

Power-Aware Memory Management

© 2003, Carla Schlatter Ellis

ESSES 2003

Outline

- Motivation for memory power/energy management and the opportunity
- Hardware power management
- OS page allocation policies
- Experimental results
- Future work, open questions

© 2003, Carla Schlatter Ellis

ESSES 2003

2

Memory: The Unturned Stone

Previous Architecture/OS Energy Studies:

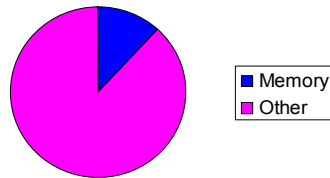
- Disk spindown policies [Douglass, Krishnan, Helmbold, Li]
- Processor voltage and clock scaling [Weiser, Pering, Lorch, Farkas et al]
- Network Interface [Stemm, Kravets]
- Mems-based storage [Nagle et al]
- Application-aware adaptation & API [Flinn&Satya]

- But where is main memory management?

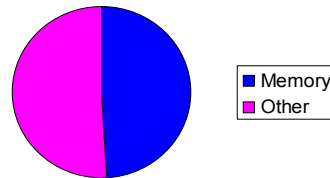
Power Aware Page Allocation [ASPLOS00]

Memory System Power Consumption

Laptop Power Budget
9 Watt Processor



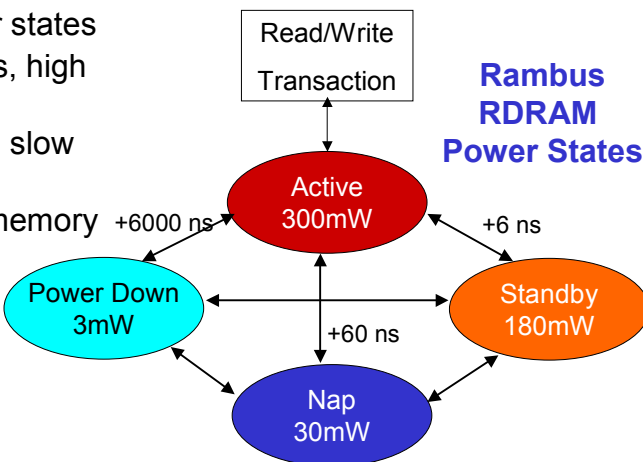
Handheld Power Budget
1 Watt Processor



- Laptop: memory is small percentage of total power budget
- Handheld: low power processor, memory is more important

Opportunity: Power Aware DRAM

- Multiple power states
 - Fast access, high power
 - Low power, slow access
- New take on memory hierarchy
- How to exploit opportunity?



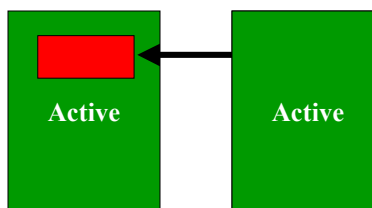
© 2003, Carla Schlatter Ellis

ESSES 2003

5

RDRAM as a Memory Hierarchy

- Each chip can be independently put into appropriate power mode
- Number of chips at each “level” of the hierarchy can vary dynamically.



Policy choices

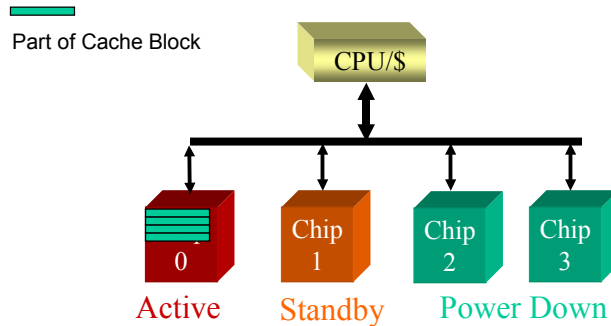
- initial page placement in an “appropriate” chip
- dynamic movement of page from one chip to another
- transitioning of power state of chip containing page

© 2003, Carla Schlatter Ellis

ESSES 2003

6

RAMBUS RDRAM Main Memory Design



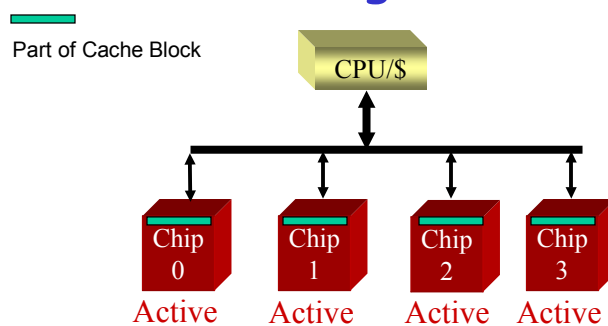
- Single RDRAM chip provides high bandwidth per access
 - Novel signaling scheme transfers multiple bits on one wire
 - Many internal banks: many requests to one chip
- Energy implication: **Activate only one chip to perform access at same high bandwidth as conventional design**

© 2003, Carla Schlatter Ellis

ESSES 2003

7

Conventional Main Memory Design



- Multiple DRAM chips provide high bandwidth per access
 - Wide bus to processor
 - Few internal banks
- Energy implication: **Must activate all those chips to perform access at high bandwidth**

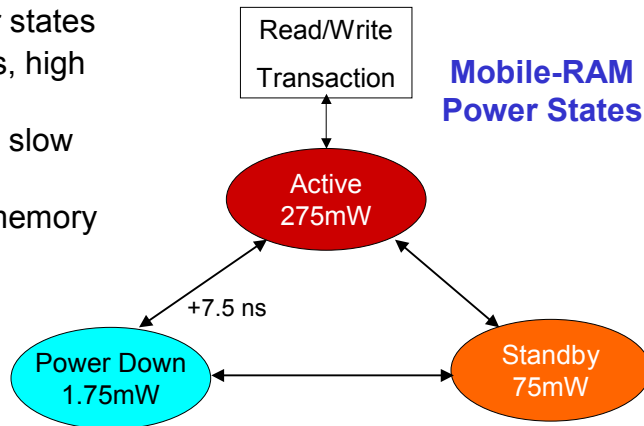
© 2003, Carla Schlatter Ellis

ESSES 2003

8

Opportunity: Power Aware DRAM

- Multiple power states
 - Fast access, high power
 - Low power, slow access
- New take on memory hierarchy
- How to exploit opportunity?



