

Tapping ZettaRAM™ for Low-Power Memory Systems

Ravi K. Venkatesan, Ahmed S. AL-Zawawi, Eric Rotenberg
ECE Department, Center for Embedded Systems Research, North Carolina State University
{rkvenkat, aalzawa, ericro}@ece.ncsu.edu

Abstract

ZettaRAM™ is a new memory technology under development by ZettaCore™ as a potential replacement for conventional DRAM. The key innovation is replacing the conventional capacitor in each DRAM cell with “charge-storage” molecules – a molecular capacitor. We look beyond ZettaRAM’s manufacturing benefits, and approach it from an architectural viewpoint to discover benefits within the domain of architectural metrics.

The molecular capacitor is unusual because the amount of charge deposited (critical for reliable sensing) is independent of write voltage, i.e., there is a discrete threshold voltage above/below which the device is fully charged/discharged. Decoupling charge from voltage enables manipulation via arbitrarily small bitline swings, saving energy. However, while charge is voltage-independent, speed is voltage-dependent. Operating too close to the threshold causes molecules to overtake peripheral circuitry as the overall performance limiter. Nonetheless, ZettaRAM offers a novel speed/energy trade-off whereas DRAM is inflexible, introducing new dimensions for architectural management of memory.

We apply architectural insights to tap the full extent of ZettaRAM’s power savings without compromising performance. Several factors converge nicely to direct focus on L2 writebacks: (i) they account for 80% of row buffer misses in the main memory, thus most of the energy savings potential, and (ii) they do not directly stall the processor and thereby offer scheduling flexibility for tolerating extended molecule latency. Accordingly, slow writes (low energy) are applied to non-critical writebacks and fast writes (high energy) to critical fetches. The hybrid write policy is combined with two options for tolerating delayed writebacks: large buffers with access reordering or L2-cache eager writebacks. Eager writebacks are remarkably synergistic with ZettaRAM: initiating writebacks early in the L2 cache compensates for delaying them at the memory controller. Dual-speed writes coupled with eager writebacks yields energy savings of 34% (out of 41% with uniformly slow writes), with less than 1% performance degradation.

1. Introduction

ZettaRAM is a new memory technology under development by the startup ZettaCore as a potential

replacement for conventional DRAM [18]. ZettaCore’s strategy is to initially leverage the large investment in silicon fabs to attain competitive memories within a few years. Accordingly, these new memories are based on conventional DRAM architectures – address decoder, wordline, access transistor, bitline, sense amp, etc. The key innovation is replacing the conventional capacitor in each DRAM cell with a new type of capacitor, which had its genesis in a DARPA-sponsored molecular electronics project [14]. Although one goal of that project was to eventually deploy individual charge-storage molecules as 1-bit memory elements, and integrate them with other molecular-scale electronics, ZettaCore currently exploits many charge-storage molecules in aggregate to create a molecular capacitor and replacement for the conventional DRAM capacitor.

Nonetheless, the aggregate molecular capacitor retains key advantages of the underlying nanotechnology from which it is derived.

1. First, in a process called *self-assembly*, the thousands of molecules that make up a molecular capacitor automatically arrange themselves into a single, uniform, dense layer (monolayer). Moreover, the charge density of the molecular capacitor is greater than the charge density of a conventional capacitor. These two factors – self-assembly and high charge density – hold the key to increasing the density of DRAM in a cost-effective manner. While conventional DRAM will certainly scale, the costs are potentially astronomical due to the need for many complex masks for constructing elaborate three-dimensional capacitor structures. These complex structures are needed in order to reduce the cell area, while maintaining an amount of charge that can be sensed. On the other hand, the aggregate molecular capacitor does not require complex masks because the same amount of charge (or more) can be packed into the desired area via a self-assembled monolayer.
2. Second, the new molecular capacitor benefits from “molecular engineering” – engineering the properties of constituent molecules with remarkable precision. Synthetic chemists can precisely tune key properties of the molecules through the choice of molecular “groups” and “linkers”, such as the speed with which electrons can be added/removed (affecting the speeds of reading

and writing), the voltage at which electrons can be added/removed (affecting read and write power consumption), retention time (how slowly charge decays), and monolayer density (affecting charge density and thus overall memory density). Engineering a molecule is highly precise, predictable/repeatable, and can be done in inexpensive laboratories, whereas tuning bulk properties of semiconductors is expensive and subject to many environmental factors. And there is tremendous flexibility in the selection of performance (by way of electron transfer rates), power consumption (by way of oxidation/reduction potentials), and other attributes of molecular capacitors, although there are certainly tradeoffs among these attributes.

In this paper, we show that the benefits of ZettaRAM extend beyond reducing fab complexity and costs. We approach the new technology from a computer architecture perspective, and find that managed ZettaRAM can operate with significantly lower power than contemporary DRAM, without sacrificing performance.

Bitline energy consumption can constitute up to 96% of overall energy consumption in DRAM [9]. Bitline energy is consumed when there is a voltage transition on the bitline, and depends on the magnitude of the voltage change. The voltages for charging (writing a “1”) and discharging (writing a “0”) are closer together for a molecular capacitor than for a conventional capacitor. This means the voltage swings in ZettaRAM are smaller than the voltage swings in conventional DRAM, for the same charge density (i.e., the same amount of charge deposited within the same cell area).

We highlight this distinction in Figure 1 and Figure 2. The graph in Figure 1 shows charge density (charge per unit area) as a function of write voltage, for a conventional capacitor used in DRAM. (Using charge per unit area ensures comparisons are with respect to the same cell area.) The amount of charge deposited on a conventional capacitor depends linearly on the applied write voltage ($Q=CV$). Thus, there is a minimum write voltage, below which not enough charge is deposited on the conventional capacitor for the sense amplifier to reliably detect a “1” during a later read operation. The minimum charge density for reliable sensing is shown with the dashed horizontal line superimposed on the graph in Figure 1. Based on where this line intersects the conventional capacitor, the minimum voltage for writing a “1” is around 1.25 V.

Thus, the overall voltage differential between fully charging and discharging a conventional capacitor is 1.25 V, as shown in Figure 1. The actual magnitude of bitline transitions depends on the nature of consecutive operations, of which there are three types – read, write 0, and write 1. In Section 3.4, we analyze individual transitions in depth for conventional DRAM and

ZettaRAM. Nonetheless, the overall voltage differential is a key predictor of energy consumption.

The graph in Figure 2 shows charge density as a function of write voltage, for a molecular capacitor used in ZettaRAM. The relationship is nonlinear and centers around a voltage called V_{ox} , the oxidation potential. When the write voltage is above V_{ox} , the molecules are charged. Conversely, when the write voltage is below V_{ox} , the molecules are discharged. Notice, to write a “1”, sufficient charge is deposited when the applied voltage is only slightly above V_{ox} . Likewise, to write a “0”, the molecular capacitor can be completely discharged at voltages only slightly below V_{ox} . The minimum voltage differential between full charging and full discharging is quite small, reducing the magnitude of bitline voltage swings. However, charging/discharging the molecules becomes exponentially slower, the closer the applied voltage is to V_{ox} . In other words, ZettaRAM presents a new performance/energy tradeoff that conventional DRAM simply cannot provide. As shown in Figure 2, the voltage differential can be expanded or contracted to favor either performance or energy, respectively.

