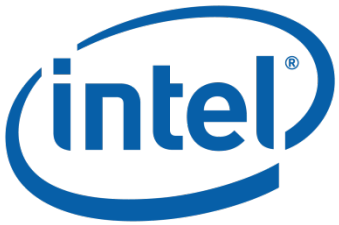
Memory Power Management via Dynamic Voltage/Frequency Scaling

Howard David (Intel)
Eugene Gorbatov (Intel)
Ulf R. Hanebutte (Intel)

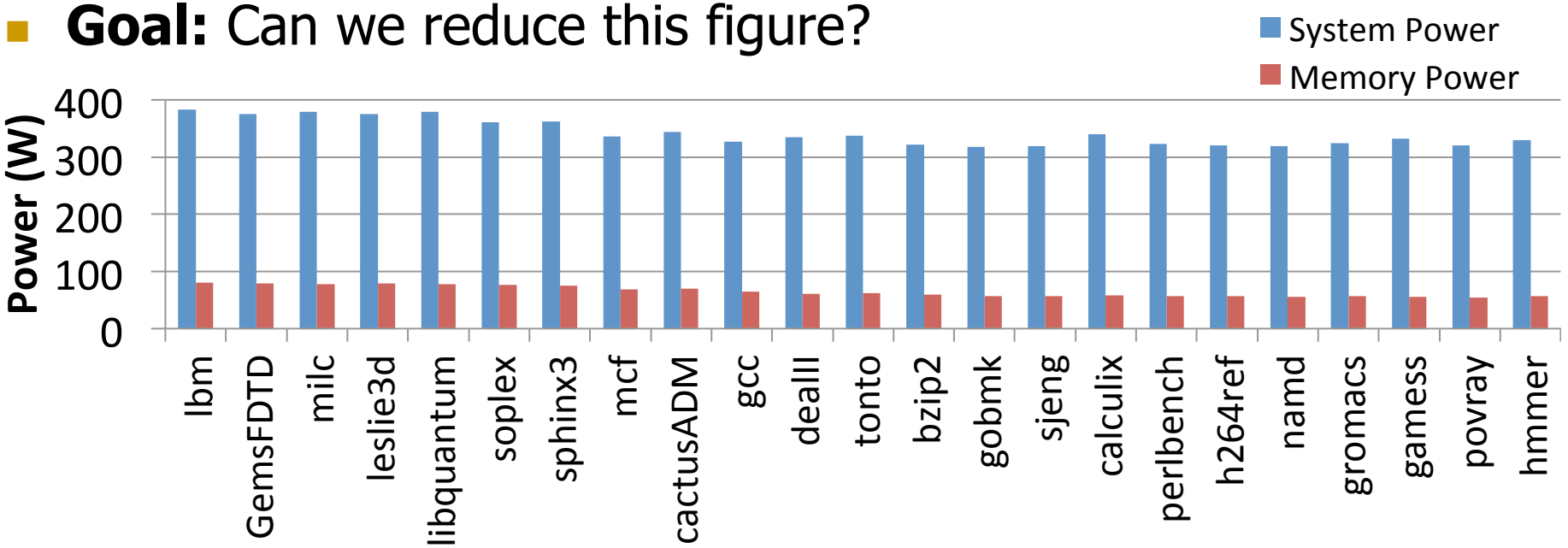
Chris Fallin (CMU)
Onur Mutlu (CMU)



SAFARI
Carnegie Mellon

Memory Power is Significant

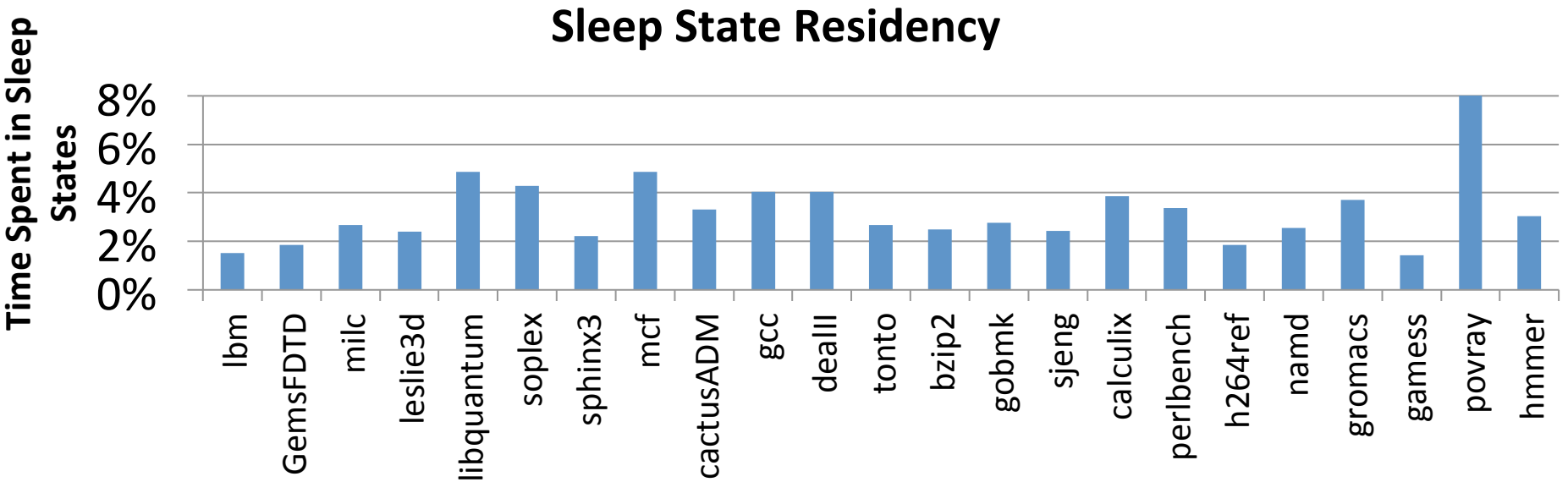
- Power consumption is a primary concern in modern servers
- Many works: CPU, whole-system or cluster-level approach
- But memory power is largely unaddressed
- Our server system*: memory is 19% of system power (avg)
  - Some work notes up to 40% of total system power
- **Goal:** Can we reduce this figure?



*Dual 4-core Intel Xeon®, 48GB DDR3 (12 DIMMs), SPEC CPU2006, all cores active.
Measured AC power, analytically modeled memory power.

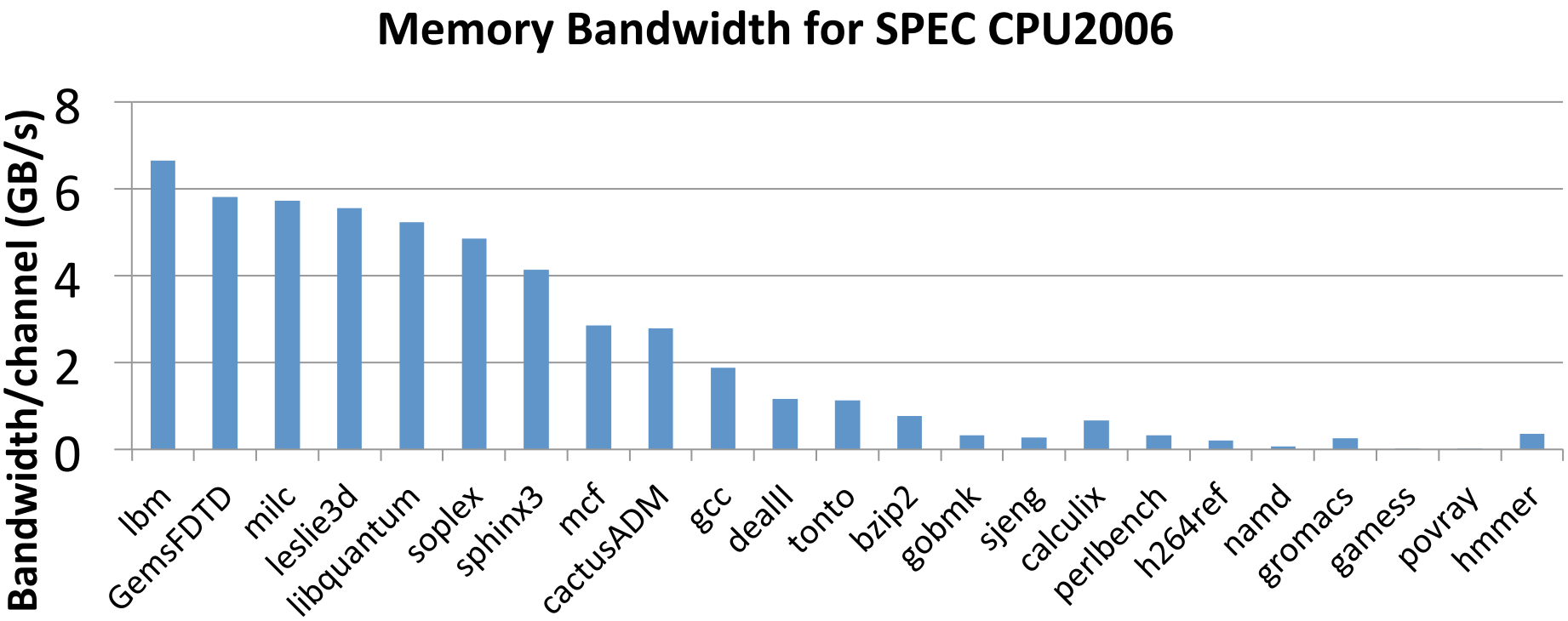
Existing Solution: Memory Sleep States?