© 2003, Carla Schlatter Ellis

ESSES 2003

9

Exploiting the Opportunity

Interaction between power state model and access locality

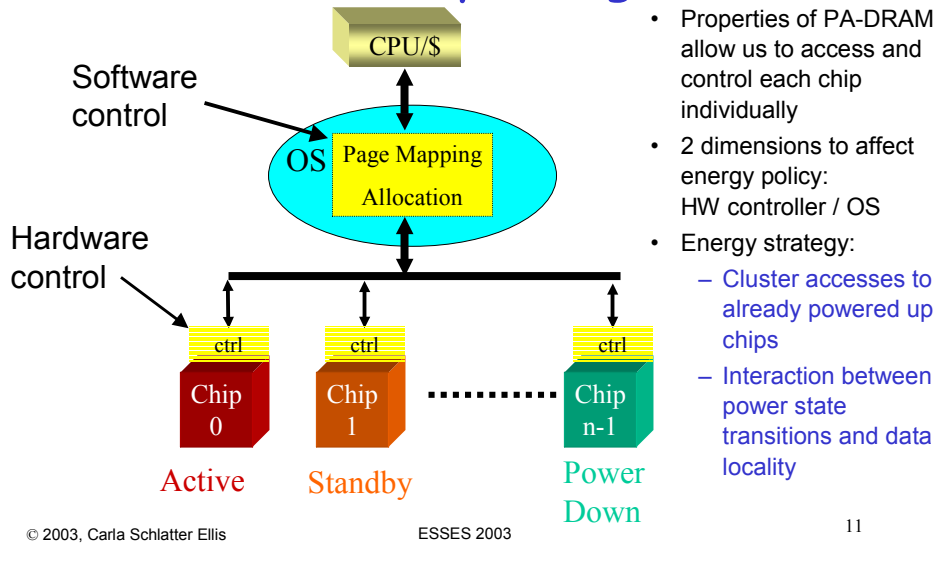
- How to manage the power state transitions?
 - Memory controller policies
 - Quantify benefits of power states
- What role does software have?
 - Energy impact of allocation of data/text to memory.

© 2003, Carla Schlatter Ellis

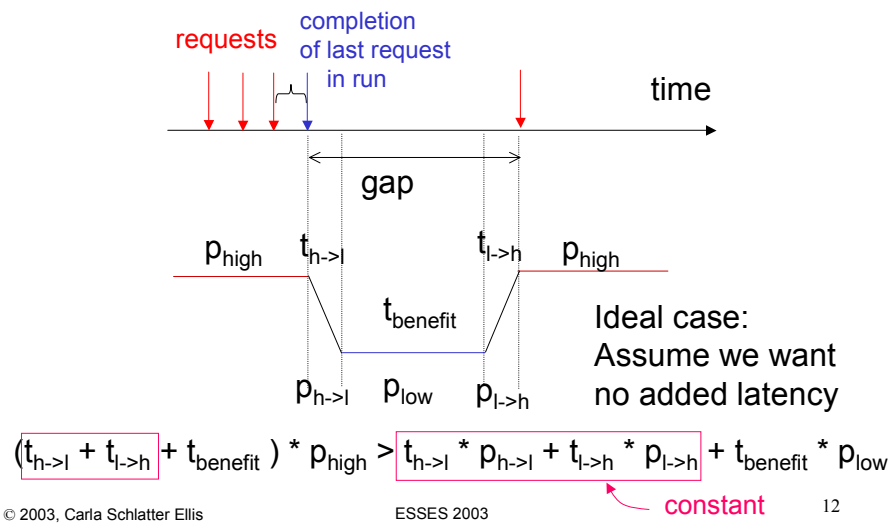
ESSES 2003

10

Power-Aware DRAM Main Memory Design



Power State Transitioning

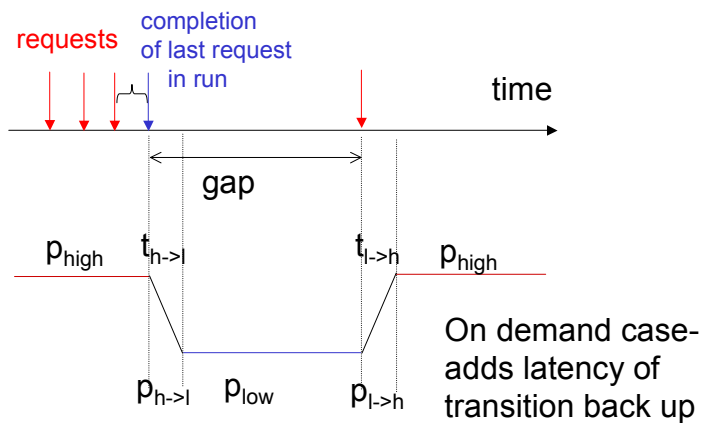


Benefit Boundary

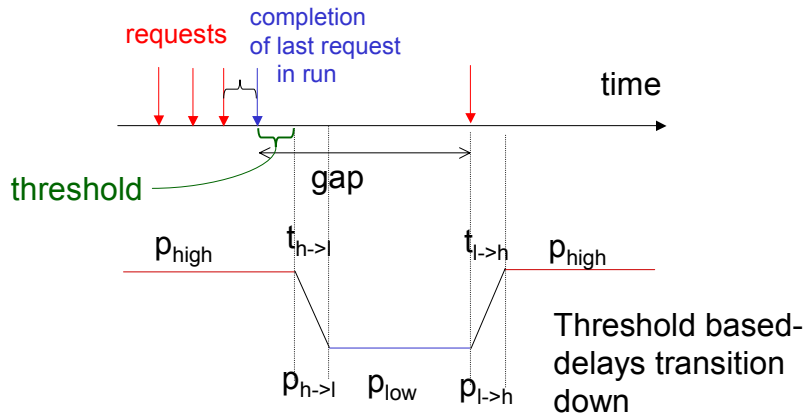
$$t_{\text{benefit}} > \frac{t_{h \rightarrow l} * p_{h \rightarrow l} + t_{l \rightarrow h} * p_{l \rightarrow h} - (t_{h \rightarrow l} + t_{l \rightarrow h}) * p_{\text{high}}}{(p_{\text{high}} - p_{\text{low}})}$$

$$\text{gap} \geq t_{h \rightarrow l} + t_{l \rightarrow h} + t_{\text{benefit}}$$

Power State Transitioning



Power State Transitioning

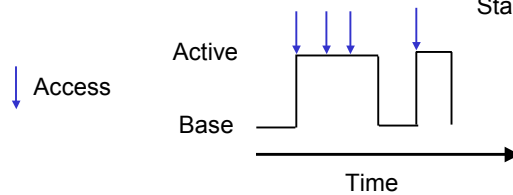
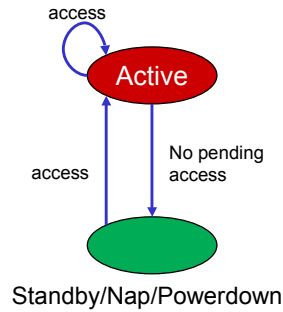