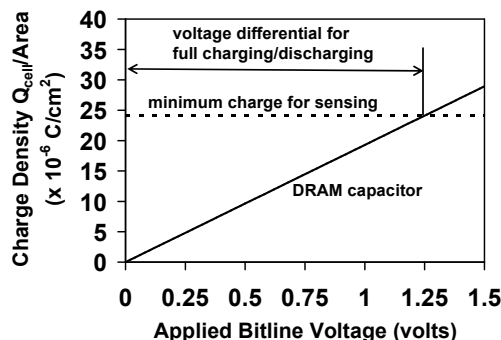


Figure 1. Charge density vs. voltage for a conventional capacitor in DRAM.

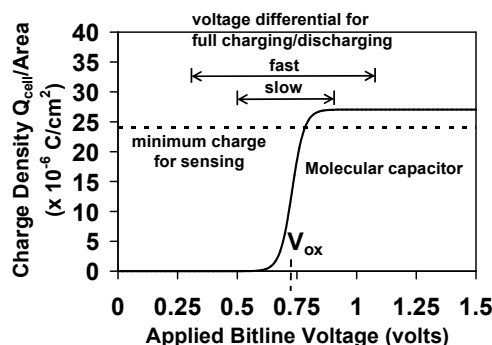


Figure 2. Charge density vs. voltage for a molecular capacitor in ZettaRAM.

To complete the overall picture, the graph in Figure 3 shows the intrinsic latency of charging/discharging the molecules as a function of voltage. The latency increases exponentially as the voltage approaches V_{ox} . Superimposed on this graph is the DRAM write latency

(dashed line). Our SPICE simulations, presented later, show that the overall speed of ZettaRAM is determined by either the speed of charging/discharging the molecules or the conventional peripheral circuitry used to access the molecular capacitor, whichever is slower. Accordingly, as shown in Figure 3, we can achieve the same performance as conventional DRAM if we use write voltages above 1.15 V (write “1”) and below 0.35 V (write “0”), since the intrinsic speed of the molecules is not the bottleneck in these regions. (Although we only described writing so far, reading is tantamount to writing a “0”. The molecular capacitor is read by discharging it, similar to reading a conventional capacitor.)

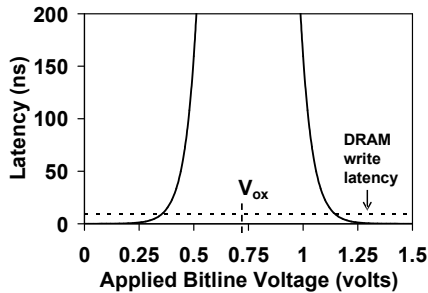


Figure 3. Intrinsic latency of charging and discharging molecules vs. voltage.

While Figure 2 shows smaller voltage swings are possible for ZettaRAM, Figure 3 shows latencies increase disproportionately. In this paper, we evaluate memory system policies for tapping most of the energy savings potential without sacrificing system-level performance.

First, a hybrid policy is applied in which slow writes delay non-critical requests and fast writes expedite critical requests. Each memory bank queues pending cache block fetch requests and cache block writeback requests, received from the L2 cache. A request must be serviced from an open memory page – an entire row of the memory bank held in the row buffer. Thus, if the needed page is not open, then the current open page must first be closed (write operation) before opening the needed page (read operation). The hybrid policy is applied to the current page as it is closed (written back to the memory array). If the L2 request that misses in the row buffer is a writeback, then the current page is closed via a slow write (low energy). L2 writebacks do not directly stall the processor, thus they are non-critical and can be deferred by slowly closing the current page, conserving energy. On the other hand, if the L2 request that misses in the row buffer is a fetch, then the current page is closed via a fast write (high energy). L2 fetches typically stall the processor, even with out-of-order execution, because the instruction scheduling window is not large enough to compensate for the memory round-trip latency.

Interestingly, writeback requests are responsible for most of the misses in the row buffer, i.e., more pages are closed on behalf of writeback requests (78% of closed

pages) than fetch requests (22% of closed pages). Therefore, conserving energy only when closing pages on behalf of writeback requests achieves most of the energy savings potential, as we will show (34% savings vs. 41% potential savings).

Moreover, because writebacks offer scheduling flexibility, there is room to explore other design parameters. Although deferred writebacks do not directly stall the processor, they can fill up the memory controller’s request queues, potentially stalling critical fetch requests. Accordingly, we investigate the effect of queue size and memory access reordering (fetch requests bypass queued writeback requests). We also explore the eager writeback policy [11] in the L2 cache, for evenly spreading out writeback requests and thereby decreasing the frequency of queue-full stalls. Interestingly, eager writebacks have little effect in the baseline system (for the SPEC2K benchmarks used), but are surprisingly effective for eliminating performance degradation otherwise caused by delayed writebacks.

Below, we summarize key results within the context of three alternative high-performance designs.

1. A ZettaRAM memory system employing only fast writes yields the same system-level performance and energy as the baseline DRAM memory system.
2. A ZettaRAM memory system employing slow writes for pages closed by non-critical requests achieves 34% bitline energy savings with less than 1% performance degradation, if the request queues are increased from 4 entries to 64 entries and memory access reordering is used. However, enlarging the queues increases system cost (each entry contains an entire cache block) and complexity. Regarding complexity, fetch requests that bypass queued writeback requests (memory access reordering) must first search the queue for possible address matches.
3. A ZettaRAM memory system employing slow writes for pages closed by non-critical requests achieves 34% bitline energy savings with less than 1% performance degradation, if the L2 cache uses the eager writeback policy [11]. This is achieved without enlarging request queues with respect to the baseline system. This is significant in terms of keeping the cost and complexity of the memory controller the same as the baseline. The eager writeback policy is simple to implement: A dirty block is written back as soon as it becomes the least-recently-used (LRU) block in the set. LRU information is already available in the set-associative L2 cache.

The rest of this paper is organized as follows. Section 2 provides background on the molecular capacitor, including basic read/write operation, our novel SPICE device model, and our novel derivation of charge density as a function of write voltage. Section 3 presents SPICE results, namely, read/write latencies and operating

voltages for both ZettaRAM and DRAM. Section 4 describes our experimental framework for system-level experiments. Results are presented in Section 5. Related work is discussed in Section 6. Finally, Section 7 summarizes the paper and discusses future work.

2. Molecular Capacitor

2.1. Molecule Description and Reading/Writing the Molecular Capacitor

A ZettaRAM memory cell is identical to a conventional DRAM memory cell except the conventional capacitor is replaced with a new capacitor-like device based on a self-assembled monolayer (SAM) of porphyrin molecules sandwiched between two electrodes. An individual porphyrin molecule is shown in Figure 4. The molecule can be positively charged by removing a single electron from the redox-active part of the molecule, referred to as *oxidation*. As such, oxidation corresponds to “writing” a logic 1. An electron can be added back to the positively charged molecule to return it to the uncharged state, referred to as *reduction*. Reduction corresponds to “writing” a logic 0.

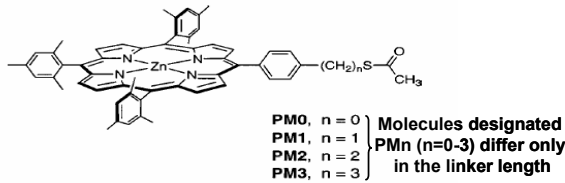


Figure 4. Single porphyrin molecule.

