Multicore DIMM: an Energy Efficient Memory Module with Independently Controlled DRAMs

Jung Ho Ahn[†], Jacob Leverich[‡], Robert S. Schreiber[†], Norman P. Jouppi[†]
[†]Hewlett-Packard Labs, [‡]Stanford University

Abstract—Demand for memory capacity and bandwidth keeps increasing rapidly in modern computer systems, and memory power consumption is becoming a considerable portion of the system power budget. However, the current DDR DIMM standard is not well suited to effectively serve CMP memory requests from both a power and performance perspective.
We propose a new memory module called a Multicore DIMM, where DRAM chips are grouped into multiple virtual memory devices, each of which has its own data path and receives separate commands (address and control signals). The Multicore DIMM is designed to improve the energy efficiency of memory systems with small impact on system performance. Dividing each memory modules into 4 virtual memory devices brings a simultaneous 22%, 7.6%, and 18% improvement in memory power, IPC, and system energy-delay product respectively on a set of multithreaded applications and consolidated workloads.

Index Terms—Multicore, memory system, DRAM, memory module.

1 Introduction

DEMAND for memory capacity and bandwidth is continually growing. Recently, the power of main memory systems in servers has equaled or exceeded that of their multi-core and many-core chip multi-processors (CMPs) [4], [11]. Current mainstream memory systems waste much of the dynamic power in memory modules, since too many bits (over 100× of a cache line) are activated per memory access, and most of them are stored back without being used.

In light of these inefficiencies we propose a new memory module, a Multicore DIMM (MCDIMM), that is designed to improve the energy efficiency of memory systems with small impact on performance. In an MCDIMM, DRAM chips are grouped into multiple virtual memory devices, each of which has its own data path and receives distinct commands (address and control signals) through a shared command path. As a result, fewer DRAM chips are involved per memory access and fewer bits are stored back.

The major attraction of our proposal comes from its simplicity. The MCDIMM is a small modification of mainstream, registered dual in-line memory modules (DIMMs). Existing DRAM chips can be used without any modification. Minimal functionality is added to the DIMM registers to provide different commands to each virtual memory device.

2 Modern Main Memory Systems

Main memory systems are built from DRAM chips. A DRAM chip contains billions of bit cells, organized as a number of two-dimensional arrays called banks. Data in a DRAM chip are accessed in 2 steps. First, a memory controller sends a command called ACTIVATE, instructing a specified bank to latch all of the bits in a given row into sense amplifiers. Then one or more column level commands (READ or WRITE) follow, causing data transfers. The number of bits transferred per READ/WRITE is determined by the data path size (typically 4, 8, or 16 bits per chip) and the burst length (typically 8). Once a sequence of read and write operations is over, a command called PRECHARGE is sent to the bank to write the data in the sense amplifiers back to their original locations and to precharge bitlines for the next ACTIVATE.

DRAMs are typically used to build dual in-line memory modules (DIMMs) having a 64-bit data path (72-bit if error correction code (ECC) bits are included). One or more DIMMs are connected to a memory controller through a shared data bus forming a memory channel. Figure 1 shows a conventional memory module which contains 8 DRAM chips, each with an 8 bit data path. Command signals (grey arrows) from a memory controller are broadcast to all DRAM chips in all DIMMs in a channel through an optional "register" per DIMM, which lessens electrical load on the memory controller and allows more pipelining. As shown in Figure 1, all DRAM chips within a DIMM [1] receive the same commands and activate the same row at the same time. As a result, all the DRAM chips within the module act as a single DRAM chip with wider data path and larger rows.

Figure 2 shows a power breakdown of a Micron 1Gb DDR3 DRAM chip [9]. The labels ×8 and ×16 mean the width of a data path and row/col means the ratio between row-level command pairs (ACTIVATE/PRECHARGE) and column-level commands (READ/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 total DRAM power consumption. Thus, it is desirable to decrease the row/col ratio in DRAM chips when a memory controller serves a sequence of memory access requests.

One possible way is to exploit the spatial locality of access requests by making a memory controller omit PRECHARGE and ACTIVATE commands between reads and writes when it detects consecutive requests to the same row. The memory controller can reorder requests to create additional opportunities [12]. The spatial locality of access requests is related to address interleaving across memory controllers, DRAM banks, and DRAM rows, so it can be improved by a careful mapping (for example, aligning OS pages with DRAM rows).

