luk et al. 2005] (version 2.4) binary instrumentation tool, is augmented to support additional pthread library APIs, such as `pthread_barriers`, and used as a functional simulator to run multithreaded applications. An event-driven timing simulator, which models in-order cores, caches, directories, and memory channels, controls the flow of program execution in the functional simulator and effectively operates as a thread scheduler.

We perform experiments with the SPLASH-2 [Woo et al. 1995], PARSEC [Bienia et al. 2008], and SPEC CPU2006 [McGhan 2006] benchmark suites. For multithreaded workloads, 64 threads are spawned per workload and each thread is mapped to a hardware thread statically. All 11 SPLASH-2 applications are used while only 6 PARSEC applications (canneal, streamcluster, blackscholes, facesim, fluidanimate, and swaptions) are used (due to a Pin limitation). PARSEC applications are executed with the simlarge dataset while our SPLASH-2 inputs are summarized in Table III. To model multiprogrammed workloads, we consolidate applications from SPEC CPU2006. The SPEC CPU2006 benchmark suite has single threaded applications consisting of integer (CINT) and floating-point (CFP) benchmarks. We made 3 groups each of integer and floating-point benchmarks, 4 applications per group, based on their L2 cache miss ratio [Henning 2007], which are listed in Table III. It also shows the number of L2 misses per instruction, which is measured by using the baseline configuration in Section 4.2. Simpoint 3.0 [Sherwood et al. 2002] is used to find several simulation phases (100 million instructions per phase) and weights. For each CINT and CFP

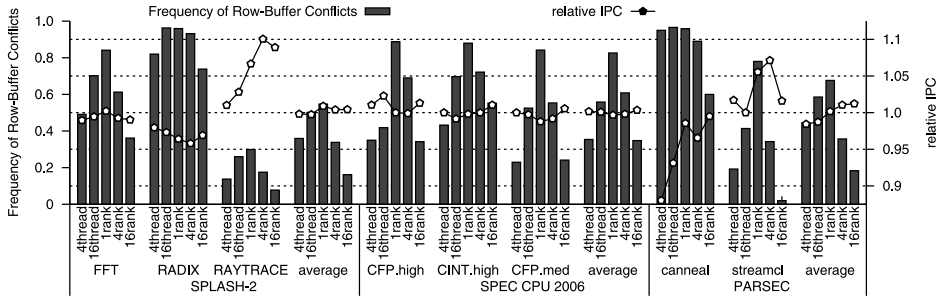


Fig. 8. Frequency of row-buffer conflicts (β) and relative IPC when the number of active threads and ranks are varied. `nthread` configurations have `n` active threads with 1 rank per memory controller. `mrank` configurations have `m` ranks per memory controller with 64 active threads. Relative IPC is the ratio of the IPC with an *open* policy over the IPC with a *closed* policy on an application.

set, 16 simulation phases per application are consolidated so that each hardware thread executes a simulation phase. The number of instances per phase of each SPEC 2006 benchmark is proportional to its weight. We simulate each workload until the end or up to 2 billion instructions. We skip initialization phases of the SPLASH-2 and PARSEC benchmark suites. gcc 4.2 is used to compile all the benchmark programs.

4. RESULTS

We evaluate the impact of rank subsetting on memory power, processor performance, and system energy-delay product using the configurations described in the previous section. In particular, we focus on understanding the interplay between subsetting DRAM ranks and utilizing DRAM power-down modes as the capacity and the reliability level of the memory systems are varied.

4.1 Impact of the Number of Ranks on the Frequency of Row-Buffer Conflicts

The parallelism-aware batching scheduling algorithm [Mutlu and Moscibroda 2008] we chose for memory access scheduling does not specify when to precharge a DRAM bank. Rixner et al. [2000] evaluated two policies called *closed* and *open* with regard to the precharge timing that showed noticeable differences on media applications. A closed-policy controller closes a row of a DRAM bank that has no pending requests. In the open-policy, an open row remains open until a request arrives for a different row of the same rank. There won't be much of performance difference between the two policies when memory controllers have enough pending requests so that there exist multiple requests to most of DRAM banks and it is not needed to speculatively open or close rows, which is the case for Ahn et al. [2006]. However, the number of pending requests from computation cores is limited due to various factors such as the lack of memory-level parallelism of applications, limited buffer sizes, or core microarchitecture. For example, the microprocessor we evaluate can have up to 64 pending memory requests since it has in-order cores with 64 concurrent threads. So unless the number of pending requests is much more than the number of DRAM banks in the system, speculative decisions are needed on deciding the bank precharging time and could have a noticeable influence on system performance and energy efficiency.

Figure 8 shows the frequencies of DRAM row-buffer conflicts (β) and the relative IPCs of 3 benchmark suites on systems that the number of active threads or the

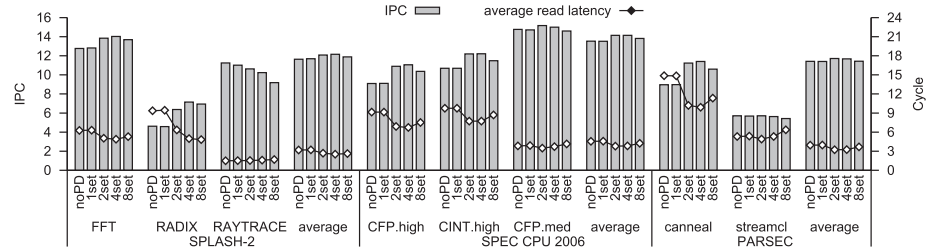
number or ranks per memory controller is varied. There are five configurations on each application. The two leftmost configurations have one rank per memory controller with 4 and 16 active threads. Three remaining configurations have 64 active threads with 1, 4, and 16 ranks per memory controller. For each suite, applications which do not access main memory frequently are not shown due to space limitations, but they are included when average values are computed. The open policy is used to measure the frequency of row-buffer conflicts. The relative IPC of an application is the ratio of the application IPC with the open policy over the one with the closed policy.