The molecule is oxidized when the voltage applied across the molecule is greater than the oxidation potential, 0.73 V for the molecule type used in this paper. It is reduced when the applied voltage is below the oxidation potential. In truth, oxidation and reduction are always taking place simultaneously – any chemical reaction is a combination of forward and reverse reactions. Equilibrium is reached, at which point the rates of the forward and reverse reactions are equal. Although the rates are balanced at equilibrium, the molecule has a strong tendency towards either the oxidized state or reduced state, depending on the applied voltage (above or below the oxidation potential, respectively).

Like reading conventional DRAM, reading ZettaRAM is destructive. To read the state of the molecules in a molecular capacitor, they are discharged (if they are initially charged). This is achieved by reducing them, i.e., the bitline is precharged to a voltage below the oxidation potential. The state of the molecules is sensed by detecting the presence (or absence) of a small voltage change on the bitline as the molecules are discharged (unless neutral), which is procedurally similar to sensing in conventional DRAMs.

An idiosyncrasy of the molecular capacitor, with regard to reading, is that the bitline is precharged to a

specific voltage below the oxidation potential called the *open circuit potential* (OCP $\sim 0.2-0.3V$) [15]. The molecular capacitor is actually an electrochemical cell (like a battery) in which the redox species is the porphyrin molecules. The OCP is a well-known artifact of electrochemical cells. Reading at the OCP prevents discharging of the “double-layer capacitance”, an internal capacitance, which would otherwise drown out discharging of the molecules themselves.

2.2. SPICE Model of Molecular Capacitor

The oxidation/reduction reactions are shown below, where A is the porphyrin molecule [13].



In non-equilibrium (charging or discharging), the net rate of oxidation or reduction – i.e., the net current – is exponentially dependent on the difference between the applied voltage and the oxidation potential. This current is expressed by the Butler-Volmer kinetic model [1], shown below, and is the basis of our SPICE model.

$$I = F \cdot k^0 \cdot \left([A] \cdot e^{(1-\alpha)\left(\frac{F}{RT}\right)(V-V_{ox})} - [A^+] \cdot e^{-\alpha\left(\frac{F}{RT}\right)(V-V_{ox})} \right) \quad (\text{EQ 2})$$

Parameters above are as follows: k^0 = standard rate constant, α = transfer coefficient, F = Faraday constant, R = gas constant, T = temperature, V = applied voltage, V_{ox} = oxidation potential, [A] = concentration of non-oxidized molecules (in moles per unit area), and $[A^+] =$ concentration of oxidized molecules.

The current I determines the *intrinsic* speed of reading and writing the molecules. Of course, when we integrate a SPICE model of the molecular capacitor into a complete memory circuit, the overall speed will be determined by several interacting components. That is, like any SPICE device model (e.g., transistor, resistor, capacitor, etc.), when the device model of the molecular capacitor is integrated into a larger circuit, the SPICE simulator correctly solves for currents and voltages at all nodes, accurately reflecting the interaction between the molecular capacitor and the rest of the circuit.

Figure 5(a) shows the SPICE model of the molecular capacitor. The voltage-controlled current source implements EQ 2. The current depends on three variables, [A], $[A^+]$, and V.

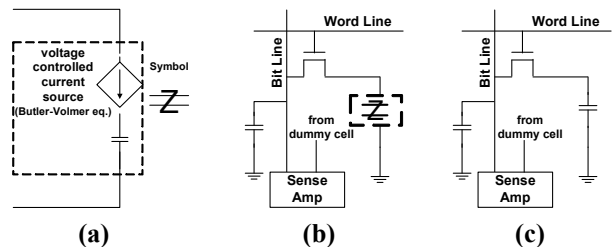


Figure 5. (a) SPICE device model of molecular capacitor. (b) ZettaRAM circuit. (c) DRAM circuit.

Figure 5(b) shows the SPICE model of the molecular capacitor integrated into a larger SPICE model of the ZettaRAM architecture, including bitline, wordline, access transistor, and sense amp.

Figure 5(c) shows the SPICE model of the DRAM architecture. The only difference between the ZettaRAM and DRAM SPICE models is the type of capacitor used inside the cell (molecular vs. conventional, respectively).

2.3. Highly Non-linear Capacitance: Charge Density Independent of Write Voltage

The oxidation/reduction reactions shown in EQ 1 eventually reach an equilibrium. The net current is zero at this equilibrium. We can derive the amount of charge ($Q_{cell} = [A^+]$) at equilibrium as a function of the write voltage, by substituting $I=0$ in the Butler-Volmer equation (EQ 2). (This gives us the effective capacitance of the molecular capacitor since capacitance expresses Q as a function of V .) Doing so yields the following $Q_{cell}(V)$.

$$Q_{cell}(V) = [A]_0 \cdot \left[\frac{1}{1 + e^{-\frac{F}{RT}(V - V_{ox})}} \right] \quad (\text{EQ 3})$$

$[A]_0$ is the total molecule concentration, equal to the sum of $[A]$ and $[A^+]$. EQ 3 is the basis for the unusual charge density graph shown earlier in Figure 2 of Section 1. The exponential term in the denominator becomes negligible as V is increased slightly above V_{ox} , such that the equilibrium charge deposited on the molecular capacitor is largely independent of write voltage, for voltages sufficiently higher than V_{ox} . This finding is significant. It means that the molecular capacitor can be nearly fully charged over a comparatively wide range of write voltages. Thus, write voltage can be tuned and perhaps even dynamically adjusted to manage the bitline-energy/write-speed tradeoff, without sacrificing the ability to sense the device since the deposited charge is the same independent of write voltage (only the speed of writing is affected by write voltage, as described earlier in Section 2.2). Conventional DRAM does not provide this flexibility because the amount of deposited charge depends linearly on the write voltage, so reliable sensing places a lower bound on the write voltage.

3. SPICE Results

In this section, SPICE simulations in a 0.18μ technology are used to determine (1) operating voltages and (2) read/write latencies, for both ZettaRAM and DRAM. We assume a 10:1 ratio between bitline capacitance and cell capacitance [8] and designed sense amps accordingly. Sense amp designs are based on [10].

3.1. DRAM

We first experimentally determine a lower bound on the DRAM write voltage (for writing a “1”), below which

not enough charge is deposited on the conventional capacitor for sensing. We call this lower bound on write voltage $V_{d_write_1}$. Searching in increments of 0.05 V, we determined $V_{d_write_1} = 1.25$ V. The graph in Figure 6 shows that writing the DRAM capacitor at 1.2 V causes sensing to fail during a later read operation, since there is too little charge.

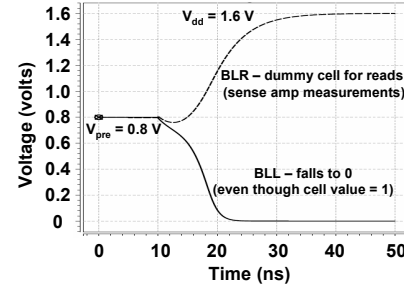


Figure 6. Writing DRAM capacitor below 1.25 V causes subsequent read operation to fail.

Next, we determine the write latency of DRAM. SPICE produces a write latency of 8.6 ns for $V_{d_write_1} = 1.25$ V. The graph is not included here, for lack of space.

Finally, we determine the read latency of DRAM. SPICE produces a read latency of 29 ns (see Figure 7).