Manuscript submitted: 22-Aug-2008. Manuscript accepted: 18-Sep-2008. Final manuscript received: 25-Sep-2008. We thank the anonymous reviewers for their constructive feedback.

1. In general, a DIMM has one or more ranks, and all the DRAM chips within a rank operate as a unit. In this paper, it is assumed that each DIMM has one rank.

Posted to the IEEE & CSDL on 10/31/2008
DOI 10.1109/L-CA.2008.13 1556-6056/08/$25.00 © 2008 Published by the IEEE Computer Society

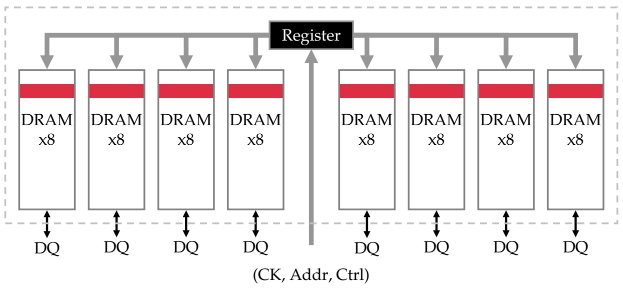

Fig. 1. A conventional DIMM which contains 8 DRAM chips, each with an 8-bit data path. All chips act together as a wide logical chip, so 10s of thousands of bits are loaded into sense amplifiers per ACTIVATE, but only 100s of bits are used per READ or WRITE.

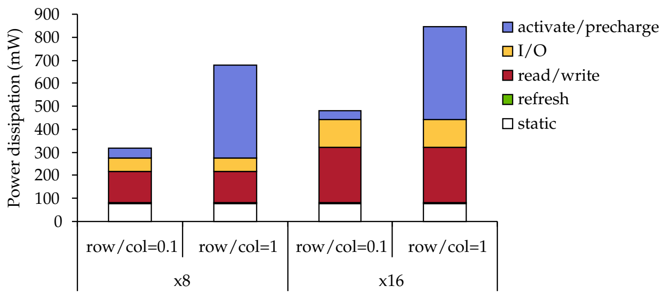

Fig. 2. DRAM power breakdown of a Micron 1Gb DDR3 DRAM chip [9]. row/col means a ratio between row-level commands (ACTIVATE/PRECHARGE) and column-level commands (READ/WRITE).

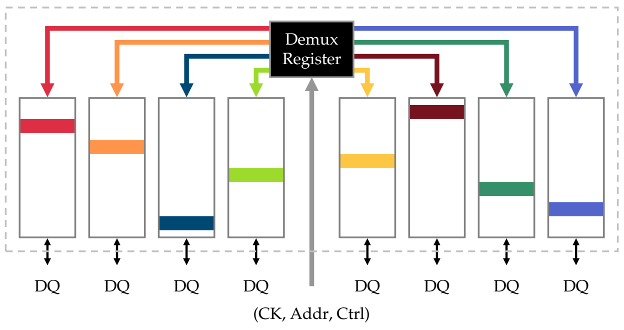

Fig. 3. An operational example of the MCDIMM. Each DRAM chip receives different commands and serves a whole cache line so only 1/8th of the bits are loaded per ACTIVATE compared to Figure 1 leading to far less activation and precharging power but 8× data transfer time.

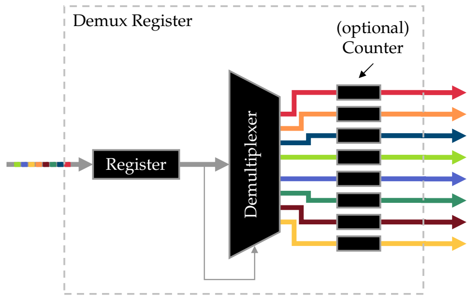

Fig. 4. Demux register microarchitecture.