Average β decreases in all three cases with increasing S or fewer active threads. This is because memory requests from the same thread on many applications have spatial locality. Requests from other threads that disrupt spatial locality due to a row-buffer conflict are less likely when there are more ranks. When, as in RADIX and canneal, there is low spatial locality, this effect is slightly less. When β is close to unity, the closed policy lowers the average access latency because the precharge delay is off the critical path. On the contrary, the open policy is beneficial if β is low since a memory request can often be served by a single column-level command. β is not always tightly coupled to the relative IPC though, because the difference in memory system access latency affects the core performance only when the latency hiding techniques in the cores (chip multithreading in our test system) cannot tolerate the cache miss latency. Since RADIX and canneal are memory bandwidth intensive applications, the closed policy provides higher IPC. On many applications, however, the open policy provides similar or even higher performance than the closed policy, especially when the system has more ranks. Considering that lower β will dissipate less main-memory power and rank subsetting will provide more DRAM banks to the system, we choose the open policy for the remainder of the evaluation.

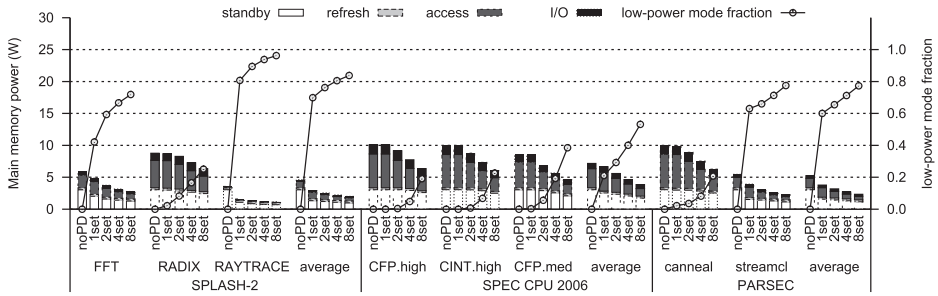
4.2 Single-Rank Performance and Power Efficiency

We first explore the optimal numbers of rank subsets that provide higher IPCs and lower system energy-delay products when we place 1 memory rank per memory controller (R = 1). We also identify major factors that improve the system energy-delay products. Figure 9 shows the performance and power result of 3 benchmark suites on a system with Multicore DIMMs where each memory controller has 1 memory rank. There are five configurations on each application. The left-most configuration has one subset per memory rank and does not exploit the low-power mode of DRAM chips, which is the baseline configuration. Four remaining configurations have their memory controllers exploit the low-power mode and the number of rank subsets are varied from 1 to 8. For each suite, applications whose performance is not sensitive to the number of rank subsets are not shown due to space limitations, but they are included when average values are computed. Off-chip memory demands depend on the instructions per cycle (IPC), the number of memory requests per instructions, and the last-level cache miss rates. Figure 9(a) shows the IPC and the average read latency while Figure 9(b) shows the memory power breakdown of applications. Static power is divided into refresh and standby power, while dynamic power is divided into chip I/O power and access power within DRAM chips performing read, write, activate, and precharge operations.

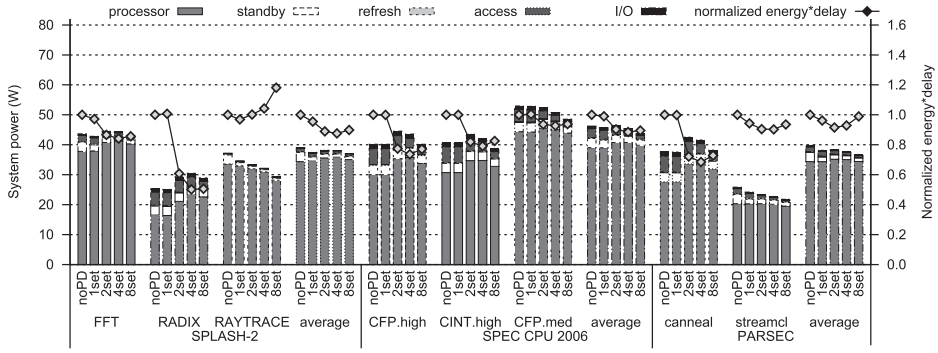
RADIX, CFP.high, CINT.high, and canneal are applications having high main-memory bandwidth demand, which consume more DRAM dynamic power than other applications. The performance of these applications, which is closely correlated with their average read latency due to their high cache miss rate, strongly depends on the number of rank subsets per rank. Except for RADIX, the IPC increases until there are 2 or 4 subsets per rank and then decreases. As explained in Section 2, it is



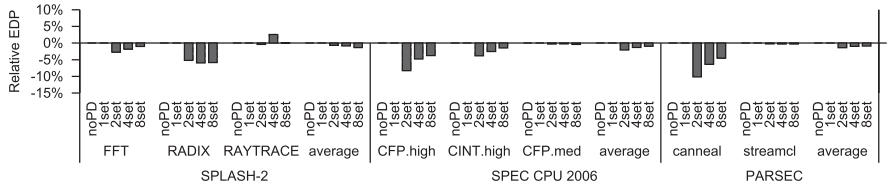
(a) IPC and average read latency. noPD configuration does not utilize DRAM power-down modes, and has 1 subset per rank. nset configurations utilize DRAM power-down modes, and have n subsets per rank.