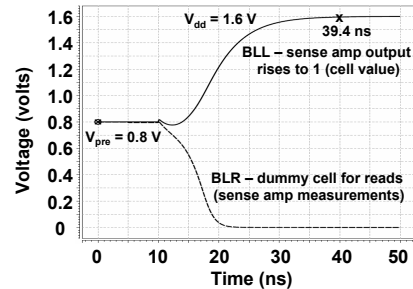


Figure 7. Read latency of DRAM is 29 ns.

3.2. ZettaRAM

In the previous subsection, we showed that the conventional capacitor of DRAM is not sufficiently charged below 1.25 V, from the standpoint of correct sensing during a later read operation. On the other hand, writing the molecular capacitor at a voltage as low as 1.0 V (and probably lower) results in correct sensing during a later read operation, as shown in Figure 8.

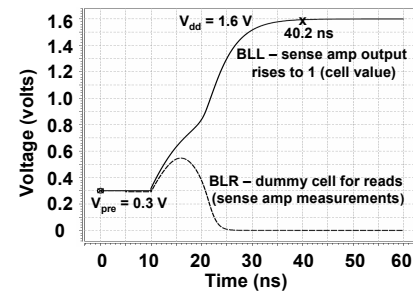


Figure 8. Writing molecular capacitor as low as 1.0 V subsequently results in correct sensing.

Next, we determine the write latencies of ZettaRAM as a function of the ZettaRAM write voltage, $V_{z_write_1}$. In the first experiment, we use the DRAM's minimum write voltage, $V_{d_write_1} = 1.25$ V. The ZettaRAM write latency at this voltage is 8.2 ns, similar to the DRAM write latency (8.6 ns) reported in the previous subsection. (We exclude the SPICE graph due to limited space.) This means that, for $V_{z_write_1} = V_{d_write_1}$, the conventional peripheral circuitry used to access the molecular capacitor is the speed limiter, not the intrinsic speed of the molecules.

The ZettaRAM molecular capacitor can be reliably written below 1.25 V, although the intrinsic speed of the molecules begins to limit overall write speed at lower voltages. The SPICE results in Figure 9 show increasing write latency with decreasing write voltage: 9 ns at 1.2 V, 29 ns at 1.1 V, and 166 ns at 1.0 V.

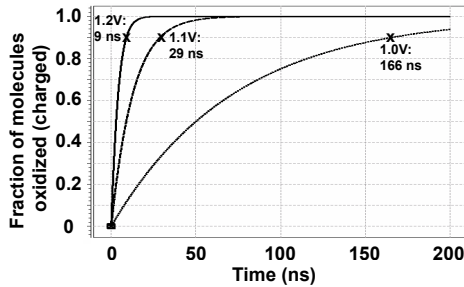


Figure 9. ZettaRAM write latency (90% of molecules oxidized) for three applied voltages.

Reading is competitive with conventional DRAM because the applied voltage is the OCP (0.3 V) which is sufficiently lower than V_{ox} (0.73 V), such that the molecule current is much faster than the sensing apparatus and thus does not limit the speed of reading. This is confirmed by SPICE simulations. The SPICE result in Figure 8 shows that the latency of reading ZettaRAM is 30 ns, similar to the read latency of DRAM (29 ns) measured in the previous subsection. Reading is procedurally similar for the conventional and molecular capacitors – it is based on sensing a small change in charge on the precharged bitline.

Reading the molecular capacitor is tantamount to writing “0”, since the read voltage is below V_{ox} , fully discharging the molecular capacitor. So far, we only discussed multiple write voltages for writing a “1”. For writing a “0”, we consider only a single write voltage equal to the read voltage. Incidentally, this is a fast write voltage. There is no point considering slower write “0” voltages between the read voltage and V_{ox} . Bitline operations always alternate between reading (open page) and writing (close page), so keeping the write “0” voltage the same as the read voltage eliminates many bitline transitions altogether. This will become clearer in Section 3.4, where we summarize bitline transitions for ZettaRAM and DRAM.

3.3. Retention Time

The retention times of the two technologies are comparable because leakage is an artifact of the access transistor, and the initial stored charge is the same. This is confirmed by the SPICE results shown in Figure 10. For example, at 40 ms, the conventional capacitor and molecular capacitor retain 32% and 51% of the initial charge, respectively. The molecular capacitor demonstrates an improved decay curve at the beginning. The retention time of both memories can be improved by applying a negative substrate bias, reducing the leakage current of the access transistor. What we want to demonstrate here is the comparable retention times of the two technologies.

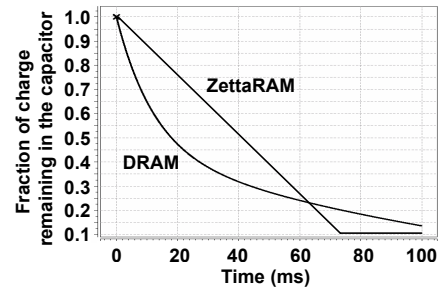


Figure 10. Retention times.

3.4. Comparison Summary and Voltage Transitions

Table 1 summarizes the important similarities and differences between conventional DRAM and ZettaRAM, in terms of operating voltages, read and write latencies, and capacitor area.

Table 2 shows bitline voltage transitions for DRAM and ZettaRAM. Because L2 requests are always serviced from the page held in the row buffer, bitline operations always alternate between reading (open page) and writing (close page). This yields only four valid transitions: read followed by write-0 or write-1, and write-0 or write-1 followed by read. The first row in the table shows the percentage breakdown of the four transitions. The benchmark *mcf* is shown (others show similar breakdowns). The second row shows the DRAM voltage differential for each transition, using the voltages derived in Section 3.1. Table entries for positive voltage transitions are highlighted, which we use in the energy accounting. Although the previous SPICE experiments used $V_{DD}=1.6$ V due to our available technology files (and a corresponding read precharge voltage of 0.8 V), for energy accounting we use $V_{DD}=V_{d_write_1}$. This adjustment minimizes DRAM energy, by applying a lower voltage differential for the higher percentage write-0→read transitions. The third and fourth rows show ZettaRAM voltage differentials, using either fast or slow writes ($V_{z_write_1_fast}=1.2$ V and $V_{z_write_1_slow}=1.0$ V). Because the write-0 and read voltages are the same (as

Table 1. Comparison of conventional DRAM and ZettaRAM attributes.

Characteristic	DRAM	ZettaRAM
Write Voltage	1.25 V (no lower)	0.73 V – 1.25 V
Area (0.18 μm technology)	0.1296 μm^2	0.1296 μm^2
Row access time (read an entire row)	29 ns	30 ns
Precharge time (write an entire row)	9 ns	function of applied voltage [9 ns @ 1.2V – 166 ns @ 1 V]
Column access time (select column)	16 ns	16 ns
Read precharge voltage	$V_{DD}/2$ (= 1.25/2 = 0.625 V)	OCP (= 0.3 V)

Table 2. Bitline voltage transitions for DRAM and ZettaRAM.