Outline

- Motivation for memory power/energy management and the opportunity
- Hardware power management
- OS page allocation policies
- Experimental results
- Future work, open questions

Dual-state HW Power State Policies

- All chips in one base state
- Individual chip Active while pending requests
- Return to base power state if no pending access



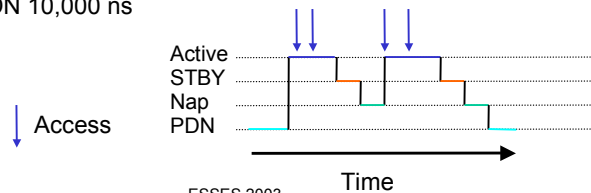
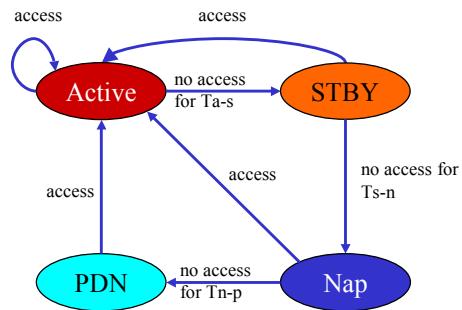
© 2003, Carla Schlatter Ellis

ESSES 2003

17

Quad-state HW Policies

- Downgrade state if no access for **threshold** time
- Independent transitions based on access pattern to each chip
- Competitive Analysis
 - rent-to-buy
 - Active to nap 100's of ns
 - Nap to PDN 10,000 ns



© 2003, Carla Schlatter Ellis

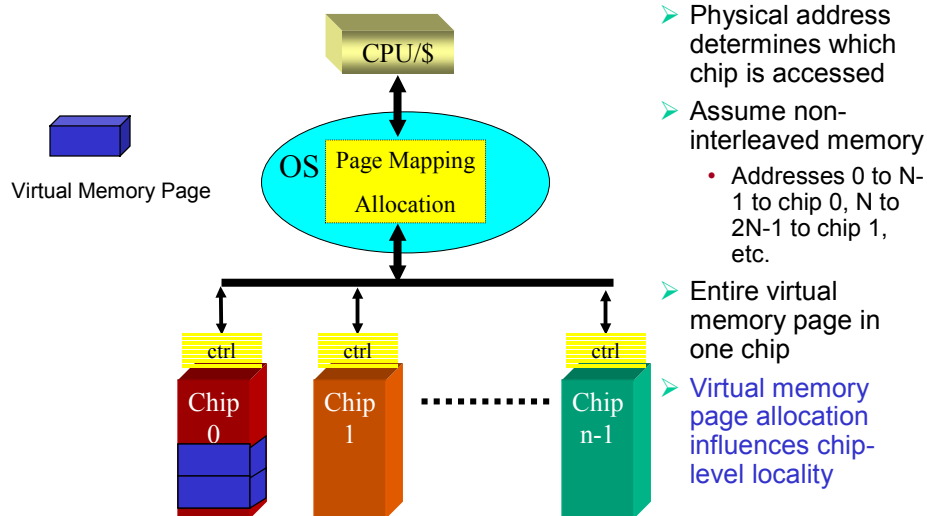
ESSES 2003

18

Outline

- Motivation for memory power/energy management and the opportunity
- Hardware power management
- OS page allocation policies
- Experimental results
- Future work, open questions

Page Allocation and Power-Aware DRAM



Page Allocation Policies

Virtual to Physical Page Mapping

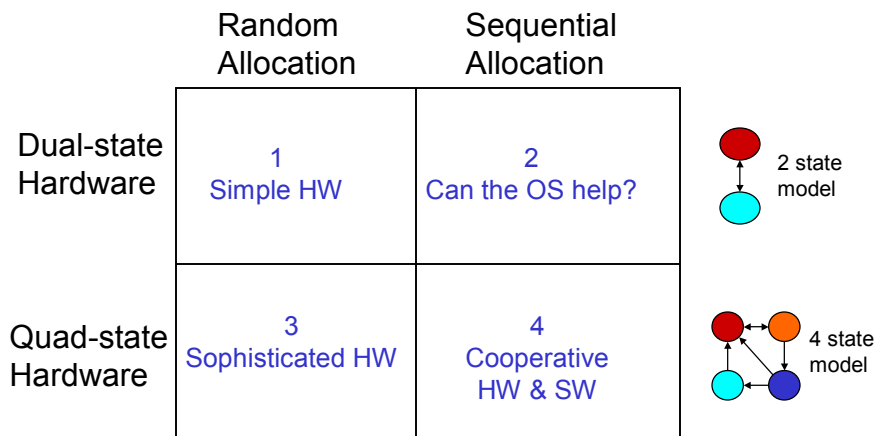
- Random Allocation – baseline policy
 - Pages spread across chips
- Sequential First-Touch Allocation
 - Consolidate pages into minimal number of chips
 - One shot
- Frequency-based Allocation
 - First-touch not always best
 - Allow (limited) movement after first-touch

Discussion: What about page
replacement policies?
Should (or *how* should) they
be power-aware?

Outline

- Motivation for memory power/energy management and the opportunity
- Hardware power management
- OS page allocation policies
- Experimental results
- Future work, open questions

The Design Space



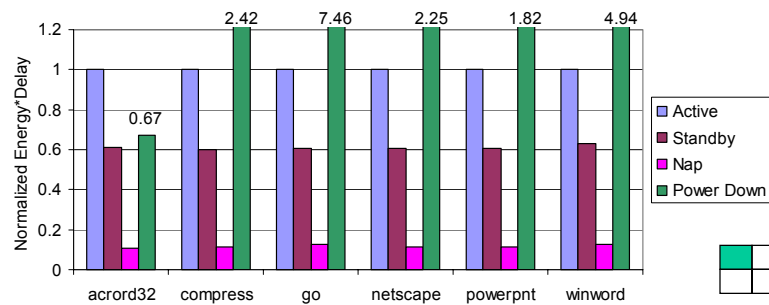
Methodology

- Metric: Energy*Delay Product
 - Avoid very slow solutions
- Energy Consumption (DRAM only)
 - Processor & Cache affect runtime
 - Runtime doesn't change much in most cases
- 8KB page size
- L1/L2 non-blocking caches
 - 256KB direct-mapped L2
 - Qualitatively similar to 4-way associative L2
- Average power for transition from lower to higher state
- Trace-driven and Execution-driven simulators