(b) Memory power breakdown and mean fraction DRAM chips stay in a low-power mode



(c) System power breakdown and energy×delay (lower is better)



(d) Energy×delay comparison between Multicore DIMM and mini-rank

Fig. 9. Memory and system level power and performance on a system with 1 rank per memory channel on 3 benchmark suites. For each suite, applications whose performance is not sensitive to the number of rank subsets are not shown due to space limitations, but they are included when average values are computed.

primarily due to interaction of two factors: access latency and effective bandwidth of memory channels. RADIX, an integer sorting application, takes advantage of higher memory bandwidth so that its performance keeps improving as S increases until there are 8 subsets per rank. In contrast, RAYTRACE doesn't stress the memory channel

bandwidth, but it is very sensitive to access latency, so its IPC drops as a memory rank is split into more subsets.[2]

FFT and CFP.med are applications with medium main memory bandwidth demand. The relationship between the IPC and the number of rank subsets is similar to that of the applications with high bandwidth demand but with smaller variation. However, compared to other applications, the fraction of time that DRAM chips stay in a low-power mode (which is shown in Figure 9(b)) substantially increases as the number of rank subsets increases. Nevertheless, on applications with low bandwidth demand, 1 subset per rank is already enough for the memory controller to put DRAMs in a low-power state most of the time. Conversely, applications with high bandwidth demand rarely leave rank subsets idle, regardless of the number of subsets, so the power-down mode is not often used.

Regardless of memory demands of the application, there are substantial savings in dynamic energy. However, since dynamic power is proportional to the performance (IPC) of an application, this reduction in dynamic power is less apparent when its performance improves. Figure 9(c) shows the system power breakdown and the system energy-delay product of workloads. The energy-delay product improves substantially on applications with high main-memory bandwidth demand, on average 25.7% among four applications when $S = 4$. Across the workloads, the energy-delay product is improved by 4.5%, 0.9%, and 3.8% on SPLASH-2, SPEC CPU 2006, and the PARSEC benchmarks by utilizing a DRAM power-down mode. Rank subsetting brings additional 8.2%, 10.6%, and 3.4% improvement when $S = 4$. It shows that the effectiveness of these two techniques is complementary. The main-memory power always improves as the number of rank subsets increases. However, the system energy-delay product degrades on most applications when S increases from 4 to 8. When memory channels are highly utilized, dynamic power is much larger than static power, and rank-subsetting provides more improvement on system energy-delay product than DRAM power-down modes.

Figure 9(d) compares the energy-delay products of systems using mini-rank with those of systems using Multicore DIMM, where negative values show that Multicore DIMM has lower EDP and positive values show that mini-rank has lower EDP. Mini-ranks dissipate more I/O power than VMDs and are hindered by timing constraints more often since all data are relayed through a shared Mini-Rank Buffer per rank. As a result, Multicore DIMM has lower EDP compared to mini-rank, as explained in Section 2.3. There is a larger difference in EDP between Multicore DIMM and mini-rank on applications with high memory bandwidth demands (maximum 6% on RADIX). In general, however, the difference in EDP of systems using Multicore DIMM and mini-rank is rather small across the workloads.

In summary, when each memory controller has 1 memory rank, 2 or 4 subsets per rank have higher IPC and lower memory access latency on average compared to other configurations. As the number of subsets per rank increases, dynamic power consumption of the memory systems decreases and DRAM chips stay in a low-power mode more frequently. Since the memory system power is substantially lower than the processor power, higher IPC due to rank subsetting affects EDP more than energy-saving by utilizing a DRAM power-down mode. Lower I/O power consumption enables Multicore

[2]The number of threads per core affects the application behavior over the number of subsets as well. When a smaller number of threads are used, more applications behave similar to RAYTRACE since it becomes harder to amortize increases in memory access latency due to rank subsetting. These simulation results are not shown due to page limitations.

DIMM to achieve better (lower) EDP than mini-rank on applications especially with high memory bandwidth demands.

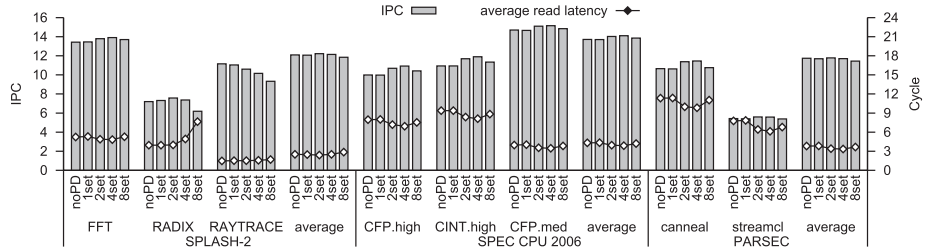
4.3 Four-Rank Performance and Power Efficiency

When more ranks are attached per memory channel so that the main-memory capacity increases, the relationship between application performance, memory power, and system energy-delay product changes in a way that it becomes more important to exploit DRAM power-down modes to achieve better EDP. Figure 10 shows the power and performance of the 3 benchmarks on a system with 2 dual-rank DIMMs per channel (R = 4). The increase in the IPC from 1 to 2 rank subsets on applications with high memory bandwidth demand is not as much as in the previous system configuration. As analyzed in Section 2, with 4 times more independent DRAM banks per channel, the activate-to-activate time constraint becomes a smaller problem as a memory controller can issue commands to other ranks, leading to high performance even without rank subsetting. Still, 2 rank subsets perform better than 1, since the timing constraints on each switch of bus ownership limit performance, and this is alleviated with multiple subsets as each DRAM transaction takes longer.

