Memory Controller Policies for DRAM Power Management

Xiaobo Fan Carla S. Ellis Alvin R. Lebeck

Department of Computer Science
Duke University
Durham, NC 27708 USA
{xiaobo,carla,alvy}@cs.duke.edu

ABSTRACT

The increasing importance of energy efficiency has produced a multitude of hardware devices with various power management features. This paper investigates memory controller policies for manipulating DRAM power states in cache-based systems. We develop an analytic model that approximates the idle time of DRAM chips using an exponential distribution, and validate our model against trace-driven simulations. Our results show that, for our benchmarks, the simple policy of immediately transitioning a DRAM chip to a lower power state when it becomes idle is superior to more sophisticated policies that try to predict DRAM chip idle time.

1. INTRODUCTION

Energy efficiency is becoming an increasingly important target for optimization in many system designs. Mobile computing devices require techniques to extend battery lifetime, while others must reduce power to meet heat or fan noise limitations (e.g., medical applications). Even desktop and server systems should be energy efficient for economical and environmental considerations. Main memory is consuming an increasing proportion of the power budget and thus motivates efforts to improve DRAM energy efficiency.

DRAM manufacturers are meeting this demand by developing DRAM chips with multiple power states such as active, standby, nap and powerdown. The chip must be in the active state to service a request. The remaining states are in order of decreasing power consumption but increasing time to transition back to active. Energy efficiency can be improved by placing the chips in a lower power state when not used. The challenge for the system designer is to utilize these modes most effectively.

In our previous work [5], we investigated memory controller policies for making DRAM chip power state transitions in conjunction with software page placement policies. The power-aware page allocation policies exploit working set locality to increase the opportunity for the memory controller to make effective transition decisions.

ISLPED'01, August 6-7, 2001, Huntington Beach, California, USA.

The goal of this work is to understand the characteristics of memory access patterns in a cache-based memory architecture and how those patterns affect the design of controller policies that transition among power states. For a memory system without caches, there is work showing potential benefits of an adaptive policy that attempts to predict the time between consecutive accesses as a basis for deciding when to make transitions [1]. By contrast, we consider the behavior of policies for memory requests generated by representative productivity applications and filtered through a 2-level cache. We consider access patterns produced by random page allocation as well as the sequential first-touch policy previously shown to be effective [5] when used in conjunction with simple power-aware controller policies. The basic question is whether simple policies are adequate to capture the relevant features of cache-filtered accesses.

To characterize memory access patterns, we define the notion of *gap* as the interval between clustered accesses. We find that most memory traces from our workload filtered by 2-level cache have gap distributions that can be approximated by an exponential distribution. They also have large average gap values (greater than 200ns). A critical parameter in the design of memory controller policies is the length of time spent in the current power state before a transition to a lower state is made. We refer to this as the *threshold*. We analyze the relationship between threshold values and a model of exponentially distributed memory access gaps. The analytical result shows that, for our benchmarks, the simple instant transition policy (threshold = 0) produces maximum benefit. Finally, we experimentally validate this theoretical conclusion through trace-driven simulation.

The remainder of this paper is organized as follows. In the next section, we provide background on power-aware memory design. We identify the primary factors that characterize memory access patterns and affect the behavior of power control. Then, we introduce our evaluation metric and our method of generating cache-filtered memory traces. Section 4 examines gap distributions and analyzes the relationship between gaps and thresholds. We present simulation results and show how close they are to the theoretical analysis. Section 5 concludes.

2. BACKGROUND

This section reviews modern DRAM power management features and appropriate memory controller policies for exploiting these features. We also identify the important characteristics of DRAM access patterns and how they interact with memory controller power management policies.

Power State Transition	Power (mW)	Time (nS)
Active	$P_a = 300$	t_{acc}=60
Standby	$P_s = 180$	-
Nap	$P_n = 30$	-
Powerdwn	$P_p = 3$	-
Stby → Act	$P_{s\to a} = 240$	$T_{s\to a} = +6$
Nap → Act	$P_{n\to a} = 165$	$T_{n\to a} = +60$
Pdn → Act	$P_{p\to a} = 152$	$T_{p\to a} = +6000$

Table 1: RDRAM Power State and Transition Values: All accesses incur the 60ns active access time. Additional delay (denoted by the +) is incurred for clock resynchronization.

