Mini-Rank: Adaptive DRAM Architecture for Improving Memory Power Efficiency

Hongzhong Zheng[*], Jiang Lin[†], Zhao Zhang[†], Eugene Gorbatov[‡], Howard David[‡] and Zhichun Zhu[*]

[*]*Department of Electrical and Computer Engineering University of Illinois at Chicago hzheng@ece.uic.edu zhu@ece.uic.edu*

[†]*Department of Electrical and Computer Engineering Iowa State University linj@iastate.edu zzhang@iastate.edu*

[‡]*Corporate Technology Group Intel Corp. Hillsboro, OR 97124 eugene.gorbatov@intel.com howard.david@intel.com*

Abstract

The widespread use of multicore processors has dramatically increased the demand on high memory bandwidth and large memory capacity. As DRAM subsystem designs stretch to meet the demand, memory power consumption is now approaching that of processors. However, the conventional DRAM architecture prevents any meaningful power and performance trade-offs for memory-intensive workloads. We propose a novel idea called mini-rank *for DDR*x *(DDR/DDR2/DDR3) DRAMs, which uses a small bridge chip on each DRAM DIMM to break a conventional DRAM rank into multiple smaller mini-ranks so as to reduce the number of devices involved in a single memory access. The design dramatically reduces the memory power consumption with only a slight increase on the memory idle latency. It does not change the DDR*x *bus protocol and its configuration can be adapted for the best performance-power trade-offs. Our experimental results using four-core multiprogramming workloads show that using x32 mini-ranks reduces memory power by 27.0% with 2.8% performance penalty and using x16 mini-ranks reduces memory power by 44.1% with 7.4% performance penalty on average for memory-intensive workloads, respectively.*

1. Introduction

The widespread use of multi-core processors and rapid increase of applications' memory footprints have dramatically increased the demand on memory bandwidth and capacity. Conventional memory designs have been stretched, and new memory designs have been proposed, to meet the demand. However, as a side effect, the memory power consumption has approached to that of processors. For example, a recent study [15] reported that the memory subsystem in a multi-core server box, which has 32 GB Fully-Buffered DIMM and 24 GB/s memory bandwidth, may consume as much as 100 Watts. Conventional DDR*x* (DDR/DDR2/DDR3) memories generally consume less power than the FB-DIMM. However, their power consumption is still significant in the overall system power profile.

This study is focused on reducing the power consumption of DDR*x* memories, which are widely used in today's computing systems. The DDR*x* memory design evolves from Synchronous DRAM (SDRAM), which uses a 64-bit memory bus for data transfer. SDRAM started with bus of frequency as slow as 66MHz, which could transfer a 64-byte cache block in eight bus cycles or 125ns. The total memory idle latency was in the range of 200ns that included precharge, row access and column access times and memory controller overhead. Nowadays, with the demand for high memory bandwidth from multi-core processors, DDR*x* bus frequency has increased beyond 400MHz besides the use of double data transfer. A DDR3-1066 memory bus, for example, runs at 533MHz and may transfer a 64-byte cache block in 7.5ns. The total memory idle latency is typically in the range of 45-70ns, as the other components have not been improved as fast as the data transfer time.

The continued use of 64-bit datapath, carried from the previous generations, is now a limit for DDR*x* memory subsystems to allow meaningful trade-offs between performance and power. A typical memory device may provide 4-bit (x4), 8-bit (x8) or 16-bit (x16) data output, where x8 devices are commonly used today. Multiple devices are needed to form a 64-bit datapath, e.g. by eight x8 devices. In DRAM's terminology, such a group of memory devices is called a *memory rank*, while a DIMM may consist of more than one ranks of devices. We call it *x64 rank* in the rest of the paper. The use of x64 rank is good for reducing the data transfer time and thus memory idle latency. However, a relatively large

number of devices must operate in tandem, consuming power with little gain nowadays given the fast increase of DDR*x* bus frequency. For multicore systems, increasing memory concurrency is more important than reducing memory idle latency, and cache block size should not be increased due to memory bandwidth contention.

We propose a new and novel design called *mini-rank* to address this issue. It uses a bridge chip called MRB (Mini-Rank Buffer) on each DIMM, between the DRAM devices and the DDR*x* bus, to break x64 ranks into x32, x16 or even smaller ranks. We have found that the power saving comes from two major sources. The DRAM operation power is reduced dramatically since fewer chips are involved for each access. Furthermore, the background power is reduced, because having more mini-ranks than x64 ranks can better utilize "fast" low power modes available on contemporary memory devices. Additionally, termination power may also be reduced depending on memory configuration.

The design has several merits crucial for implementation. It does not require any changes to the DRAM devices; such changes would incur tremendous costs to DRAM manufacturing. It is compatible with the existing DDR*x* bus design (unlike FB-DIMM), avoiding any bus re-design issues and cost. Furthermore, the mini-rank design can be adaptive, i.e. adapting the data width for the best performance-power trade-offs; and the MRBs can be bypassed to make the DIMMs compatible with existing memory controllers or existing processors with built-in memory controllers.

The mini-rank structure increases the memory idle latency slightly. For example, with DDR3-1066 memories, using x32 mini-ranks may increase the time of transferring a 64-byte cache block from 7.5ns to 16.9ns. Nevertheless, in multi-core systems with high memory concurrency, memory queuing delay is usually longer than memory idle latency and therefore the overall performance is only slightly affected. Additionally, the mini-rank design can better support memory concurrency, which offsets the performance penalty and may even improve performance for some applications.

The bridge chip, MRB, introduces additional power that has to be accounted for. We have done a preliminary design and power estimation using an I/O model based on standard On Die Termination (ODT) mechanism and Synopsys Design Compiler. We have developed a performance and power simulator for DDR*x* memories and integrated it into the M5 simulator [1]. Using multiprogramming workloads constructed from SPEC CPU2000 programs, our experiments show that significant power/energy saving can be achieved with both DRAM open and close page modes. For x8 device configurations with close page mode and memory-intensive workloads, the average memory power reduction is 27.0% and 44.1% with x32 and x16 mini-ranks, respectively. The corresponding average performance penalty is only 2.8% and 7.4%, respectively. For non-memory-intensive workloads,

Power State	Power (mW)	Comments
Background		
ACT_STBY	964	Any bank open or with operations
PRE_STBY	836	All banks precharged and w/o any operation
ACT_PDN	578	7.5ns exit latency; any bank active and CKE disabled
PRE_PDN_FAST	321	7.5ns exit latency; all banks precharged, CKE disabled and fast mode
PRE_PDN_SLOW	128	11.25ns exit latency; all banks precharged, CKE disabled and slow mode
SREF_FAST	77	938ns exit latency; CKE disabled and all other signals/inputs are floating
SREF_SLOW	38	6938ns exit latency; CKE disabled and all other signals/inputs are floating; registers closed
Operation		
ACT_PRE	514	Active and precharge operation (1st bank 643mW)
Read/Write		
RD	1864	Read burst out
WR	2121	Write burst in
I/O		
RD_OUTPUT	477	Driving read output
WR_TERM	554	Terminating write output
RD_TERM_OTH	1179	Terminating read/write output of other DIMMs within the channel;
WR_TERM_OTH	1306	available for multi-DIMMs per channel system.

Table 1. The power mode of 1GB/rank DDR3-1066 used in our experiments. The I/O power is dependent on system design and termination mechanism; and the WR_TERM power is counted as part of memory controller power.

the average power reduction is 9.2% and 13.5% power, respectively; and the average performance penalty is 1% and 2.3%, respectively.