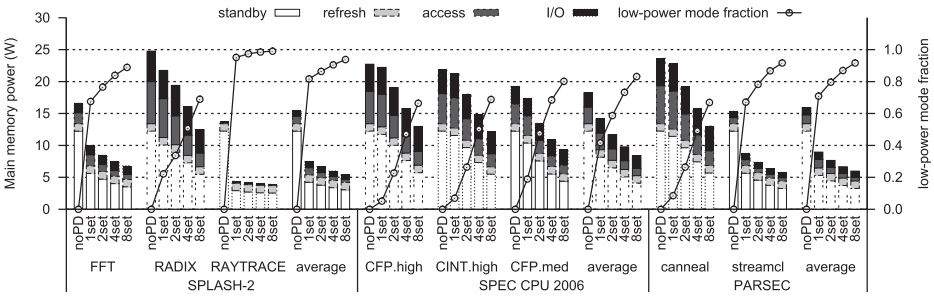
With 4 ranks per channel, static memory power (such as standby and refresh power) and I/O power increase substantially, becoming a significant part of the total memory power as shown in Figure 10(b). Since the peak bandwidth per channel is the same as with 1 rank per channel, banks are idle more often, hence it is more likely that the memory controller can exploit low-power modes. I/O power increases since there are more termination resistors per data bus, and sometimes I/O power even surpasses the access power within DRAM chips, highlighting the need for more energy-efficient technologies such as differential signaling or point-to-point connections. The total memory power becomes comparable to the processor power on applications with high memory demand (Figure 10(c)). However, since the performance of these applications varies less than in the single rank case as the number of rank subsets changes less than before, the energy-delay product improves less as multiple subsets are used: 3.8% on SPLASH-2 with 4, 12.3% on SPEC CPU 2006 with 4, and 1.8% on PARSEC with 2 rank subsets all compared to the configuration utilizing a low-power mode but no rank subsetting. Rather, there are bigger savings by utilizing DRAM low-power modes even without rank subsetting: 16.1% and 13.6% on SPLASH-2 and PARSEC. The SPEC CPU 2006 benchmarks access main memory more often than others, so a 6.9% improvement in energy-delay product from putting DRAMs in a low-power mode is less than the additional improvement due to the rank subsets.

Figure 10(d) shows that the difference in the energy-delay product between Multicore and mini-rank DIMMs diminishes compared to the 1 rank per channel configuration since data bus I/O power increases a lot, meaning that the additional power due to the Mini-Rank Buffer becomes less important. There are configurations that mini-rank outperforms Multicore DIMM due to uneven distributions of memory accesses across subsets as explained in Section 2.3, which is most pronounced in RADIX. With 8 subsets per rank, mini-rank has relatively lower energy-delay product than Multicore DIMM on average in all three workloads. However in this configuration, both have worse EDP compared to the configurations with 2 or 4 subsets, in other words, on the configurations where rank subsetting is beneficial, Multicore DIMM and mini-rank provide similar performance.

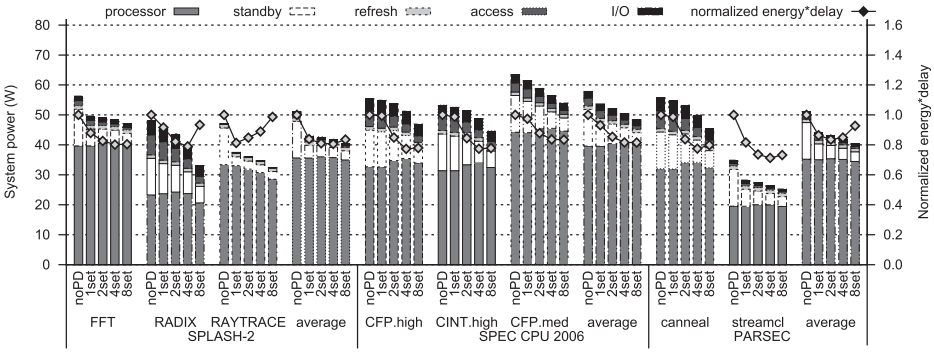
As the number of ranks per memory controller increases from 1 to 4, the number of DRAM banks in the memory system quadruples as well, becoming 128 and surpassing the number of concurrent threads (64) of the CPU. Additional increase in the number of



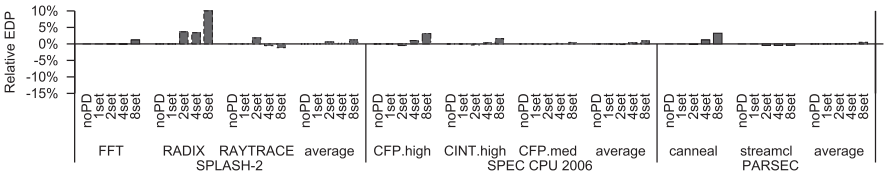
(a) IPC and average read latency. noPD configuration does not utilize a DRAM power-down mode, and has 1 subset per rank. nset configurations utilize a DRAM power-down mode, and have n subsets per rank.