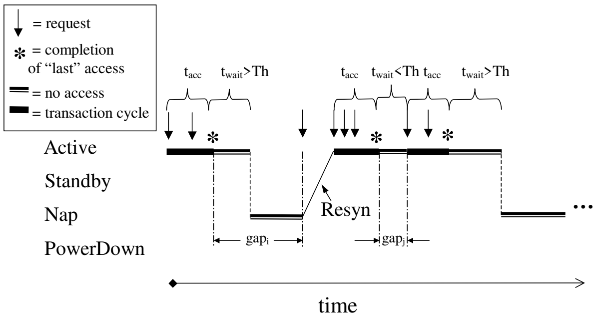

Figure 1: Power Management

2.1 Rambus DRAM

Memory technology has developed to respond to the needs of mobile computer designers to limit power consumption in the face of increasing demand for performance. One concrete example is Direct Rambus DRAM (RDRAM)[7]. The Direct Rambus technology delivers high bandwidth (1.6GB/sec per device), using a narrow bus topology operating at a high clock rate. As a result, each RDRAM chip can be activated independently. RDRAM offers four power modes: active, standby, nap, and powerdown. Because of the narrow topology, each chip can be independently set to an appropriate power state.

An RDRAM device must be in the active state to perform a read or write transaction, which takes 60ns and consumes 300mW. A chip that is not servicing a memory request can be in any of the lower power states. However, these states incur additional delay for clock resynchronization. Standby is fast and uses 60% of the power of active mode. Greater power savings can be achieved by using nap mode (10% of the power of active) with an additional resynchronization time required to transition to the active state in order to service a memory request. Powerdown mode has the minimal power consumption (1% of active), but a significant delay for clock synchronization (100 times that needed by nap mode) to enter the active state. Table 1 shows the power states with the power cost values used in this study as well as the possible transitions and additional transition times into active mode [7, 4].

2.2 DRAM Power Management

The challenge for the memory controller designer is to utilize these modes effectively. It is not only the availability of these power states but the ability to transition between them dynamically on a per-chip basis that gives the RDRAM its potential for power management. The key for the memory controller policy is to determine when the benefit of transitioning to a low power state is greater than the penalty for transitioning back to the active state.

The time between DRAM accesses is the important characteristic that influences the memory controller policy design. Furthermore, we note that any DRAM chip access that arrives during the service time of the previous access can immediately be serviced and will increase the time the chip is in the active state. We call a sequence of such DRAM chip accesses *clustered accesses* since the DRAM chip can not transition to a lower power state. Therefore, it is actually the time between clustered accesses—called the *gap*—that a DRAM chip could reside in a lower power state.

By monitoring the *gap* we can establish a memory controller power management policy that exploits the tradeoff between potential energy savings and resynchronization costs. The power management policy works as follows (see Figure 1): a given DRAM chip remains in the active state until the *gap* exceeds a *threshold* amount of time, it then transitions to the low power state until the next access. The key to this policy is to determine the appropriate *threshold* value to maximize energy efficiency. However, this depends on the DRAM access characteristics in terms of *gap*. The following section outlines our methodology for exploring the relationship between *gap* and *threshold*.

3. METHODOLOGY

To evaluate energy efficiency, we use the Energy•Delay product ($E \bullet D$) [2]. This metric captures our goal of achieving high performance (seconds) while minimizing energy consumption (Joules). Although total system energy consumption is important, it is highly dependent on specific design choices (e.g., processor, display type, wireless network interface, etc.). Therefore, we concentrate only on DRAM energy consumption, and ignore the energy consumed by all other system components.

To fully explore the relationship between DRAM access *gaps* and the memory controller *threshold* values, we use a combination of trace-driven simulation (described below) and analytic evaluation (see Section 4). The trace-driven simulator is used to both characterize the DRAM access patterns and to validate our analytic model.

The trace-driven simulator processes instruction and data address traces of personal productivity applications [6] and uses a simplified RDRAM model. This simulator models a two-level cache hierarchy with a 16KB L1 and a 256KB L2 cache, both caches are direct-mapped with 32B blocks and can support 8 outstanding misses. Higher associative caches do not qualitatively change our results. We model the individual RDRAM chips and their associated power state, but do not model memory bus contention or the internal DRAM banks. In these studies we only model the transition from the lower power state to active. The transitions from active to lower power states do not incur any delay or energy consumption.