2. DRAM Power Model

A memory access involves a combination of up to three DRAM operations, namely precharge, activation (row access), and data read/write (column access), of multiple DRAM devices (chips) [23], [28]. The scheduling of those operations is called memory access scheduling. In the close page mode, the common sequence is activation, data read/write, and then precharge for the next access. In the open page mode, the common sequence for a row buffer miss is precharge, activation, and data read/write; and only data read/write is needed for a row buffer hit.

Table 1 summarizes a representative DRAM power model used in this study, whose parameters are calculated based on the voltage and current values from a DDR3-1066 data sheet [18], [19], [21] (details will be presented in Section 4). The power consumption of a device is classified into four
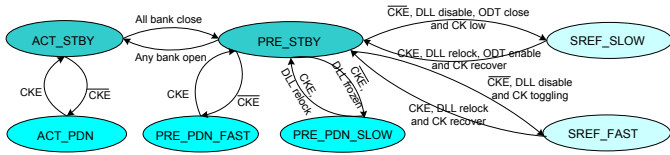
Figure 1. Background power state transition (ODT: On Die Termination [19]).

categories: background, operation, read/write and I/O [21]. The operation power is for a device to perform activation and precharge, the read/write power is for data read and write, and the I/O power is for driving the data bus and terminating data. Data termination is needed at the ranks other than the one being read or written. The background power will be discussed below. All devices in a rank operate in tandem.

The background power is the power that a DRAM chip consumes all the time with or without operations. Today's DRAMs support multiple low power modes to reduce the background power. There are three power levels, standby, power-down and self-refresh, and seven power modes regarding the background power of DDR3 as shown in Figure 1. The control signal CKE (ClocK Enable) is to assert or de-assert the clock signal. The two standby states, *active standby* and *precharge standby*, have high background power and make the rank ready for accesses. The three power-down modes, *active power-down*, *precharge power-down fast* and *precharge power-down slow*, have a lower power rate but with a moderate exit latency. The slow mode has DLL (Delay lock loop) frozen, consuming 60% less power than the fast mode with a slight increase of the exit latency. The self-refresh fast/slow modes, with CKE disabled and self-refresh enabled, have the lowest power rate but very long exit latency. The two modes are not used in this study.

3. Mini-Rank Design and Implementation

3.1. Design and Implementation

Overview. In general, the memory subsystem consists of a set of memory channels and DIMMs (Dual In-line Memory Modules). Each DIMM consists of multiple DRAM devices (chips). A DRAM device has a small number of data I/O pins, usually 4 or 8 or 16, for the reasons of package and cost. The most commonly used devices today are x8 devices. To form the 64-bit datapath in DDR*x* (72-bit with ECC), eight x8 devices (nine with ECC) are ganged together to form an x64 rank, and all devices in the rank operate simultaneously.

To break an x64 rank into multiple smaller ranks (x32, x16 or x8), our proposed *mini-rank* design adds a bridge chip, called *mini-rank buffer* (MRB), to each DIMM. With x8 devices, a x32 mini-rank has only four devices and a x16 mini-rank only two devices. The same principle also

applies to x4 and x16 devices. Figure 2 shows the mini-rank design and the structure of mini-rank buffer for x8 devices. The DRAM devices accept memory commands and most of control signals directly from the DDR*x* bus, while all data connections go through the MRB. The MRB has two major functions: (1) to relay data transfer between DDR*x* bus and the DRAM devices and (2) to produce chip select signals to the devices. It connects to the DDR*x* bus and has private data connection with each device.

As shown in the right part of Figure 2, the MRB has six components: data interface with DRAM chips and DDR*x* bus, DDR*x* control interface, buffer entries for read/write data, command/address re-decode logic and delayed loop lock (DLL). The read and write data entries are separated for concurrent accesses and further possible scheduling. All control signal decoding and data burst in/out can be pipelined. The DLL is used to reduce the clock skew. Although Figure 2 shows two x32 mini-ranks, the MRB can support dynamic configuration of the mini-rank architecture; for example, as four x16 mini-ranks. The configuration, memorized in config-uration registers (not shown), can decide to which devices a read or write command will be routed, and how many cycles it takes to burst the data from/to the devices. Additionally, a simple extension of the MRB can allow the bypass from MRB chip (with one cycle penalty).

Memory Access Scheduling. The use of mini-rank changes two parameters for memory access scheduling, the interval between the read/write command and the data burst and the number of data burst cycles. Figure 3 illustrates the difference of scheduling a single 64-byte read request with and without using x32 mini-ranks, assuming the accessed bank is in the precharge state. The scheduling for write accesses is similar. In the example, t_{RCD} and t_{CL} are multiple cycles not shown fully in the figure. With conventional DDR*x* memories, the data are burst for four bus cycles (the 4th row), which is 7.5ns if the memory is DDR3-1066. With x32 mini-rank, the data are first burst into the MRB for eight bus cycles (the 5th row), and then onto the x64 bus for four cycles (the 6th row) in a pipelined manner. There is one extra cycle latency for the MRB to buffer the last trunk of data, therefore the latency is increased by five bus cycles over the conventional DDR*x* memories. The command and address wires are directly connected to each device. Therefore, there is no extra latency for buffering the command/address signals.

3.2. Modeling of MRB Power Consumption

Modeling Methodology. We first model MRB using Verilog and then break its power consumption over three portions of the MRB: (1) I/O interface with DRAM chips and DDRx bus, including data, control and address pins; (2) DLL (delayed loop lock); and (3) non-I/O logics, including SRAM data entries and re-decode logic. We estimated the
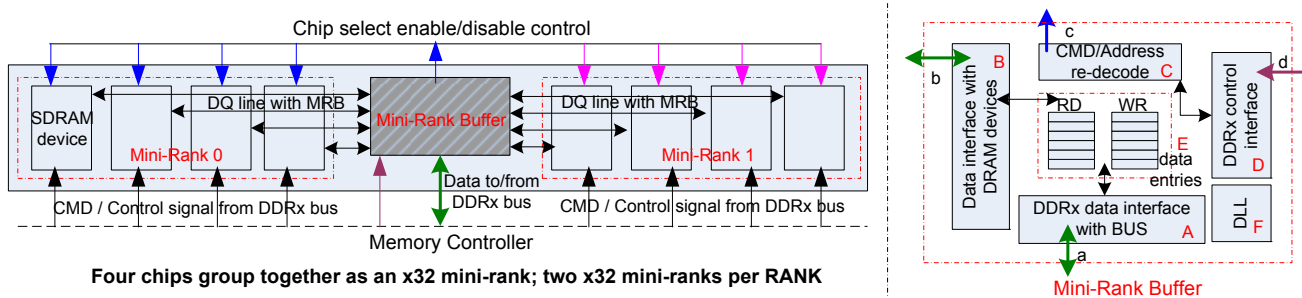
Figure 2. The Mini-rank design with x8 device. In the right part: (a) data to/from DDRx bus, (b) data to/from DRAM device, (c) chip select enable/disable control, and (d) Control signal from DDRx bus.
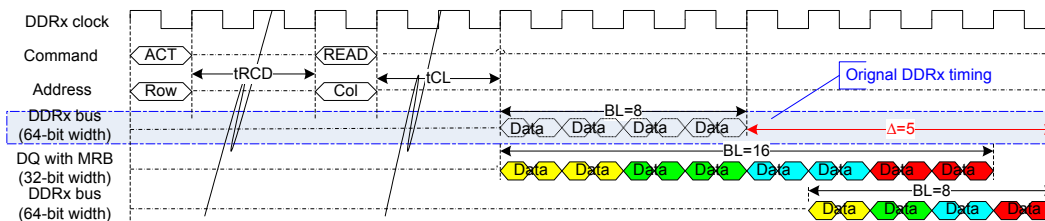

Figure 3. x32 mini-rank timing.

