Fine-Grained Activation for Power Reduction in DRAM

This DRAM architecture optimization, which appears transparent to the memory controller, significantly reduces power consumption. With trivial additional logic, using the posted-CAS command enables a finer-grained selection when activating a portion of the DRAM array. Experiments show that, in a high-use memory system, this approach can reduce total DRAM device power consumption by up to 40 percent.

Elliott Cooper-Balis
Bruce Jacob
University of Maryland

DRAM has become the ubiquitous solution for memory in all types of systems, from the world's fastest supercomputer to the latest mobile smart phone. Its widespread adoption is due primarily to device standardization, particularly the package pin-out and the device's operating protocol. Unfortunately, device standardization also tends to inhibit dramatic—or, in many cases, even moderate—modifications. Consequently, evolution in DRAM technology comes in small, incremental steps. In this article, we discuss an optimization to the DRAM device architecture that requires no modification in protocol, and a low-cost modification in control logic that doesn't affect the device's interoperability with standardized memory controllers and interfaces.

This optimization is similar in concept to Fujitsu's fast-cycle RAM (FCRAM), which partitions the DRAM storage array within the device, reducing access times and power consumption. Unfortunately, the proprietary nature of this design and the required nonstandard bus width has inhibited its widespread adoption.

However, we can employ the fundamental idea behind the FCRAM while still adhering to all adopted standards for DRAM memory systems. The existing posted-CAS (column-address strobe) command, introduced in 2003, gives a DRAM device earlier access to the entire address of requested data: the column address is sent to the DRAM device one cycle after the row address—far earlier than necessary.[1] With the addition of a single decoder to the device control logic, the address obtained via this command can serve to activate a smaller portion of a row within the data array. This achieves the same power benefits as the FCRAM without requiring changes to the physical bus or operating protocol; all that's required is the posted-CAS command.

A significant portion of the power dissipated during typical DRAM device operation is during row activation. Therefore, activating fewer bits can significantly reduce the overall power dissipated by the memory system. While executing the SPEC benchmark suite on the University of Maryland's cycle-accurate DRAM simulator (DRAMsim),[2] our proposed architecture reduced the DRAM system's power consumption by 9 to 40 percent.

Published by the IEEE Computer Society

Modern DRAM architecture

Since the 1970s, DRAM devices have used a split-addressing mechanism that divides the target address into two components, allowing a narrow (and thus inexpensive) address bus. The DRAM array consists of rows and columns that are accessed and addressed using separate commands, which typically arrive several clock cycles apart. Data in the array is represented as charge stored in capacitors and must be sensed before it can be read or written. The act of sensing this charge is called an *activation* and is performed upon receiving a row-address strobe (RAS) command. Activations are always performed on an entire DRAM row simultaneously. Once an activation is complete, the data remains within the sense amplifiers to be read or written with either a CAS or CAS-W (CAS write) command, respectively. After reading or writing data, the memory controller must prepare the DRAM array for the next activation via a precharge operation (a PRE command), one of which is always paired with a previous activation. Activation and precharge operations always apply to the data array's entire row, regardless of how much data is actually needed. In a typical DRAM device, a row of the data array has 8,192 bits, yet a column, the addressable portion of that row, is typically between 4 and 32 bits.[3]

In modern DRAM devices (see Figure 1), the storage array isn't a monolithic structure but rather consists of thousands of smaller cores, typically 256 Kbits apiece.[4,5] This prevents having word- and bitlines that span the entire array length, which would represent unnecessarily large loads. Although these cores' architecture and organization vary between manufacturers, the concept spans all DRAM devices because it's a physical necessity. This concept shouldn't be confused with the multiple, independent arrays in modern devices, called *banks*. A typical DRAM device has four or eight banks, which allow requests to be handled in parallel, thereby reducing conflicts and allowing increased bus utilization.

The DRAM protocol also includes a refresh (REF) command. Due to the nature of the capacitors used to store individual bits of data, the representative charge leaks, causing the intended value to dissipate beyond recognition. The REF command resolves this issue by reading a row and placing it back into the data array, thereby refreshing it. This occurs once every 64 ms. Thus, in a device with 8,192 rows, the memory controller typically issues a REF command every 7.8 μs.[3]

A commodity dual in-line memory module (DIMM), the form of DRAM memory that is most familiar to consumers, can contain between one and four ranks of DRAM. In a standard JEDEC-style double data-rate

The modern memory system

Channel: A group of one or more ranks of DRAM that handle requests from the memory controller. These ranks operate independently from other channels.

Dual in-line memory module (DIMM): A printed circuit board containing individual DRAM devices grouped together to form one or more ranks of memory. The average consumer is most familiar with this form of DRAM.

Rank: A group of DRAM devices operating together to service requests from the memory controller. Each device shares the same command and address bus, but each of the device's data buses (typically 4 to 16 bits wide) are grouped to form a larger, monolithic data bus. A JEDEC standardized memory system is organized by ranks that operate on a 64-bit data bus.

Bank: An independent array of DRAM cells inside a DRAM device. A typical DDR3 DRAM device has eight banks.

Row: A group of bits in the DRAM array that are sensed and precharged together when receiving an *activate* or *precharge* command, respectively. A typical DDR3 DRAM device has 16,384 rows.

Column: The smallest addressable portion of the DRAM device. Depending on the DRAM device, a column can range from 4 to 16 bits and even as high as 32 bits for high-end graphics memory.

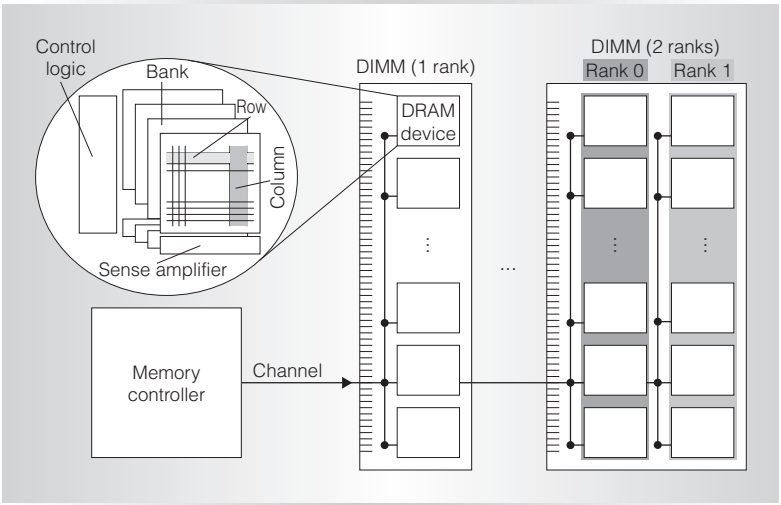

Figure A. Elements of a modern memory system.