- Most memory energy-efficiency work uses sleep states
  - Shut down DRAM devices when no memory requests active
- But, even low-memory-bandwidth workloads keep memory awake
  - Idle periods between requests diminish in multicore workloads
  - CPU-bound workloads/phases rarely completely cache-resident



Memory Bandwidth Varies Widely

- Workload memory bandwidth requirements vary widely



- Memory system is provisioned for peak capacity

  → often underutilized

Memory Power can be Scaled Down

- DDR can operate at multiple frequencies → reduce power
  - Lower frequency directly reduces switching power
  - Lower frequency allows for lower voltage
  - Comparable to CPU DVFS

CPU Voltage/ Freq.	System Power
↓ 15%	↓ 9.9%

Memory Freq.	System Power
↓ 40%	↓ 7.6%

- Frequency scaling increases latency → reduce performance
  - Memory storage array is asynchronous
  - But, bus transfer depends on frequency
  - When bus bandwidth is bottleneck, performance suffers

Observations So Far

- Memory power is a significant portion of total power
  - 19% (avg) in our system, up to 40% noted in other works
- Sleep state residency is low in many workloads
  - Multicore workloads reduce idle periods
  - CPU-bound applications send requests frequently enough to keep memory devices awake
- Memory bandwidth demand is very low in some workloads
- Memory power is reduced by frequency scaling
  - And voltage scaling can give further reductions

DVFS for Memory

- **Key Idea:** observe memory bandwidth utilization, then adjust memory frequency/voltage, to reduce power with minimal performance loss

  → **Dynamic Voltage/Frequency Scaling (DVFS) for memory**

- **Goal in this work**:
  - Implement DVFS in the memory system, by:
  - Developing a simple control algorithm to exploit opportunity for reduced memory frequency/voltage by observing behavior
  - Evaluating the proposed algorithm on a real system

Outline

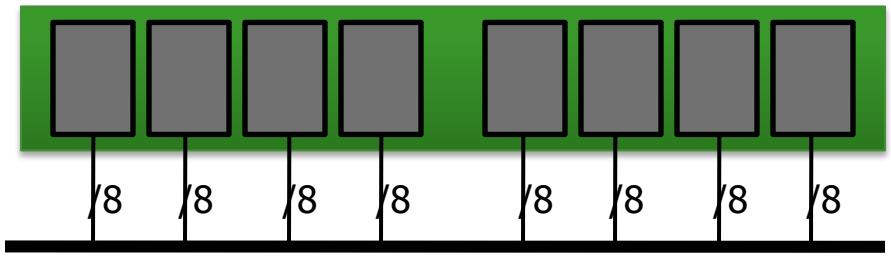
- Motivation
- Background and Characterization
  - DRAM Operation
  - DRAM Power
  - Frequency and Voltage Scaling
- Performance Effects of Frequency Scaling
- Frequency Control Algorithm
- Evaluation and Conclusions

Outline

- Motivation
- Background and Characterization
  - DRAM Operation
  - DRAM Power
  - Frequency and Voltage Scaling
- Performance Effects of Frequency Scaling
- Frequency Control Algorithm
- Evaluation and Conclusions

DRAM Operation

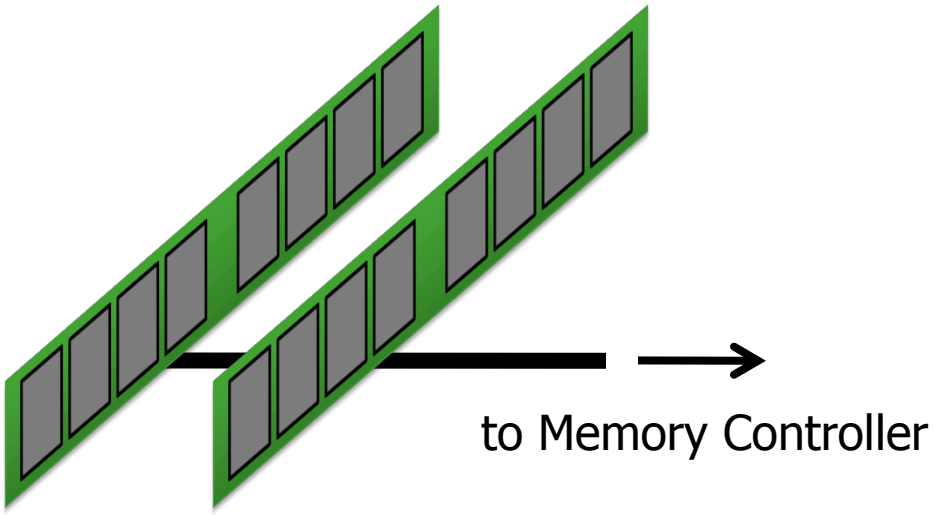
- Main memory consists of DIMMs of DRAM devices
- Each DIMM is attached to a memory bus (channel)



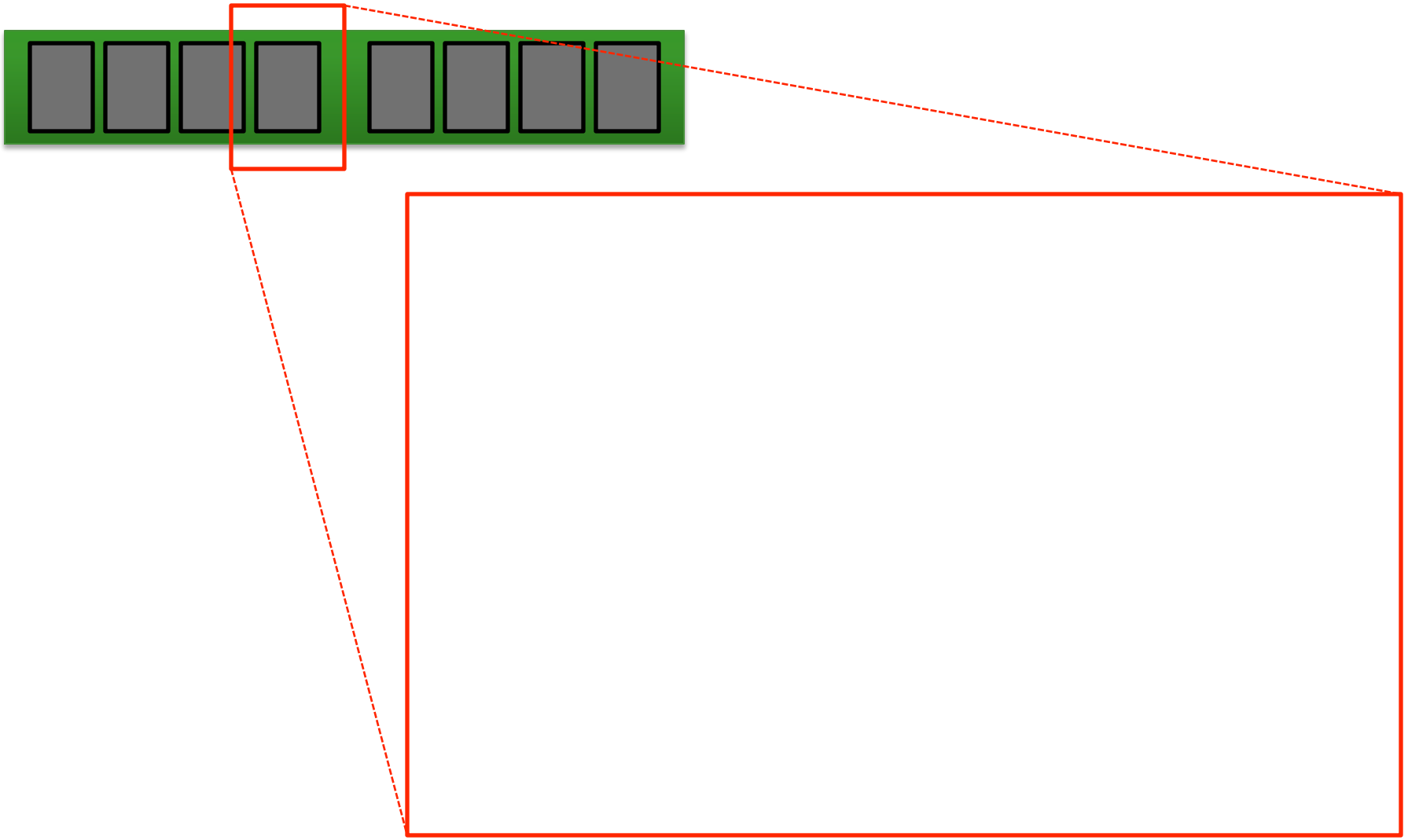
Memory Bus (64 bits)

DRAM Operation

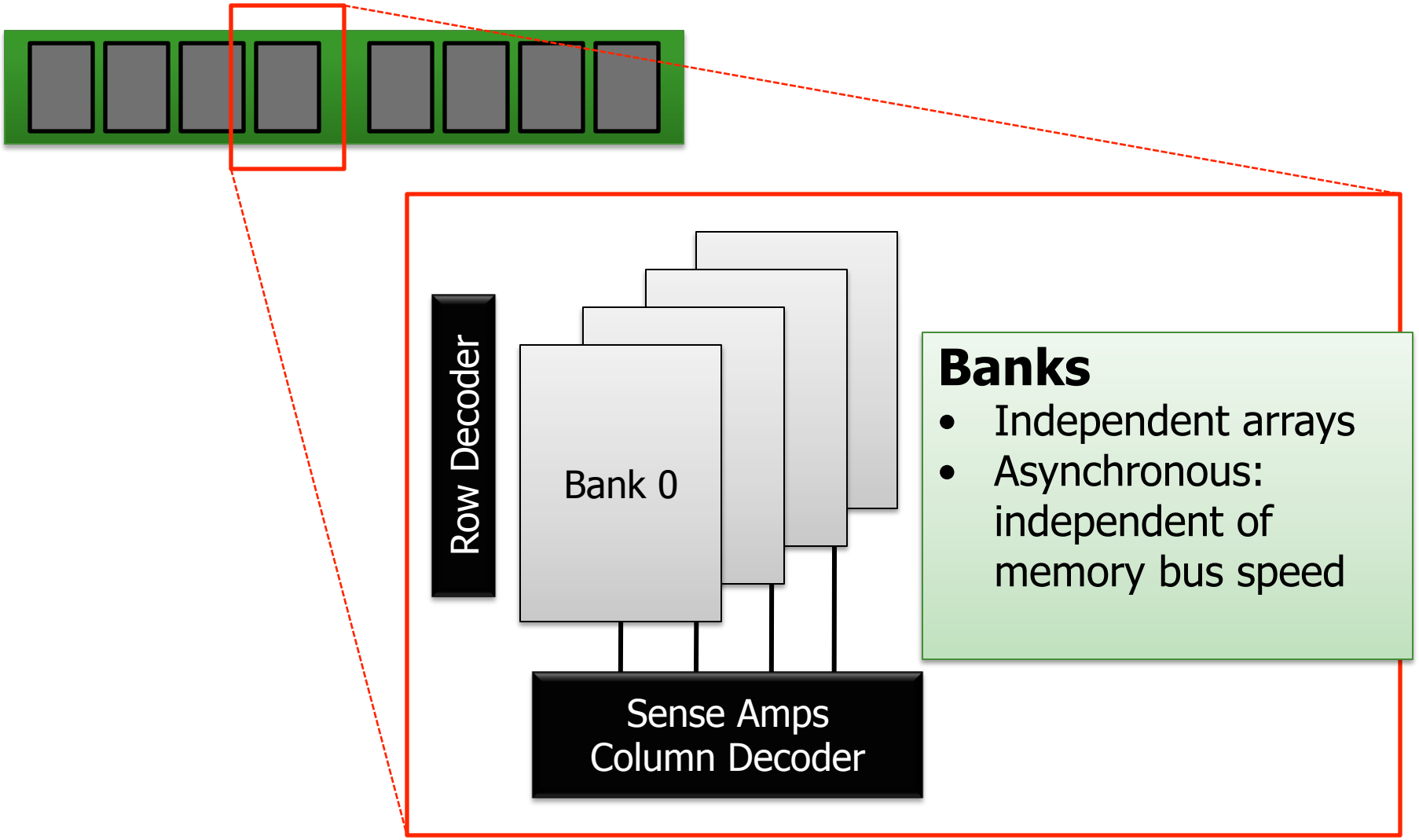
- Main memory consists of DIMMs of DRAM devices
- Each DIMM is attached to a memory bus (channel)
- Multiple DIMMs can connect to one channel