(b) Memory power breakdown and mean fraction DRAM chips in a low-power mode



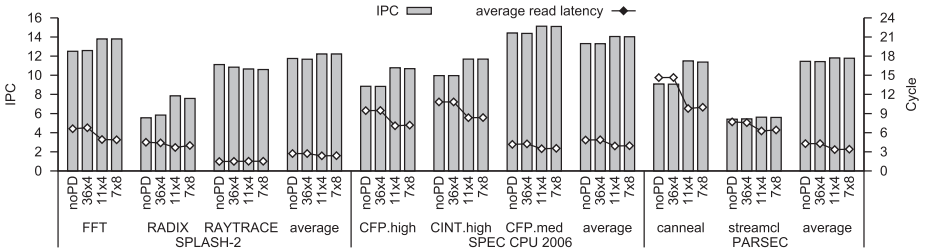
(c) System power breakdown and energy×delay (lower is better)



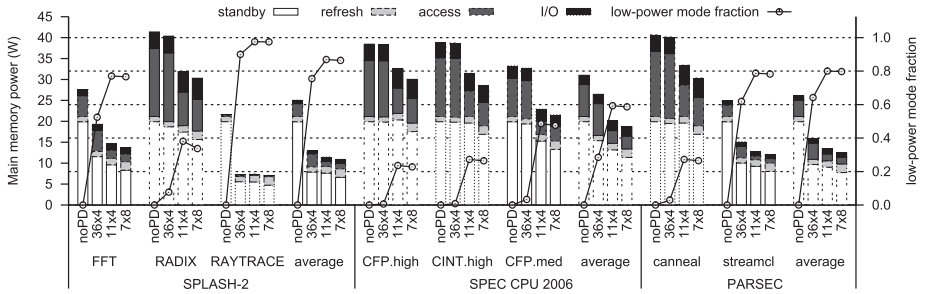
(d) Energy×delay comparison between Multicore DIMM and mini-rank

Fig. 10. Memory and system level power and performance on a system with 4 ranks per memory channel on 3 benchmark suites. For each suite, applications whose performance is not sensitive to the number of rank subsets are not shown due to space limitations, but they are included when average values are computed.

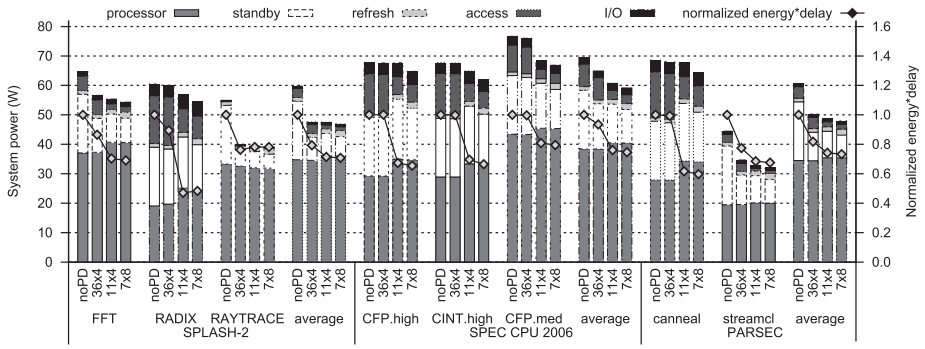
DRAM banks by rank subsetting helps less on improving the IPC. Rather, serialization in cache line transfers due to rank subsetting lowers the IPC when $S = 4$ or 8. As a result, utilizing a DRAM power-down mode provides more impact on EDP compared to rank subsetting in general. Still, the applications with high memory bandwidth



(a) IPC and average read latency. noPD configuration does not utilize DRAM power-down modes, and has 1 subset per rank. nset configurations utilize DRAM power-down modes, and have n subsets per rank.



(b) Memory power breakdown and mean fraction DRAM chips in a low-power mode



(c) System power breakdown and energy×delay (lower is better)

Fig. 11. Power and performance of applications on systems with chipkill-level reliability. There are four configurations per application: both noPD and 36x4 for the conventional system with 36 ×4 4Gb DRAM chips per rank while power-down modes are applied only to 36x4, 11x4 with 11 ×4 4Gb DRAM chips per MCDIMM rank, and 7x8 with 7 ×8 8Gb DRAM chips per MCDIMM rank.

demands exploit rank subsetting effectively. The difference in EDP by the choice of specific rank subsetting implementation decreases as more ranks are deployed.

4.4 Power and Performance of Chipkill-Level Reliability

Both rank subsetting and exploiting DRAM low-power modes equivalently improve the energy-delay products of systems with chipkill-level reliability because supporting chipkill increases the number of DRAM chips to be involved per memory access and to provide the same memory capacity. Figure 11 shows the performance and energy efficiency of 4 different memory systems supporting chipkill level reliability. On each application, the first two columns (noPD and 36x4) have the values for a conventional chipkill solution, in which each rank consists of 36 ×4 4Gb DRAM chips (32 chips for data and 4 for parity and other information). The first column does not utilize the

DRAM low-power mode while the second column does. Since each DRAM chip has a prefetch length of 8, a minimum transfer size $36 \times 8 \times 4 = 1152$bits of data should be read or written. The cache line size of the system is 72B = 576bits including ECC in internal caches, so half of the data are not used. Although burst chopping [Micron Technology Inc. 2006] can be used to save I/O power by not transferring unused data, substantial DRAM dynamic energy is still wasted. In the second memory system, denoted `11x4`, each Multicore DIMM rank consists of 2 subsets, each with 11 $\times$4 4Gb DRAM chips. As a result, 2/9 = 22.2% more DRAM chips are used for error correction over a system that only provides parity or ECC, but only 11 chips, less than 1/3 of DRAM chips compared to the conventional chipkill DIMMs, are used per memory access. The third system, denoted `7x8`, has 2 Multicore DIMM VMDs (subsets) per rank, each with 7 $\times$8 8Gb DRAM chips. Three more DRAM chips are used for error correction, but it needs fewer than 1/5th as many DRAM chips per access compared to `36x4`. All configurations have the same data capacity. The `36x4` configurations have 2 ranks per channel, and the last two have 4 ranks (two dual-rank DIMMs) per channel and 2 subsets per rank.