	Bitline Transition			
	read \rightarrow write 0	read \rightarrow write 1	write 0 \rightarrow read	write 1 \rightarrow read
% of all transitions, benchmark = <i>mcg</i>	28.46%	21.48%	28.48%	21.58%
Conventional DRAM ΔV	$-(V_{DD}/2)$ = -0.625	$+(V_{d_write_1}-V_{DD}/2)$ = 0.625	$+(V_{DD}/2)$ = 0.625	$-(V_{d_write_1}-V_{DD}/2)$ = -0.625
Fast ZettaRAM ΔV ($V_{z_write_1_fast} = 1.2$ V)	0	$+(V_{z_write_1_fast}-V_{ocp})$ = 0.9	0	$-(V_{z_write_1_fast}-V_{ocp})$ = -0.9
Slow ZettaRAM ΔV ($V_{z_write_1_slow} = 1.0$ V)	0	$+(V_{z_write_1_slow}-V_{ocp})$ = 0.7	0	$-(V_{z_write_1_slow}-V_{ocp})$ = -0.7
Baseline DRAM ΔV ($V_{read} = V_{d_write_0} = V_{ocp}$)	0	$+(V_{d_write_1}-V_{read})$ = 0.95	0	$-(V_{d_write_1}-V_{read})$ = -0.95

discussed in Section 3.2), two of the transitions incur no voltage change.

The lack of any write-0 \rightarrow read transitions gives ZettaRAM a substantial energy advantage over conventional DRAM. Conceivably, the same strategy of unifying the read potential and the write-0 potential may be applicable in future DRAMs. To level the playing field, we enhance the DRAM by lowering the read potential from $V_{DD}/2$ and raising the write-0 voltage from 0 V, both to V_{ocp} . (Like ZettaRAM, the enhanced DRAM sense amp senses logic “0” via the absence of a bitline shift.) This enhanced DRAM is the baseline for all architectural experiments. Voltage differentials for this baseline DRAM are shown in the last row of Table 2.

4. Experimental Framework

4.1. Memory Simulator: Modeling Timing

The interleaved ZettaRAM memory system, shown in Figure 11, is modeled after synchronous DRAM (SDRAM) [12].

The ZettaRAM memory system has 4 independent ports, with each port tied to a bank. The memory controller maps physical addresses to memory addresses (bank id, row id, and column id) and schedules pending memory requests. The memory controller maintains a separate queue of pending memory requests for each bank. There are two types of memory requests initiated by the L2 cache, fetch block and writeback block.

Memory access reordering is used by default. Fetch requests circumvent queued writeback requests unless there is an address match. Where indicated, we also

investigate configurations with memory access reordering disabled.

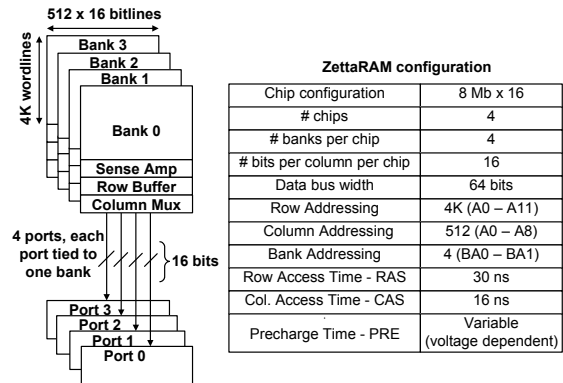


Figure 11. Interleaved ZettaRAM memory system

A ZettaRAM *page* is a row in memory that is read into the row buffer to service memory requests. The memory controller can use one of two different policies to manage pages – open page policy and close page policy. In the close page policy, a page is “closed” after servicing the memory request, i.e., the page is immediately written back into its memory array. In the open page policy, a page is left “open” after reading the page into the row buffer, i.e., the data is held in the row buffer (cached). By keeping the page open, subsequent accesses to the same page do not suffer the penalty of opening the page. However, if there is a request to a different page in the same bank, the open page policy suffers the penalty of closing the current page before opening the new page, thus sometimes increasing the wait time of fetch and writeback requests. Nonetheless, we find that open page

policy significantly outperforms close page policy, so we consider only open page policy in our simulations.

4.2. Memory Simulator: Modeling Energy

Bitline energy, i.e., energy required to charge the bitline when opening or closing a page, can constitute up to 96% of the total memory system energy [9]. Thus, in our experiments, we measure bitline energy consumption in the main memory. We track the voltage states of all bitlines in order to measure the energy required to charge the bitlines for a particular memory operation.

Assuming a single voltage supply (V_{DD}), the energy to charge a bitline is $E_{\text{bitline}} = C_{\text{BL}} \cdot V_{\text{DD}} \cdot (\Delta V_{\text{BL}}) = C_{\text{BL}} \cdot V_{\text{DD}} \cdot (V_{z_write_1} - V_{\text{ocp}})$. Thus, dynamically adjusting the write-1 voltage yields linear energy scaling. If we use a dedicated voltage supply for charging the bitline ($V_{z_write_1}$), then $E_{\text{bitline}} = C_{\text{BL}} \cdot V_{z_write_1} \cdot (V_{z_write_1} - V_{\text{ocp}})$. Now, dynamically adjusting the write-1 voltage yields quadratic energy scaling. In this paper, we assume dual voltage supplies for the dual write voltages ($V_{z_write_1_fast}$ and $V_{z_write_1_slow}$). The supplies can be implemented using high-efficiency DC-DC converters [3]. Dual voltages were implemented in drowsy caches and selected in one to two cycles via a MUX [6], a technique we borrow.

The analytical model $C_{\text{BL}} \cdot V_{\text{DD}} \cdot (\Delta V_{\text{BL}})$ is derived by integrating power across the voltage supply ($V_{\text{DD}} \times I$), which yields the overall energy consumed, as opposed to integrating power across only the bitline capacitor ($V_{\text{BL}} \times I$). The analytical model was compared against SPICE simulations, and they match exactly.

4.3. Cycle-level Simulator

Our memory simulator is integrated with a detailed cycle-level processor simulator. The SimpleScalar ISA (PISA) [2] and compiler (gcc-based) are used. The processor configuration is given in Table 3. The cache and bus configurations are based on the Pentium® 4 processor [7]. The L1 instruction and data caches each allow up to 32 outstanding misses. The L2 cache allows up to 8 outstanding fetch requests at a time. Increasing the number of L2 MSHRs beyond 8 provided only minor performance benefits. The maximum number of outstanding L2 writeback requests is only limited by the buffering in the memory controller.

Table 3. Processor configuration.

4-issue OOO superscalar, 7-stage pipeline	
Frequency	1 GHz
Reorder Buffer	128 entries
Issue queue, LSQ	64 entries
Branch predictor	gshare, 2^{16} entries
Functional units	4, universal
Bus	400 MHz 64-bit
L1 cache (split - I & D)	8 KB, 4-way, 64 B line size
L2 cache (unified)	256 KB, 8-way, 128 B line, writeback
Hit Latency	L1: 2 ns, L2: 10 ns
MSHRs	L1: 32, L2: 8

4.4. Benchmarks

We use eight different integer benchmarks from the SPEC2000 benchmark suite with reference inputs. We used SimPoint to determine the appropriate starting simulation point for each benchmark [16]. 100 million instructions are then simulated from this simulation point. The SimPoints chosen for each benchmark are shown in Table 4. Table 4 also shows the rates of L1 and L2 cache misses (per 1000 instructions) and L2 writebacks (per 1000 instructions) to main memory for each benchmark.

Table 4. SPEC2000 benchmarks.