I/O power by calculating the DC power using Thevenin equivalent circuit. The DLL power is derived from the power difference of DRAM device in two states with DLL enabled and disabled. The non-I/O logic power is estimated by the Synopsys Design Compiler [26], which takes our Verilog code as input. It includes both static power and dynamic power and assumes average transistor activity level. Our simulator provides the activity ratio of the MRB chips, which is factored into the calculation of the actual I/O power and non-I/O logic power.

Modeling Details. We model the I/O power using the standard ODT (On Die Termination) of DDR3 technology [21]. Figure 4-(a) shows the structure of typical ODT mechanism for one DQ (data bitline) between MRB and DRAMs. The output drivers at either the MRB or a device have an impedance of R_{ON} that pulls the DQ towards V_{DDQ} for a "1" and V_{SSQ} for a "0". The termination on the die is functionally a pull-up resistor R_{TTPU} and a pull-down resistor R_{TTPD} where $R_{TTPU} = R_{TTPD} = 2 \times R_{TT}$. R_{TT} is the Thevenin equivalent termination value selected for the device. According to DDR3 data sheet [19], for point-to-point systems, all output drivers are set to 34Ω (R_{ON}) and all terminations are set to 120Ω (R_{TT}). The read/write DC power of the output driver and the termination is calculated based on Thevenin equivalent circuit showed in Figure 4-(b) and (c). With mini-rank, the MRB consumes the termination power but no such power is needed for the devices because of their private connections to the MRB. R_{Z1}/R_{Z2} (34Ω) is the output drive impedance for MRB or devices, and R_{S1} (15Ω) is the impedance to reduce transmission line reflection. The

value shown in the table of Figure 4 is the peak power and has to be de-rated with read/write utilization, which is from the simulation, to calculate the overall power. Finally, the non-I/O logic power is from the synthesis result by Synopsys Design Compiler using $130nm$ standard cell library.

Figure 4-(d) shows an example of the MRB power breakdown with memory sub-system configuration of four channels, two DIMMs per channel, two ranks per DIMM and two mini-ranks per rank (x8 devices) with 50% read and 30% write channel bandwidth utilization (emulating memory intensive workloads). The I/O power, non-I/O logic power, and DLL power make up 85.9%, 8.7% and 5.4% of the MRB power, respectively; the total is 936mW. Not surprisingly, the I/O power dominates in the MRB power. Nevertheless, the I/O power between the MRB and the DDRx bus, which is 58.1% of whole MRB power, is also needed in conventional DDRx memories. The actual power increase is 394mW per DIMM for the given access frequency.

3.3. ECC Support in the Mini-rank Architecture

With the mini-rank architecture, there is a large design space to support ECC. For simplicity, we only briefly discuss the design options and leave the issue for future study.

Error Correcting Code (ECC) is a standard reliability feature commonly used in memory structures. DIMMs with ECC supports single-bit error correction and double-bit error detection. There are 8 ECC bits for every 64 data bits stored in a separate and extra chip on each DIMM. The ECC bits are transferred to the memory controller in parallel with the 64 data bits through the DDRx bus. With x4 devices, each
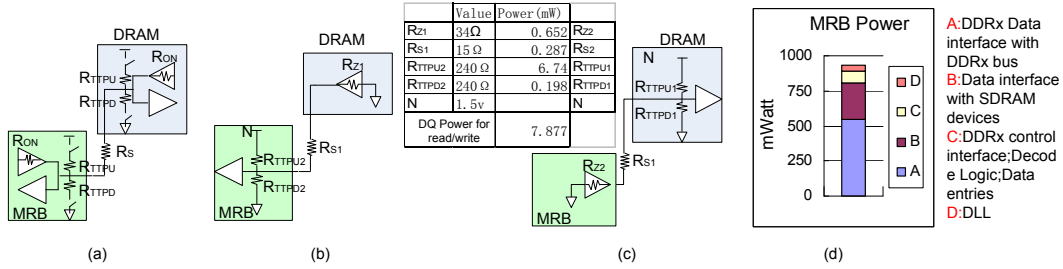
Figure 4. Mini-rank Buffer power consumption: (a) ODT of MRB and DRAMs; (b) DRAM read; (c) DRAM write; and (d) MRB power breakdown (assuming 50% read, 30% write).

rank consists of sixteen devices for data and two for ECC; with x8 devices, there are eight devices for data and one for ECC. With x16 devices, the ECC support is tricky because with an extra device for ECC, only half of its capacity may be utilized.

For ECC DIMMs and mini-rank, it is undesirable to use devices dedicated to ECC storage. If so, such a device will be shared by at least two mini-ranks and therefore becomes a bottleneck of accesses. A natural design is to distribute ECC bit blocks over all devices as in RAID-5. In this case, it is no longer the case that a rank consists of a fixed set of devices. Each access will touch one more device than the simple mini-rank design. For example, with x8 devices and x32 mini-rank configuration, each access will touch five devices instead of four. Therefore, there will be a moderate decrease of power saving from mini-rank without ECC support.

Another approach is to store ECC bits along with their associated data bits in the same page, which we call *embedded ECC*. It requires no change of DRAM devices, but affects the memory data layout and therefore changes the address translation in memory controller. For example, consider a x8 device with 8K bits per page. Without ECC, each page stores 128 64-bit data blocks. With embedded ECC, each page will store 113 blocks of 72 bits (64 data bits and 8-bit ECC bits) with 56 unused bits (0.7% of the page size). Address translation, i.e. breaking a physical memory address into device address including channel, DIMM, rank and bank indices and page and column addresses, must be done carefully. In the conventional address translation, the page size is a power of two so a simple split of the memory address is sufficient. With embedded ECC, an division would be needed in the translation if not done carefully, which is not desired.

There was a classic problem in memory system design: given n memory modules of m words each, where n is not a power of two (so chosen to reduce module conflicts), how to break a memory address into module and word addresses without using a division, and meanwhile avoid or minimize unused holes in the memory data layout. In the mathematical form, the problem becomes mapping a given integer x (the address) into a pair of integers (u, v). There were a number of solutions. One solution appeared in prime memory system [8], where n is chosen to be a prime number. It was proven that the address mapping can be done using $u = x \mod m$ and $v = x \mod p$, where the mod operations can be done using parallel "cycle adders". In fact, it works as long as p and m are coprime, i.e. their greatest common divisor is one. In our problem, n and m have different meanings: assuming cache line interleaving, n is the number of pages in the system and is a power of two, while m is the number of cache lines in a page and is not necessarily a power of two[1]. Therefore, as long as m is chosen to be an odd number, which must be a coprime of n, the address translation from x to (u, v) can be done using the above method; u is further split into device address fields. An 8Kb page has 113 72-bit blocks, so all blocks useful. An 4Kb can hold 56 such blocks, so one block will be left unused.