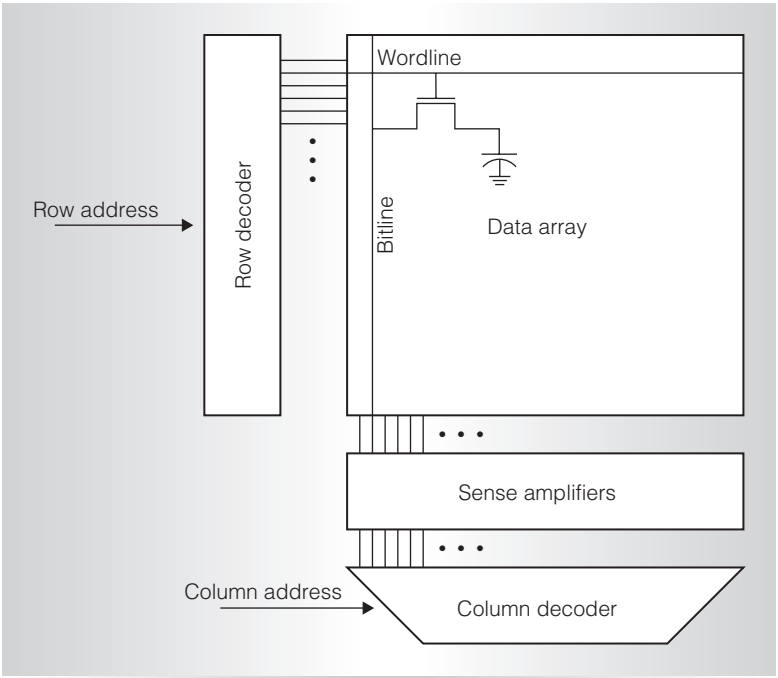

Figure 1. The modern DRAM device architecture. A large array of capacitances is addressed with a row and column decoder. The addressed bits are sent to the sense amplifier, where data may be read or written.

(DDR) DRAM memory system, a *rank* contains multiple DRAM devices grouped together and operated in lock step. When a command is sent to a particular rank, all devices in that rank receive the same command. For example, a 1-Gbyte rank of storage with a 64-bit-wide data bus can contain eight ×8 1-Gbit DRAM devices, all working in unison to handle requests from the memory controller. In such a system, a 64-byte cache fill will discharge, sense, and recharge 65,536 capacitors and precharge 65,536 bit lines to read 512 bits of data.

Later generations of DDR have additional specialized commands, one of which is the posted-CAS command. The memory controller sends this command immediately following the RAS command instead of waiting until the row has been completely activated. The DRAM device buffers the command and column address and delays their execution until the data is available in the sense amplifiers. This command was introduced to simplify scheduling by the memory controller and to relieve strain on the command bus.[1]

Figure 2 displays the difference between a standard read cycle and one using the posted-CAS command. In Figure 2b, the memory controller sends the CAS command immediately after the RAS command, yet the column access doesn't occur until the sense operation has been completed. The timing constraint used to specify this delay is *tAL*, for additive latency. Note the implicit PRE command in both cases. The point at which the memory controller issues the PRE command depends on the memory system's row-buffer management policy: a closed-page management system precharges the row immediately after a CAS or CAS-W (which is the case in both Figures 2a and 2b), whereas an open-page management system leaves a row activated until an explicit PRE command is received (not shown in either Figure 2a or 2b).

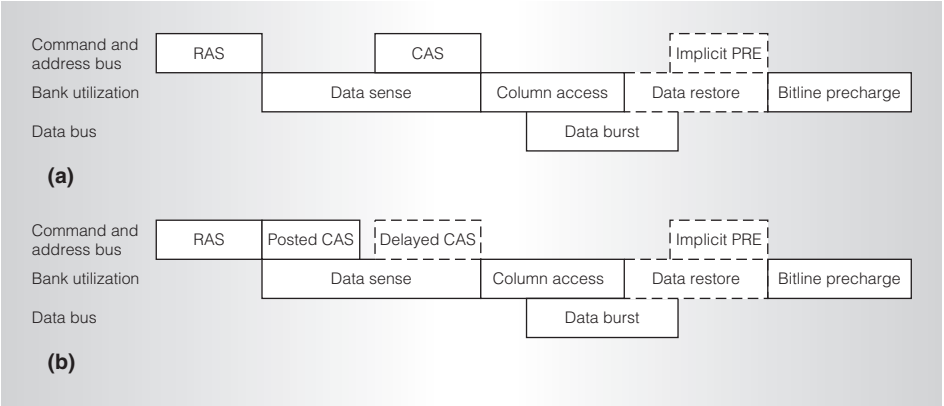

Figure 2. The difference between a typical read cycle (a) and a read cycle using the posted-CAS (column-address strobe) command (b). (PRE: precharge; RAS: row-address strobe.)

Power consumption in DRAM systems

Within an individual DRAM device, the power consumed during operation falls into three distinct components: background power, activation power, and read/write burst power.[6] Background power encompasses the power consumed by the control logic as well as the power dissipated from refreshing the data array. This value depends on the device's state (that is, whether a row is activated and whether the device is in a power-down state). Activation power refers to power consumed from activating an array row and from precharging the array's bitlines. These quantities are grouped together because they're always executed as a pair. The read/write burst power is consumed when data moves either into or out of the device.

DRAM power dissipation is application specific; the usage patterns in the memory system determine the ratio of the three components. Under light use, the background power dominates the total power dissipated. Conversely, under heavy use, the activation power is the most dominant. The total read/write burst power consumed depends on both the memory system's activity and the burst length. For example, in a system with a 2.4-GHz processor and a single rank of typical 400-MHz DDR2 DRAM (800 Mbps), varying the total cache size and cache block size while executing the equake benchmark demonstrates how different loads on the memory system impact the breakdown of power consumption within the DRAM (see Figure 3a). A larger cache size reduces the load on the memory system, leading to fewer activations and more opportunities to place the DRAM devices in power-down mode, thereby reducing the background power as well. A larger cache block size provides similar benefits to increasing cache size and also amortizes the high cost of DRAM row activations over more read and write data transferred.

The relative effects of various cache configurations are application dependent: although the equake benchmark results (Figure 3a) suggest that cache block size has a greater impact on power consumption than total cache size, the art benchmark results (Figure 3b) show the opposite. There is no ideal solution for all situations.

Fujitsu fast-cycle RAM

Fujitsu's FCRAM module targets mobile and embedded systems because of its low power consumption and performance comparable to that of standard DDR devices. The FCRAM achieves lower overall power consumption during operation by partitioning data arrays into smaller subarrays. When an activation is issued to the data array, instead of activating an entire row, the FCRAM activates a subarray and sends a smaller portion of the bits to the sense amplifiers, thereby consuming less power and taking less time to complete. Activating a smaller portion of a row requires more address bits than when simply activating an entire row. The FCRAM accomplishes this by essentially moving bits from the column address to the row address, thereby performing a portion of the column access early.[1] This is evident from the relative sizes of the column and row addresses for similar specified parts of DDR and FCRAM (see Figure 4).[7,8]

This simple and effective optimization to the DRAM architecture results in both faster access times and reduced power consumption. However, despite its benefits, the device has never achieved widespread use. Its adoption has been hindered by the fact that it is proprietary to Fujitsu and doesn't adhere to the JEDEC standard for memory systems. This requires nontrivial physical changes to other parts of the system, such as PCB board traces, sockets, and packaging.