Methodology Continued

- Trace-Driven Simulation
 - Windows NT personal productivity applications (Etch at Washington)
 - Simplified processor and memory model
 - Eight outstanding cache misses
 - Eight 32Mb chips, total 32MB, non-interleaved
- Execution-Driven Simulation
 - SPEC benchmarks (subset of integer)
 - SimpleScalar w/ detailed RDRAM timing and power models
 - Sixteen outstanding cache misses
 - Eight 256Mb chips, total 256MB, non-interleaved

Dual-state + Random Allocation (NT Traces)



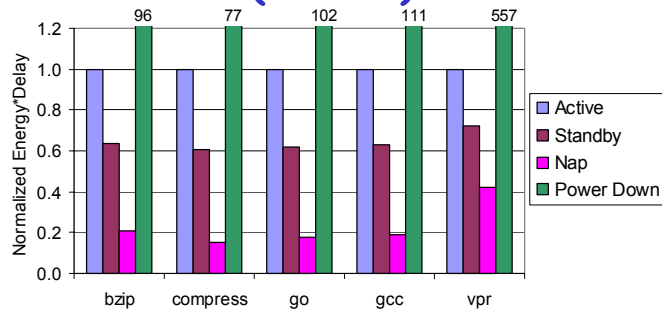
- Active to perform access, return to **base state**
- Nap is best ~85% reduction in E*D over full power
- Little change in run-time, most gains in energy/power

© 2003, Carla Schlatter Ellis

ESSES 2003

27

Dual-state + Random Allocation (SPEC)



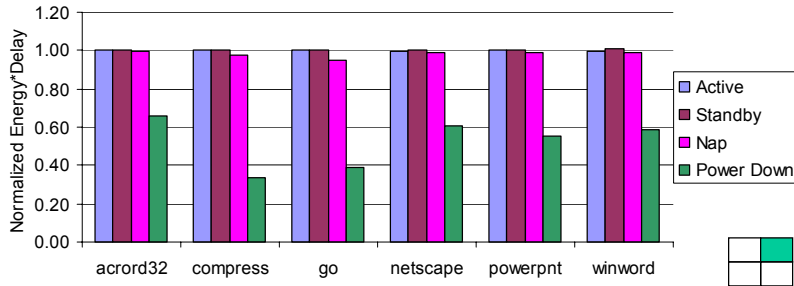
- All chips use same base state
- Nap is best 60% to 85% reduction in E*D over full power
- **Simple HW provides good improvement**

© 2003, Carla Schlatter Ellis

ESSES 2003

28

Benefits of Sequential Allocation (NT Traces)



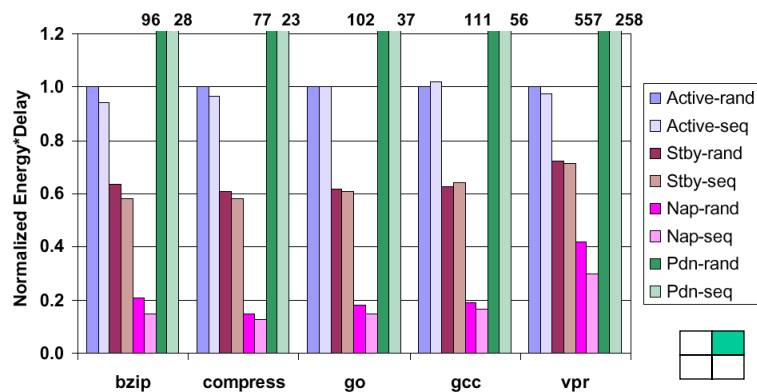
- Sequential normalized to random for same dual-state policy
- Very little benefit for most modes
 - Helps PowerDown, which is still really bad

© 2003, Carla Schlatter Ellis

ESSES 2003

29

Benefits of Sequential Allocation (SPEC)



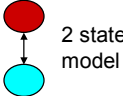
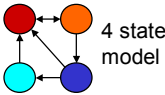
- 10% to 30% additional improvement for dual-state nap
- Some benefits due to cache effects

© 2003, Carla Schlatter Ellis

ESSES 2003

31

Results (Energy*Delay product)

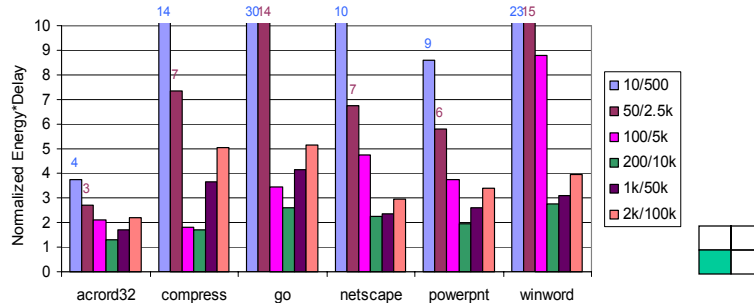
	Random Allocation	Sequential Allocation	
Dual-state Hardware	Nap is best 60%-85% improvement	10% to 30% improvement for nap. Base for future results	 2 state model
Quad-state Hardware	What about smarter HW?	Smart HW and OS support?	 4 state model

© 2003, Carla Schlatter Ellis

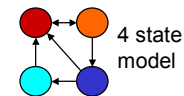
ESSES 2003

32

Quad-state HW + Random Allocation (NT) - Threshold Sensitivity



- Quad-state random vs. Dual-state nap sequential (best so far)
- With these thresholds, sophisticated HW is not enough.

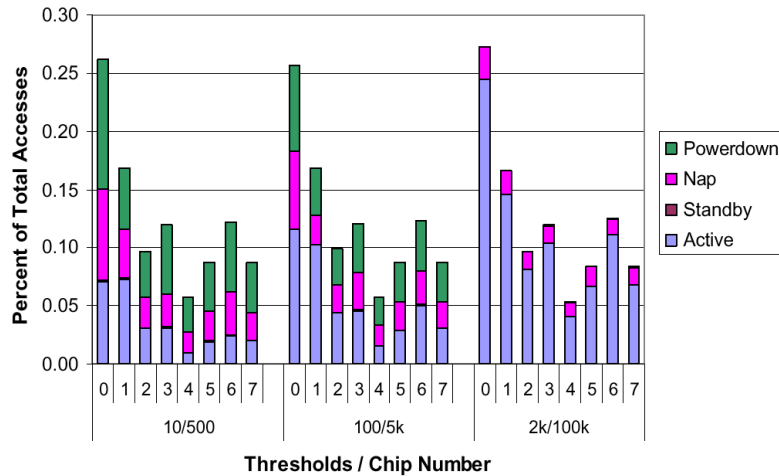


© 2003, Carla Schlatter Ellis

ESSES 2003

33

Access Distribution: Netscape



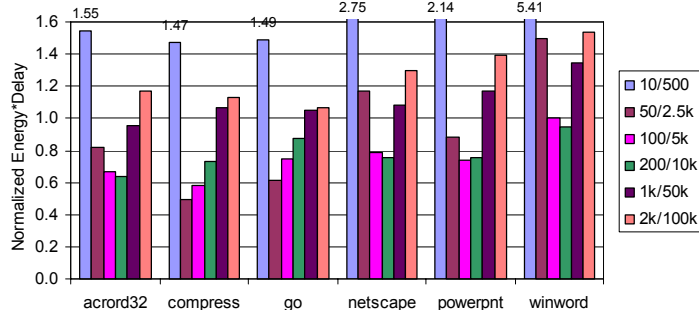
- Quad-state Random with different thresholds