Although embedded ECC complicates memory address translation and may have other implications, it has unique merits. First, there will be no need to manufacture two types of DIMMs. The same DIMM can be used with or without ECC given proper memory controller support. Second, it allows other forms of error detection or correction, not necessarily having 8 redundant bits per 64 data bits, to help improve system reliability. The checking function can be implemented in the MRBs to avoid extra traffic on memory bus. Third, it is an efficient solution to supporting ECC with x16 devices, which might gain popularity in the future. Finally, the ratio of power saving from using mini-rank will increase slightly because an access to conventional ECC memories involves an extra device and therefore consumes more power than non-ECC memories.

4. Experimental Setup

4.1. Simulation Environment

We use M5 [1] as the base architectural simulator and extend its memory part to simulate DDR*x* DRAM systems in details. The simulator keeps tracking the states of each

---

1. If page interleaving is used, n is the number of banks in the system and m the number of pages in a bank.

Parameter	Value
Processor	4 cores, 3.2 GHz, 4-issue per core, 16-stage pipeline
Functional units	4 IntALU, 2 IntMult, 2 FPALU, 1 FPMult
IQ, ROB and LSQ size	IQ 64, ROB 196, LQ 32, SQ 32
Physical register num	228 Int, 228 FP
Branch predictor	Hybrid, 8k global + 2K local, 16-entry RAS, 4K-entry and 4-way BTB
L1 caches (per core)	64KB Inst/64KB Data, 2-way, 64B line, hit latency: 1 cycle Inst/3-cycle Data
L2 cache (shared)	4MB, 4-way, 64B line, 15-cycle hit latency
MSHR entries	Inst:8, Data:32, L2:64
Memory	4 channels, 2-DIMMs/channel, 2-ranks/DIMM, 8-banks/rank
Memory controller	64-entry buffer, 15ns overhead
DDR3 channel bandwidth	1066MT/s (Mega Transfers/second), 8byte/channel, 8.5GB/s/channel
DDR3 DRAM latency	8-8-8, precharge 15ns, row access 15ns, column access 15ns

Table 2. Major simulation parameters.

Parameters	Values
Normal voltage	1.5V
Active power-down current	45mA
Burst refresh current	255mA
Precharge standby current	65mA
Operating burst read current (x4, x8/x16)	220/280mA
Operating burst write current (x4, x8/x16)	240/350mA
Active standby current (x4, x8/x16)	75/80mA
Operating one bank active-precharge current (x4, x8/x16)	115/140mA
Self refresh fast mode current	7mA
Self refresh slow mode current	3mA
Precharge power-down fast mode current	25mA
Precharge power-down slow mode current	10mA

Table 3. Parameters for calculating power for 1Gbit/device DDR3-1066 (8-8-8).

memory channel, DIMM, rank and bank. Based on current memory states, memory commands are issued according to the hit-first policy, under which row buffer hits are scheduled before row buffer misses. Reads are scheduled before write operations under normal conditions. However, when outstanding writes occupy more than half of the memory buffer, writes are scheduled first until the number of outstanding writes drops below one-fourth of the memory buffer size. The memory transactions are pipelined whenever possible and XOR-based mapping [29], [16] is used as default configuration. Table 2 shows the major simulation parameters.

To estimate the power consumption of DDRx DRAM devices, we follow the Micron power calculation methodology [21], [18]. A rank is the smallest power management unit for DDRx DRAMs. At the end of each memory cycle, the simulator checks each rank state and calculates the energy consumed during the cycle accordingly. The parameters used to calculate the DRAM power and energy are listed in Table 3 and x16 device has higher value on precharge/activation and read/write operation than those of x4/x8 device to drive its wide data path [18], [19], [20]. For cases that the electrical

Workload	Applications
MEM-1	swim,applu,art,lucas
MEM-2	fma3d,mgrid,galgel,equake
MEM-3	swim,applu,galgel,equake
MEM-4	art,lucas,mgrid,fma3d
MIX-1	ammp,gap,wupwise,vpr
MIX-2	mcf,parser,twolf,facerec
MIX-3	apsi,bzip2,ammp,gap
MIX-4	wupwise,vpr,mcf,parser
ILP-1	vortex,gcc,sixtrack,mesa
ILP-2	perlbmk,crafty,gzip,eon
ILP-3	vortex,gcc,gzip,eon
ILP-4	sixtrack,mesa,perlbmk,crafty

Table 4. Workload mixes.

current values presented in data-sheet are from the maximum device voltage, they are de-rated by the normal voltage [21].

4.2. Workload Construction

In order to limit the simulation time while still emulating the representative behavior of program execution, we select representative simulation points of 100 million instructions for the benchmark programs. The simulation points are picked up according to SimPoint 3.0 [24]. Each processor core is single-threaded and runs one program. The simulation stops when any application commits 100 million instructions. We classify the twenty-six benchmarks of the SPEC2000 suite into MEM (memory-intensive), MIX, and ILP (compute-intensive) applications. The MEM applications are those having memory bandwidth usage higher than 10GB/s when four instances of the application run on a quad-core processor with the DDR3-1066 memory system (four channels and close page mode). The ILP applications are those with memory bandwidth usage lower than 2GB/s; and the MIX applications are those with memory bandwidth usage between 2GB/s and 10GB/s. Note that program *mcf* is usually classified as a MEM workload. Using the classification method here, it falls into the MIX category because the program has very low ILP degree that makes it have low memory bandwidth usage although its cache miss rate is high. We use the memory bandwidth usage to classify applications because the memory power consumption of an application is closely related to its bandwidth usage. Table 4 shows twelve four-core multi-programming workloads randomly selected using these applications.

There are are several metrics and methodologies for comparing performance of multicore/multithreaded systems ([27] for a recent example). We use SMT speedup [25] as performance metric, which is calculated as $\sum_{i=1}^{n}(IPC_{multi}[i]/IPC_{\text{single}}[i])$, where n is the total number of cores, $IPC_{multi}[i]$ is the IPC value of the application running on the ith core under the multi-core execution and $IPC_{single}[i]$ is the IPC value of the same application under single-core execution. We also have results using harmonic mean of the speedups [17], which are

consistent with the results to be reported but will not be presented due to space limit.

4.3. DRAM Configurations

We use two types of base DRAM configurations in our experiments:

- **CC-Slow** with cache line interleaving, close page mode, auto precharge, and with "precharge power-down slow" as the preferred low power mode.
- **PO-Fast** with page interleaving, open page mode, no auto precharge, and with "active power-down" as the preferred low power mode.

PO-Fast is preferred by programs with good spatial locality in their memory accesses, as many accesses become DRAM row buffer hits that do not require precharge and activiation. CC-Slow is preferred by the other programs. With CC-Slow, in most time a device stays in the "precharge standby" mode, which may directly transit to the "precharge power-down slow" and "precharge power-down fast" modes. The former one consumes 128mW power (per device) and its exit latency is 11.25ns; the latter one consumes 321mW with 7.5ns exit latency. Therefore, the former one is a clear winner in power efficiency and is used in CC-Slow. With PO-fast, in most time a device stays in the "active standby" mode and it may directly transit to the "active power-down" mode, which consumes 578mW with an exit latency of 7.5ns. In certain cases, it may make sense for the device to transit to "precharge standby" and then to one of the precharge power-down modes. However, that sequence and the reverse to the "active standby" mode is not only time consuming but also power consuming. We have verified through experiments that PO-Fast is the best configuration for MEM workloads; and CC-Slow is the best configuration for MIX and ILP workloads. In both configurations, a device transits to the preferred low power mode immediately when there is no pending access.