Other solutions for reducing DRAM power consumption

In addition to Fujitsu's FCRAM, various other solutions have been proposed to reduce DRAM power consumption (see the ''Related work'' sidebar). These solutions range from software and operating-system-based approaches to circuit- and architecture-level modifications. However, although the demand for low-power memory is great, the current near-future outlook leaves much to be desired.[9]

Most of these solutions tend to fall into one of two categories, with the memory controller representing the dividing line. At higher abstraction levels (operating

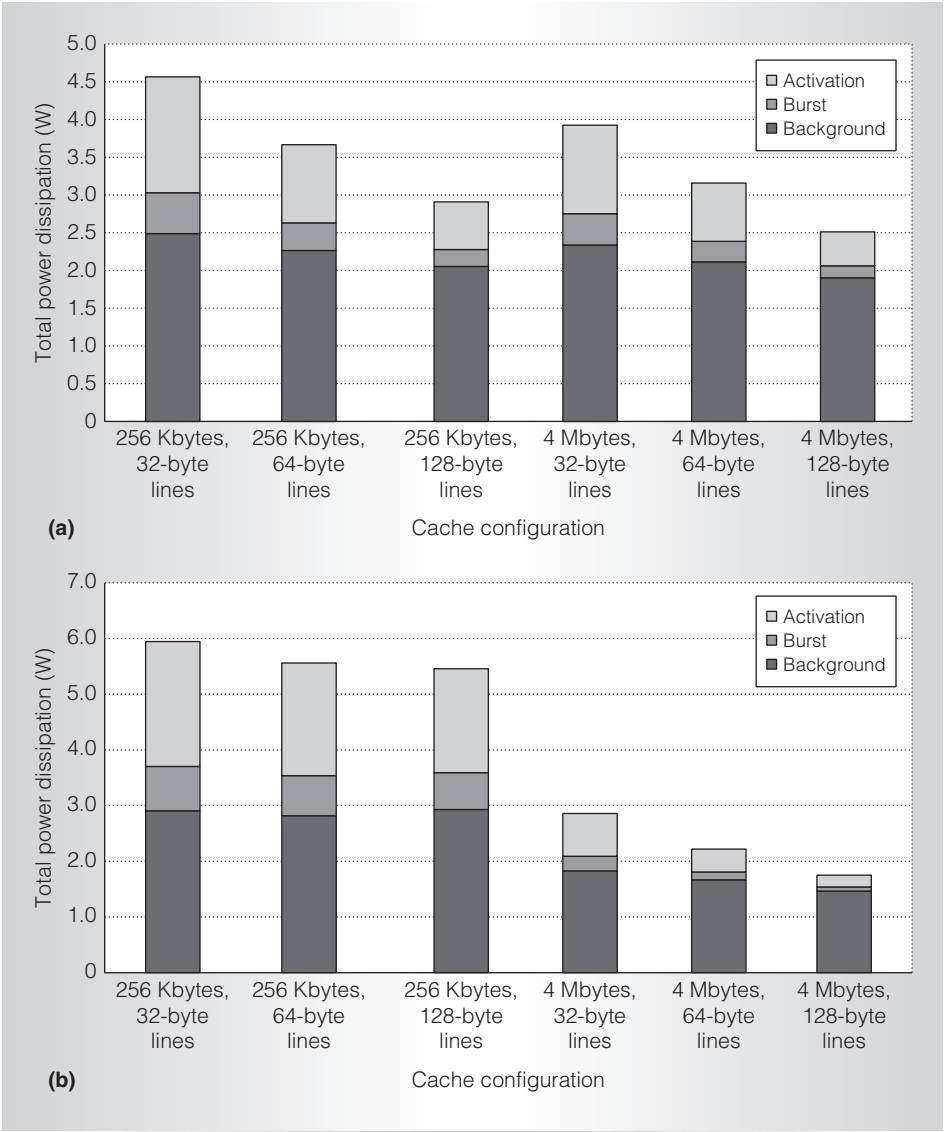

Figure 3. Cache configurations and the impact on power components in the memory system for a 2.4-GHz CPU executing the equake benchmark (a) and the art benchmark (b). The equake benchmark results suggests that cache block size has a greater impact on power consumption than total cache size, but the art benchmark results suggest the opposite.

system, compiler), the solutions are typically independent of the DRAM architecture and therefore require no physical changes to the system. At lower abstraction levels (DRAM system architecture, DRAM circuit design), the solutions typically require nonstandard changes—as with Fujitsu's FCRAM.

An ideal solution to this issue is one that, like software solutions, is transparent to the host system yet provides the benefits of hardware solutions. Our proposed architecture satisfies both of these constraints.

Fine-grained activation with the posted-CAS command

Successive generations of the DDR standard have introduced increasingly complex commands to account for higher clock frequency and complex scheduling

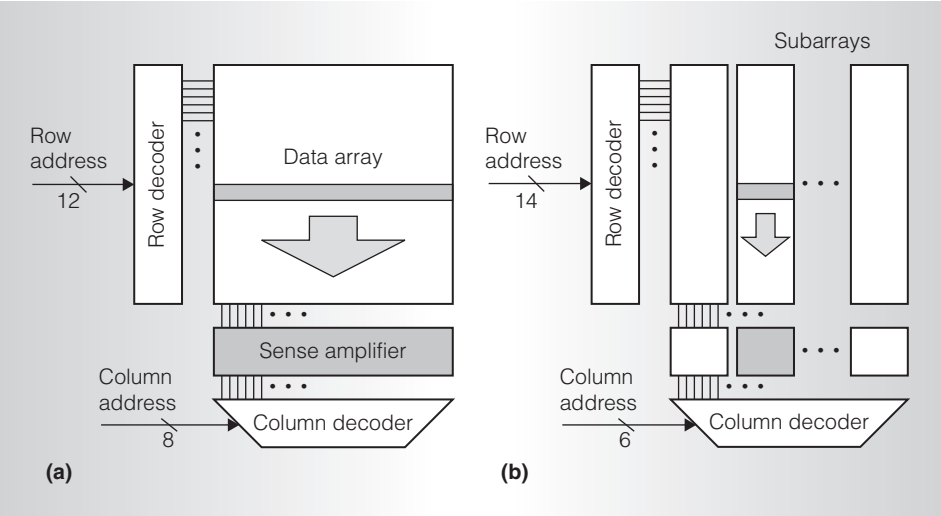

Figure 4. The architectural difference between standard DRAM (a) and Fujitsu's fast-cycle RAM (FCRAM) (b).

algorithms. The second-generation DDR protocol introduced the posted-CAS command. To implement this command, additional logic on the DRAM device buffers the column address and delays the command until the row has been completely activated. The *tAL* timing parameter controls this timing delay (http://www.jedec.org). When using the posted-CAS command, the memory controller sends the RAS and CAS commands in successive clock cycles instead of waiting until the row has finished activating to execute the CAS command. This scheme simplifies the memory controller's command-bus scheduling by combining the activation, column-access, and precharge operations into two adjacent clock cycles.[10]

Although the mechanism was initially implemented for scheduling purposes, there are other benefits from early access to the column address. As we mentioned earlier, current memory systems must activate an entire row of the data array before any access is possible, regardless of how little of the row is actually required. However, when using the posted-CAS command, the column address is available near the beginning of an activation. Thus, the device can select a smaller portion of the row to activate, thereby reducing the power dissipated by both activation and precharge operations.