Inside a DRAM Device



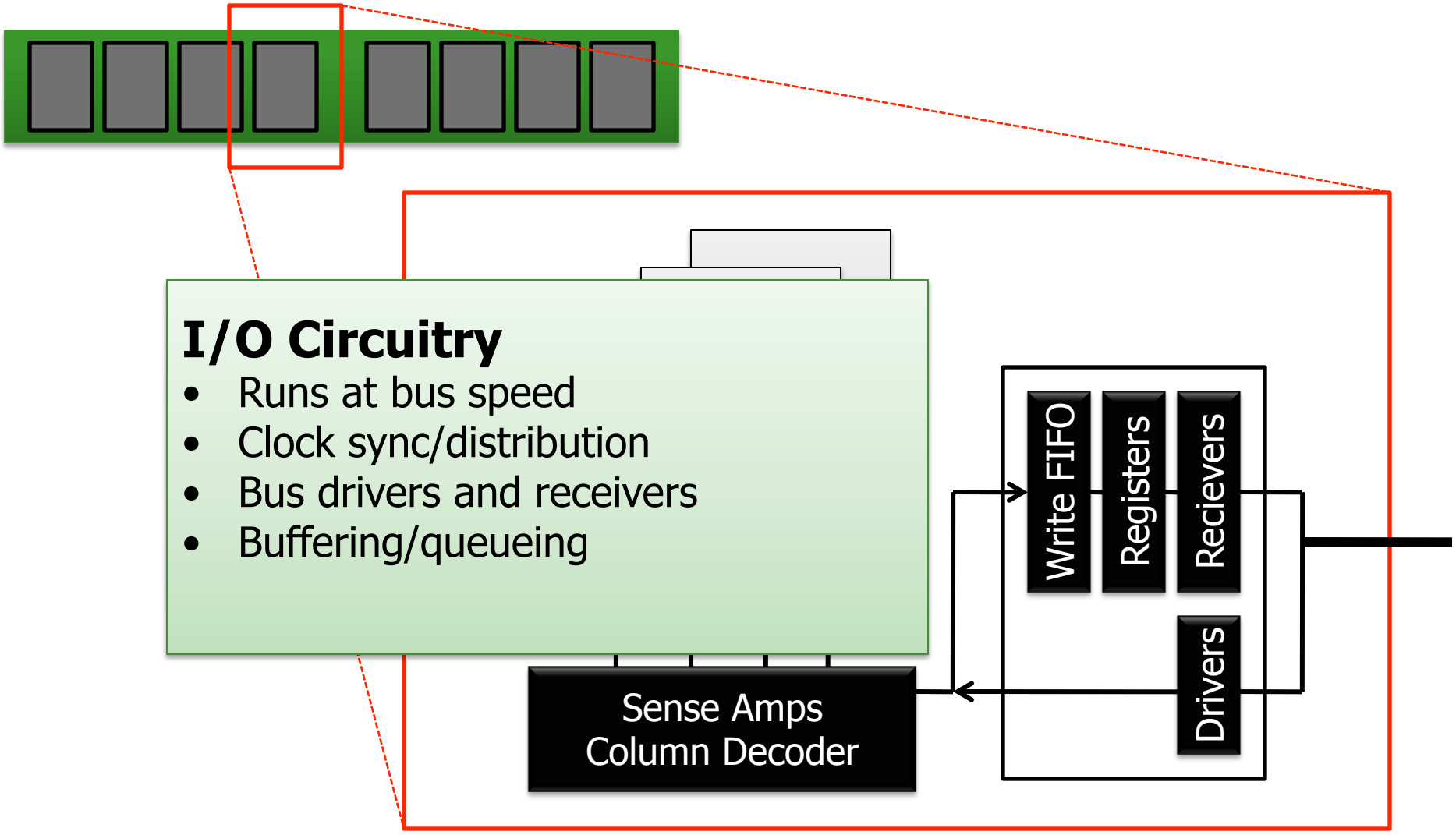
Inside a DRAM Device



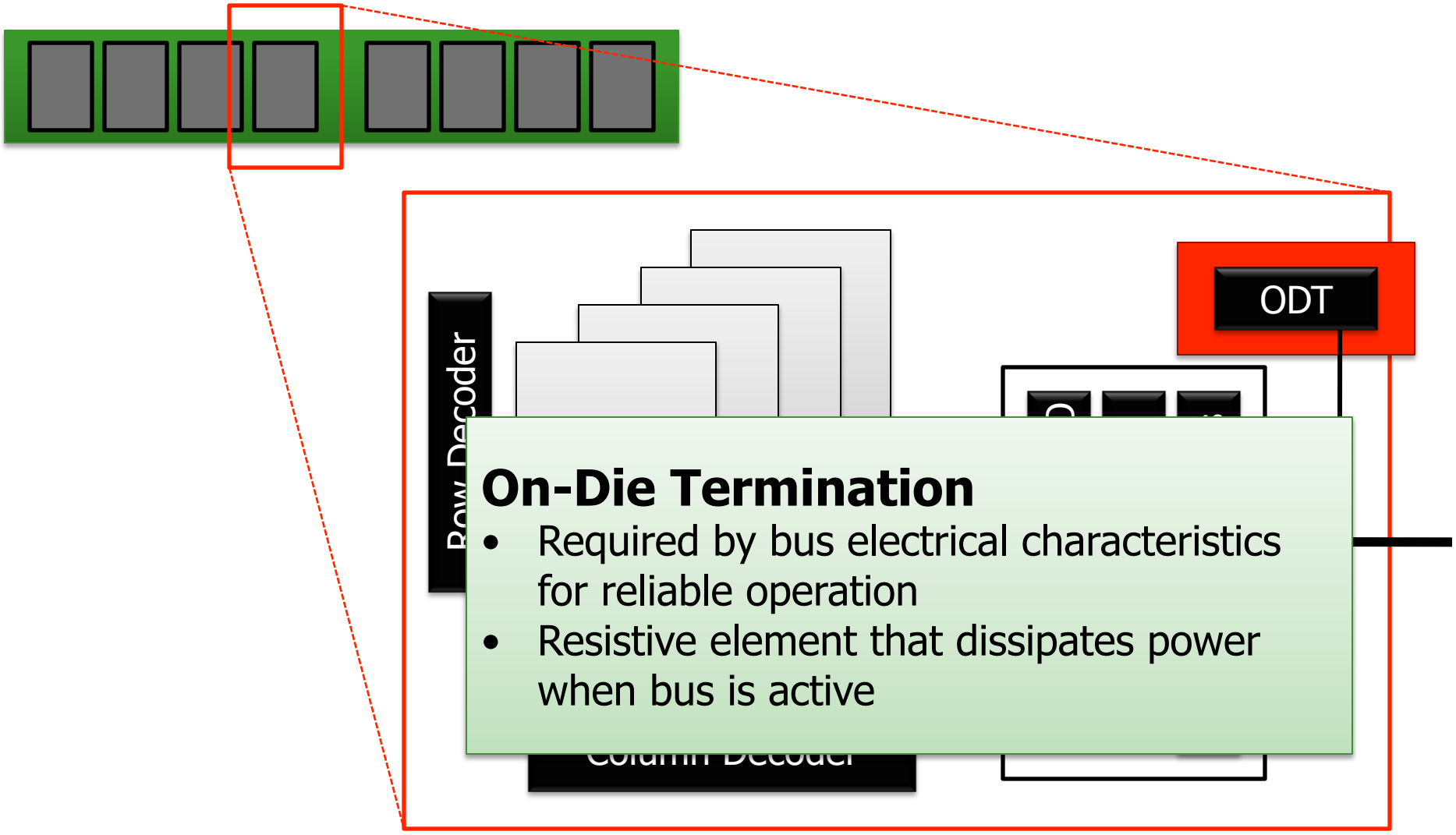
Banks

- Independent arrays
- Asynchronous: independent of memory bus speed

Inside a DRAM Device



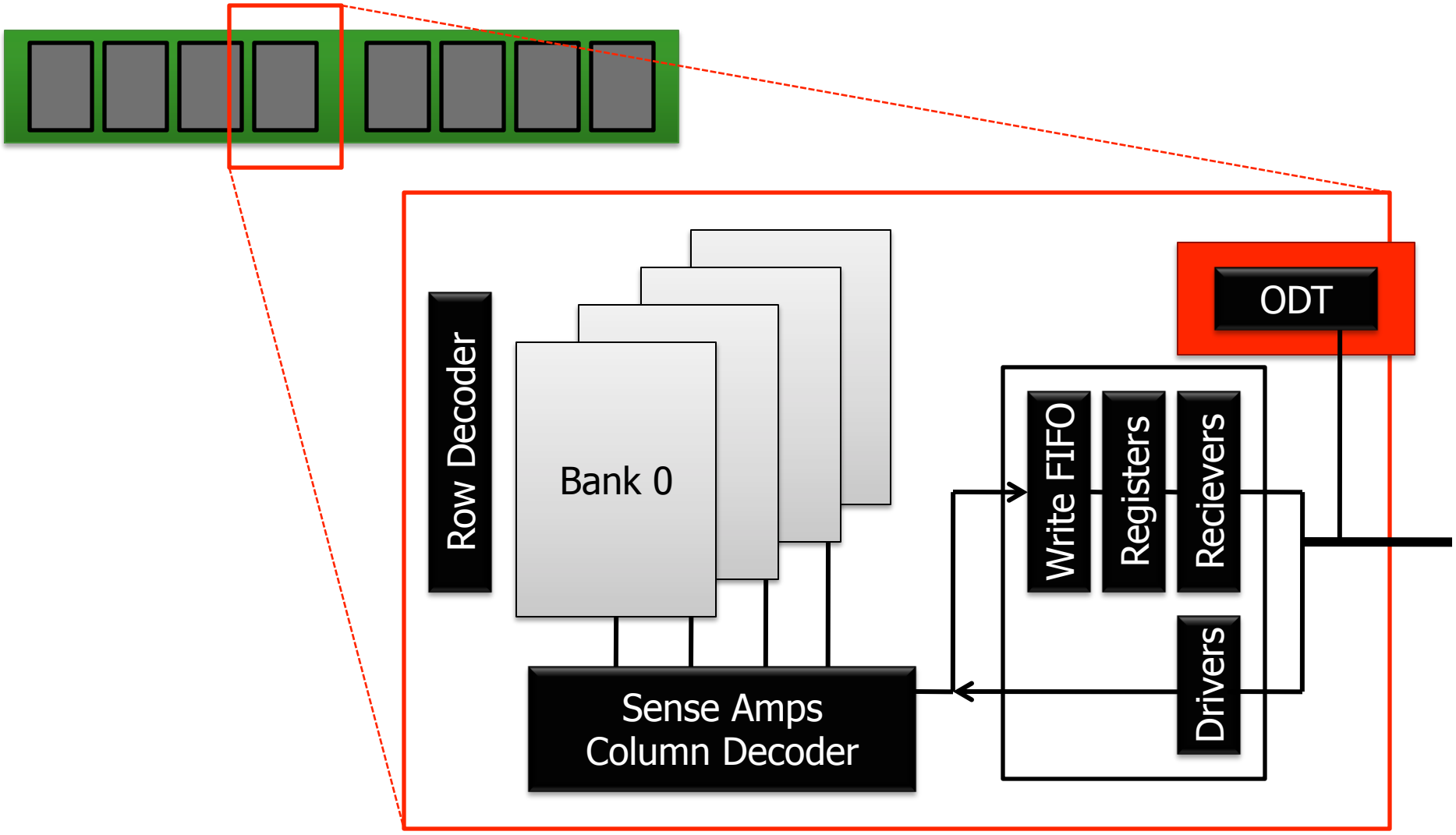
Inside a DRAM Device



On-Die Termination

- Required by bus electrical characteristics for reliable operation
- Resistive element that dissipates power when bus is active