© 2003, Carla Schlatter Ellis

ESSES 2003

34

Quad-state HW + Sequential Allocation (NT) - Threshold Sensitivity



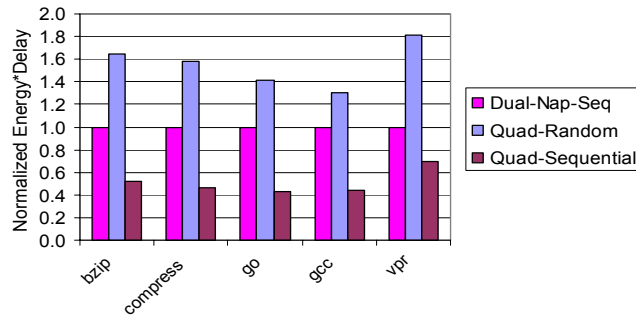
- Quad-state vs. Dual-state nap sequential
- Bars: active->nap / nap ->powerdown threshold values
- Additional 6% to 50% improvement over best dual-state

© 2003, Carla Schlatter Ellis

ESSES 2003

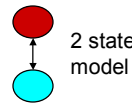
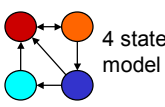
36

Quad-state HW (SPEC)



- Base: Dual-state Nap Sequential Allocation
- Thresholds: 0ns A->S; 750ns S->N; 375,000 N->P
- Quad-state + Sequential 30% to 55% additional improvement over dual-state nap sequential
- HW / SW Cooperation is important

Summary of Results (Energy*Delay product, RDRAM, ASPLOS00)

	Random Allocation	Sequential Allocation	
Dual-state Hardware	Nap is best dual-state policy 60%-85%	Additional 10% to 30% over Nap	 2 state model
Quad-state Hardware	Improvement not obvious, Could be equal to dual-state	Best Approach: 6% to 55% over dual-nap-seq, 80% to 99% over all active.	 4 state model

Conclusion

- New DRAM technologies provide opportunity
 - Multiple power states
- Simple hardware power mode management is effective
- Cooperative hardware / software (OS page allocation) solution is best

Questions?

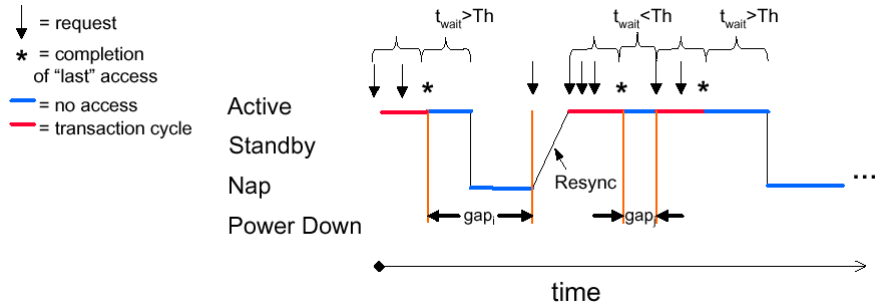
Outline Part 2

- More on memory controller design
 - How to determine best thresholds?
- Other possible OS page allocation policies
- Other OS policies – context switches
- Interaction with other system components (memory and DVS)

Controller Issues

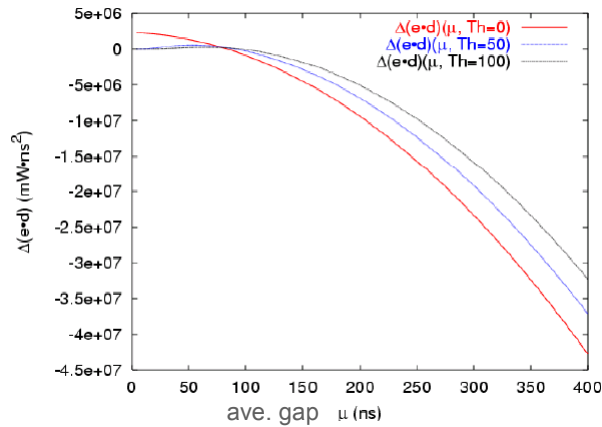
- Thresholds – it is not obvious what they should be / how to set them
- Are more sophisticated policies needed (in cache-based systems)?
- How much information about the access patterns is needed?

Determining Thresholds in Power State Transitions



If (gap > benefit boundary) threshold = 0
 // but gap unknown
 else threshold = ∞

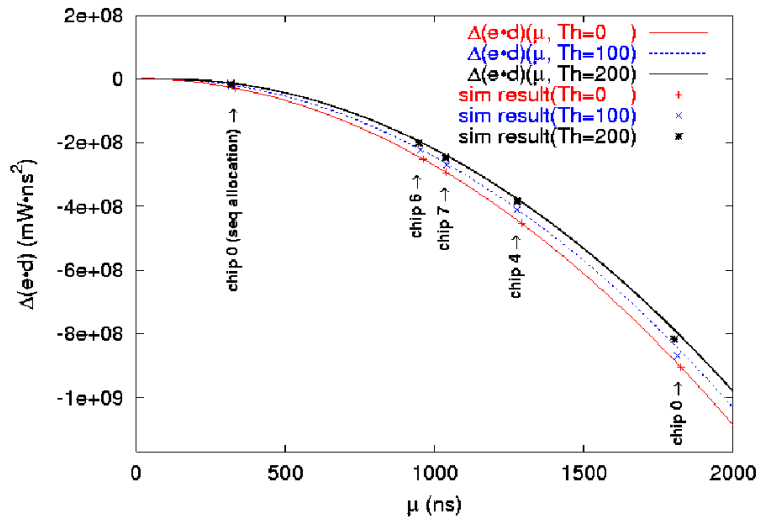
Change in $E \cdot D$ vs. Ave. Gap



Lower is better

Based on analytical model, for exponential gap distributions, large average gap, $Th = 0$ is best [ISLPED01]
 // unfortunately, real gap distributions are not generally exponential