In traditional chipkill systems, memory power can surpass the processor power, amplifying the importance of improving energy efficiency of processor-memory interfaces. The `36x4` configurations clearly performs worse than others since its effective per-rank bandwidth is lower while `11x4` and `7x8` obtain benefits from having multiple rank subsets; more banks and less frequent timing constraint conflicts, as shown in Section 4.3. There are major savings in DRAM dynamic power on configurations with high-reliability MCDIMMs (Figure 11(b)) since far fewer DRAM chips are used per access. This also helps memory controllers to idle subsets so that the static power of MCDIMMs can be even lower than conventional chipkill DIMMs unless the whole memory system is largely idle like on RAYTRACE and streamcl. This is true even though systems with high-reliability MCDIMMs have more DRAM chips than `36x4`. Therefore both subsetting DRAM ranks and exploiting DRAM low-power modes are equally important to enhancing energy-delay product. By utilizing DRAM low-power modes, system energy delay product on `36x4` is improved by 16.3%. Rank-subsetting leads to additional improvement on system energy delay product: 12.0% and 13.1% on `11x4` and `7x8` compared to the `36x4` with a low-power mode utilized.

5. RELATED WORK

A large body of computer architecture research aims to improve the performance, energy efficiency, and reliability of main-memory systems. One key idea is to group together memory accesses with the same access type and similar addresses by reordering, in order to minimize the performance degradation due to various timing constraints on DRAM accesses. There have been proposals to exploit these characteristics to achieve higher performance on vector [Mathew et al. 2000], stream [Rixner et al. 2000], and single-core and multicore processors [Mutlu and Moscibroda 2007; Nesbit et al. 2006]. Higher performance typically leads to higher energy efficiency by reducing execution time and saving static power. Throughout this article, we use one of the latest proposals called Parallelism-Aware Batch Scheduling [Mutlu and Moscibroda 2008] in the evaluation.

Multiple power states were introduced in Rambus DRAM (RDRAM [Rambus 1999]). Some studies try to exploit these low power states in RDRAMs by allocating and migrating OS pages in order to put DRAM chips into a low power state for longer periods [Huang et al. 2003; Lebeck et al. 2000]. Modern DDR*x* DRAM chips also have multiple low power states. Hur and Lin [2008] suggest ways to exploit them in a memory scheduler. These are complementary to our idea of saving dynamic energy per memory access, which can also be synergistic as shown in Section 4. Ghosh and Lee [2007]

suggest a memory controller design with smart refresh to save refresh power. This is complementary to our idea as well.

Among the proposals advocating rank subsetting, module threading [Ware and Hampel 2006] relies on high speed signaling. The memory controller outputs separate chip select signals for selecting a subset of devices. Multicore DIMM [Ahn et al. 2008] replaces a register per memory rank with a demux register, which routes or demultiplexes address and command signals to the selected subset. Zheng et al. called a subset a mini-rank [Zheng et al. 2008] and proposed a design in which all mini-ranks in a memory rank send and receive data to/from a memory controller through a mini-rank buffer. They did not consider processor power and reliability in their evaluation of their architecture. The key difference between Multicore DIMM and mini-rank is the placement of the data mux and address/command demux. Mini-rank has a demux per memory rank while Multicore DIMM has one per memory channel. As a result, mini-rank is more costly in energy and component count. Multicore DIMM has one address/command demux per memory rank, while mini-rank does not have any. Since address and command signals must be registered per rank due to signal integrity issues, the incremental cost of the demux register is minimal. Both proposals need chip select signals per rank subset.

There are recent proposals to improve the efficiency of main-memory systems either by reducing OS page sizes [Sudan et al. 2010] or by modifying the internal microarchitecture of DRAM chips [Udipi et al. 2010]. Sudan et al. [2010] suggested to collocate cache blocks that are frequently utilized into the same row of the DRAM bank by reducing OS page sizes and providing hardware/software mechanisms for data migration within main memory in order to reduce DRAM row-buffer conflicts. Udipi et al. [2010] proposed changes to the DRAM microarchitecture to alleviate the overfetch problem while sacrificing area efficiency. Their techniques either delay DRAM row activation until both row-level and column-level commands reach the DRAM chip in order to activate DRAM cells that only correspond to the cache line to be accessed, broaden the internal DRAM data path so that an entire cache line can be fetched from a small portion of a bank in a DRAM chip, or provide a checksum logic per cache line to provide RAID-style fault tolerance. Rank subsetting does not require changes to OS or DRAM chip microarchitecture and can be supplementary to these techniques.

6. CONCLUSION

Memory power is becoming an increasingly significant percentage of system power. In this article, we holistically assessed the effectiveness of rank subsetting on the performance, energy efficiency, and reliability at the system level rather than only looking at the impact on individual components. We also quantified the interactions between DRAM power-down modes and rank subsetting. For single-rank and four-rank memory systems, across the SPLASH-2, SPEC CPU 2006, and PARSEC benchmarks, we found that power-down modes without rank subsetting could save an average of 3.5% and 13.1% in system energy-delay product. When rank subsetting is added, additional average savings of 7.7% and 6.6% are obtained. The cost of rank subsetting is very low: for example in the Multicore DIMM approach a latch on each DIMM is converted to a demux latch. Thus given the insignificant investment required, these system energy-delay product savings are remarkably high returns.

We compared Multicore DIMM and mini-rank both qualitatively and quantitatively using a wide range of workloads. This article presented the first results for mini-rank systems where the impact of reduced IPC due to longer memory accesses were taken into account by calculating the increased processor power dissipation.