Another way to save dynamic energy in a memory system is to decrease the number of bits activated per memory access. For most use cases, too many bits are latched per ACTIVATE, wasting dynamic energy. The size of a row of a DRAM bank is typically 8Kb, and each DRAM chip in the memory module activates its corresponding row. So in a DIMM consisting of 8 DRAM chips, 8×8Kb = 64Kb of DRAM cells are activated. This is much larger than a cache line, which is typically 64B = 0.5Kb. So over 99% of the activated DRAM cells are unused for a cache line transaction if row/col = 1. The MCDIMM improves energy efficiency by activating fewer DRAM chips per cache line transaction. This is further described in the following section.

3 Energy Efficient Memory Architecture

Figure 3 shows a Multicore DIMM (MCDIMM). It is similar to the conventional memory module shown in Figure 1, except that the register is replaced with a new device called a demux register. Instead of just repeating/broadcasting command signals, the demux register receives a command signal and routes (demultiplexes) it into the proper DRAM chip. Hence each chip receives different commands through a shared command bus (illustrated by different colors) and transfers data, independently of and concurrently with other chips, with its own data bus. In this configuration, each DRAM communicates to the memory controller over a separate narrow data bus. It shares the bus with corresponding chips in other MCDIMMs.

Only one DRAM chip is involved per memory access, saving energy for activation and precharging. Furthermore, this new memory module effectively has more banks with smaller pages compared to conventional memory modules. More banks typically cause fewer bank conflicts [2] on random access sequences, leading to higher performance. However, it takes more time to deliver the same amount of data since the data bus is narrower. This additional serialization latency can affect system performance negatively if it cannot be hidden.

The microarchitecture of the demux register is shown in Figure 4. A demultiplexer routes an incoming command to the proper destination using chip select signals. A register placed prior to the demultiplexer forwards command signals with good signal quality and timing margin. Optionally, the demux register can translate a single command from the memory controller into multiple commands using counters. For example, if the burst length of the DRAM chip cannot be set long enough to cover an entire cache line, the demux register can translate a column-level command like READ or WRITE into multiple READ/WRITE commands to the DRAM chip. This translation saves command bandwidth from the memory controller. The demux register doesn't dissipate much power, since commands and addresses are delivered with lower data rates, the functionality of the device is simple, and the consumed I/O power will be the same as before, since the command bandwidth has not changed. This is different from an advanced memory buffer (AMB) in a fully-buffered DIMM (FBDIMM) [5] where both data and commands are serialized and daisy chained, consuming much more power. Compared to a conventional DIMM register, the demux register only needs additional pins over a conventional register and additional traces on the DIMM PCB.

The serialization latency problem can be mitigated by grouping multiple DRAM chips into a *virtual memory device*. Figure 5 shows an example configuration where two memory chips are grouped together forming four virtual memory devices in the MCDIMM. For systems requiring ECC, ×9 or ×18 chips (RDRAM [5] and RLDRAM [10] for example) can be used. Some high availability systems use Chipkill [5] to pro-

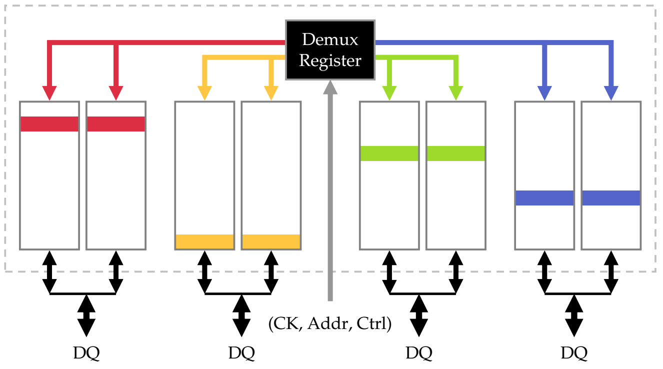

Fig. 5. Multiple DRAMs can be grouped into a *virtual memory device*.

tect against single memory-chip failure. MCDIMM does not support Chipkill, but we expect that other techniques, such as duplication cache [1], can provide similar functionality.

Ware and Hampel [16] suggest module threading for less power per access and higher data bus utilization. While their scheme is similar to ours, they rely on higher-speed signaling and their memory controller outputs separate chip select signals for selecting a subset of devices. In contrast, we do not change the DRAM chip timing but augment the functionality of a DIMM register to hold and route commands to selected virtual memory devices.