Model Validation

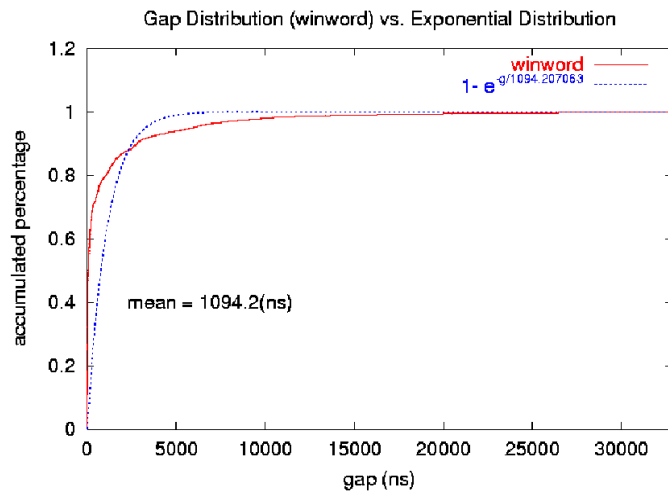


© 2003, Carla Schlatter Ellis

ESSES 2003

46

Access Patterns Not Always Exponential

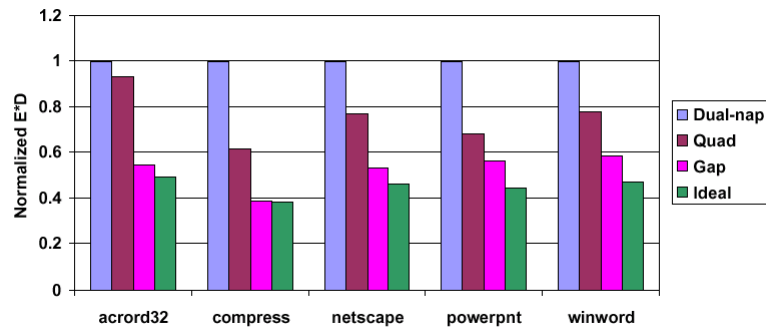


© 2003, Carla Schlatter Ellis

ESSES 2003

47

History-based Prediction in Controller



- Sequential page allocation, NT traces
- Ideal: offline policy, delay = all active, minimize power
- Gap policy: History-based prediction
 - If predicted gap > benefit boundary, immediately transition

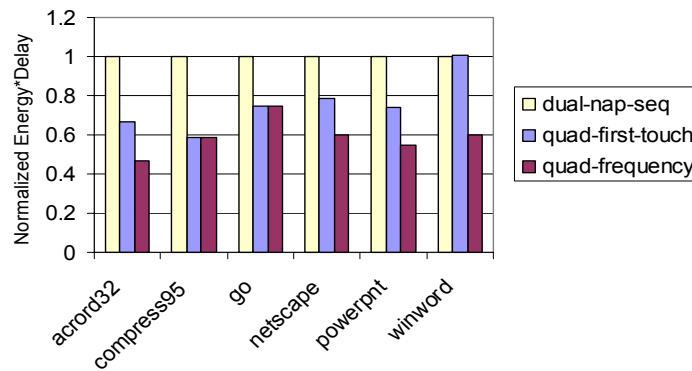
Outline Part 2

- More on memory controller design
 - How to determine best thresholds?
- Other possible OS page allocation policies
- Other OS policies – context switches
- Interaction with other system components (memory and DVS)

Better Page Allocation Policies?

- Intuitively, first-touch will not always be best
- Allow movement after first-touch as “corrections”
- Frequency-based allocation
- Preliminary results
 - Offline algorithm: sort by page count
 - Allocate sequentially in decreasing order
 - Packs most frequently accessed pages into first chip
 - Provides insight into potential benefits (if any) of page movement and motivate an on-line algorithm

Frequency vs. First-Touch (NT)



- Base: dual-state nap sequential
- Thresholds: 100 A->N; 5,000 N->PDN
- Opportunity for further improvements beyond first-touch

Hardware Support for Page Movement

- Data collection hardware
 - Reserve n pages in chip 0 (n=128)
 - 10-bit saturating counter per physical page
- On-line Algorithm
 - Warmup for 100ms, sample accesses for 2ms
 - Sort counts, move 128 most frequent pages to reserved pages in hot chip, repack others into minimum number of chips
- Preliminary experiments and results
 - Use 0.011ms and 0.008mJ for page move
 - 10% improvement for winword
 - Need to consider in execution-driven simulator

Outline Part 2

- More on memory controller design
 - How to determine best thresholds?
- Other possible OS page allocation policies
- Other OS policies – context switches
- Interaction with other system components (memory and DVS)

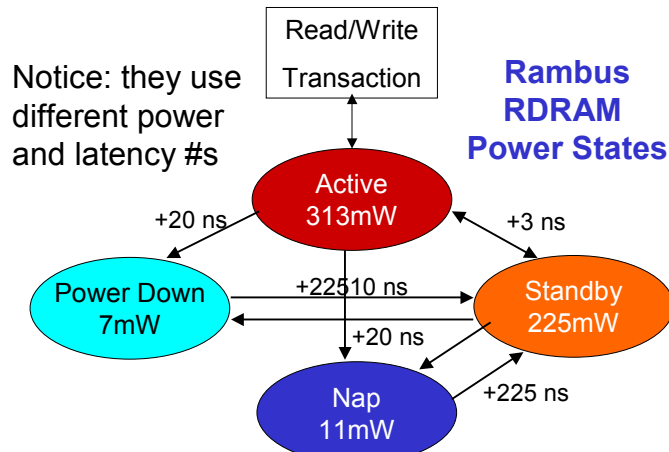
Power-Aware Virtual Memory Based On Context Switches

Huang, Pillai, Shin, “Design and Implementation of Power-Aware Virtual Memory”, USENIX 03.

Basic Idea

- Power state transitions under SW control (not HW controller)
- Treated explicitly as memory hierarchy: a process's active set of nodes is kept in higher power state
- Size of active node set is kept small by grouping process's pages in nodes together – “energy footprint”
 - Page mapping - viewed as NUMA layer for implementation
 - Active set of pages, α_i , put on preferred nodes, ρ_i
- At context switch time, hide latency of transitioning
 - Transition the union of active sets of the next-to-run and likely next-after-that processes to standby (pre-charging) from nap
 - Overlap transitions with other context switch overhead

Rambus RDRAM



© 2003, Carla Schlatter Ellis

ESSES 2003

56

RDRAM Active Components

	Refresh	Clock	Row decoder	Col decoder
Active	X	X	X	X
Standby	X	X	X	
Nap	X	X		
Pwrdn	X			

© 2003, Carla Schlatter Ellis

ESSES 2003

57

Determining Active Nodes

- A node is active iff at least one page from the node is mapped into process i 's address space.
- Table maintained whenever page is mapped in or unmapped in kernel.
- Alternatives rejected due to overhead:
 - Extra page faults
 - Page table scans
- Overhead is only one incr/decr per mapping/unmapping op

count	n_0	n_1	...	n_{15}
p_0	108	2		17
...				
p_n	193	240		4322

Implementation Details

Problem:

DLLs and files shared by multiple processes (buffer cache) become scattered all over memory with a straightforward assignment of incoming pages to process's active nodes – large energy footprints afterall.