Because the command and logic already exist to send and buffer the column address, no modifications to the protocol, bus, or memory controller are necessary. The architectural modifications include the addition of a single one-hot decoder and a small amount of control circuitry for each selectable division in the data array. The decoder uses the column address to select only the required portion of the row. The higher the degree of the decoder, the finer the grain of access allowed to the row in the data array. Although it's possible to activate the individual column contained in the request, our experiments suggest there will be diminishing returns after decoder sizes of 5 to 32. Beyond this point, the decoder's cost will outweigh the overall power reduction.

Fine-grained activation architecture

Figure 5 shows our proposed DRAM device architecture. We introduced an additional n-to-2^n one-hot decoder, the *row-division decoder*, to select only the necessary portion of the row by using the upper n bits of the column address received from the posted-CAS command. We also added a single AND gate to the wordline of each division in the row, for every row. This AND gate uses the input to the wordline driver and the row-division decoder to determine

Related work

Researchers have investigated scheduling-based power management at the operating-system,[1] compiler,[2,3] and memory controller levels.[4-6] Effectively managing the memory controller can provide significant reductions in power dissipation, as Hur and Lin have shown by throttling performance, managing power-down modes, and using adaptive history-based scheduling.[7] The memory controller's row-buffer management policy and address-mapping scheme also impact the power dissipated within the memory system.[4] Panda et al. have shown how to reduce power with address-mapping schemes,[8] whereas Delaluz et al. and Fan, Ellis, and Lebeck have demonstrated the effectiveness of managing the power-down mode.[5,9] These proposals target the kernel, compiler, or memory-controller level, and are independent of the DRAM architecture. However, although they are effective, they typically have a negative impact on performance.

Researchers have also proposed various memory architectures—such as Fujitsu's fast-cycle RAM (FCRAM)[10]—at both the system and circuit levels, to reduce the power footprint in the memory system. Bhattacharjee et al. have proposed interconnecting memory modules using H-trees.[11] This allows switching off portions of the memory system, thus saving significant power. However, although these architectures might effectively reduce power consumption, they require nonstandard physical changes to the system, thereby making them less likely to be adopted.

JEDEC (http://www.jedec.org) has standardized a low-power DRAM module that can be used with other standard parts, but its performance is greatly reduced, usually lagging a generation or more in speed behind typical DRAMs.[12] JEDEC has also implemented, at the protocol level, timing parameters that prevent excessive power consumption. For example, JEDEC introduced the *tFAW* and *tRRD* timing parameters in DDR2 to limit the current draw within a DRAM device by spreading out bank activations in time.[4]

References

1. V. Delaluz et al., ''Scheduler-Based DRAM Energy Management,'' *Proc. 39th Ann. Design Automation Conf.* (DAC 02), ACM Press, 2002, pp. 697-702.
2. V. Delaluz et al., ''Energy-Oriented Compiler Optimizations for Partitioned Memory Architectures,'' *Proc. Int'l Conf. Compilers, Architecture, and Synthesis for Embedded Systems,* ACM Press, 2000, pp. 138-147.
3. R. Saied and C. Chakrabarti, ''Scheduling for Minimizing the Number of Memory Accesses in Low Power Applications,'' *Proc. Workshop VLSI Signal Processing,* IEEE Press, 1996, pp. 169-178.
4. B. Jacob, S.W. Ng, and D.T. Wang, *Memory Systems: Cache, DRAM, Disk,* Morgan Kaufmann, 2007.
5. V. Delaluz et al., ''DRAM Energy Management Using Software and Hardware Directed Power Mode Control,'' *Proc. 7th Int'l Symp. High-Performance Computer Architecture* (HPCA 01), IEEE CS Press, 2001, pp. 159-169.
6. S. Irani, S. Shukla, and R. Gupta, ''Online Strategies for Dynamic Power Management in Systems with Multiple Power-Saving States,'' *ACM Trans. Embedded Computing Systems,* vol. 2, no. 3, 2003, pp. 325-346.
7. I. Hur and C. Lin, ''A Comprehensive Approach to DRAM Power Management,'' *Proc. 14th Int'l Symp. High-Performance Computer Architecture* (HPCA 08), IEEE CS Press, 2008, pp. 305-316.
8. P.R. Panda and N.D. Dutt, ''Low-Power Memory Mapping through Reducing Address Bus Activity,'' *IEEE Trans. Very Large Scale Integration (VLSI) Systems,* vol. 7, no. 3, 1999, pp. 309-320.
9. X. Fan, C. Ellis, and A. Lebeck, ''Memory Controller Policies for DRAM Power Management,'' *Proc. Int'l Symp. Low Power Electronics and Design,* ACM Press, 2001, pp. 129-134.
10. *128 M-Bit (4-Bank × 1 M-Word × 32-Bit) Single Data Rate I/F FCRAM,* part no. MB81ES123245-10, data sheet DS05-11440-2E, Fujitsu, 2006; http://www.datasheetdir.com/MB81ES123245-10+download.
11. S. Bhattacharjee and D.K. Pradhan, ''LPRAM: A Low Power DRAM with Testability,'' *Proc. Asia and South Pacific Design Automation Conf.* (ASP-DAC 04), IEEE Press, 2004, pp. 390-393.
12. L. Mason, ''Low Power DRAM Roadmap Faces Rocky Road and Fuzzy Guardrails,'' *Denali Memory Report,* blog, 27 June 2008; http://www.denali.com/wordpress/index.php/dmr/2008/06/27/low_power_dram_roadmap_faces_rocky_road_.

which wordline driver should be operating (Figure 6). The row and column decoders' functionality remains unchanged. The row address received from the RAS command drives the row decoder, which raises a single wordline, thereby selecting which row to move to the sense amplifiers. The column decoder uses the column address to determine which bits to read or write into the currently active row.

We've modified the sense amplifiers' timing controllers to use the row-division decoder's output to enable or disable sensing by gating their timing input with the row-division decoder's output, which requires a single extra gate per controller.[5] When the row-division decoder selects a particular row division, the respective sense amplifiers are active while the rest remain idle, thus avoiding wasting power through sensing the precharged, yet unchanged, values on the bitlines.