For timing considerations (necessary to compute energy consumption), we use a simplified processor model that exe-

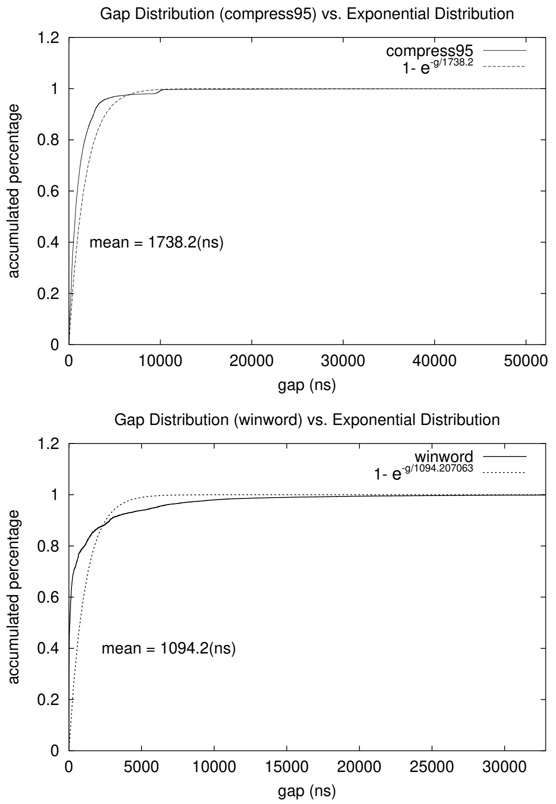

Figure 2: Examples of Gap Distribution

cutes one instruction per cycle, and never stalls due to long latency operations (i.e., execution only stalls when the maximum number of outstanding misses is reached). We assume a 500Mhz processor clock, the level one cache takes 2 cycles to access, while the level two cache incurs an additional 10 cycles. We simulate a non-interleaved main memory system with eight 32Mb RDRAM chips, for a total main memory capacity of 32MB.

4. EVALUATION

Recognizing that gap and threshold are two factors that may affect our power control effectiveness, we study the relationship between $E \bullet D$ and these two factors.

4.1 DRAM Access Characteristics

The first step is to capture the distribution of access gaps in the execution of different benchmarks within our cache-based architecture. We observe cache misses from each individual memory chip and measure the time between clustered misses. Figure 2 is one of the examples showing the gap distributions of one chip for the compress95 and winword benchmarks. Most of our benchmarks have similar distributions regardless of the number of memory chips and physical page placement policy.

Related studies trying to statistically characterize cache misses have not been successful because the distributions that best characterize the behavior do not have finite variance [8]. For our purposes it is sufficient to approximate the

Benchmark	Result
Compress95	Pass
Go	Pass
Netscape	Pass
Acroread	Fail
PowerPoint	Fail
Winword	Fail

Table 2: Chi-Square Test Results

gap distribution. Thus, Figure 2 also plots the exponential distribution with the same mean gap size.

From Figure 2 we observe that the gap distributions for compress95 and winword match the general shape of the exponential. Table 2 shows the results of applying the Chi-Square test of our observed data to the exponential distribution of gap sizes [3]. The Chi-Square test reveals that three of our applications pass the test with significance $\alpha = 0.05$, whereas three fail this test. Nonetheless, using the exponential as an approximation of the real gap distribution is sufficient as it produces results consistent with simulation (see Section 4.3). Modeling the gap distribution with the exponential allows us to perform analysis and more extensively explore the design space. We assert that the errors inherent in this approximation results in a pessimistic bias in the results.

4.2 Analysis

We use the always-active policy as a baseline for comparison. For simplicity, we choose *nap* as the low power state in our 2-state threshold waiting control policy. Let g denote the gap between clustered memory accesses and Th be the waiting threshold before transitioning to the low power state. Assuming the memory access gap follows an exponential distribution, its density function is:

$$p(g) = \frac{1}{\mu} e^{-\frac{g}{\mu}}$$

where μ is the mean gap. Then the mean time of staying in low power is:

$$t_{nap} = \int_{Th}^{\infty} p(g)(g - Th)dg = \mu e^{-\frac{Th}{\mu}}$$

With P_a as the *active* state power consumption and P_n as the *nap* state power consumption, the mean energy savings from staying in low power is:

$$\Delta e_1 = (P_a - P_n)t_{nap} = \mu(P_a - P_n)e^{-\frac{Th}{\mu}}$$

Let $T_{n\to a}$ represent the resynchronization time from *nap* to *active*. The mean energy cost for resynchronization is:

$$\begin{aligned} \Delta e_2 &= \frac{1}{2}(P_a + P_n)T_{n\to a}\int_{Th}^{\infty} p(g)dg \\ &= \frac{1}{2}(P_a + P_n)T_{n\to a}e^{-\frac{Th}{\mu}} \end{aligned}$$