Inside a DRAM Device

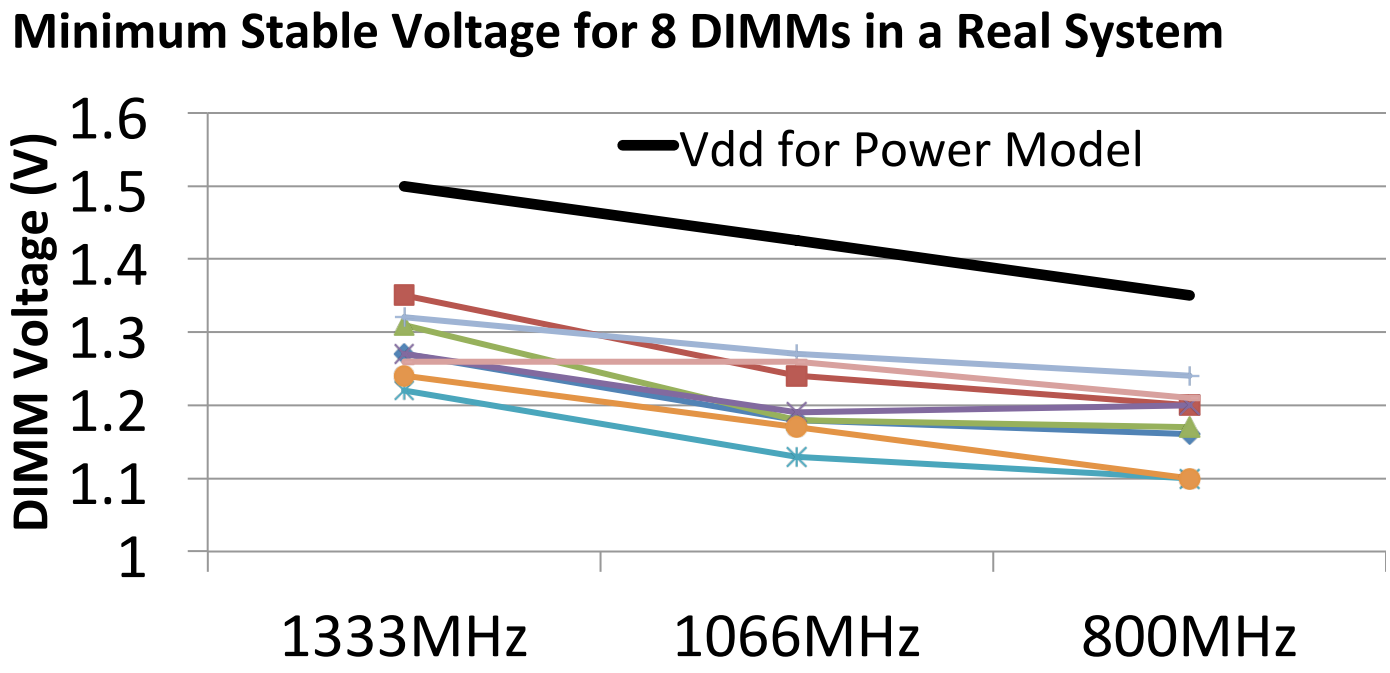


Effect of Frequency Scaling on Power

- **Reduced memory bus frequency:**
- Does not affect bank power:
  - Constant energy per operation
  - Depends only on utilized memory bandwidth
- Decreases I/O power:
  - Dynamic power in bus interface and clock circuitry reduces due to less frequent switching
- Increases termination power:
  - Same data takes longer to transfer
  - Hence, bus utilization increases
- Tradeoff between I/O and termination results in a net power reduction at lower frequencies

Effects of Voltage Scaling on Power

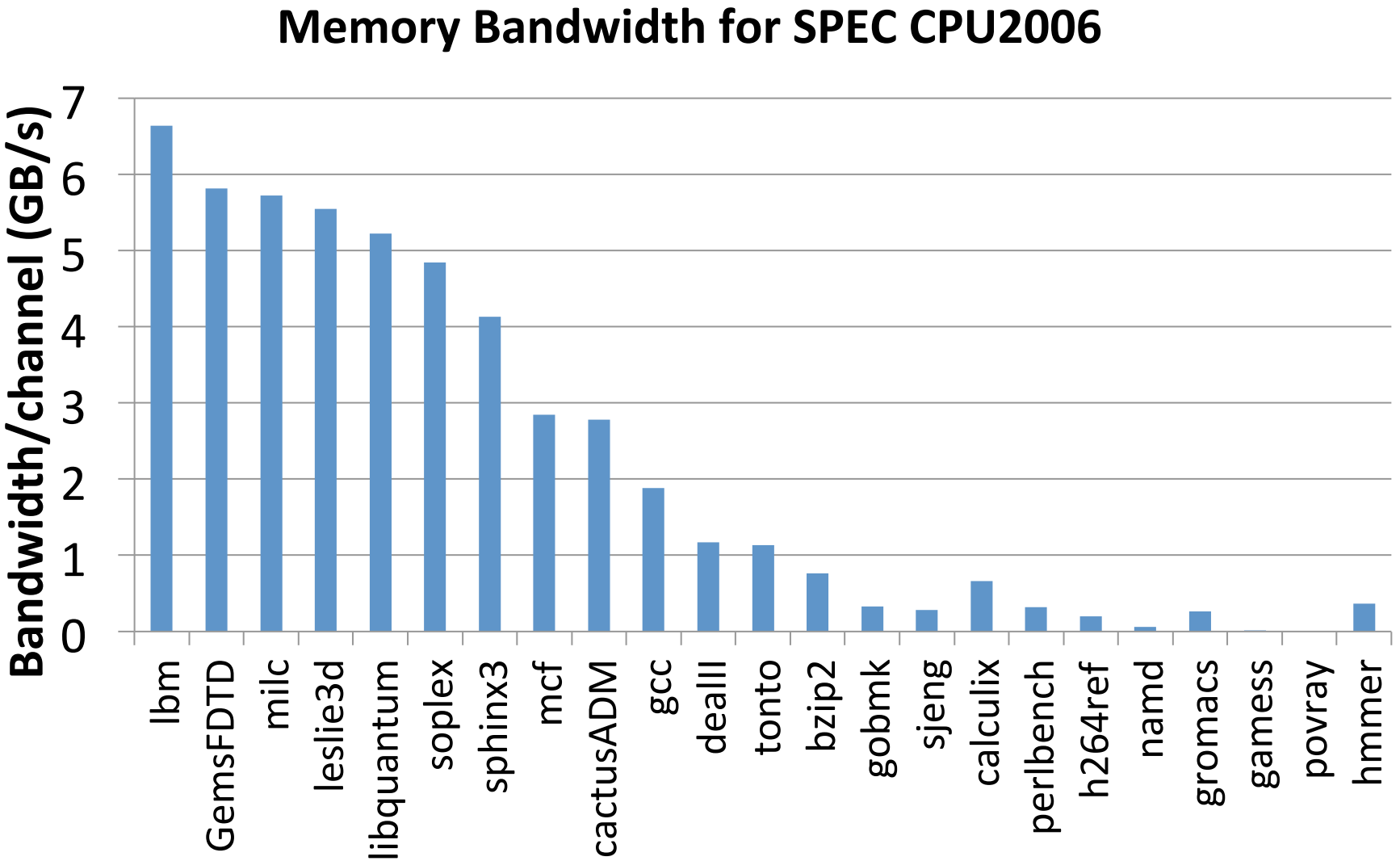
- Voltage scaling further reduces power because all parts of memory devices will draw less current (at less voltage)
- Voltage reduction is possible because stable operation requires lower voltage at lower frequency:



Outline

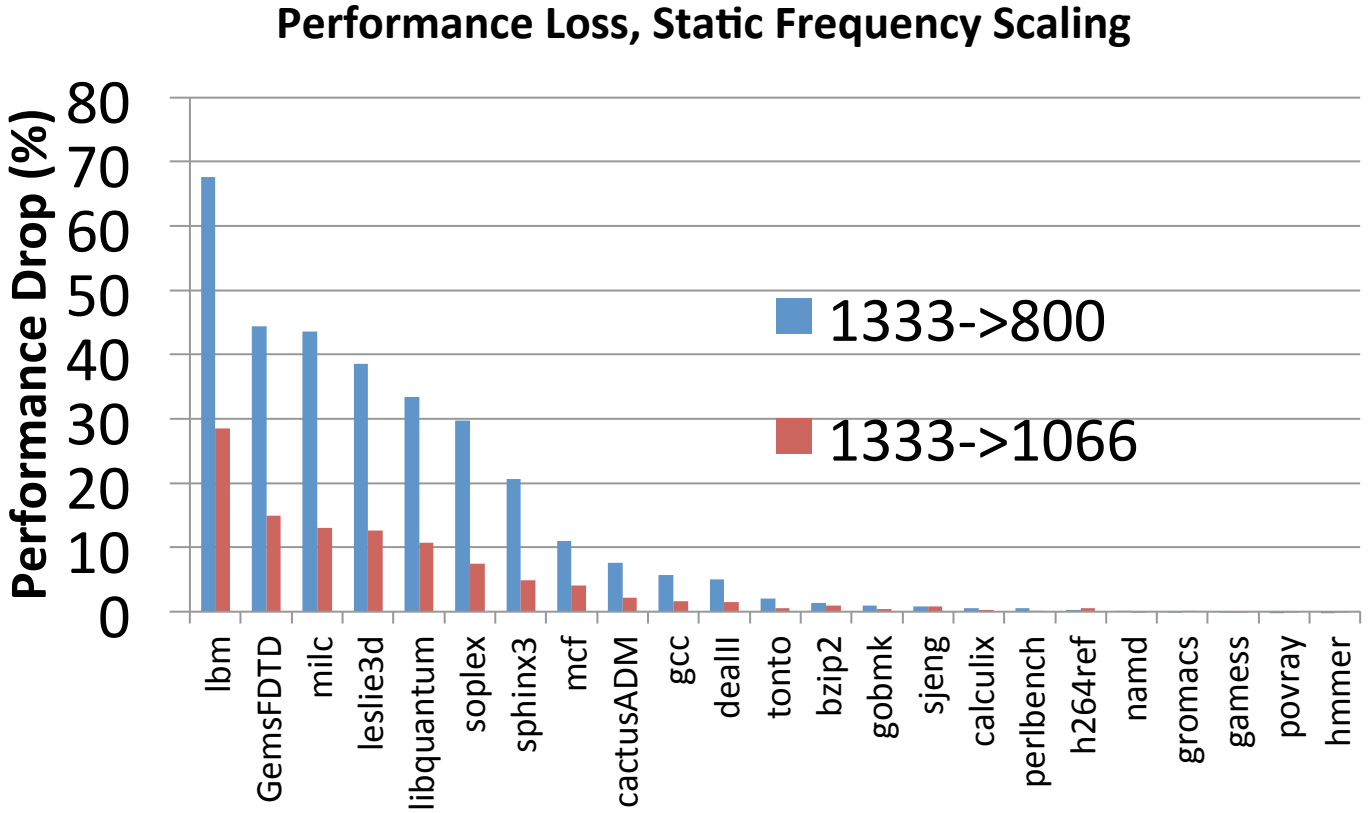
- Motivation
- Background and Characterization
  - DRAM Operation
  - DRAM Power
  - Frequency and Voltage Scaling
- Performance Effects of Frequency Scaling
- Frequency Control Algorithm
- Evaluation and Conclusions

How Much Memory Bandwidth is Needed?



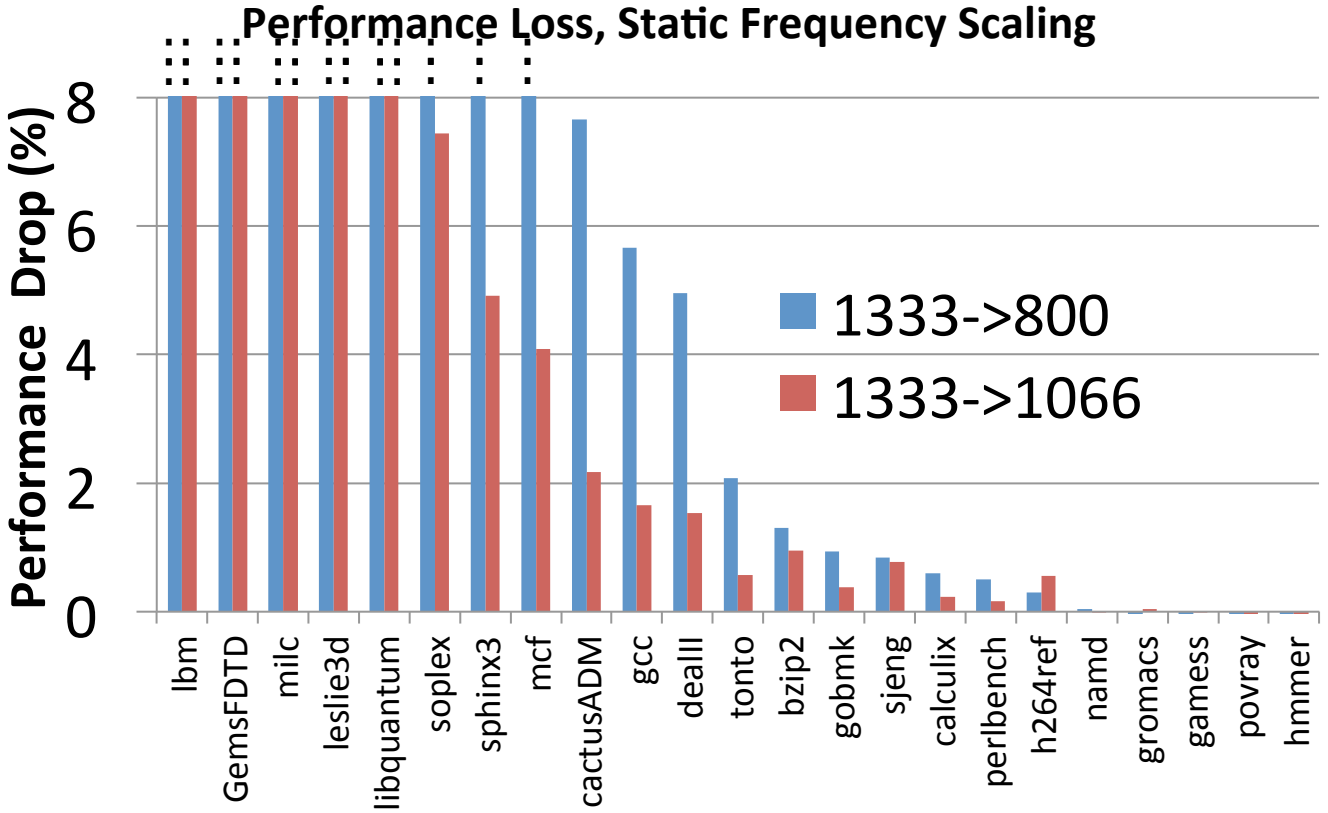
Performance Impact of Static Frequency Scaling

- Performance impact is proportional to bandwidth demand
- Many workloads tolerate lower frequency with minimal performance drop



Performance Impact of Static Frequency Scaling

- Performance impact is proportional to bandwidth demand
- Many workloads tolerate lower frequency with minimal performance drop