4 Experimental Setup

A multicore, multithreaded architecture is used for experiments. There are 8 cores, each running 4 hardware threads simultaneously. Each core has 32KB 8-way set-associative L1 instruction and data caches and a 1MB L2 cache. The main memory subsystem consists of 4 memory channels each with a memory controller and an MCDIMM. Each MCDIMM contains 8 4Gb future DDR3 DRAM chips. There is a crossbar between L2 caches and memory controllers. The memory controller processes requests in order and keeps a row open until there is a request accessing a different row in the same bank.

32nm process technology is assumed for both processor and memory. The power and area of the 90nm Niagara [7] core is scaled as in [15] and used for processor power. CPU clock frequency is fixed at 2GHz. CACTI 5 [15] is used for modeling the timing, area, and power of the caches and the main memory chips. The power consumption of CPU (except the caches) is estimated to be 22.3W, with half of this static (including leakage) and half varying in proportion to IPC. The page size of the DRAM chips is 8Kb. The data transfer rate of a 64-bit memory module is 2Gbps, so it takes 8 CPU cycles to transfer a 64B cache line. Command and address signals are transferred at half the speed of the CPU clock and each memory command (both column level and row level commands) takes 2 CPU cycles.

We use the NAS Parallel Benchmarks (NPB) [6], SPEC CINT2006 [8], and SPECjbb2000 [14] applications. We skip the initialization phase of multithreaded benchmarks (NPB and SPECjbb2000). For the SPEC CINT2006 benchmarks, we used the SimPoint tool [13] to find simulation phases (320 million instructions per phase) and weights. We did timing simulation for 10 billion instructions. The HP Labs COTSon simulator [3], based on AMD's SimNow infrastructure, is used for performance evaluation with timing simulation modified for our target system.

5 Evaluation

Figure 6(a) shows the instructions per cycle (IPC) and the average read latency for 8 NPB applications, 3 SPEC CINT2006 mixtures, and the SPECjbb benchmark on 4 different memory system configurations. The NPB and SPECjbb applications are configured to have 32 threads while 12 single threaded SPEC CINT2006 benchmarks are consolidated into 3 mixtures: 8 instances of 429.mcf, 462.libquantum, 471.omnetpp, and 473.astar for high memory demand (CINT.high), 8 of 403.gcc, 445.gobmk, 464.h264ref, and 483.xalancbmk for medium memory demand (CINT.med), and 8 of 400.perlbench, 401.bzip2, 456.hmmer, and 458.sjeng for low memory demand (CINT.low). The number of instances per phase of each CINT2006 benchmark is proportional to its weight. Figure 6(b) shows the memory power breakdown while Figure 6(c) shows the system power breakdown and the energy-delay product. The geometric mean of the 12 applications is shown on the rightmost part of each figure. N in Nxic stands for the number of virtual memory devices per MCDIMM (1xic is equivalent to a conventional DIMM). The time it takes to transfer a cache line is proportional to N. So it takes 64 CPU cycles to transfer a cache line from a virtual memory device when $N = 8$. The energy-delay product is normalized to the configuration with $N = 1$ per application. Because each thread is executed in order, the IPC and the average read latency are highly correlated. We can explain the performance and the power dissipation trends of the applications on different configurations by combining their memory access and working set characteristics. Factors affecting memory system performance while the number of virtual memory devices is varied are explained in Section 3.

On most of the NPB applications and CINT.high, performance improves when N changes from 1 to 2 and again when N changes from 2 to 4. This is because the increased number of virtual memory devices provides more banks, which yields better performance despite the latency penalty due to increased serialization latency. When only a single DRAM chip serves a cache line, the serialization latency is not compensated by the larger number of banks, giving lower performance. On ft.B and is.C in particular, the dynamic power used to perform memory reads and writes is relatively high, because memory is more frequently accessed than in the other applications. The performance of ua.C, SPECjbb, CINT.med, and CINT.low is mostly insensitive to N since their cache hit rates are so high that memory accesses are infrequent. Figure 6(b) shows that on average, even though the dynamic power saving is 51% when N is 4, the total memory power saving is limited to 22%, since the standby power is a significant part of the total memory power. Here, it is assumed that DRAM chips stay in an active standby mode [9] when they are idle. DRAM power-down modes can be used to save standby or static power with minimal performance penalty when memory demands are low.