Therefore the mean energy cost (increased energy consumption) for each gap is:

$$\begin{aligned} \Delta e &= \Delta e_2 - \Delta e_1 && (1) \\ &= [\frac{1}{2}(P_a + P_n)T_{n\to a} - (P_a - P_n)\mu]e^{\frac{-Th}{\mu}} && (2) \end{aligned}$$

The mean increased delay for each gap is simply:

$$\Delta d = \int_{Th}^{\infty} T_{n \to a} p(g) dg = T_{n \to a} e^{\frac{-Th}{\mu}} \tag{3}$$

With these per-gap mean delay and energy changes, we compute the change of total $E \bullet D$ product in one run. From Figure 1, the per-gap mean energy consumption with the always-active policy is:

$$e_0 = P_a(t_{acc} + \mu) \tag{4}$$

and the mean delay is:

$$d_0 = t_{acc} + \mu \tag{5}$$

With the power transition policy, the per-gap mean energy consumption e and mean delay d are:

$$e = e_0 + \Delta e \qquad d = d_0 + \Delta d$$

Let D_0 and E_0 denote the original runtime and energy consumption with the always-active policy, D and E are those with power state transition. Assume n is the number of gaps in the same run. Since $E = ne$, $D = nd$, $E_0 = ne_0$ and $D_0 = nd_0$, the total change of $E \bullet D$ is calculated by:

$$\begin{aligned} \Delta(E \bullet D) &= E \bullet D - E_0 \bullet D_0 & (6) \\ &= n^2(d_0 \Delta e + \Delta d e_0 + \Delta d \Delta e) & (7) \end{aligned}$$

We define per-gap mean change in energy delay product as $\Delta(e \bullet d)$.

$$\Delta(e \bullet d) = d_0 \Delta e + \Delta d e_0 + \Delta d \Delta e \tag{8}$$

From Equation 7 we see the change in total energy delay product $\Delta(E \bullet D)$ is linear to the per-gap mean change $\Delta(e \bullet d)$. So we use this $\Delta(e \bullet d)$ as the metric to evaluate the control policy. If it is positive, the policy is worse than the always-active policy; if it is negative, the policy is better. As $\Delta(e \bullet d)$ decreases, the benefit increases.

From Equations 2-5 and Equation 8 we have:

$$\Delta(e \bullet d) = f(\mu, Th) \tag{9}$$

Now, we can use this analytic result to explore the parameter space. We use the parameter values in Table 1 to model a single RDRAM chip. By substituting these parameter values into our formulas, we obtain Figure 3 and Figure 4. Figure 3 shows $\Delta(e \bullet d)$ as a function of μ (mean *gap*) with different fixed *threshold* values. Figure 4 shows $\Delta(e \bullet d)$ as a function of *threshold* with different mean *gap* values. Since our empirical gap distributions have relatively large average gap values, we first focus on the case where μ is large. As we can see from the two graphs, when *threshold* is fixed and μ is large enough, $\Delta(e \bullet d)$ is a monotonically decreasing function of μ; while with fixed μ, $\Delta(e \bullet d)$ increases as *threshold* increases. Threshold 0 produces maximum benefit on $\Delta(e \bullet d)$. When μ is large, the energy savings can overcome the extra transition cost. Because of the memoryless property of exponential distribution, waiting for a threshold amount of time does not provide any knowledge about the future access. Therefore the instant transition policy is the best policy for the distributions with large mean gaps.