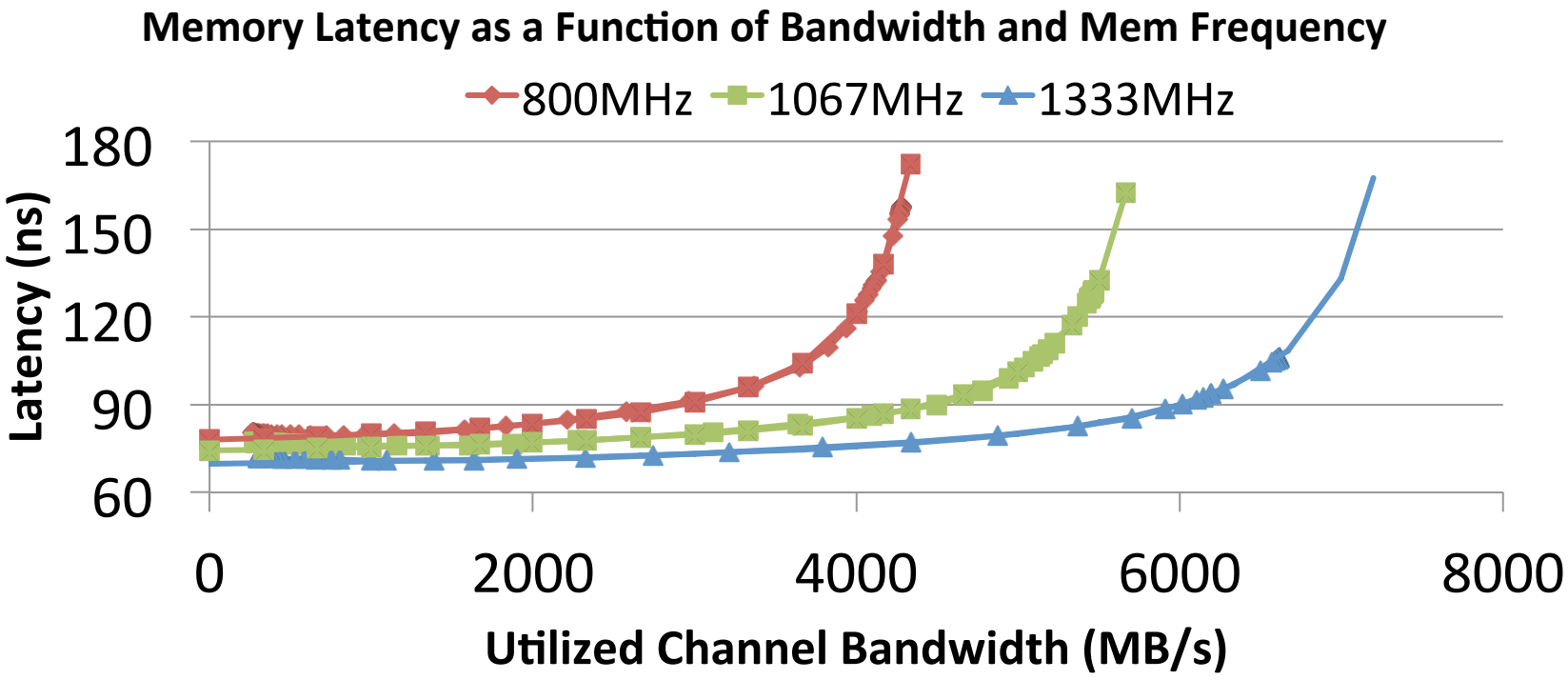


Outline

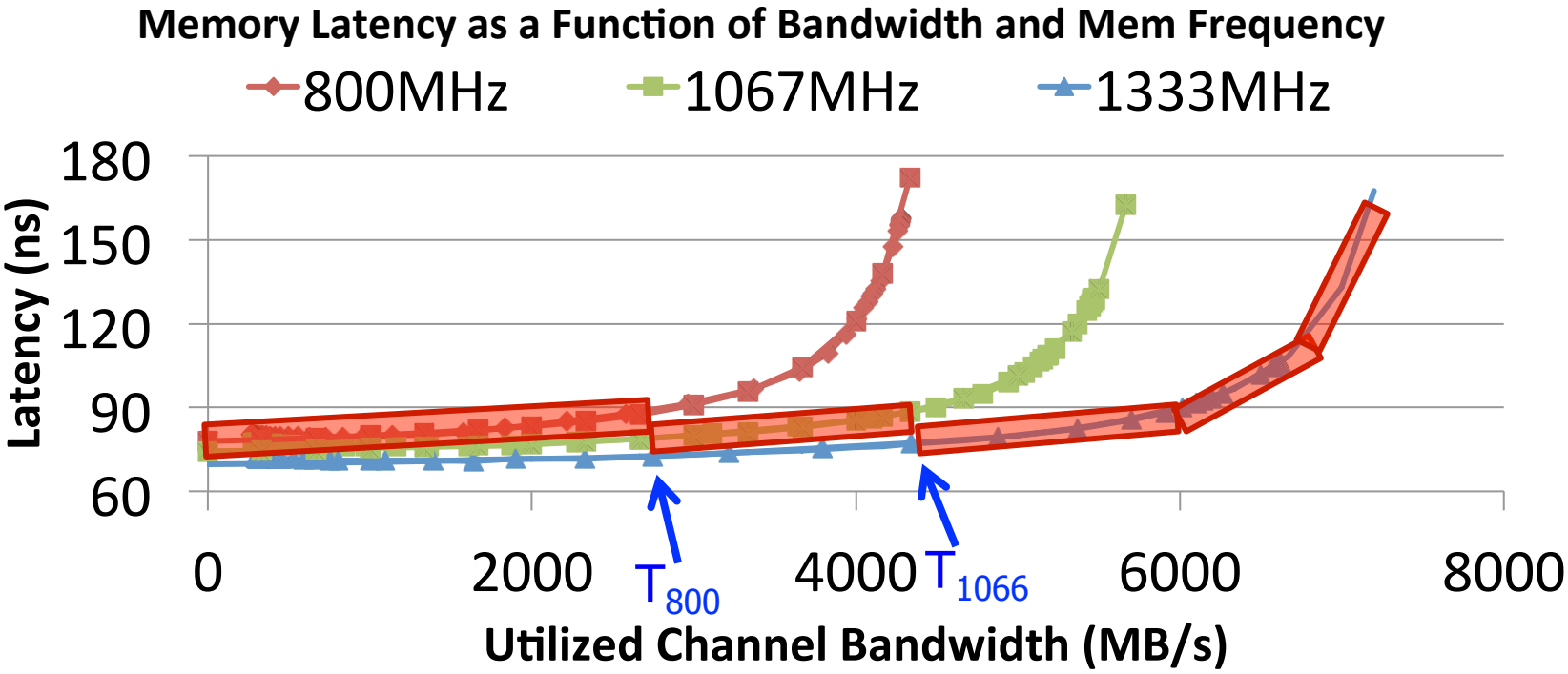
- Motivation
- Background and Characterization
  - DRAM Operation
  - DRAM Power
  - Frequency and Voltage Scaling
- Performance Effects of Frequency Scaling
- Frequency Control Algorithm
- Evaluation and Conclusions

Memory Latency Under Load

- At low load, most time is in array access and bus transfer
  → small constant offset between bus-frequency latency curves
- As load increases, queueing delay begins to dominate
  → bus frequency significantly affects latency



Control Algorithm: Demand-Based Switching



After each epoch of length T_{epoch}:

Measure per-channel bandwidth BW

if BW < T_{800} : switch to 800MHz

else if BW < T_{1066} : switch to 1066MHz

else : switch to 1333MHz

Implementing V/F Switching

- **Halt Memory Operations**
  - Pause requests
  - Put DRAM in Self-Refresh
  - Stop the DIMM clock
- **Transition Voltage/Frequency**
  - Begin voltage ramp
  - Relock memory controller PLL at new frequency
  - Restart DIMM clock
  - Wait for DIMM PLLs to relock
- **Begin Memory Operations**
  - Take DRAM out of Self-Refresh
  - Resume requests

Implementing V/F Switching

- **Halt Memory Operations**
  - Pause requests
  - Put DRAM in Self-Refresh
  - Stop the DIMM clock
- **Transition Voltage/Frequency**
  - Begin voltage ramp


Memory frequency already adjustable statically


Voltage regulators for CPU DVFS can work for memory DVFS


Full transition takes ~20µs

Outline

- Motivation
- Background and Characterization
  - DRAM Operation
  - DRAM Power
  - Frequency and Voltage Scaling
- Performance Effects of Frequency Scaling
- Frequency Control Algorithm
- Evaluation and Conclusions

Evaluation Methodology

- **Real-system evaluation**
  - Dual 4-core Intel Xeon®, 3 memory channels/socket
  - 48 GB of DDR3 (12 DIMMs, 4GB dual-rank, 1333MHz)

- **Emulating memory frequency for performance**
  - Altered memory controller timing registers (tRC, tB2BCAS)
  - Gives performance equivalent to slower memory frequencies

- **Modeling power reduction**
  - Measure baseline system (AC power meter, 1s samples)
  - Compute reductions with an analytical model (see paper)

Evaluation Methodology

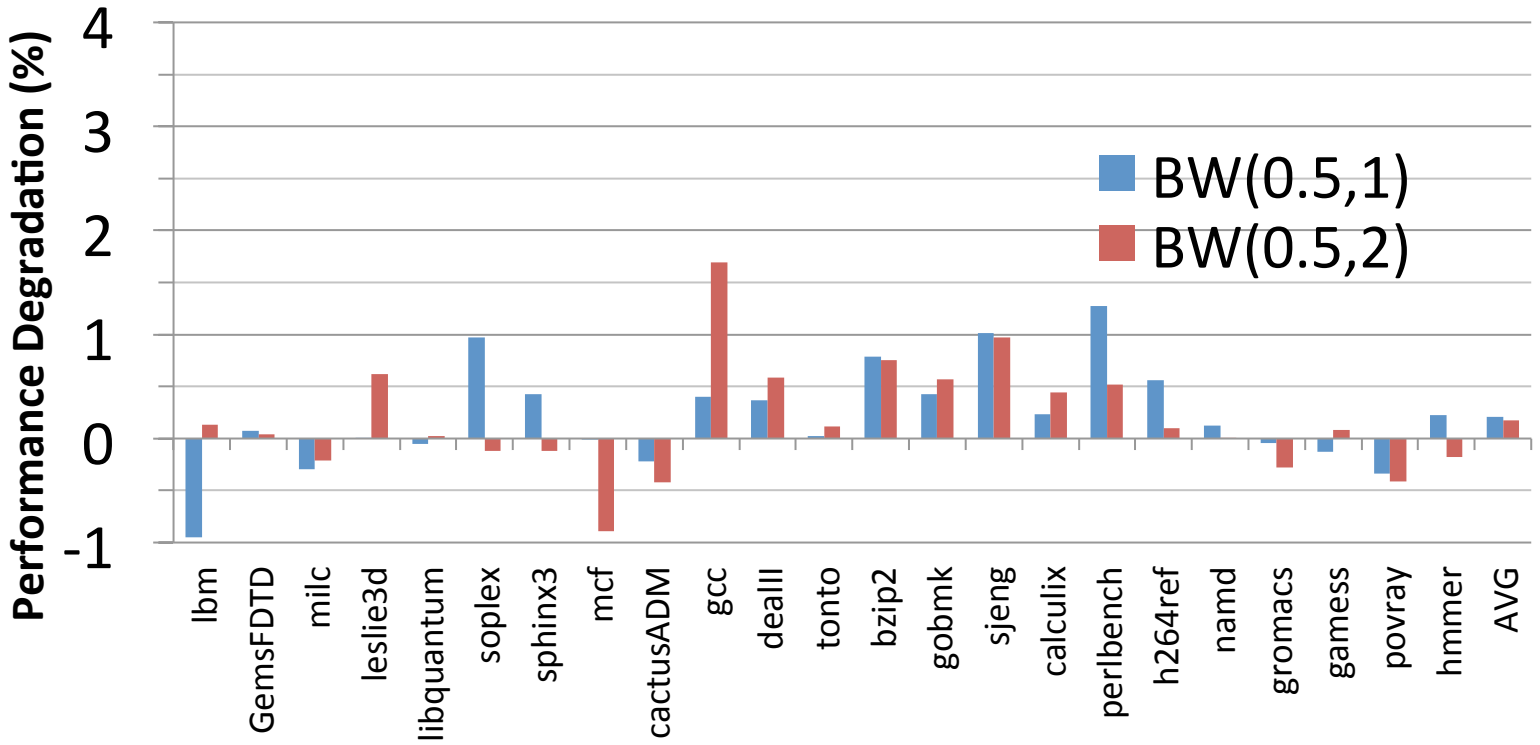
- **Workloads**
  - SPEC CPU2006: CPU-intensive workloads
  - All cores run a copy of the benchmark

- **Parameters**
  - T_{epoch} = 10ms
  - Two variants of algorithm with different switching thresholds:
  - BW(0.5, 1): T_{800} = 0.5GB/s, T_{1066} = 1GB/s
  - BW(0.5, 2): T_{800} = 0.5GB/s, T_{1066} = 2GB/s