	SimPoint (billions of instr.)	L1 misses*	L2 misses*	writebacks*	writebacks that close page*
bzip	1	84.8	13.3	4.6	2.8
gap	209.5	87.8	4.2	1.8	1.2
gcc	11	98.8	9.6	3.13	2.4
gzip	48.7	97.0	4.7	1.91	1.5
mcf	31.7	208.6	80.3	31.84	23.8
parser	1.7	58.9	5.4	2.12	1.5
twolf	3.2	110.5	22.8	7.61	4.9
vortex	5.8	81.2	7.5	2.9	2.4

* per 1000 instructions

5. Results

5.1. DRAM Energy and Performance

Figure 12 shows (a) bitline energy consumption and (b) execution times, for DRAM operating at 1.25 V. Within the DRAM memory controller, the pending request queue for each bank is fixed at 4 entries. Memory access reordering is used in the baseline unless otherwise indicated. Since 1.25 V is the lowest reliable write voltage for DRAM, we use this system as our baseline and all ZettaRAM performance and energy measurements are normalized with respect to this baseline.

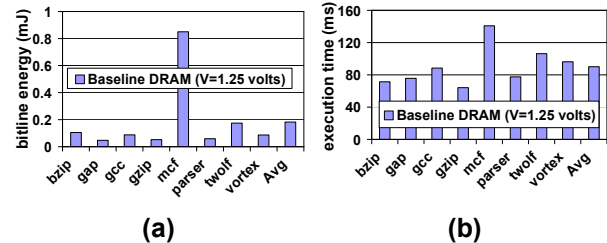


Figure 12. (a) Bitline energy consumption and (b) execution times, for DRAM operating at 1.25 V.

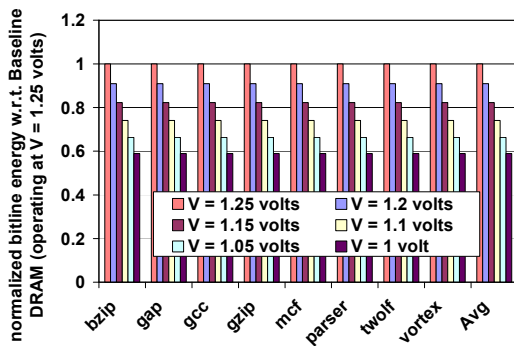
5.2. ZettaRAM Tradeoff Between Bitline Energy and System Performance

Next, we quantify the tradeoff between system performance and bitline energy as the ZettaRAM write voltage is manipulated. For each experiment, only a single fixed write voltage is used to close pages, regardless of the type of request causing a page to close.

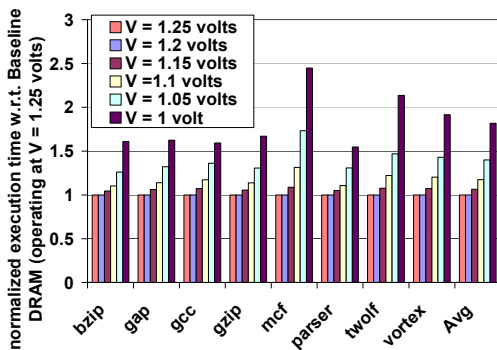
Figure 13 shows normalized (a) bitline energy consumption and (b) execution times, for ZettaRAM operating at fixed write voltages of 1.0 V through 1.25 V in 0.05 volt increments. At 1.25 V and 1.2 V, the execution times for ZettaRAM and the baseline DRAM are equal because the intrinsic speed of the molecules is fast enough above 1.2 V, such that the write latency is dictated by the peripheral circuitry. Thus, when we employ the hybrid policy later, we use 1.2 V as the voltage for *fast writes* (high energy). However, at lower voltages, overall write latency and thereby system performance is mainly determined by the intrinsic speed of the molecules.

From Figure 13(a), lowering the write voltage from 1.25 V to 1.0 V reduces bitline energy by 41%. However, as expected, execution time increases by 50-150%, as shown in Figure 13(b). This is because write latency increases exponentially with decreasing write voltage.

In the next subsections, we evaluate memory system policies for tapping most of the energy savings potential without sacrificing system-level performance.



(a) normalized bitline energy



(b) normalized execution time

Figure 13. ZettaRAM with various write voltages.

5.3. Hybrid Write Policy

Both fetch and writeback requests cause the current page to close when they miss in the row buffer. We propose handling these fetches and writebacks differently. Since fetch requests are timing critical, the current page should be closed using a fast write. On the other hand,

writebacks offer more scheduling flexibility because they do not directly stall the processor, so we propose closing the current page using a slow write in this case. A potential downside of this approach is less energy savings than employing slow writes uniformly.

Fortunately, most of the energy savings potential rests with writebacks that miss in the row buffer. The graph in Figure 14(a) shows that 71-82% of all closed pages are closed on behalf of writebacks that miss in the row buffer. Only 18-29% of all closed pages are due to fetches that miss in the row buffer. Writebacks exhibit significantly lower locality than fetches, with respect to the row buffer. Figure 14(b) shows that fetches hit 80-90% of the time, whereas writebacks hit only 18-40% of the time (29% on average). All of this implies that employing slow writes only when closing pages on behalf of writeback requests will probably achieve most of the energy savings potential. This is confirmed by experiments that follow.

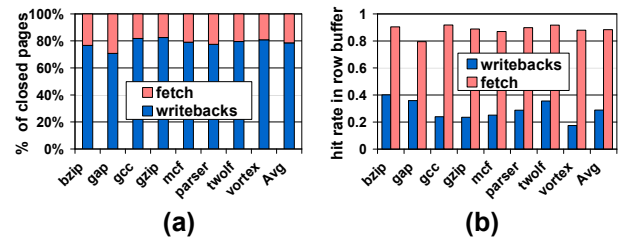


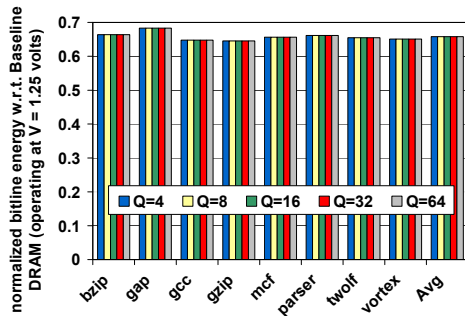
Figure 14. (a) Percentage of closed pages that are closed due to writebacks vs. fetches. (b) Row buffer hit rates for writebacks and fetches.

Although delayed writebacks do not directly stall the processor, they may fill the memory controller's request queues, stalling future fetch requests. Fortunately, writebacks offer scheduling flexibility. We investigate the effect of queue size for tolerating delayed writebacks, in combination with the default policy of memory access reordering (fetch requests bypass queued writeback requests). Fast and slow writes are done at 1.2 V and 1.0 V, respectively.

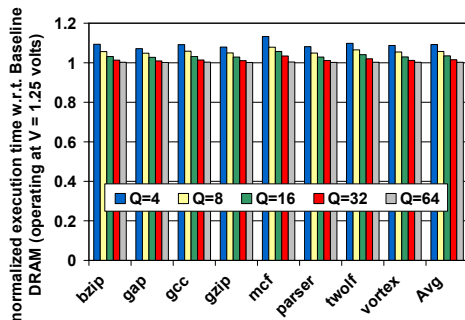
Figure 15 shows (a) bitline energy consumption and (b) execution times, for ZettaRAM using the hybrid write policy and queue sizes of 4, 8, 16, 32 and 64 entries. A ZettaRAM memory system employing slow writes for pages closed by non-critical requests (writebacks) achieves 34% bitline energy savings with less than 1% performance degradation, if the pending request queue contains 64 entries and memory access reordering is used. As predicted, most of the energy savings potential is tapped by focusing only on non-critical writeback requests: 34% savings on average (Figure 15a), compared to 41% savings when slow writes are applied uniformly (Figure 13a, V=1.0). The residual performance degradation at smaller queue sizes (4-32) can be attributed to an increase in the number of queue-full stalls with respect to the baseline DRAM, caused by delayed writebacks. Nonetheless, the performance degradation

with a queue size of 4 has been dramatically reduced, from an average slowdown of 80% with uniformly slow writes (Figure 13b, $V=1.0$) to only 10% with hybrid fast/slow writes (Figure 15b).