The costs involved in the additional row-division decoder depend on the desired

activation granularity. In the simplest case, in which a row is divided into two separate parts, the highest-order bit in the column address and a single inverter can activate the respective portions; such a situation will reduce the activation power dissipation by half. Table 1 outlines the costs involved when using a row-division decoder of varying degrees coupled with the control gates that it drives. Relative to the DRAM device's overall power consumption, the row-division decoder and additional gates dissipate a negligible amount of power (less than 1 percent, even in the most extreme case). The on-die area consumed by the decoder depends on the fabrication technology used to create the device. We created transistor-level schematics for the row-division decoder in Cadence Design Systems' Virtuoso, and we calculated the power consumption values using TSMC's 0.25-µm technology. Although 0.25-µm technology is clearly outdated, we used it as a conservative means of determining the added logic's power consumption. Even when modeling outdated technology, the power consumption from the added logic is insignificant compared to the rest of the memory system. With a smaller feature size and more advanced technology, power consumption from additional logic would be even less than the outlined values, thereby producing more desirable results.

Because the storage array within the DRAM device is divided into numerous smaller cores, our additional costs involved in physically partitioning the storage array for the proposed mechanism are negligible. This partitioning has already been implemented to prevent unnecessarily long word- and bitlines. The modifications required to implement fine-grained activation are independent of the physical architecture of the storage array within a DRAM.

If the memory controller doesn't have access to, or isn't issuing, posted-CAS commands, the device will still function properly. The row-division decoder is only enabled upon receiving a posted-CAS command; otherwise, it's disabled, placing all output lines high and thereby driving all divisions of the row. Operating under this condition is identical to the original DRAM device architecture and ensures maximum interoperability with other standardized parts.

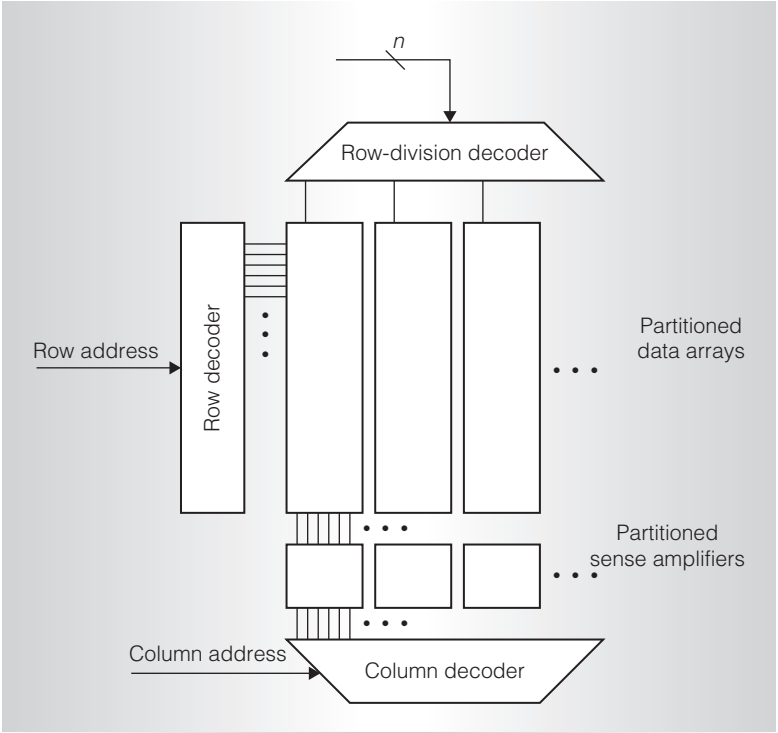

Figure 5. Our proposed DRAM device architecture. In this architecture, we added both a one-hot decoder (the row-division decoder) and a single AND gate to the wordline of each division in a row.

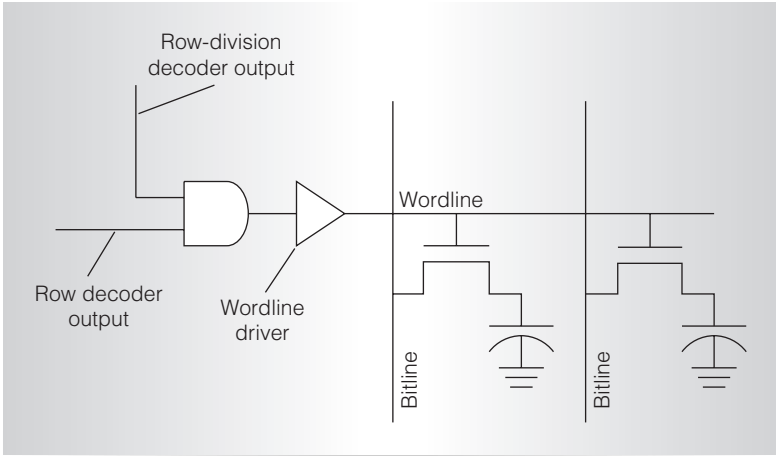

Figure 6. The selection mechanism, which includes a single AND gate. This AND gate uses the input to the wordline driver and the row-division decoder to determine which wordline driver should be operating.

The fine-grained activation architecture will work only with a closed-page row-buffer management policy. An open-page

Table 1. Costs for various decoders and control gates.

Row-division decoder	Power (μW)
1 to 2	8.9
2 to 4	85.2
3 to 8	89.4
4 to 16	216.3
5 to 32	371.9

management policy takes advantage of an entire row being active to satisfy subsequent read or write requests to the same row. This wouldn't be possible with the proposed architecture, because only a portion of the row is activated with each request. The posted-CAS command isn't intended for use with an open-page management policy, so this isn't a limitation of our approach. The additive delay used to postpone a CAS command cannot be turned off at the command level. When additive latency is enabled, all CAS commands are delayed by the amount specified by *tAL*. Therefore, using the posted-CAS command with an open-page management policy encounters an unnecessary latency penalty.[10]

DRAM power simulation

To characterize the power dissipation of our architecture, we modified DRAMsim to provide the ability to specify the granularity at which a row is activated. We used SPEC benchmarks to show the benefits of various activation granularities under different workloads. The test system consisted of single-, dual-, and quad-core CPUs running at 2.4 GHz with either a 256-Kbyte or 4-Mbyte cache, and a single 2-Gbyte rank of storage consisting of 16 ×4 1-Gbyte DRAM devices operating at 400 MHz (800 Mbps). We tested a wide variety of configurations to demonstrate the memory use patterns created by different system setups and the resulting impact of implementing fine-grained activation.

The memory controller placed the DRAM devices into power-down mode when all queues were empty, thereby reducing background power during inactive periods. We used several address-mapping schemes to determine a request's physical location. Varying the address-mapping scheme has a small, but noticeable, impact on each power component's relative size. For example, an address-mapping scheme that exploits the parallel nature of the internal banks will have a larger activation power component relative to the background power compared to a scheme that maps all requests to a single bank.