In all configurations, four DDRx channels are used. There are three variants for each configuration: (1) 4CH-2D-1R with x4 devices, two DIMMs per channel, one rank per DIMM, and sixteen devices per rank; (2) 4CH-2D-2R with x8 devices, two DIMMs per channel, two ranks per DIMM, and eight devices per rank; (3) 4CH-2D-4R with x16 devices, two DIMMs per channel, four ranks per DIMM and four devices per rank. In all variants, there are sixteen devices per DIMM.

5. Performance Evaluation and Analysis

5.1. Power Reduction and Performance Impact

Figure 5 presents the average power reduction and performance impact across all mini-rank configurations. In general,

the average power rate drops significantly and the performance penalty is limited.

Results for CC-Slow and MEM Workloads. The mini-rank structure achieves the most significant power saving and smallest performance loss on the MEM workloads. Those workloads are memory-intensive but their performance is not sensitive to the slight increase of memory idle latency, because the queueing delay is significant in the overall memory latency. As for x8 devices, the average memory power savings are 27.0%, 44.1% and 55.6% and the average performance losses are 2.8%, 7.4% and 16.0%, for x32, x16, and x8 mini-rank configurations, respectively. The maximum energy saving is 48.2% with x8 mini-ranks. As for x4 devices, the average power savings are 20.7%, 38.1%, 51.2% and 63.1% and the performance losses are 1.4%, 5.2%, 13.6% and 20.5%, respectively, with x32, x16, x8 and x4 mini-ranks. The maximum energy saving is 48.2% with x8 mini-ranks. As for x16 devices, the power savings are 20.6% and 36.8% and the performance losses are 3.7% and 8.8% with x32 and x16 mini-ranks, respectively. The maximum energy saving is 32.0% with x16 mini-ranks.

These significant power savings are mainly from the reductions of operation power, background power, and termination power. Obviously, the operation power will drop because each memory access touches fewer devices with mini-ranks. It is less obvious that background power also drops because there are more mini-ranks than conventional ranks, and therefore there are more opportunities to find devices to enter fast low power modes. The reduction of termination power is another source of power saving since fewer DQs (date bit lines) are connected to the shared data bus. Detailed breakdown of the power saving will be given in Section 5.2.

Results for CC-Slow and MIX and ILP Workloads. The MIX workloads are moderately memory-intensive and more sensitive to the increase of memory idle latency than the MEM workloads. The power saving is still outstanding but the performance loss is higher. As for x8 devices, the average power saving is 25.1%, 39.5% and 48.1% with x32, x16 and x8 mini-ranks, respectively. For x4 devices, the average power saving is 24.4%, 41.6%, 52.4% and 60.3% with x32, x16, x8 and x4 mini-ranks, respectively; and for x16 devices, the average power saving is 15.3% and 27.9% for x32 and x16 mini-ranks, respectively. However, for all devices, only the x32 mini-rank configuration can limit the performance loss to around 5%; other configurations have performance losses higher than 10%.

The ILP workloads are not memory-intensive and insensitive to memory idle latency. There is visible relative power saving from using mini-ranks and negligible performance loss. With x32 mini-ranks, the power saving is 12.8%, 9.2% and 5.4% for using x4, x8 and x16 devices, respectively, with less than 1% performance loss. With x16 mini-ranks, the power saving is 19.8%, 13.5% and 8.2%, respectively
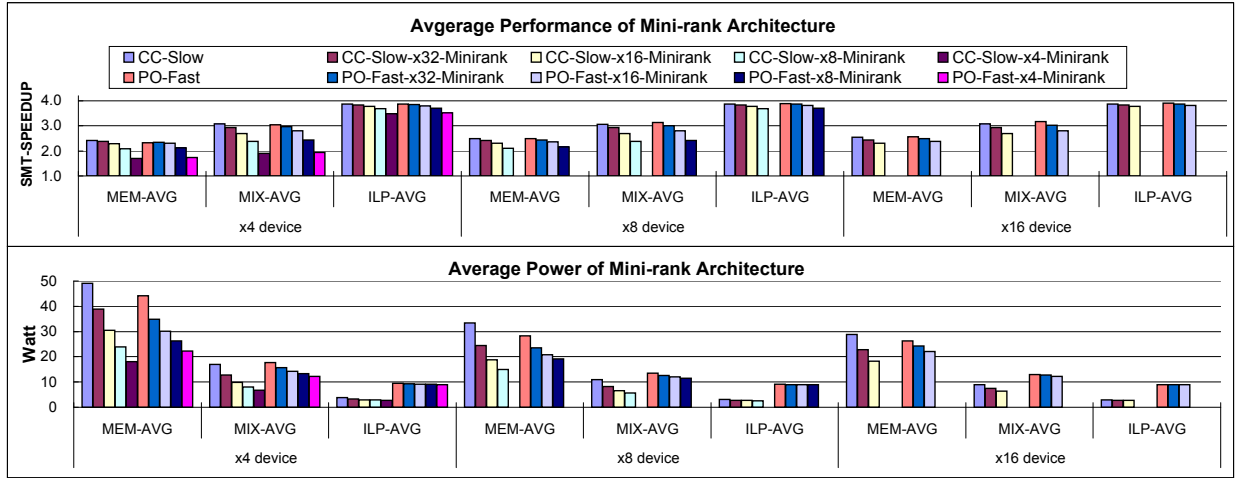
Figure 5. Overview of power saving and performance impact with varying mini-rank configurations.

with about 2.0% performance loss.

Results with PO-Fast and MEM Workloads. The trend of power saving and performance impact are similar to those of CC-Slow. For simplicity, we only discuss the MEM workloads. The power saving is somewhat less significant for two reasons: The open page mode already reduces the operation power (fewer precharge and activation operations with row buffer hits), and the preferred "active power-down" low power mode consumes more power than the counterpart in CC-Slow (578mW vs. 128mW). For x8 devices, the average power saving is 16.9%, 26.1% and 32.4% and the performance loss is 2.8%, 7.4% and 16.0% with x32, x16 and x8 mini-ranks, respectively. For x4 devices, the saving is 21.5%, 32.1%, 40.6% and 49.5%, and the penalty is 0.4%, 1.2%, 8.7% and 25.1% with x32, x16, x8 and x4 mini-ranks, respectively. For x16 devices, the saving is 9.8% and 16.1% with penalty of 2.7% and 8.1% with x32 and x16 mini-ranks, respectively.

The mini-rank architecture is more effective in power saving with CC-Slow than with PO-Fast, because CC-Slow can utilize the increased number of mini-ranks better in reducing the background power. With x8 devices and x16 mini-ranks, CC-Slow and PO-Fast consume 18.7W and 20.9W power on average, respectively. With x8 devices and x8 mini-ranks, CC-Slow and PO-Fast consume 14.9W and 19.1W power on average, respectively.

5.2. Breakdown of Power Reduction

Figure 6 shows the detailed memory power consumption of mini-rank structures breakdown into five parts: (1) MRB's non-I/O logic and MRB's I/O with devices, (2) device or MRB's I/O power with DDR*x* bus, (3) device read/write, (4) device operation, and (5) device background. The reference configurations are CC-Slow and PO-Fast with conventional ranks, which do not have part one power consumption and

their part two power consumption is the device I/O power with DDR*x* bus. For mini-ranks, their part two power consumption is the MRB I/O power with DDR*x* bus. Due to space limit, we only discuss the results with x8 devices.