Enlarging the queues increases system cost (each entry contains an entire cache block, thus 4 64-entry queues costs 31 KB more than 4 4-entry queues) and complexity. Fetch requests that circumvent queued writeback requests must first search the queue for possible address matches. With a larger queue, the complexity for checking these conflicts increases. In Section 5.5, we measure the impact of not reordering memory accesses, to reduce complexity. But first, we explore eager writebacks as an alternative to large queues in the next subsection.



(a) normalized bitline energy



(b) normalized execution time

Figure 15. ZettaRAM with hybrid write policy and various request queue sizes.

5.4. De-clustering L2 Writeback Requests

To avoid the complexity of larger queues in the memory controller, we can alternatively employ techniques that evenly distribute writeback requests from the L2 cache. One such technique is eager writeback [11], in which a writeback is issued as soon as a dirty block becomes the LRU block in its set, instead of waiting for the block to be evicted. Figure 16 shows the arrival time (in cycles) of the next request to a bank after a writeback request starts closing a page, for the hybrid write policy with 4 queue entries (top graph) and the hybrid write policy with 4 queue entries in conjunction with the eager writeback policy in the L2 cache (bottom graph). The measurements are for *mcf* (other benchmarks show

similar patterns). We can see that the L2 eager writeback policy de-clusters the writeback requests to memory. For example, once a writeback request starts closing a page, the next request does not arrive for at least 100 cycles. In comparison, without eager writeback, about a quarter of all next requests arrive between 0 and 100 cycles.

Thus, with eager writebacks, we can probably do well with a small queue, in spite of delaying writebacks in the memory controller. *Effectively, issuing the writeback early from the L2 cache compensates for delaying it in the memory controller.*

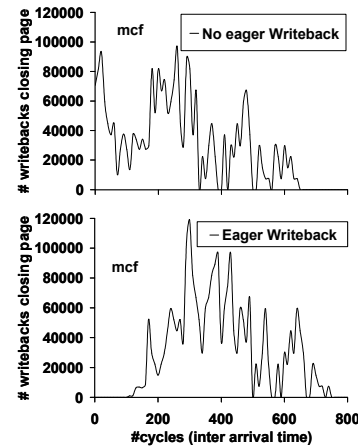


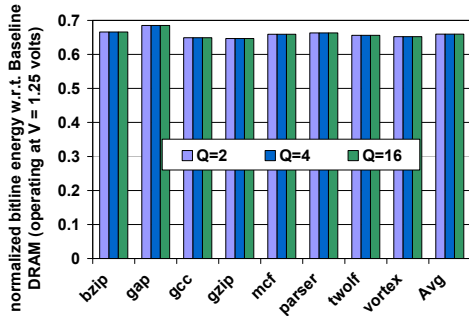
Figure 16. Arrival time (in cycles) of next request after a writeback request starts closing a page.

The results in Figure 17 confirm our prediction. The L2 cache implements eager writeback for all configurations, including the baseline DRAM system to which energy and performance are normalized. We note that L2 eager writebacks improve performance of the baseline DRAM system by only a negligible amount for these benchmarks, 0.6-1.3%, and bitline energy consumption is unaffected.

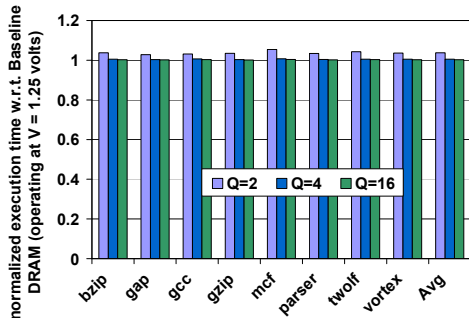
Figure 17(b) shows that L2 eager writebacks are surprisingly effective for eliminating performance degradation otherwise caused by delayed writebacks in the memory controller. A ZettaRAM memory system employing slow writes for pages closed by non-critical requests achieves 34% bitline energy savings with less than 1% performance degradation, with the L2 eager writeback policy. This is achieved without enlarging the request queue size with respect to the baseline system (4 entries). This is significant in terms of keeping the cost and complexity of the memory controller the same as the baseline. Results are also presented for queue sizes of 2 and 16. The queue size of 2 entries degrades performance by 4%, whereas 16 entries performs only slightly better than 4 entries.

The eager writeback policy may increase the number of L2 writeback requests, by occasionally writing back dirty blocks before the final store has occurred to the block before eviction. Fortunately, it is rare. Figure 18

shows the percentage increase in the number of L2 writeback requests and L2 writeback requests that close a page in memory, because of premature writebacks. On average, there is only a 0.84% increase in the number of writeback requests and a 0.16% increase in the number of writeback requests that close a page. Of course, extra writebacks are accounted for in the primary results in Figure 17.



(a) normalized bitline energy



(b) normalized execution time

Figure 17. ZettaRAM (hybrid write policy, various queue sizes) in conjunction with L2 eager writebacks.

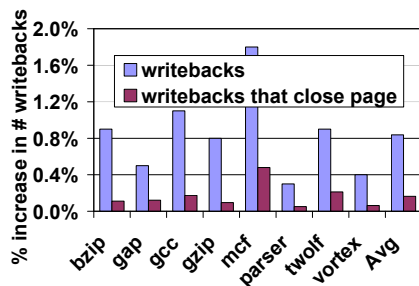


Figure 18. Percentage increase in L2 writebacks.

5.5. Effect of Memory Access Reordering

From the previous subsections, we arrive at two competing alternatives for achieving high performance and low energy with a ZettaRAM memory system: (1) hybrid policy with a large queue vs. (2) hybrid policy with a small queue and L2 eager writebacks. In all previous experiments, memory access reordering was used by default. We now measure the impact of disabling

memory access reordering for the two competing alternatives. The increase in execution time when reordering is disabled is shown in Figure 19 (“ZettaRAM, large Q, no reorder” and “ZettaRAM, small Q, eager WB, no reorder”). As one might expect, memory access reordering is performance-critical for the ZettaRAM that achieves latency tolerance via the large queue (execution time increases by 5-12% without reordering), but not performance-critical for the ZettaRAM that achieves latency tolerance via eager writebacks (execution time increases by less than 1.2% without reordering).

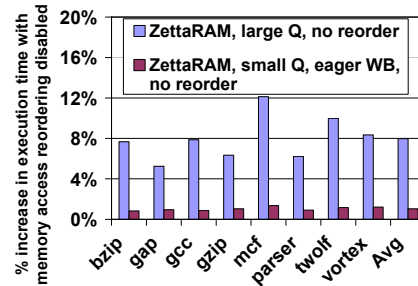


Figure 19. Impact of disabling memory access reordering for two ZettaRAM configurations.