The DRAM power consumption model and timing constraints are based on Micron Technologies' MT47H256M4-25 device.[3] During operation, the simulation properly adhered to all timing constraints, and we calculated the power consumption with a V_{DD} of 1.9 V. In the typical JEDEC style, using 16 of these devices results in a single 2-Gbyte rank of storage with a 64-bit-wide data bus.

Results

With the introduction of fine-grained activation, the memory system experienced a power reduction of 9 to 33 percent for single-core processors, and 11 to 41 percent for quad-core processors. The determining factor in the reduction was the activation power component's size relative to the total power dissipation. A memory system under light use, with numerous cycles between each activation won't experience much benefit, because most of the power consumed is from other power components, such as the power dissipated when operating the control logic or refreshing the data array's rows. On the other hand, a saturated memory system will experience a far greater benefit, because most of its power is dissipated during activations. The variations in benefits observed between each situation were almost entirely based on how active the memory system was during execution.

For a single-core processor running at 2.4 GHz (Figure 7), we observed power reductions between 9 and 33 percent, depending on the activation granularity. We observed the greatest benefit during execution of the equake benchmark, which simulates seismic activity. Incidentally, this benchmark has the highest bus utilization with respect to the other benchmarks tested, and therefore places the greatest load on the memory system. As a comparison, the equake benchmark dissipates 1.5 W activating the data array's

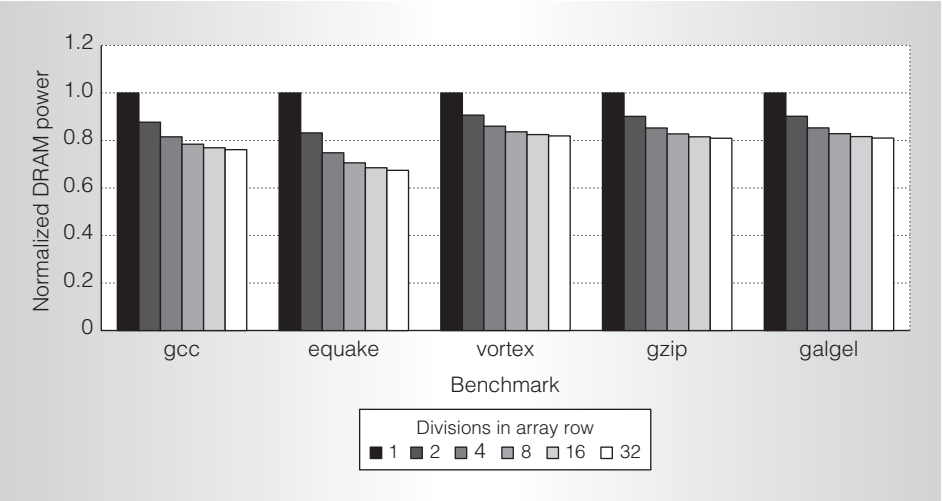

Figure 7. A single-core 2.4-GHz processor with 2-Gbyte DDR2 (double data rate two) DRAM devices operating at 400 MHz, for various activation granularities.

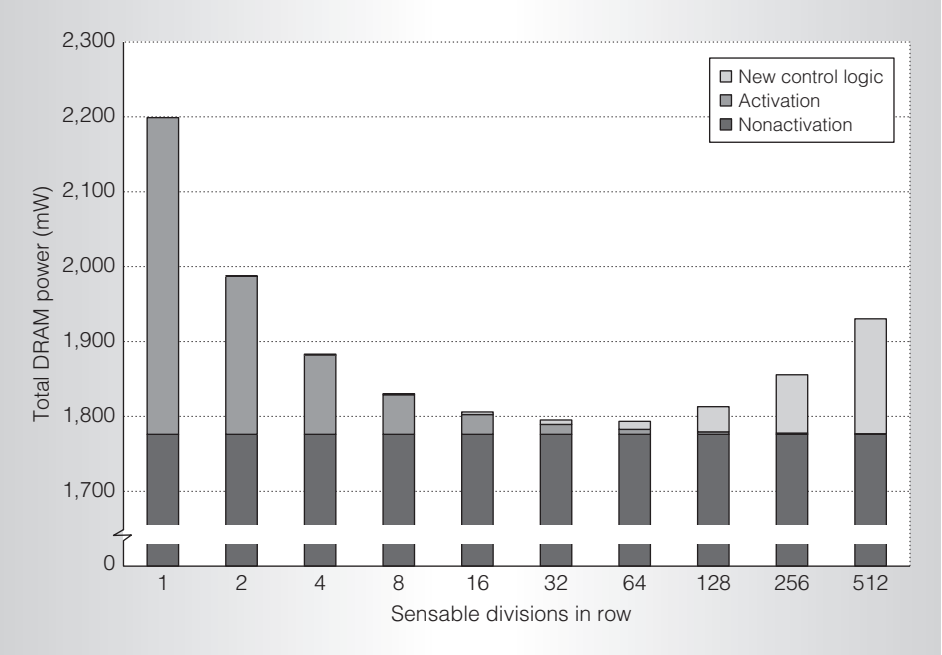

Figure 8. Example of costs of extraneous decoder logic while executing the galgel benchmark on a 2.4-GHz processor with 2-Gbyte DDR2 DRAM devices operating at 400 MHz.

rows from a total of 4.6 W (32 percent), whereas gzip, the least-active benchmark, dissipates 0.4 W activating rows of 2.3 W total (17 percent).

Intuition suggests that the costs of partitioning and selecting divisions in a row will, at some point, begin to impact the total reduction observed with the new architecture. The transistor-level simulations show that small-to-medium row-division decoders dissipate power of less than 1 percent of the total, thereby making the decoder's cost virtually unnoticeable (see Table 1). However, as Figure 8 demonstrates, if we increase the decoder size beyond the point of diminishing returns, its cost does

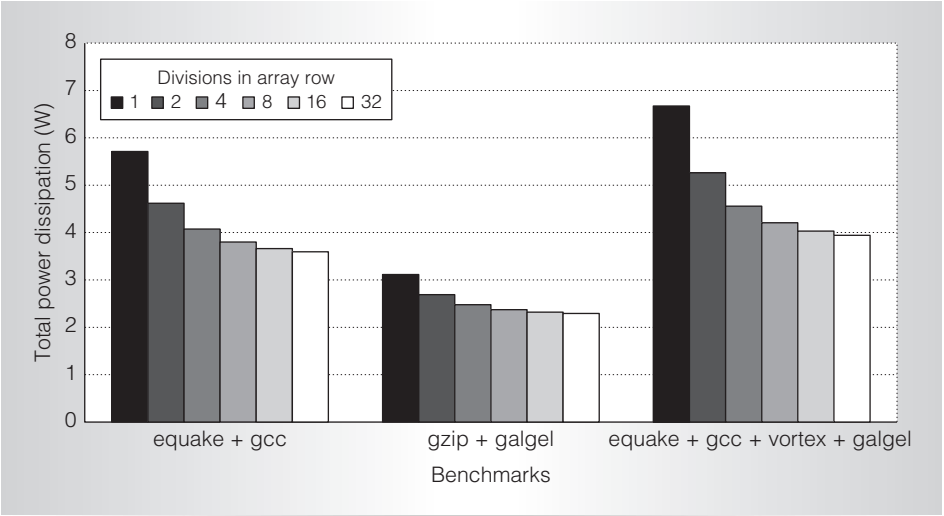

Figure 9. Dual- and quad-core 2.4-GHz CPUs with 2-Gbyte DDR2 DRAM devices operating at 400 MHz, running SPEC benchmarks on each core while varying activation granularity.

indeed become noticeable. (Note that the vertical axis in Figure 8 is not zero-oriented.)