Device Operation Power and Background Power. First of all, the device operation power (part 4) drops proportionally as the size (data width) of mini-ranks scales down. This is because the number of devices involved per access is proportional to the data width. For example, for the MEM-1 workload with CC-Slow, the operation power is 11.7W, 5.9W, 2.9W and 1.4W with normal ranks and x32, x16, x8 mini-ranks, respectively. Second, the background power (part 5) also drops significantly as the size of mini-ranks scales down, because the mini-ranks will stay in the preferred low power mode for a longer time when the number of mini-ranks increases. The "precharge standby" high power mode consumes 6.5 times of the power of the "precharge power-down slow" low power mode. The average background power reduction for MEM workloads and with CC-Slow is 25.5%, 46.9% and 61.8% for x32, x16 and x8 mini-rank configurations, respectively. On average a device stays in the low power mode for 19%, 44%, 66% and 81% of time with the conventional rank and the x32, x16 and x8 mini-ranks, respectively. For ILP workloads, the reduction is 6.2%, 8.9% and 10.2%, respectively. It is less significant because the light memory traffic of ILP workloads already makes the devices stay in the low power mode for about 97% of time. The measured average bandwidth usage is 128MB/s per channel from ILP workloads vs. 5.9GB/s per channel from the MEM workloads. Nevertheless, 3% of time in the high power mode translates to 16% of total power consumption, so there is still room of improvement.

Device/MRB I/O Power with the DDR*x* Bus. This power (part 2) drops with the use of mini-rank because of the reduction of termination power. For the MEM workloads,
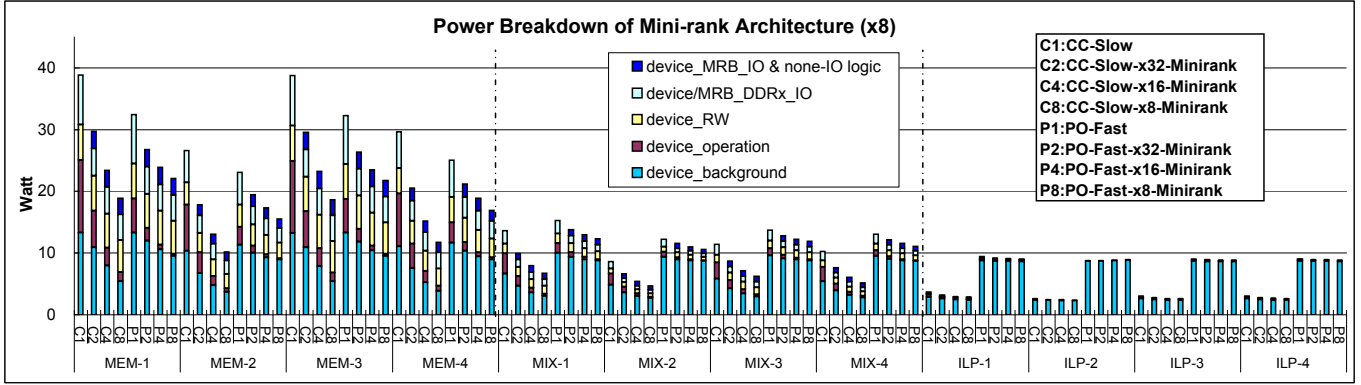
Figure 6. Power breakdown with mini-rank configurations (x8 device).

this power is 6.8W with conventional ranks and 4.8W, 4.5W and 4.3W with x32, x16 and x8 min-ranks. This reduction happens when each DIMM has more than one conventional rank. In our configuration with x8 devices, each channel has two DIMMs and there are two ranks per DIMM. The DQs (data bit lines) of all of the four ranks are directly connected to the DDRx bus. When one rank is being accessed, the 128 DQs of the other DIMMs must be terminated. With mini-ranks of any size, only 64 DQs of the MRB have to be terminated.

Device Read/Write Power. This power (part 3) changes only slightly because this power is proportionally to the number of bits being read and written. There is a moderate decrease by moving from x32 to x16 and then to x8 mini-ranks. For example, this power is 4.8W, 4.5W and 4.2W with x32, x16 and x8 mini-ranks for MEM workloads on average, respectively. This reduction is partially from the performance loss and thus the increase of program execution time as the size of mini-rank scales down.

MRB I/O Power to Devices and non-I/O Logic Power. Finally, we need to pay attention to this power (part 1) because it is the extra power introduced by MRBs. Like device I/O power, this power is roughly proportional to the memory access frequency. For the MEM workloads, this power (of all MRBs) on average is 2.4W, 2.2W and 2.1W with x32, x16, and x8 mini-ranks, respectively. By contrast, for the ILP workloads, it is 65mW, 64mW and 62mW, respectively. The MRBs has its own low power mode in which the DLL is frozen, and they can enter the low power mode with the ILP workloads. Our simulation statistics show that the DIMM (and thus the MRB) is idle for 95% of time with the ILP workloads, and only 6% and 42% of time with the MEM and MIX workloads, respectively. As Figure 6 shows, this part of power is fully offset by the reduction of operation power, background power, and the termination power.

The trends of changes with PO-Fast are similar. In fact, the use of mini-rank can cut the device operation power more deeply than with CC-Slow for the MEM-workloads, from

4.3W with conventional ranks to 1.6W, 0.64W and 0.29W with x32, x16, and x8 mini-ranks, respectively. Again, the major reason is having less devices involved in each memory accesses. Additionally, using mini-rank increases the number of logically independent row buffers (while their sizes are reduced) and improves row buffer hit ratios, which further reduces the device operation power. The average hit ratio is 62.4%, 70.9%, 77.4% and 77.9% for conventional ranks and x32, x16, and x8 mini-ranks, respectively.

5.3. Analysis of Performance Impact

The left part of Figure 7 compares the performance of conventional ranks and mini-ranks for each MEM workload, with CC-Slow and x8 device configuration. The performance metric is SMT speedup [25]. The results with PO-Fast (not shown) are similar. To gain insights, the right part of Figure 7 shows the breakdown of memory latency for read accesses into four parts: (1) the memory controller overhead (T_{MCO}), (2) the DRAM operation latency ($T_{operation}$), (3) the additional latency introduced by MRB (T_{MRB}), and (4) the queuing delay ($T_{queuing}$). T_{MCO} is a fixed latency of 15ns (48 processor cycles for our simulation configuration); $T_{operation}$ is the memory idle latency excluding the memory controller overhead (T_{MCO}) and including the delay caused by DRAM refreshes, whose minimum is 120 processor cycles with CC-Slow. T_{MRB} is 30, 78, 178 processor cycles with x32, x16, and x8 mini-ranks, respectively.

For the MEM workloads, the extra latency introduced by MRB is a small component (8.5% on average with x32 mini-ranks) in the overall memory latency because of the significant queueing delay. Furthermore, having more mini-ranks helps reduce $T_{queuing}$. For example, $T_{queuing}$ is reduced from 174 processor cycles to 169, 160, and 133 processor cycles with x32, x16, and x8 mini-ranks. For the MIX workloads, the queuing delay is less significant than for the MEM workloads and thus T_{MRB} has more weight. For the ILP workloads, this weight is even higher; however, the overall performance penalty is smaller than the MIX
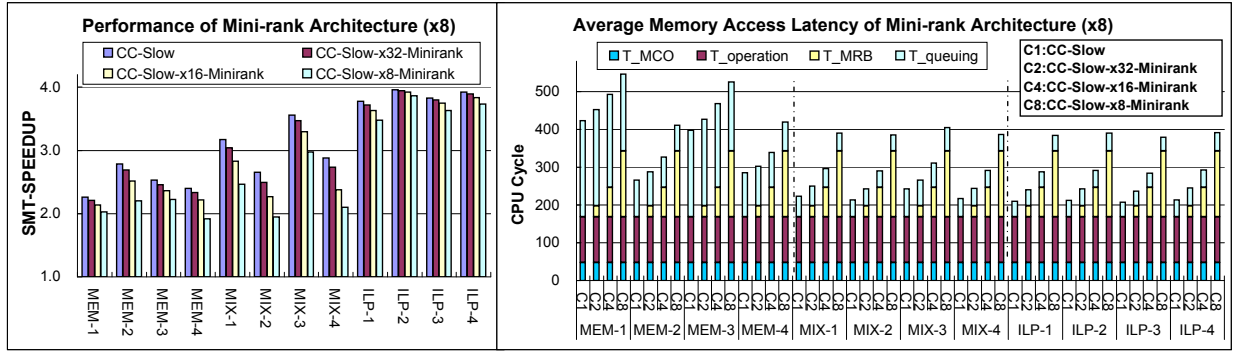
Figure 7. Performance impact with mini-rank configurations (x8 device).

workloads because now memory stall is a smaller factor in the overall performance.