In Figure 6(c) it can be seen that increasing the number of virtual memory devices per MCDIMM decreases the system power in general. On average, the configuration with 2 DRAM chips per virtual memory device ($N = 4$) saves 22% of the main memory power compared to the baseline configuration. Moreover, the energy-delay product is improved significantly when N is either 2 or 4, 15% and 18% better than conventional DIMMs. A maximum improvement of 55% in the energy-delay product is observed on is.C when there are 4 virtual memory devices per module. When N is 8, although the system power is further improved, the energy-delay product is higher than

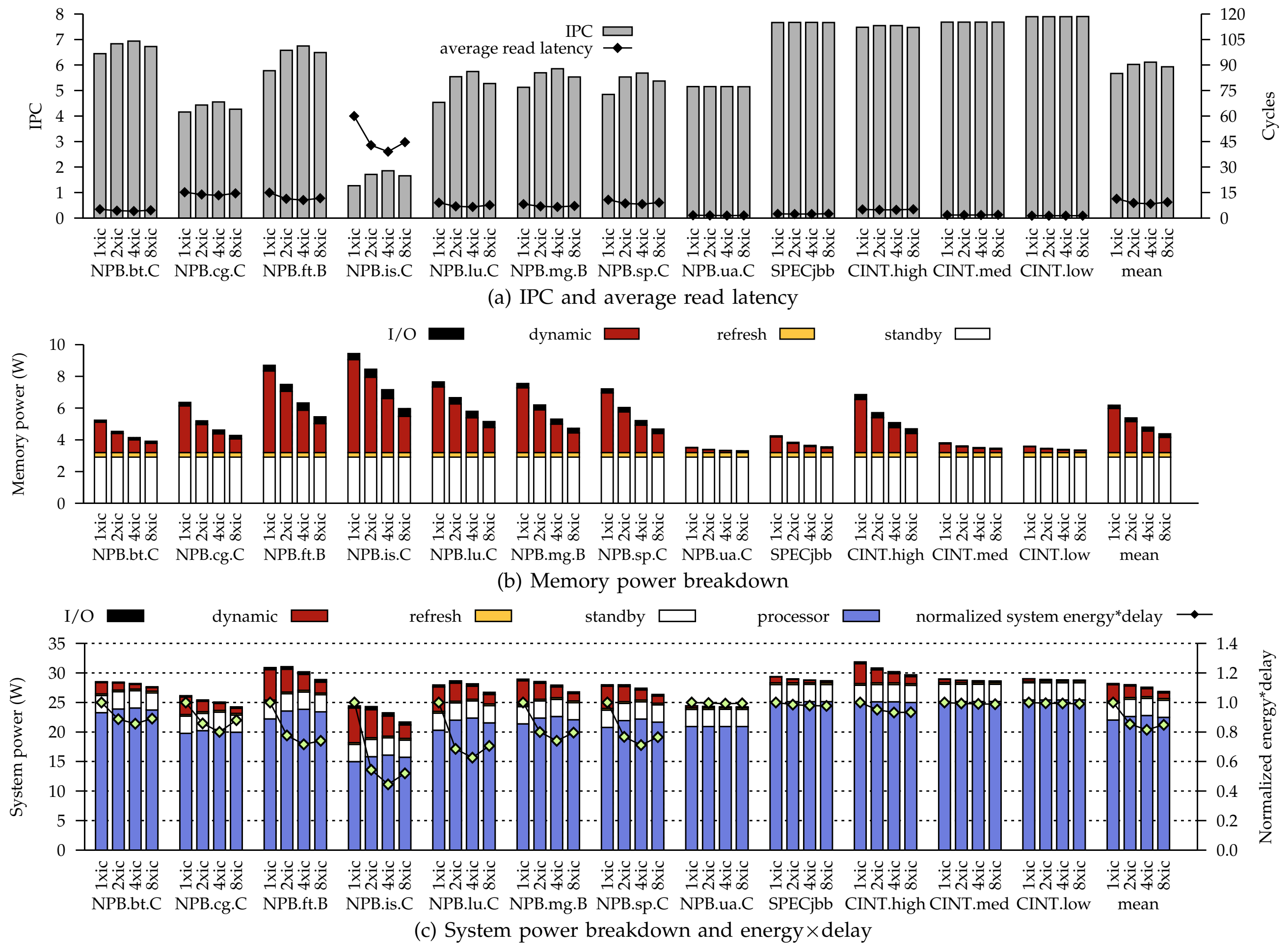

Fig. 6. (a) IPC and average read latency, (b) memory power breakdown, and (c) system power breakdown and energy-delay product of 8 NPB applications, 3 SPEC CINT2006 mixtures, the SPECjbb benchmark, and their geometric mean values. N in Nxic stands for the number of virtual memory devices per MCDIMM. 1xic is equivalent to a conventional DIMM.

when N is 4 since the performance is degraded more than the power saving.

6 Conclusion

A Multicore DIMM provides several advantages by controlling each DRAM device independently in a memory module. First, fewer bits are latched in the sense amplifiers on row activation, leading to lower energy per access, especially when the number of data accesses per row activation is small. Second, existing DRAM chips can be used without any modification, leading to less cost premium. Third, an increased number of virtual memory devices provides more banks, which yields better performance until the increased serialization latency negates the benefits of more banks. Given all these advantages and the relatively low cost of adoption, we believe Muliticore DIMMs provide compelling value.

References