To sum up, the ZettaRAM with hybrid fast/slow writes and L2 eager writebacks taps most of the energy savings potential of ZettaRAM with little performance loss and the least complexity in the memory controller.

6. Related Work

There have been a lot of studies on energy management of main memories (DRAMs). Delaluz et al. [4] and Fan et al. [5] have proposed energy management schemes based on the main memory switching between four different operating modes (active, standby, nap and power down). Techniques have also been proposed to reduce row-buffer conflicts and increase row buffer hit rates [17]. This in turn results in less bitline state transitions because the data remains in the row buffer for a longer period of time, leading to lesser energy consumption in main memory. Itoh et al. quantified the energy consumption in main memory and concluded that bitline energy consumption is the main component of the total memory system energy consumption [8][9].

7. Summary and Future Work

We explored the energy benefits of ZettaRAM, a new memory technology based on conventional DRAM architectures but featuring a new molecular capacitor that replaces the conventional capacitor. Some of the key advantages of the molecular capacitor are (i) self-assembly and high charge density, thus cost-effectively scaling the density of DRAM, and (ii) precise control of molecules’ attributes, yielding tremendous flexibility in controlling performance and energy consumption as guided by architectural explorations.

This paper explored ZettaRAM from the point of view of computer architects. Using a novel SPICE device model for the molecular capacitor, we first confirmed via SPICE simulations and other analyses that ZettaRAM and conventional DRAM are comparable in most respects when operating at nominal voltages, but ZettaRAM derives key benefits from being able to reliably perform writes at low voltages whereas DRAM fails to deposit sufficient charge at these lower voltages. We explored the tradeoff that ensues, namely that lower voltage swings significantly reduce bitline energy at the expense of degraded molecule speed and in turn system performance. Nonetheless, this tradeoff presents itself for the first time and provides opportunities to intelligently manage ZettaRAM.

We systematically applied architectural principles to almost completely eliminate performance losses while achieving nearly the full power savings potential. Several key insights were revealed in the process. We decided to focus on L2 writeback requests and found that several factors nicely converge to make this a nearly optimal choice. First, it turns out that writeback requests account for about 80% of all closed pages, and thus most of the energy savings potential. Second, writebacks do not directly stall the processor and thereby offer significant scheduling flexibility, in terms of tolerating longer latency. Based on these insights, we proposed a hybrid write policy, in which slow writes (low energy) are applied to non-critical writebacks and fast writes (high energy) are applied to critical fetches. The hybrid write policy is then combined with one of two techniques for tolerating delayed writeback requests: using a large pending request queue with memory access reordering or implementing the L2 eager writeback policy. Eager writeback is remarkably synergistic with low-power ZettaRAM: initiating writebacks early in the L2 cache compensates for delaying them at the memory controller. For SPEC2K benchmarks, applying slow writes non-uniformly coupled with small queues and the eager writeback policy yields bitline energy savings of 34% (out of a possible 41% with uniformly slow writes), with less than 1% performance degradation.

A single choice of molecule was used in this paper, although hundreds have been characterized by ZettaCore and we plan to expand our architecture research to include other synthesized molecules in the future. We are also investigating novel uses of other various ZettaCore devices and look to exploit them in memories and logic.

8. Acknowledgments

We thank Ken Mobley and Werner Kuhr for valuable discussions on modeling the molecular capacitor, Rhett Davis for insights on multiple supply voltages, and the anonymous reviewers for their valuable comments. This research was supported by a grant from ZettaCore.

9. References

- [1] A. J. Bard and L. R. Faulkner. *Electrochemical Methods: Fundamentals and Applications*, John Wiley and Sons, 2001, pp. 92-96.
- [2] D. Burger, T. Austin, and S. Bennett. Evaluating Future Microprocessors: The SimpleScalar Toolset. Tech. Rep. CS-TR-96-1308, CS Department, Univ. of Wisc.-Madison, July 1996.
- [3] A. P. Chandrakasan and R. W. Brodersen. Minimizing Power Consumption in Digital CMOS Circuits. *Proc. of the IEEE*, 83(4):498-523, April 1995.
- [4] V. Delaluz, A. Sivasubramaniam, M. Kandemir, N. Vijaykrishnan, and M. J. Irwin. Scheduler-based DRAM Energy Management. *Design Automation Conference*, June 2002.
- [5] X. Fan, C. S. Ellis, and A. R. Lebeck. Memory Controller Policies for DRAM Power Management. *Int'l Symp. on Low Power Electronics and Design*, Aug. 2001.
- [6] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge. Drowsy Caches: Simple Techniques for Reducing Leakage Power. *Int'l Symp. on Comp. Architecture*, July 2002.
- [7] G. Hinton, D. Sager, M. Upton, D. Boggs, D. Carmean, A. Kyker, P. Roussel. *The Microarchitecture of the Pentium® 4 Processor*. Intel Technology Journal Q1, 2001.
- [8] K. Itoh, K. Sasaki, Y. Nakagome. Trends in Low-Power RAM Circuit Technologies. *Proc. of the IEEE*, 83(4):524-543, April 1995.
- [9] K. Itoh. *VLSI Memory Chip Design*, Springer Series in Advanced Microelectronics, 2001, pp. 117-123.
- [10] K. Itoh. *VLSI Memory Chip Design*, Springer Series in Advanced Microelectronics, 2001, p. 403.
- [11] H. S. Lee, G. S. Tyson, M. K. Farrens. Eager Writeback – a Technique for Improving Bandwidth Utilization. *33rd Int'l Symp. on Microarchitecture*, pp. 11-21, 2000.
- [12] Micron SDRAM 8Mx16x4 Part No. MT48LC32M16A2TG-75, 2003.
- [13] K. M. Roth, D. T. Gryko, P. C. Clausen, J. Li, J. S. Lindsey, W. G. Kuhr, D. F. Bocian. Comparison of Electron-Transfer and Charge-Retention Characteristics of Porphyrin-Containing Self-Assembled Monolayers Designed for Molecular Information Storage. *J. Phys. Chem. B*, 106, 8639-8648, 2002.
- [14] K. M. Roth, N. Dontha, R. B. Dabke, D. T. Gryko, C. Clausen, J. S. Lindsey, D. F. Bocian, W. G. Kuhr. Molecular Approach Toward Information Storage Based on the Redox Properties of Porphyrins in Self-Assembled Monolayers. *J. Vac. Sci. Technology B*, 18, 2359–2364, 2000.
- [15] K. M. Roth, J. S. Lindsey, D. F. Bocian, W. G. Kuhr. Characterization of Charge Storage in Redox-Active Self-Assembled Monolayers. *Langmuir*, 18, 4030–4040, 2002.
- [16] T. Sherwood, E. Perelman, G. Hamerly and B. Calder. Automatically Characterizing Large Scale Program Behavior. *10th Int'l Conf. on Architectural Support for Programming Languages and Operating Systems*, Oct. 2002.
- [17] Z. Zhang, Z. Zhu, and X. Zhang. A Permutation-based Page Interleaving Scheme to Reduce Row-buffer Conflicts and Exploit Data Locality. *33rd Int'l Symp. on Microarchitecture*, pp. 32-41, Dec. 2000.
- [18] ZettaCore - <http://www.zettacore.com>

ZettaRAM™ mark is a trademark of ZettaCore Inc.
ZettaCore™ mark is a trademark of ZettaCore Inc.