5.4. Energy Saving of Mini-rank Architecture

The memory energy is calculated as the product of power rate and program execution time. Figure 8 shows the energy saving by using mini-rank with x8 devices. All workloads show energy savings. This is also true for x4 and x16 devices (not shown). Not surprisingly, the MEM workloads has the most significant saving. The average saving is 26.2%, 40.9% and 48.2% with x32, x16 and x8 mini-ranks, respectively. For the MIX workloads, the average saving is 22.1%, 33.9% and 37.8%, respectively; and for the ILP workloads, it is 8.1%, 10.7% and 11.6%, respectively. We avoid using EDP (Energy Delay Product) as a metric because our simulation does not report processor energy consumption. Contemporary processors are increasingly responsive in power saving when system performance is bounded, consuming less power as memory performance degrades. Therefore, we expect that the improvements in EDP will also be significant with those processors.

5.5. Power and Performance Impact of Mini-rank Architecture with Different Memory Configurations

Figure 9 shows the power saving and performance impact of mini-rank architecture with three more DRAM configurations: (1) single channel, single DIMM per channel and single rank per DIMM (1CH-1D-1R); (2) dual channel, single DIMM per channel and single rank per DIMM (2CH-1D-1R); and (3) eight channel, dual DIMM per channel and dual rank per DIMM (8CH-2D-2R). All use x8 devices. The first two represent low-end computing systems like desktops; and the last one represents workstations and servers (and so does the 4CH-2D-2R configuration in the previous experiments). The power saving and performance impact with 8CH-2D-2R are similar to those with 4CH-2D-2R.

However, with 1CH-1D-1R and 2CH-2D-2R, there are gains in both power saving and performance for the MEM

workloads. The average performance gain with 1CH-1D-1R is 8.0%, 12.2% and 13.6% and the average power saving is 6.7%, 20.4% and 30.3%, with x32, x16, and x8 mini-ranks, respectively. The source of performance gain is the increased memory concurrency from having more mini-ranks. For MEM workloads, the bandwidth utilization is improved that more than offsets the MRB latency. The average read latency reduces from 1511 cycles without mini-rank to 1334, 1232 and 1221 cycles with x32, x16 and x8 mini-ranks, respectively. As for 2CH-1D-1R, the performance gain is 5.8%, 5.5% and 1.6% and the power saving is 8.6%, 13.8% and 36.3% with x32, x16, and x8 mini-ranks, respectively.

6. Related Work

Most of related works focus on how to utilize low power modes of DRAM memories, particularly for Rambus and Direct Rambus memories. Lebeck et al. propose power-aware page allocation policies to increase the chances that DRAM chips can be put into low power modes [13]. Delaluz et al. propose compiler-directed and hardware-assisted approaches, such as clustering data with similar lifetime and predicting inter-access time, to exploiting memory low power modes [3]. Fan et al. build an analytical model to approximate memory chips idle time and study when to make power mode transitions [6]. Delaluz et al. present an OS-level scheduling policy by tracing the memory usage of each process to reduce the energy consumption [4]. Huang et al. propose a power-aware virtual memory system that can effectively manages energy footprint of each process through virtual memory remapping [10]. Li et al. build a model to estimate performance loss of low power management and propose performance guaranteed low power management schemes for both memory and disk [14]. Zhou et al. use page miss ratio curve as guidance for memory allocation of multiprogramming systems and memory energy management [30]. Huang et al. propose memory traffic shaping method by data migration and address remapping to maximize the time that memory chips in low power modes [11]. Pandey et al. explore the unique memory behavior of DMA accesses to aggregate
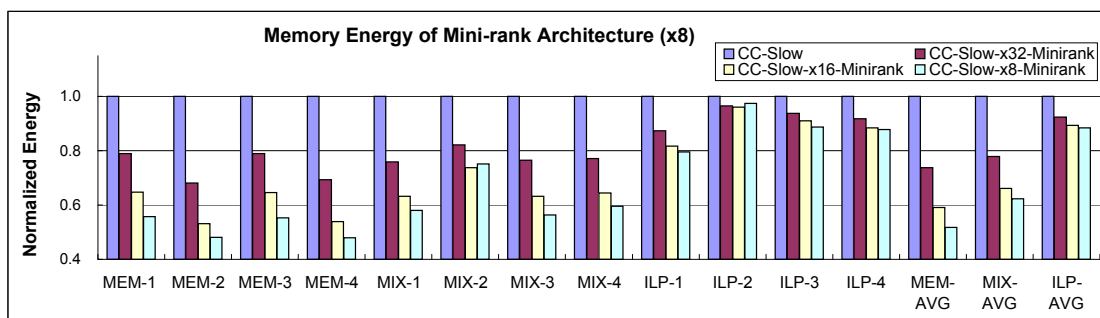
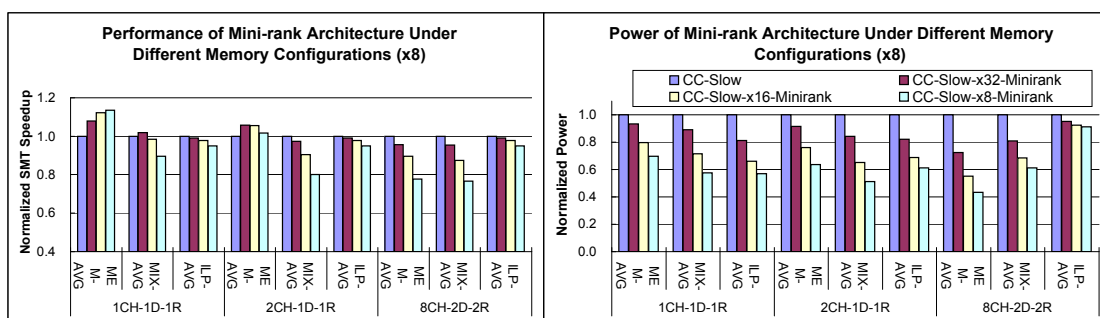Figure 8. Memory energy with mini-rank configurations (x8 device).

Figure 9. Power and performance of mini-rank architecture under different memory configurations (x8 device).

DMA requests from different I/O buses together in order to maximize the memory low power duration [22].