[1] N. Aggarwal, J. E. Smith, K. K. Saluja, N. P. Jouppi, and P. Ranganathan, “Implementing High Availability Memory with a Duplication Cache,” in *Micro*, Nov 2008.
[2] J. Ahn, M. Erez, and W. J. Dally, “The Design Space of Data-Parallel Memory Systems,” in *SC*, Nov 2006.
[3] A. Falcon, P. Faraboschi, and D. Ortega, “Combining Simulation and Virtualization through Dynamic Sampling,” in *ISPASS*, Apr 2007.
[4] HP Server Power Calculators, “http://h30099.www3.hp.com/configurator/powercalcs.asp.”
[5] B. Jacob, S. W. Ng, and D. T. Wang, *Memory Systems: Cache, DRAM, Disk*. Morgan Kaufmann, 2007.
[6] H. Jin, M. Frumkin, and J. Yan, “The OpenMP Implementation of NAS Parallel Benchmarks and Its Performance,” NASA Ames Research Center,” NAS-99-011, 1999.
[7] P. Kongetira, K. Aingaran, and K. Olukotun, “Niagara: A 32-Way Multithreaded Sparc Processor,” *IEEE Micro*, vol. 25, no. 2, 2005.
[8] H. McGhan, “SPEC CPU2006 Benchmark Suite,” in *Microprocessor Report*, Oct 2006.
[9] Micron Technology Inc., *DDR3 SDRAM Datasheet*, 2008, http://www.micron.com/ products/dram/ddr3/.
[10] ——, *RLDRAM Datasheet*, 2008, http://www.micron.com/products/dram/rldram/.
[11] U. Nawathe, *et al.*, “An 8-Core 64-Thread 64b Power-Efficient SPARC SoC,” in *ISSCC*, Feb 2007.
[12] S. Rixner, W. J. Dally, U. J. Kapasi, P. R. Mattson, and J. D. Owens, “Memory Access Scheduling,” in *ISCA*, Jun 2000.
[13] T. Sherwood, E. Perelman, G. Hamerly, and B. Calder, “Automatically Characterizing Large Scale Program Behavior,” in *ASPLOS*, Oct 2002.
[14] SPEC, “Spec jbb2000. http://www.spec.org/osg/jbb2000/.”
[15] S. Thoziyoor, J. Ahn, M. Monchiero, J. B. Brockman, and N. P. Jouppi, “A Comprehensive Memory Modeling Tool and its Application to the Design and Analysis of Future Memory Hierarchies,” in *ISCA*, Jun 2008.
[16] F. A. Ware and C. Hampel, “Improving Power and Data Efficiency with Threaded Memory Modules,” in *ICCD*, Oct 2006.