Implementation Details

Solutions:

- DLL Aggregation
 - Special case DLLs by allocating Sequential first-touch in low-numbered nodes
- Migration
 - Kernel thread – kmigrated – running in background when system is idle (waking up every 3s)
 - Scans pages used by each process, migrating if conditions met
 - Private page not on
 - Shared page outside $\cap \rho_i$

Process	ρ	α							
<i>syslog</i>	14	0(3)	8(5)	9(51)	10(1)	11(1)	13(3)	14(76)	
<i>login</i>	11	0(12)	8(7)	9(112)	11(102)	12(5)	14(20)	15(1)	
<i>startx</i>	13	0(21)	7(12)	8(3)	9(7)	10(12)	11(25)	13(131)	14(43)
<i>X</i>	12	0(125)	7(23)	8(47)	9(76)	10(223)	11(19)	12(1928)	13(82)
			14(77)	15(182)					
<i>sawfish</i>	10	0(180)	7(5)	8(12)	9(1)	10(278)	13(25)	14(5)	15(233)
<i>vim</i>	10,15	0(12)	9(218)	10(5322)	14(22)	15(4322)			
...

Table 2: A snapshot of processes' node usage pattern

Process	ρ	α					
<i>syslog</i>	14	0(108)	1(2)	11(13)	14(17)		
<i>login</i>	11	0(148)	1(4)	11(98)	15(9)		
<i>startx</i>	13	0(217)	1(12)	13(25)			
<i>X</i>	12	0(125)	1(417)	9(76)	11(793)	12(928)	13(169) 14(15)
<i>sawfish</i>	10	0(193)	1(281)	10(179)	13(25)	14(11)	15(50)
<i>vim</i>	10,15	0(12)	1(240)	10(5322)	15(4322)		
...

Table 3: Effect of aggregating pages used by DLLs.

Process	ρ	α		
<i>syslog</i>	14	0(15)	14(125)	
<i>login</i>	11	0(76)	11(183)	
<i>startx</i>	13	0(172)	13(82)	
<i>X</i>	12	0(225)	1(2)	12(2220)
<i>sawfish</i>	10	0(207)	1(56)	10(436)
<i>vim</i>	10,15	0(12)	1(240)	10(5322) 15(4322)
...

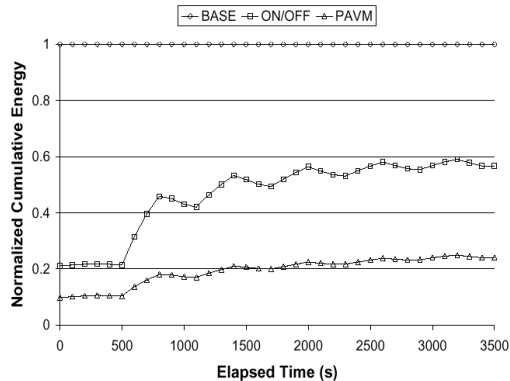
Table 4: Effect of library aggregation with page migration.

Evaluation Methodology

- Linux implementation
- Measurements/counts taken of events and energy results calculated (not measured)
- Metric – energy used by memory (only).
- Workloads – 3 mixes: light (editing, browsing, MP3), poweruser (light + kernel compile), multimedia (playing mpeg movie)
- Platform – 16 nodes, 512MB of RDRAM
- Not considered: DMA and kernel maintenance threads

Results

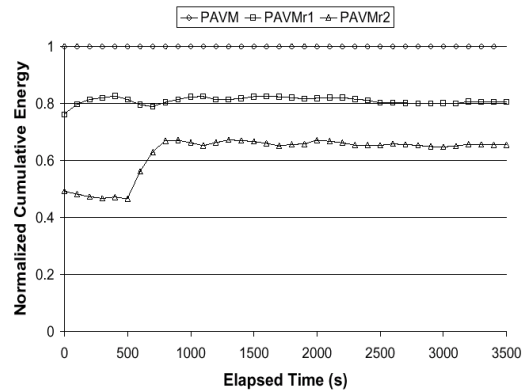
- Base – standby when not accessing
- On/Off – nap when system idle
- PAVM



(b) Poweruser workload

Results

- PAVM
- PAVMr1 - DLL aggregation
- PAVMr2 – both DLL aggregation & migration



(b) Poweruser workload

Results

	Light	Poweruser	Multimedia
<i>Base</i>	4100 mW	4118 mW	4230 mW
<i>On/Off</i>	892 mW	2324 mW	3991 mW
<i>PAVM</i>	465 mW	986 mW	2687 mW
<i>PAVMr1</i>	397 mW	791 mW	2442 mW
<i>PAVMr2</i>	237 mW	646 mW	1725 mW

Conclusions

- Multiprogramming environment.
- Basic PAVM: save 34-89% energy of 16 node RDRAM
- With optimizations: additional 20-50%
- Works with other kinds of power-aware memory devices

Discussion: Alternatives to
this scheme for migration?
Replication? On mapping?

Outline Part 2

- More on memory controller design
 - How to determine best thresholds?
- Other possible OS page allocation policies
- Other OS policies – context switches
- Interaction with other system components (memory and DVS)



Tension between Slow&Steady and Bursty



- Bursty request patterns are “good for” power state transition devices
- Slow&Steady is the idea behind DVS
- What happens when the two execution behaviors *depend on* each other?
 - The CPU generates the memory request stream and stalls when not supplied in time.
- We consider this in the context of power-aware DRAM and voltage scaling CPUs

Architectural Model

CPU model based on

Xscale:

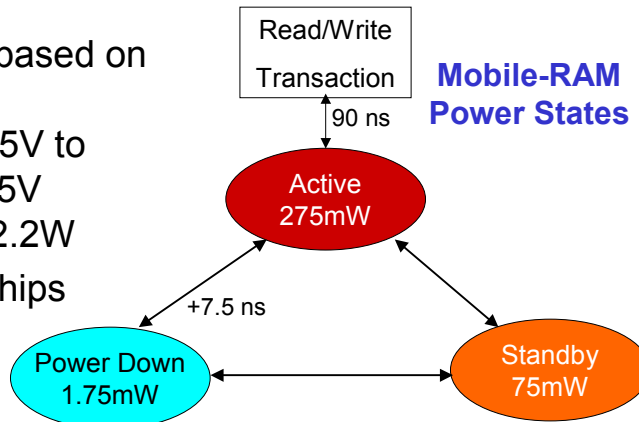
50MHz, .65V to

1GHz, 1.75V

15mW to 2.2W

Memory: 2 chips

64MB



© 2003, Carla Schlatter Ellis

ESSES 2003

70

Methodology

- PowerAnalyzer simulator modified with memory model
- Workload – multimedia applications from the MediaBench suite.
 - MPEG decoder used in presented results
 - 15 frames/sec, period of 66ms
 - Input file with 3 frames of the 3 types (I-, B-, P-frames)
 - All Xscale frequencies will meet deadline.
 - Synthetic program to investigate miss rates

© 2003, Carla Schlatter Ellis

ESSES 2003

71

What effect does the memory technology have on DVS?

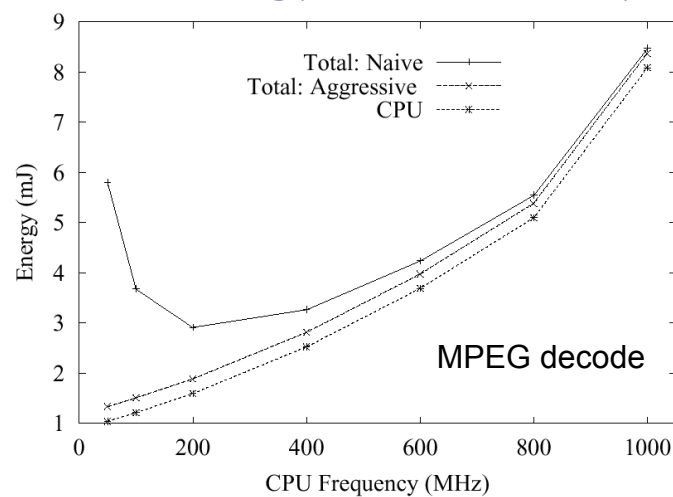
- Scheduling idea: pick slowest speed you can without violating deadline (or other performance constraint)
- Consider memory effects
 - Old fashioned all-active memory
 - Naïve – goes into powerdown when CPU halts
 - Aggressive – managed power-states
- Previous work investigated performance impact on high-end. [Martin, Pouwelse]

© 2003, Carla Schlatter Ellis

ESSES 2003

72

Effect of PA-Memory with DVS on Total Energy (CPU+Memory)

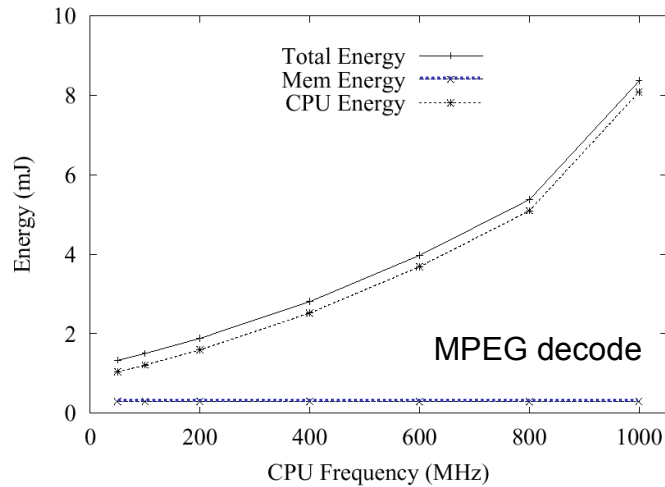


© 2003, Carla Schlatter Ellis

ESSES 2003

75

XScale and Mobile-RAM



© 2003, Carla Schlatter Ellis

ESSES 2003

76

What effect does DVS have on the design of the memory controller?

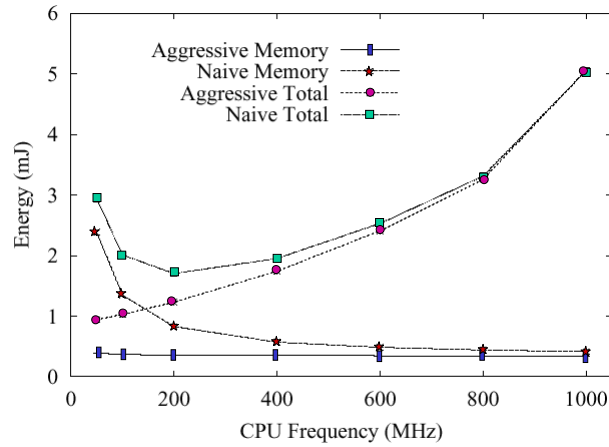
- Stretching out or speeding up execution changes the length of gaps in the memory access pattern
 - Latency tolerance at different speeds
 - Best memory controller policy may depend on average gap

© 2003, Carla Schlatter Ellis

ESSES 2003

77

Affect of Miss Rates on Controller Policy (Synthetic Program)



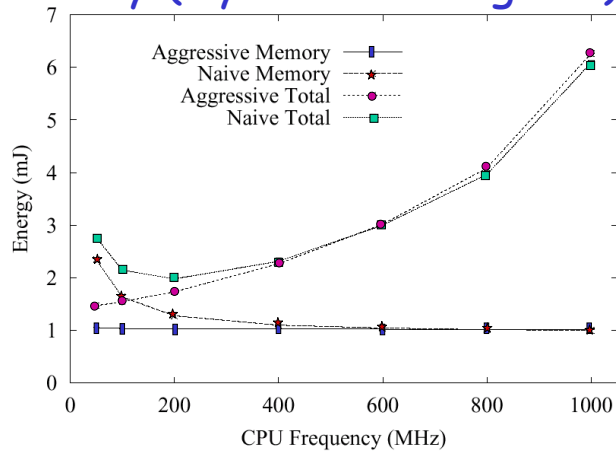
a) 2% Miss Ratio

© 2003, Carla Schlatter Ellis

ESSES 2003

78

Affect of Miss Rates on Controller Policy (Synthetic Program)



c) 16% Miss Ratio

© 2003, Carla Schlatter Ellis

ESSES 2003

79

DVS/PA-Memory Synergy

- With power-aware memory considered, the lowest speed/voltage is not necessarily lowest energy choice.
- Memory access behavior must enter into the speed-setting decision
- The best memory controller policy may depend on the speed setting. Memory controller policy should be adaptive in that case.

Questions?