Figure 9 shows the result of increasing the total number of CPU cores while using the fine-grained architecture. The total power dissipation within the memory system is greatest with a quad-core CPU, but the greatest benefit of using our architecture occurs in this case as well. This is because the memory system is least idle when requests are coming from four separate threads, further showing that the greater the activity is in the memory system, the greater the benefit will be when using the proposed architecture. Nevertheless, although this benefit is significant, all systems employing DRAM within the memory system will experience some benefit; it's only a matter of how much.

Different total cache sizes and block sizes will place varying loads on the memory system.[1,11] Figure 10 gives examples of how different cache configurations can impact the benefit of using a fine-grained activation architecture. Again, the determining factor in the reduction of power observed is the memory system's usage pattern. A smaller cache will place a greater load on the memory system, given its higher miss rate relative to a larger cache. Conversely, larger block sizes will amortize the costs of accessing the DRAM array over more data transferred. Although Figure 10a suggests that a larger total cache size has a greater benefit on DRAM power dissipation, Figure 10b suggests otherwise; the results are clearly application dependent.

This data suggests that, in certain cases, simply increasing the total cache size or cache block size can provide similar, if not greater, benefits in power reduction in the memory system. Although this could decrease DRAM power dissipation, increasing the total size or block size will greatly increase the power dissipated by the cache, thereby making total system power reductions negligible, if not increasing system power altogether. In a real system, regardless of the total size and the cache block size, misses occur and requests are sent to the DRAM, making the proposed architectural changes an effective addition to any system that uses DRAM in the memory system.

Changing the address-mapping scheme used by the memory controller affects memory system power dissipation and the ratio between each of the power breakdown components. The disparity between total power consumption while using each scheme is a result of the mapping's exploitation of the parallel nature of multiple banks within the DRAM

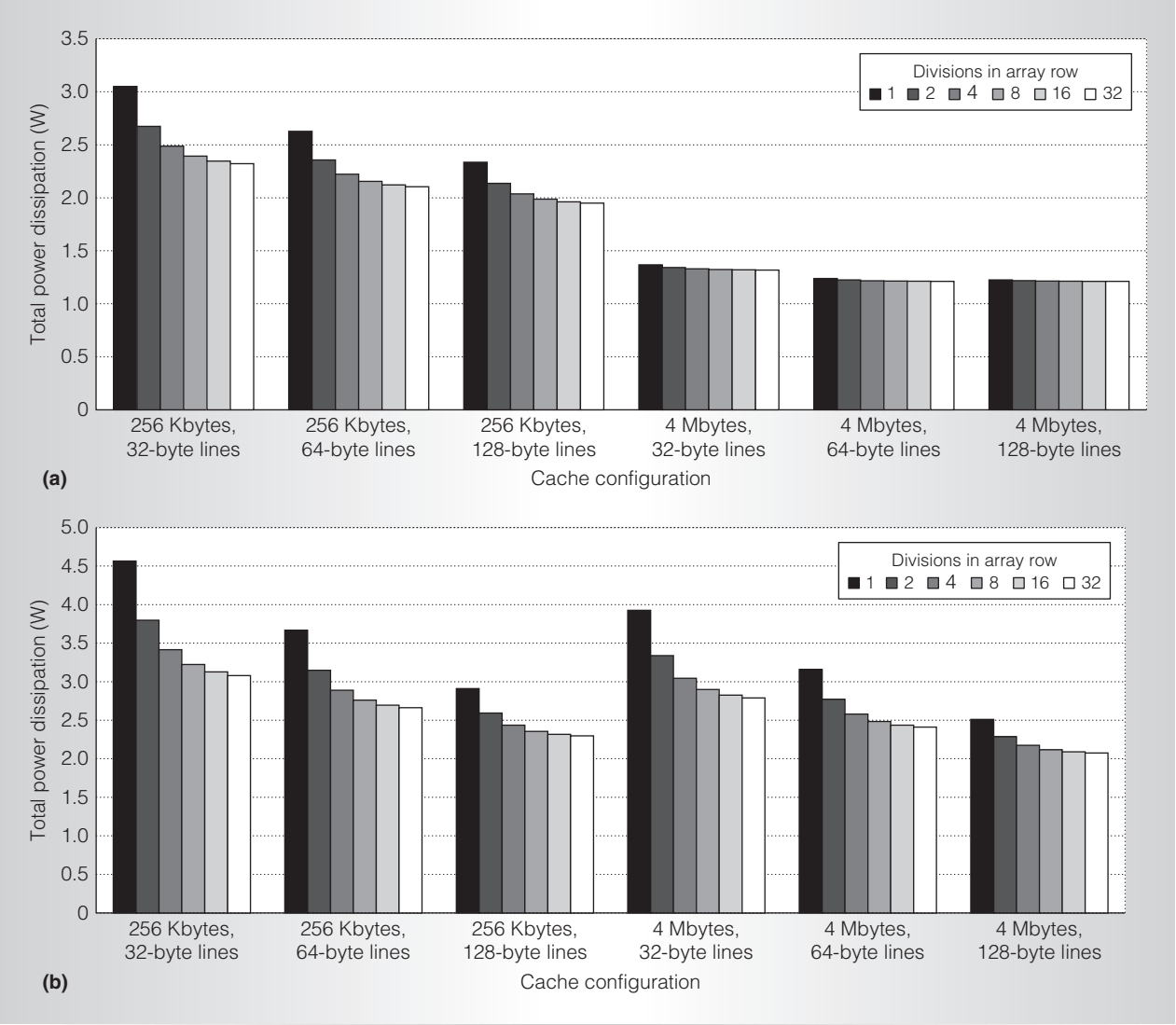

Figure 10. Variations in benefit when changing the cache configuration while running the gcc benchmark (a) and the equake benchmark (b), on a 2.4-GHz processor with 2-Gbyte DDR2 DRAM devices operating at 400 MHz.

device. For example, four successive requests might all be issued to a single bank with one address-mapping scheme or to each of the independent banks using another scheme. In the latter situation, background power would account for less of the total power dissipated while satisfying the four requests and could be placed into power-down mode sooner than in the first situation. Figure 11a shows four different address-mapping schemes. Figure 11b compares the power dissipation of these four schemes. These address-mapping schemes range from a combination of commercially used translations (Scheme A) to mappings created solely for DRAMsim (Scheme D).