Recently, Fan et al. explore power management mechanisms at data center level [7] and Hur and Lin study the power saving potential of a memory scheduling scheme and evaluates a scheme to predict throttling delay for memory power control [12]. Mrinmoy et al. propose adaptive refresh method to reduce the refresh power [9]. Bruno et al. propose algorithms that dynamically adjust memory chip power states to limit power consumption of Rambus DRAM [5]. Different from those works, our proposed mini-rank architecture is to reduce operation power as well as exploit low power opportunities for significant power/energy improvement on memory sub-system. And with more mini-ranks in memory subsystems, there will be more opportunities to employ the previously proposed low power techniques for Rambus in DDR*x* memories. Additionally, Cuppu and Jacob [2] have studied the performance impact of memory design parameters including the trade-off between latency and concurrency. The mini-rank design involves such a trade-off but for power efficiency rather than performance.

7. Conclusion

In this study, we have proposed a novel mini-rank DRAM architecture for power optimization by reducing the number of devices involved in each memory access and taking advantage of fast low power modes. Our simulation results show

significant power/energy efficiency improvement across all device configurations. The proposed mini-rank architecture only requires a minor change to the memory controller, while keeps memory devices and bus design untouched. It can also provide dynamically configurable features for best power and performance trade-offs for varying workloads behavior, power and performance requirements.

Acknowledgment

References

[1] N. L. Binkert, R. G. Dreslinski, L. R. Hsu, K. T. Lim, A. G. Saidi, and S. K. Reinhardt. The m5 simulator: Modeling networked systems. *IEEE Micro*, 26(4):52–60, 2006.

[2] V. Cuppu and B. Jacob. Concurrency, latency, or system overhead: Which has the largest impact on uniprocessor DRAM-system performance? In *Proceedings of the 28th International Symposium on Computer Architecture*, pages 62–71, 2001.

[3] V. Delaluz, M. Kandemir, N. Vijaykrishnan, A. Sivasubramaniam, and M. J. Irwin. DRAM energy management using software and hardware directed power mode control. In *Proceedings of the 7th International Symposium on High-Performance Computer Architecture*, pages 159–170, 2001.

[4] V. Delaluz, A. Sivasubramaniam, M. Kandemir, N. Vijaykr-ishnan, and M. J. Irwin. Scheduler-based DRAM energy management. In *Proceedings of the 39th conference on Design automation*, pages 697–702, 2002.

[5] B. Diniz, D. Guedes, J. Wagner Meira, and R. Bianchini. Limiting the power consumption of main memory. In *Proceedings of the 34th International Symposium on Computer Architecture*, pages 290–301, 2007.

[6] X. Fan, C. Ellis, and A. Lebeck. Memory controller policies for DRAM power management. In *Proceedings of the 2001 International Symposium on Low Power Electronics and Design*, pages 129–134, 2001.

[7] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *Proceedings of the 34th International Symposium on Computer Architecture*, pages 13–23, 2007.

[8] Q. S. Gao. The Chinese Remainder Theorem and the prime memory system. In *Proceedings of the 20th International Symposium on Computer Architecture*, pages 337–340, 1993.

[9] M. Ghosh and H.-H. S. Lee. Smart Refresh: An Enhanced Memory Controller Design for Reducing Energy in Conventional and 3D Die-Stacked DRAMs. In *Proceedings of the 40th International Symposium on Microarchitecture*, Dec. 2007.

[10] H. Huang, P. Pillai, and K. G. Shin. Design and implementation of power-aware virtual memory. In *Proceedings of the USENIX Annual Technical Conference 2003 on USENIX Annual Technical Conference*, pages 57–70, 2003.

[11] H. Huang, K. G. Shin, C. Lefurgy, and T. Keller. Improving energy efficiency by making DRAM less randomly accessed. In *Proceedings of the 2005 International Symposium on Low Power Electronics and Design*, pages 393–398, 2005.

[12] I. Hur and C. Lin. A comprehensive approach to dram power management. In *Proceedings of the 13th International Symposium on High-Performance Computer Architecure*, 2008.

[13] A. R. Lebeck, X. Fan, H. Zeng, and C. Ellis. Power aware page allocation. In *Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 105–116, 2000.

[14] X. Li, Z. Li, F. David, P. Zhou, Y. Zhou, S. Adve, and S. Kumar. Performance directed energy management for main memory and disks. In *Proceedings of the 11th International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 271–283, 2004.

[15] J. Lin, H. Zheng, Z. Zhu, E. Gorbatov, H. David, and Z. Zhang. Software thermal management of DRAM memory for multi-core systems. In *Proceedings of the 2008 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 337–348, 2008.

[16] W. Lin, S. K. Reinhardt, and D. Burger. Reducing DRAM latencies with an integrated memory hierarchy design. In *Proceedings of the Seventh International Symposium on High-Performance Computer Architecure*, pages 301–312, 2001.

[17] K. Luo, J. Gummaraju, and M. Franklin. Balancing thoughput and fairness in SMT processors. In *IEEE International Symposium on Performance Analysis of Systems and Software*, 2001.

[18] Micron Technology, Inc. DDR3 SDRAM System-Power Calculator. http://download.micron.com/downloads/misc/ddr3_power_calc.xls.

[19] Micron Technology, Inc. MT41J128M8BY-187E. http://download.micron.com/pdf/datasheets/dram/ddr3/1Gb%20DDR3%20SDRAM.pdf.

[20] Micron Technology, Inc. MT47H64M16HR-25E. http://download.micron.com/pdf/datasheets/dram/ddr2/1GbDDR2.pdf.

[21] Micron Technology, Inc. TN-41-01: Calculating Memory System Power For DDR3. http://download.micron.com/pdf/technotes/ddr3/TN41_01DDR3%20Power.pdf, Aug. 2007.

[22] V. Pandey, W. Jiang, Y. Zhou, and R. Bianchini. DMA-Aware Memory Energy Management. In *Proceedings of the 12th International Symposium on High-Performance Computer Architecture*, pages 133–144, 2006.

[23] S. Rixner, W. J. Dally, U. J. Kapasi, P. Mattson, and J. D. Owens. Memory access scheduling. In *Proceedings of the 27th International Symposium on Computer Architecture*, pages 128–138, June 2000.

[24] T. Sherwood, E. Perelman, G. Hamerly, and B. Calder. Automatically characterizing large scale program behavior. In *Proceedings of the Tenth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 45–57, Oct. 2002.

[25] A. Snavely, D. M. Tullsen, and G. Voelker. Symbiotic job-scheduling with priorities for a simultaneous multithreading processor. In *Proceedings of the 2002 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 66–76, 2002.

[26] Synopsys Corp. Synopsys design compiler. http://www.synopsys.com/products/logic/design_compiler.html.

[27] J. Vera, F. J. Cazorla, A. Pajuelo, O. J. Santana, E. Fernandez, and M. Valero. A novel evaluation methodology to obtain fair measurements in multithreaded architectures. In *Workshop on Modeling Benchmarking and Simulation*, 2006.

[28] D. T. Wang. *Modern DRAM Memory Systems: Performance Analysis and a High Performance, Power-Constrained DRAM Scheduling Algorithm*. PhD thesis, University of Maryland, Department of Electrical and Computer Engineering, 1993.

[29] Z. Zhang, Z. Zhu, and X. Zhang. A permutation-based page interleaving scheme to reduce row-buffer conflicts and exploit data locality. In *Proceedings of the 33rd International Symposium on Microarchitecture*, pages 32–41, 2000.

[30] P. Zhou, V. Pandey, J. Sundaresan, A. Raghuraman, Y. Zhou, and S. Kumar. Dynamic tracking of page miss ratio curve for memory management. In *Proceedings of the 11th International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 177–188, 2004.