When μ is small (the part left of the crossover point in Figure 3, the line $\Delta(e \bullet d)(\mu = 50, Th)$ in Figure 4), a larger threshold performs better than a smaller threshold but worse than the original always-active policy. This is because with

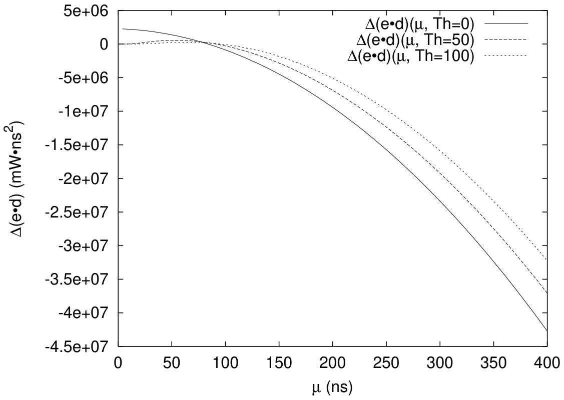

Figure 3: $\Delta(e \bullet d) = f(\mu, Th), Th = 0, 50, 100ns$

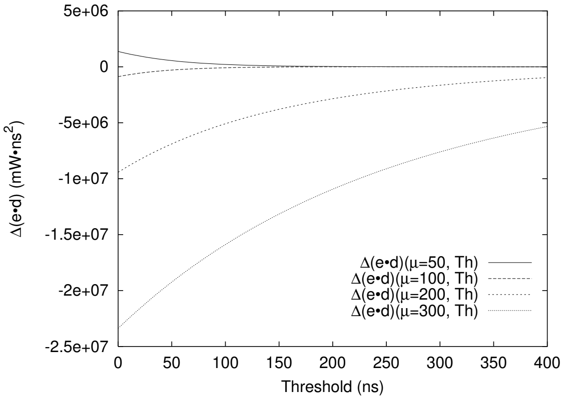

Figure 4: $\Delta(e \bullet d) = f(\mu, Th), \mu = 50, 100, 200, 300ns$

a small mean value most gaps cannot bring a potential energy saving large enough to cover the transition cost. The larger threshold causes fewer power state transitions and thus avoids some resynchronization costs. The always-active (no transition) policy is the best for this case.

4.3 Validation

To validate our analytic model we use trace-driven simulation of both random and sequential first-touch page allocation policies [5]. By comparing $\Delta(e \bullet d)$ obtained from the simulation to that obtained from the model, we can gauge the accuracy of the model. For this analysis we focus only on the transitions from active to nap.[1] Therefore, with sequential first-touch page allocation, only one DRAM chip is used. In an actual implementation, the unused chips would transition to the powerdown state, independent of the active to nap threshold. For random allocation, although all eight DRAM chips are active, for brevity we consider only the four chips with the smallest average *gap*. We present results for only the benchmarks compress95 and winword. We note that winword produces the largest error between the simulation and the model. The other benchmarks produce results similar to these two benchmarks.

Figure 5 shows the $\Delta(e \bullet d)$ values obtained from both simulation and the model. Table 3 shows the raw simula-

[1] Given our average gap sizes, our analytic results suggest no viable role for the intermediate standby state.

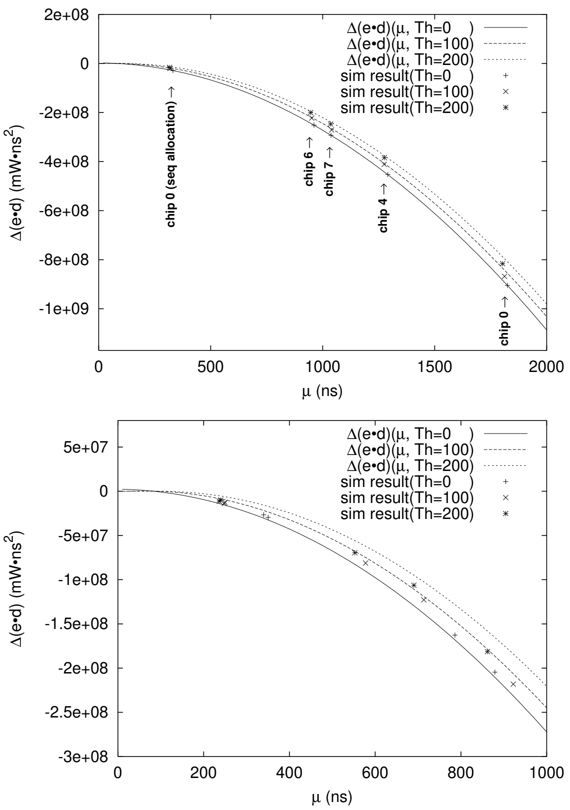

Figure 5: Model Validation (compress95 and winword):

tion data and the relative difference between the model and the simulation for compress95. Each row of the table corresponds to one DRAM chip for the given threshold, hence only one chip per threshold with sequential first-touch and four chips per threshold with random page allocation.