    → More aggressive frequency/voltage scaling

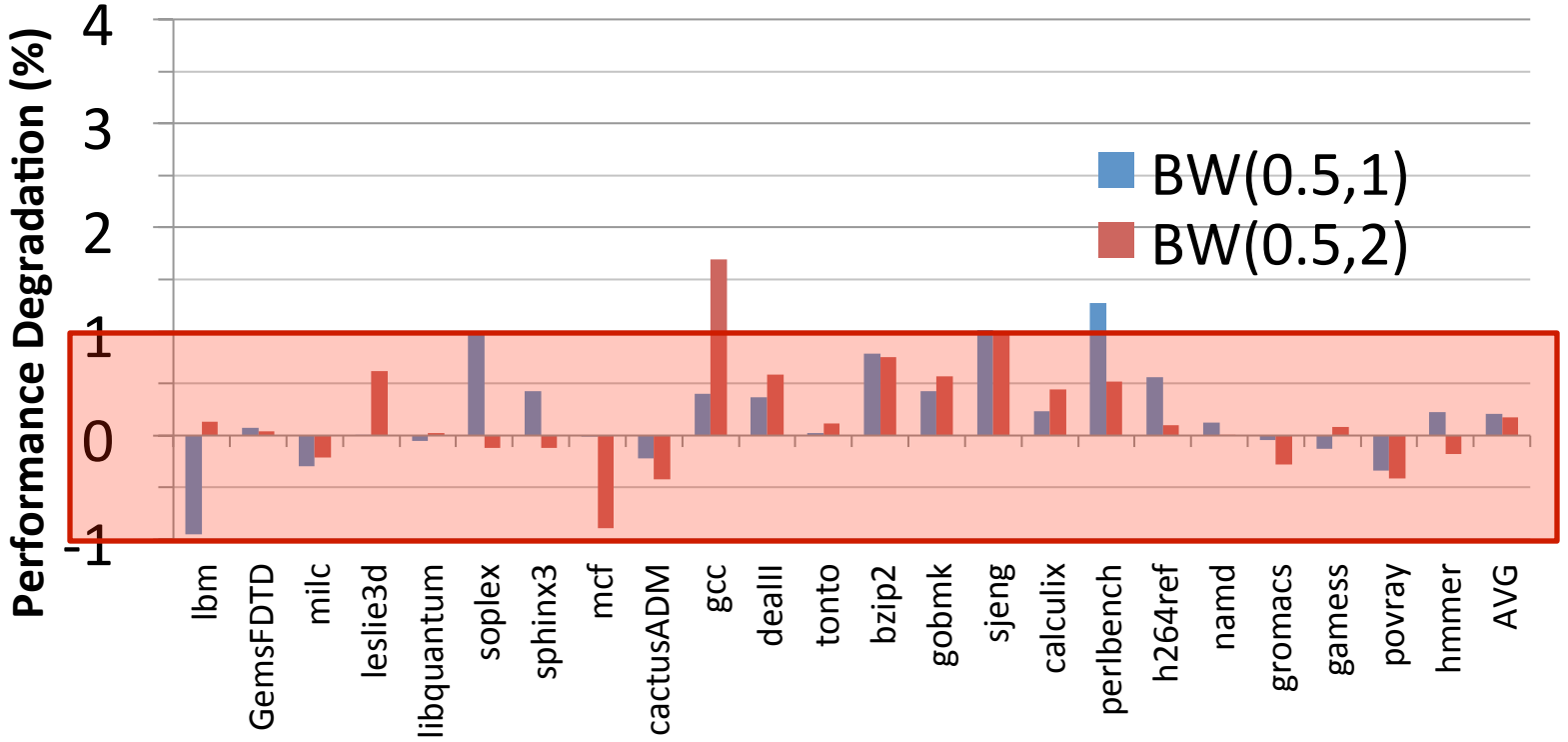
Performance Impact of Memory DVFS

- Minimal performance degradation: 0.2% (avg), 1.7% (max)



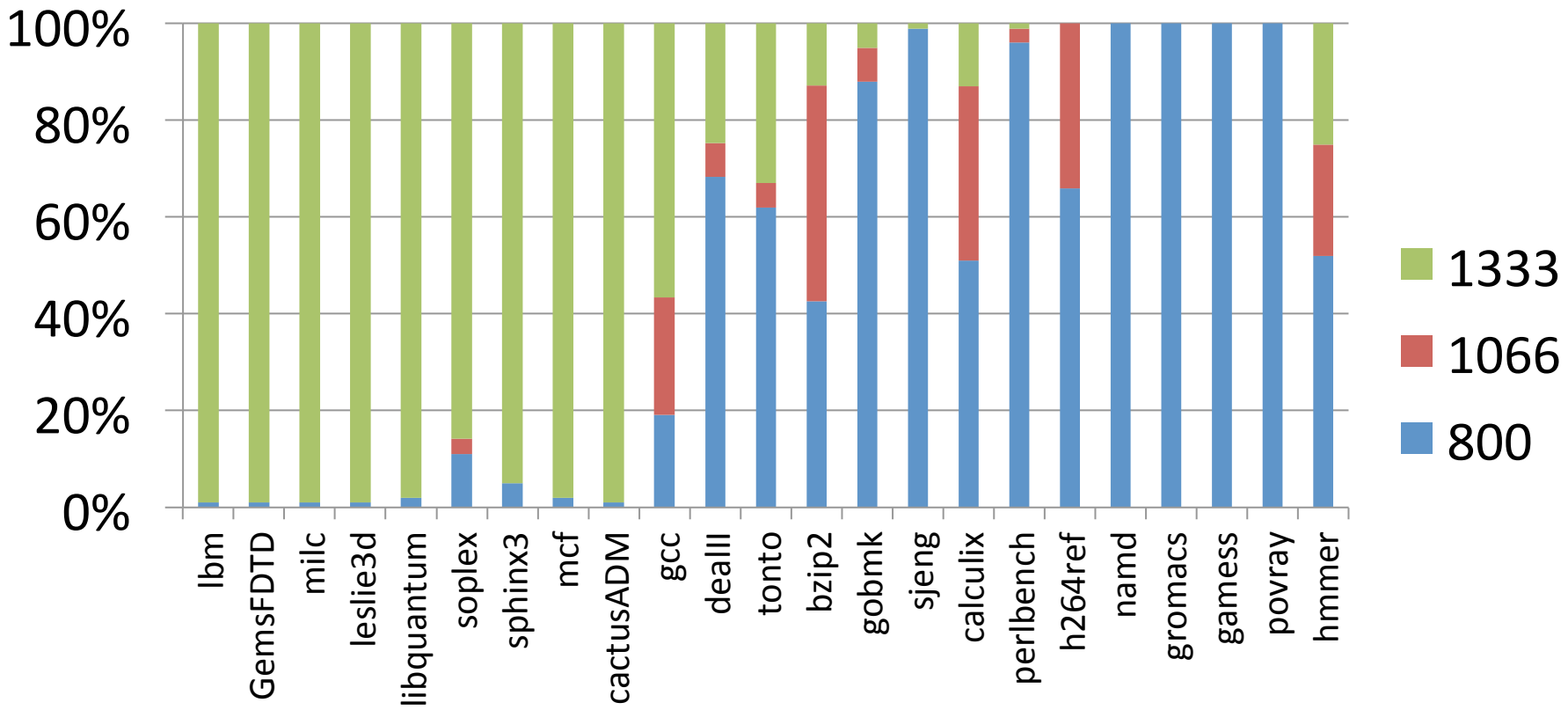
Performance Impact of Memory DVFS

- Minimal performance degradation: 0.2% (avg), 1.7% (max)
- Experimental error ~1%



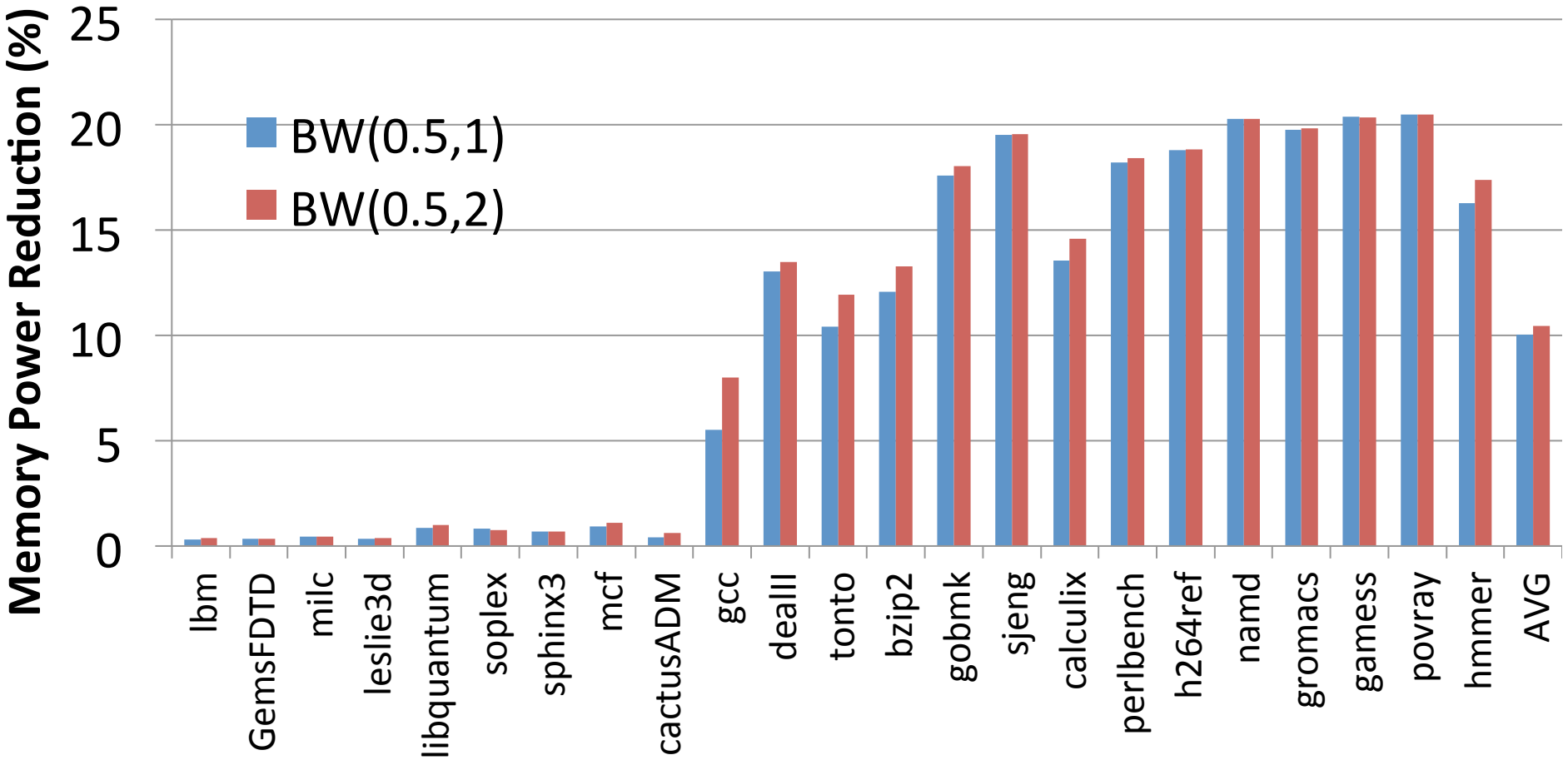
Memory Frequency Distribution

- Frequency distribution shifts toward higher memory frequencies with more memory-intensive benchmarks