Rank subsetting increases the amount of data read out of a single chip, so it also can increase the probability of a large number of bit errors when an entire chip fails.

In this article we extended the MCDIMM design for high-reliability systems. Enhancing reliability involves a tradeoff between energy and code rate. Traditional chipkill solutions optimize code rate at the cost of reduced energy efficiency. With the cost of powering datacenters now exceeding their capital cost over their lifetimes, solutions which strike a balance between code rate and energy efficiency make more sense.

In summary, we expect rank subsetting, especially reliability enhanced Multicore DIMM, to be a compelling alternative to existing processor-memory interfaces for future DDR systems due to their superior energy efficiency, tolerance for DRAM timing constraints, similar or better system performance, and ability to provide high reliability.

REFERENCES

AHN, J., EREZ, M., AND DALLY, W. J. 2006. The design space of data-parallel memory systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*.

AHN, J., LEVERICH, J., SCHREIBER, R. S., AND JOUPPI, N. P. 2008. Multicore DIMM: An energy efficient memory module with independently controlled DRAMs. *Comput. Architec. Lett. 7,* 1.

AHN, J., JOUPPI, N. P., KOZYRAKIS, C., LEVERICH, J., AND SCHREIBER, R. S. 2009. Future scaling of processor-memory interfaces. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*.

AMD. 2007. *BIOS and Kernel Developer's Guide for AMD NPT Family 0Fh Processors*. http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/32559.pdf.

BARROSO, L. A. 2005. The price of performance. *Queue 3,* 7, 48–53.

BIENIA, C., KUMAR, S., SINGH, J. P., AND LI, K. 2008. The PARSEC benchmark suite: Characterization and architectural implications. In *Proceedings of the International Conference on Parallel Architectures and Compiler Techniques*.

BOHR, M. 2009. The new era of scaling in an SoC world. In *Proceedings of the International Solid-State Circuits Conference*.

DELL, T. J. 1997. The benefits of chipkill-correct ECC for PC server main memory. White paper, IBM Microelectronics Division.

FRAILONG, J. M., JALBY, W., AND LENFANT, J. 1985. XOR-Schemes: A flexible data organization in parallel memories. In *Proceedings of the International Conference on Parallel Processing*.

GEE, J., HILL, M. D., PNEVMATIKATOS, D. N., AND SMITH, A. J. 1993. Cache performance of the SPEC92 benchmark suite. *IEEE Micro 13*.

GHOSH, M. AND LEE, H.-H. S. 2007. Smart Refresh: An enhanced memory controller design for reducing energy in conventional and 3D DieStacked DRAMs. In *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*.

HENNESSY, J. AND PATTERSON, D. A. 2011. *Computer Architecture: A Quantitative Approach* 5th Ed. Morgan Kaufmann.

HENNING, J. L. 2007. Performance counters and development of SPEC CPU2006. *SIGARCH Comput. Architect. News 35,* 1.

HO, R., MAI, K., AND HOROWITZ, M. A. 2001. The future of wires. *Proc. IEEE 89,* 4.

HUANG, H., PILLAI, P., AND SHIN, K. G. 2003. Design and implementation of power-aware virtual memory. In *Proceedings of the Annual Conference on USENIX Annual Technical Conference*.

HUR, I. AND LIN, C. 2008. A comprehensive approach to DRAM power management. In *Proceedings of the 14th IEEE International Symposium on High Performance Computer Architecture*.

INTEL PRESS. 2009. *The Problem of Power Consumption in Servers*.

JACOB, B., NG, S. W., AND WANG, D. T. 2007. *Memory Systems: Cache, DRAM, Disk*. Morgan Kaufmann.

JEDEC. 2007. DDR3 SDRAM specification. JESD79-3B, http://www.jedec.org/download/ search/JESD79-3B.pdf.

JOHNSON, T. AND NAWATHE, U. 2007. An 8-core, 64-thread, 64-bit power efficient SPARC SoC (Niagara2). In *Proceedings of the International Symposium on Physical Design*.

KANG, U. ET AL. 2009. 8Gb 3D DDR3 DRAM using through-silicon-via technology. In *Proceedings of the International Solid-State Circuits Conference*.

KELTCHER, C., MCGRATH, K., AHMED, A., AND CONWAY, P. 2003. The AMD Opteron processor for multiprocessor servers. *IEEE Micro 23,* 2.

KONGETIRA, P., AINGARAN, K., AND OLUKOTUN, K. 2005. Niagara: A 32-way multithreaded Sparc processor. *IEEE Micro 25*, 2.

LEBECK, A. R., FAN, X., ZENG, H., AND ELLIS, C. 2000. Power aware page allocation. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems*.

LI, S., AHN, J., STRONG, R. D., BROCKMAN, J. B., TULLSEN, D. M., AND JOUPPI, N. P. 2009. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *Proceedings of the 41st Annual IEEE/ACM International Symposium on Microarchitecture*.

LIM, K., RANGANATHAN, P., CHANG, J., PATEL, C., MUDGE, T., AND REINHARDT, S. 2008. Understanding and designing new server architectures for emerging warehouse-computing environments. In *Proceedings of the 35th International Symposium on Computer Architecture*.

LUK, C.-K., COHN, R., MUTH, R., PATIL, H., KLAUSER, A., LOWNEY, G., WALLACE, S., REDDI, V. J., AND HAZELWOOD, K. 2005. Pin: Building customized program analysis tools with dynamic instrumentation. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation*.