The frequency of activations in the memory system is the determining factor regarding the benefit from implementing this new architecture. Systems such as servers, whose memory access patterns have a low degree of locality, will have a higher frequency of activations and, therefore, will benefit the most. In the end, all systems that use DRAM in their memory system will see some reduction in power dissipation with this new architecture; the only issue is how much. MICRO

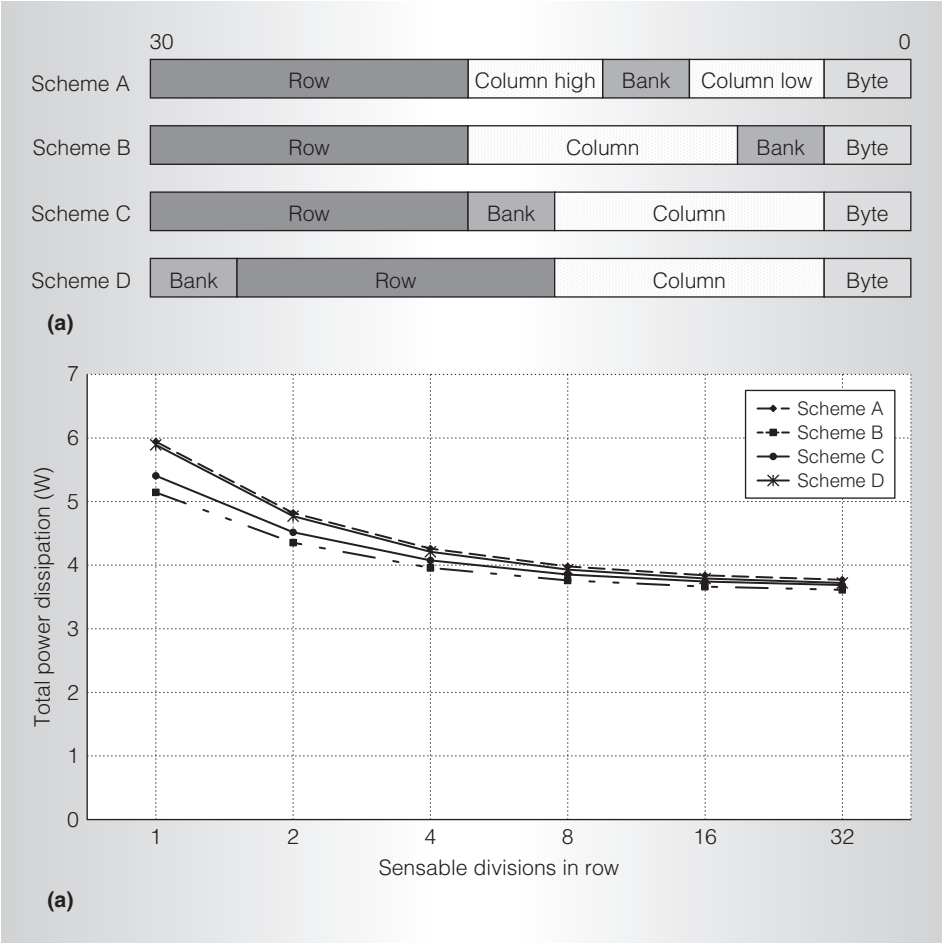

Figure 11. Comparison of four different address-mapping schemes of 2 Gbytes of space used by the simulator (a) and the impact of these schemes on power dissipation while using a fine-grained architecture (b).

References

1. B. Jacob, S.W. Ng, and D.T. Wang, *Memory Systems: Cache, DRAM, Disk,* Morgan Kaufmann, 2007.
2. D. Wang et al., ''DRAMsim: A Memory-System Simulator,'' *ACM SIGARCH Computer Architecture News,* vol. 33, no. 4, 2005, pp. 100-107.
3. *DDR2 SDRAM,* part no. MT47H256M4, data sheet, Micron Technologies, 2004; http://download.micron.com/pdf/datasheets/dram/ddr2/1GbDDR2.pdf.
4. R.J. Baker, *CMOS Circuit Design, Layout, and Simulation,* 2nd ed., Wiley-IEEE Press, 2005.
5. B. Keeth et al., *DRAM Circuit Design: Fundamental and High-Speed Topics,* 2nd ed., Wiley-IEEE Press, 2007.
6. ''Calculating Memory System Power for DDR2,'' tech. note TN-47-04, Micron Technologies, 2005; http://download.micron.com/pdf/technotes/ddr2/tn4704.pdf.
7. *128 M-Bit (4-Bank × 1 M-Word × 32-Bit) Single Data Rate I/F FCRAM,* part no. MB81ES123245-10, data sheet DS05-11440-2E, Fujitsu, 2006; http://www.datasheetdir.com/MB81ES123245-10+download.
8. R. Saied and C. Chakrabarti, ''Scheduling for Minimizing the Number of Memory Accesses in Low Power Applications,'' *Proc. Workshop VLSI Signal Processing,* IEEE Press, 1996, pp. 169-178.
9. L. Mason, ''Low Power DRAM Roadmap Faces Rocky Road and Fuzzy Guardrails,'' *Denali Memory Report,* blog, 27 June 2008;

http://www.denali.com/wordpress/index.php/dmr/2008/06/27/low_power_dram_roadmap_faces_rocky_road_.
10. B. Davis, B. Jacob, and T. Mudge, ''The New DRAM Interfaces: SDRAM, RDRAM, and Variants,'' *Proc. High Performance Computing,* LNCS 1940, Springer, 2000, pp. 26-31.
11. V. Cuppu and B. Jacob, ''Concurrency, Latency, or System Overhead: Which Has the Largest Impact on Uniprocessor DRAM-System Performance?'' *Proc. 28th Ann. Int'l Symp. Computer Architecture* (ISCA 01), ACM Press, 2001, pp. 62-71.

Elliott Cooper-Balis is pursuing a PhD in computer engineering at the University of Maryland, College Park. His research interests include next-generation memory systems, with a focus on architectural simulation. He has an MS in computer engineering from the University of Maryland, College Park.

Bruce Jacob is a professor at the University of Maryland, College Park. His research interests include memory-system design, DRAM architectures, virtual-memory systems, and microarchitectural support for real-time embedded systems. He has a PhD in computer science and engineering from the University of Michigan, Ann Arbor. He is a member of the IEEE Computer Society, IEEE, and the ACM.

Direct questions and comments about this article to Elliott Cooper-Balis, 1416 A.V. Williams Building, ECE Dept., University of Maryland, College Park, MD 20742; ecc17@umd.edu.