From these results we see that for compress95, in most cases, the results from the model are within 5% of the simulation results. For winword, the error is larger, approaching 50% in some cases. Although this error seems large, the qualitative result is the same for both simulation and the model—zero threshold performs best. Furthermore, the trend in the results is the same, zero threshold performs best and increasing the threshold decreases the energy benefits. We note that the relative difference increases as the threshold increases, and is generally larger for sequential first-touch page allocation than for random page allocation.

For sequential-first-touch allocation, the absolute $\Delta(e \bullet d)$ values are small, so small changes produce large relative errors. This page allocation policy benefits mostly from the unused chips entering powerdown. Nonetheless, a zero threshold is the best solution for both the simulation results and the model. The increasing relative difference as the threshold increases is due to the approximated *gap* distribution differing from the real distribution. For larger thresholds this difference is magnified, while for the zero threshold $\Delta(e \bullet d)$ is distribution independent (see Equations 2-3).

Th	**Average Gap(ns)**	$\Delta(e \bullet d)(10^8 mW \cdot ns^2)$ analysis	$\Delta(e \bullet d)(10^8 mW \cdot ns^2)$ simulation	**Diff**
Sequential First-touch Page Allocation				
0	331.3	-0.289	-0.286	1.0%
100	318.1	-0.184	-0.204	9.8%
200	316.5	-0.128	-0.161	20.5%
Random Page Allocation				
0	961.1	-2.515	-2.514	0.04%
0	1037.1	-2.928	-2.927	0.04%
0	1290.8	-4.534	-4.525	0.01%
0	1824.5	-9.047	-9.046	0.01%
100	950.5	-2.200	-2.228	1.3%
100	1039.5	-2.658	-2.712	2.0%
100	1274.8	-4.075	-4.114	1.0%
100	1811.9	-8.429	-8.679	2.9%
200	946.0	-1.949	-2.003	2.7%
200	1035.3	-2.382	-2.469	3.5%
200	1275.4	-3.759	-3.839	2.1%
200	1803.0	-7.881	-8.163	3.5%

Table 3: Comparison between Analysis and Simulation (Compress95): For random allocation each row for a given threshold represents an individual chip (1 of 4), while there is only one active chip for sequential.

5. CONCLUSION

Modern DRAM chips provide power management features to help meet the increasing demand for energy efficient computing. The challenge is to develop memory controller policies that best exploit these features. This paper explores DRAM power management policies for cache-based systems using analytic modeling validated with trace-driven simulation. Our results reveal that, for most workloads on cache-based systems, DRAM chips should immediately transition to a lower power state when they become idle and will not benefit from sophisticated power management policies.

Acknowledgements

This work supported in part by NSF Grants CCR-0082914, EIA-99-72879, EIA-99-86024, NSF CAREER Award MIP-97-02547, Duke University, and equipment donations from Intel and Microsoft.

6. REFERENCES

[1] V. Delaluz, M. Kandemir, N. Vijaykrishnan, A. Sivasubramaniam, and M.J. Irwin. DRAM Energy Management Using Software and Hardware Directed Power Mode Control. In *HPCA 2001*, January 2001.

[2] Ricardo Gonzalez and Mark Horowitz. Energy Dissipation in General Purpose Microprocessors. In *Proceedings of the IEEE International Symposium on Low Power Electronics*, October 1995.

[3] R. Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling.* Wiley-Interscience, April 1991.

[4] D. Lammers. IDF: Mobile Rambus spec unveiled. *EETimes Online*, February 1999. //www.eetimes.com/story/OEG19990225S0016.

[5] Alvin R. Lebeck, Xiaobo Fan, Heng Zeng, and Carla S. Ellis. Power aware page allocation. In *Proceedings of Ninth International Conference on Architectural Support for Programming Languages and Operating System (ASPLOS IX)*, November 2000.

[6] Dennis C. Lee, Patrick J. Crowley, Jean-Loup Baer, Thomas E. Anderson, and Brian N. Bershad. Execution characteristics of desktop applications on Windows NT. In *Proceedings of the 25th Annual International Symposium on Computer Architecture*, pages 27–38, June 1998.

[7] Rambus. *RDRAM*, 1999. http://www.rambus.com.

[8] Harold S. Stone. *High-Performance Computer Architecture*, chapter Memory System Design, pages 76–84. Addison Wesley, 1993.