MARTIN, M. M. K., SORIN, D. J., BECKMANN, B. M., MARTY, M. R., XU, M., ALAMELDEEN, A. R., MOORE, K. E., HILL, M. D., AND WOOD, D. A. 2005. Multifacet's general execution-driven multiprocessor simulator (GEMS) toolset. *SIGARCH Comput. Architect. News 33,* 4.

MATHEW, B. K., MCKEE, S. A., CARTER, J. B., AND DAVIS, A. 2000. Design of a parallel vector access unit for SDRAM memory systems. In *Proceedings of the 6th IEEE International Symposium on High Performance Computer Architecture*.

MCGHAN, H. 2006. SPEC CPU2006 Benchmark suite. In *Microprocessor Report*.

MICRON TECHNOLOGY INC. 2006. *DDR3 SDRAM Datasheet*. Rev. K 04/10 EN, http://www.micron.com/products/dram/ddr3/.

MICRON TECHNOLOGY INC. 2007. Calculating memory system power for DDR3. Tech. rep. TN-41-01, Micron.

MICRON TECHNOLOGY INC. 2008. RLDRAM datasheet. http://www.micron.com/products/ dram/rldram/.

MUKHERJEE, S. S., EMER, J., AND REINHARDT, S. K. 2005. The soft error problem: An architectural perspective. In *Proceedings of the 11th IEEE International Symposium on High Performance Computer Architecture*.

MUTLU, O. AND MOSCIBRODA, T. 2007. Stall-time fair memory access scheduling for chip multiprocessors. In *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*.

MUTLU, O. AND MOSCIBRODA, T. 2008. Parallelism-aware batch scheduling: Enhancing both performance and fairness of shared DRAM systems. In *Proceedings of the 35th International Symposium on Computer Architecture*.

NESBIT, K. J., AGGARWAL, N., LAUDON, J., AND SMITH, J. E. 2006. Fair queuing memory systems. In *Proceedings of the 38th Annual IEEE/ACM International Symposium on Microarchitecture*.

NOSE, K. AND SAKURAI, T. 2000. Analysis and future trend of short-circuit power. *IEEE Trans. Comput.-Aid. Des. Integ. Circ. Syst. 19,* 9.

PAN, H., ASANOVIĆ, K., COHN, R., AND LUK, C.-K. 2005. Controlling program execution through binary instrumentation. *SIGARCH Comput. Architect. News 33,* 5.

PETERSON, W. W. AND WELDON, E. J. 1972. *Error-Correcting Codes* 2nd Ed. MIT Press.

RAMBUS. 1999. RDRAM, http://www.rambus.com.

RIXNER, S., DALLY, W. J., KAPASI, U. J., MATTSON, P. R., AND OWENS, J. D. 2000. Memory access scheduling. In *Proceedings of the 27th International Symposium on Computer Architecture*.

SARWATE, D. V. AND SHANBHAG, N. R. 2001. High-speed architectures for Reed-Solomon decoders. *IEEE Trans. VLSI Syst. 9,* 5.

SCHROEDER, B., PINHEIRO, E., AND WEBER, W.-D. 2009. DRAM errors in the wild: A large-scale field study. In *Proceedings of the ACM SIGMETRICS Conference*.

SHEN, J. P. AND LIPASTI, M. H. 2005. *Modern Processor Design: Fundamentals of Superscalar Processors*. McGraw Hill.

SHERWOOD, T., PERELMAN, E., HAMERLY, G., AND CALDER, B. 2002. Automatically characterizing large scale program behavior. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems*.

SUDAN, K., CHATTERJEE, N., NELLANS, D., AWASTHI, M., BALASUBRAMONIAN, R., AND DAVIS, A. 2010. Micro-Pages: Increasing DRAM efficiency with locality-aware data placement. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems*.

THOZIYOOR, S., AHN, J., MONCHIERO, M., BROCKMAN, J. B., AND JOUPPI, N. P. 2008a. A comprehensive memory modeling tool and its application to the design and analysis of future memory hierarchies. In *Proceedings of the 35th International Symposium on Computer Architecture*.

THOZIYOOR, S., MURALIMANOHAR, N., AHN, J., AND JOUPPI, N. P. 2008b. Cacti 5.1. Tech. rep. HPL-2008-20, HP Labs.

UDIPI, A. N., MURALIMANOHAR, N., CHATTERJEE, N., BALASUBRAMONIAN, R., DAVIS, A., AND JOUPPI, N. P. 2010. Rethinking DRAM design and organization for energy-constrained multi-cores. In *Proceedings of the 37th International Symposium on Computer Architecture*.

WARE, F. A. AND HAMPEL, C. 2006. Improving power and data efficiency with threaded memory modules. In *Proceedings of the International Conference on Computer Design*.

WOO, S. C., OHARA, M., TORRIE, E., SINGH, J. P., AND GUPTA, A. 1995. The SPLASH-2 programs: Characterization and methodological considerations. In *Proceedings of the 22nd International Symposium on Computer Architecture*.

XAMBO-DESCAMPS, S. 2003. *Block Error-Correcting Codes: A Computational Primer*. Springer.

ZHANG, Z., ZHU, Z., AND ZHANG, X. 2000. A permutation-based page interleaving scheme to reduce row-buffer conflicts and exploit data locality. In *Proceedings of the 32nd Annual IEEE/ACM International Symposium on Microarchitecture*.

ZHENG, H., LIN, J., ZHANG, Z., GORBATOV, E., DAVID, H., AND ZHU, Z. 2008. Mini-Rank: Adaptive DRAM architecture for improving memory power efficiency. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*.

Received July 2010; revised January 2011, June 2011; accepted August 2011