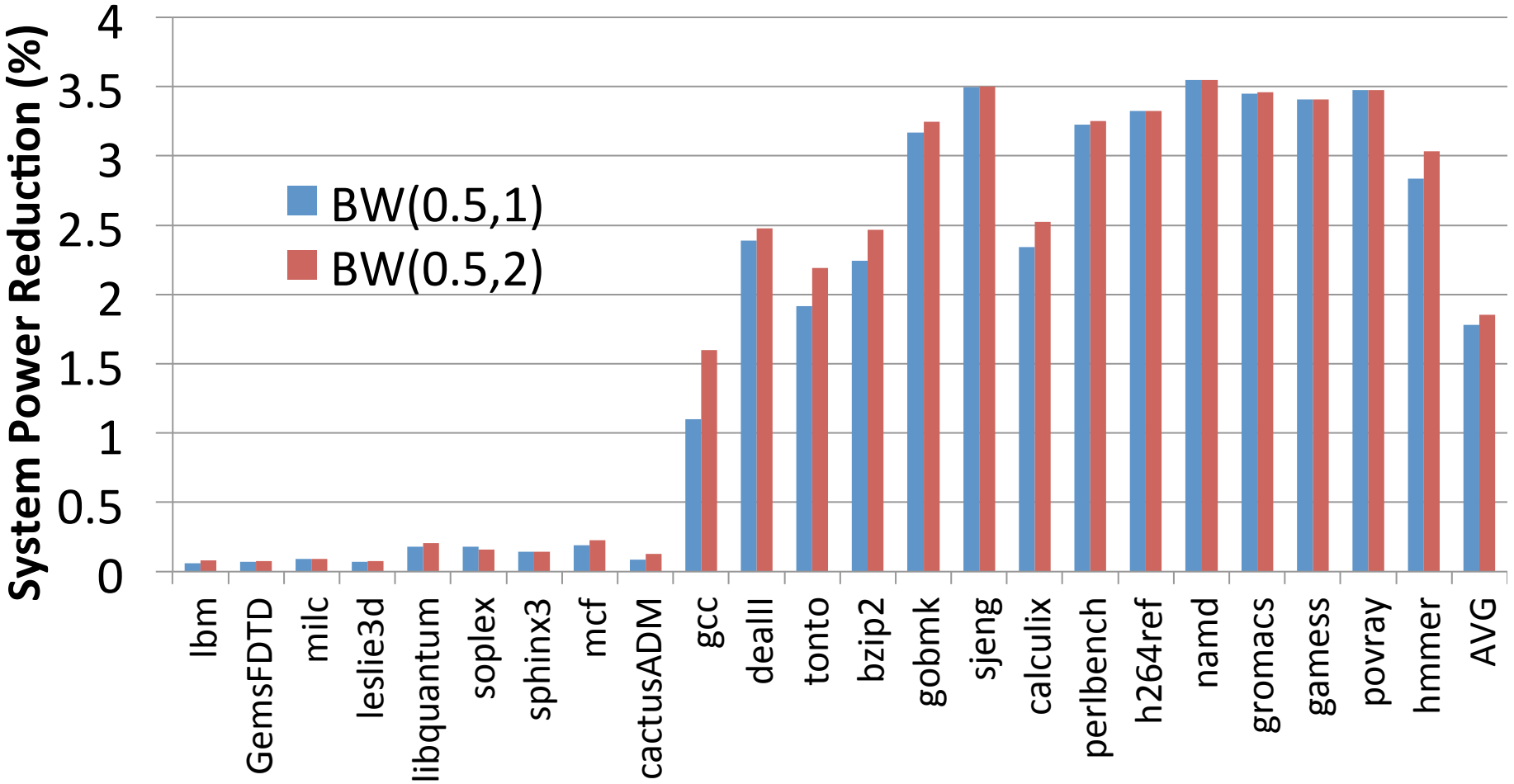
Memory Power Reduction

- Memory power reduces by 10.4% (avg), 20.5% (max)



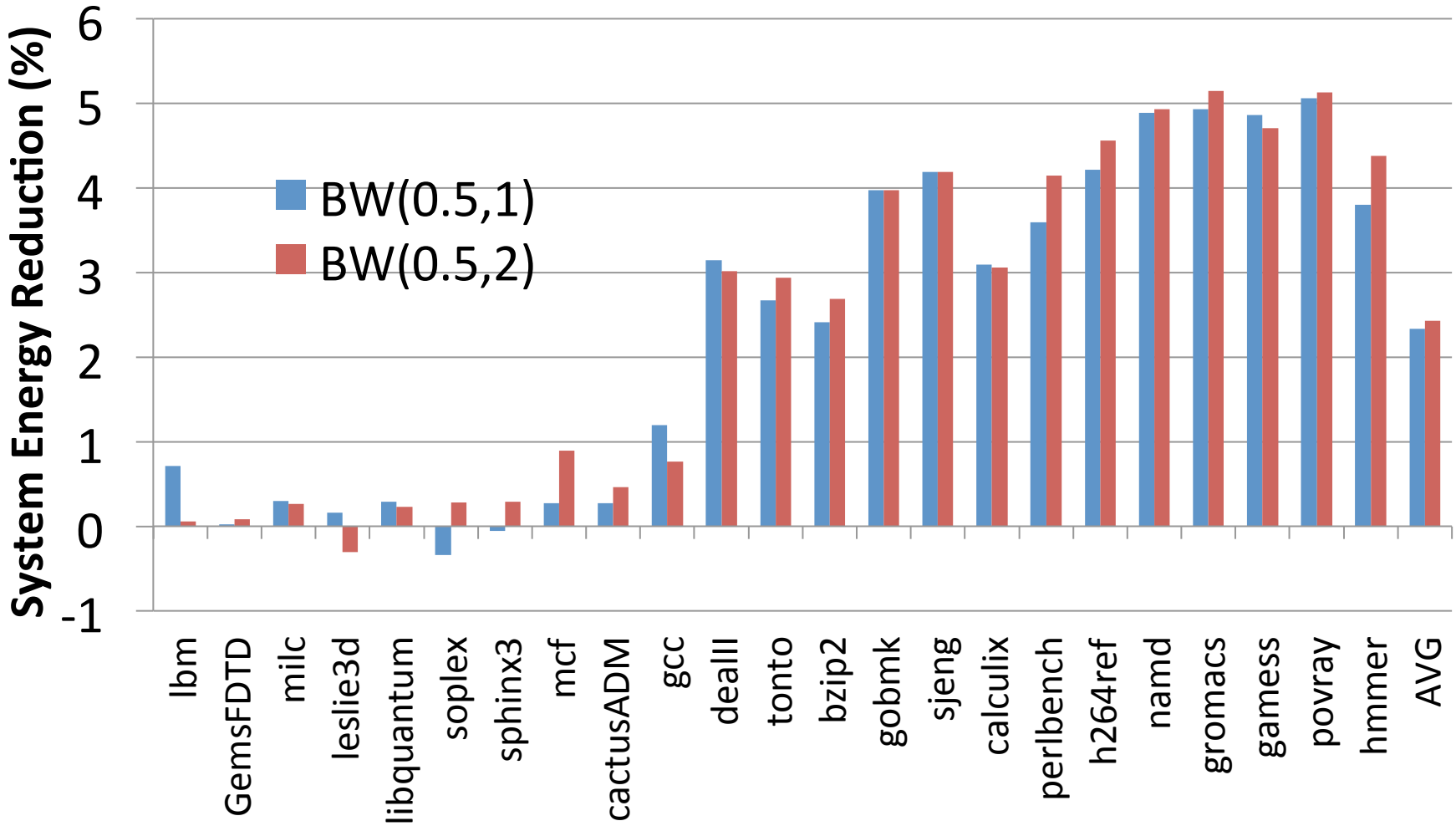
System Power Reduction

- As a result, system power reduces by 1.9% (avg), 3.5% (max)



System Energy Reduction

- System energy reduces by 2.4% (avg), 5.1% (max)



Related Work

- **MemScale** [Deng11], concurrent work (ASPLOS 2011)
  - Also proposes Memory DVFS
  - Application performance impact model to decide voltage and frequency: requires specific modeling for a given system; our bandwidth-based approach avoids this complexity
  - Simulation-based evaluation; our work is a real-system proof of concept

- **Memory Sleep States** (Creating opportunity with data placement [Lebeck00,Pandey06], OS scheduling [Delaluz02], VM subsystem [Huang05]; Making better decisions with better models [Hur08,Fan01])
- **Power Limiting/Shifting** (RAPL [David10] uses memory throttling for thermal limits; CPU throttling for memory traffic [Lin07,08]; Power shifting across system [Felter05])

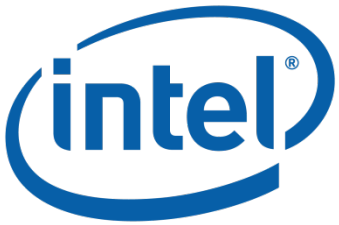
Conclusions

- Memory power is a significant component of system power
  - 19% average in our evaluation system, 40% in other work
- Workloads often keep memory active but underutilized
  - Channel bandwidth demands are highly variable
  - Use of memory sleep states is often limited
- Scaling memory frequency/voltage can reduce memory power with minimal system performance impact
  - 10.4% average memory power reduction
  - Yields 2.4% average system energy reduction
- Greater reductions are possible with wider frequency/ voltage range and better control algorithms

Memory Power Management via Dynamic Voltage/Frequency Scaling

Howard David (Intel)
Eugene Gorbatov (Intel)
Ulf R. Hanebutte (Intel)

Chris Fallin (CMU)
Onur Mutlu (CMU)



SAFARI
Carnegie Mellon

Why Real-System Evaluation?

- Advantages:
  - Capture all effects of altered memory performance
    - System/kernel code, interactions with IO and peripherals, etc
  - Able to run full-length benchmarks (SPEC CPU2006) rather than short instruction traces
  - No concerns about architectural simulation fidelity
- Disadvantages:
  - More limited room for novel algorithms and detailed measurements
  - Inherent experimental error due to background-task noise, real power measurements, nondeterministic timing effects

- For a proof-of-concept, we chose to run on a real system in order to have results that capture all potential side-effects of altering memory frequency

CPU-Bound Applications in a DRAM-rich system

- We evaluate CPU-bound workloads with 12 DIMMs: what about smaller memory, or IO-bound workloads?

- 12 DIMMs (48GB): are we magnifying the problem?
  - Large servers can have this much memory, especially for database or enterprise applications
  - Memory can be up to 40% of system power [1,2], and reducing its power in general is an academically interesting problem

- CPU-bound workloads: will it matter in real life?
  - Many workloads have CPU-bound phases (e.g., database scan or business logic in server workloads)
  - Focusing on CPU-bound workloads isolates the problem of varying memory bandwidth demand while memory cannot enter sleep states, and our solution applies for any compute phase of a workload

[1] L. A. Barroso and U. Holzle. "The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines." Synthesis Lectures on Computer Architecture. Morgan & Claypool, 2009.
[2] C. Lefurgy et al. "Energy Management for Commercial Servers." IEEE Computer, pp. 39—48, December 2003.

Combining Memory & CPU DVFS?

- Our evaluation did not incorporate CPU DVFS:
  - Need to understand effect of single knob (memory DVFS) first
  - Combining with CPU DVFS might produce second-order effects that would need to be accounted for

- Nevertheless, memory DVFS is effective by itself, and mostly orthogonal to CPU DVFS:
  - Each knob reduces power in a different component
  - Our memory DVFS algorithm has neligible performance impact → negligible impact on CPU DVFS
  - CPU DVFS will only further reduce bandwidth demands relative to our evaluations → no negative impact on memory DVFS

Why is this Autonomic Computing?

- Power management in general is autonomic: a system observes its own needs and adjusts its behavior accordingly

  → Lots of previous work comes from architecture community, but crossover in ideas and approaches could be beneficial

- This work exposes a new knob for control algorithms to turn, has a simple model for the power/energy effects of that knob, and observes opportunity to apply it in a simple way

- Exposes future work for:
  - More advanced control algorithms
  - Coordinated energy efficiency across rest of system
  - Coordinated energy efficiency across a cluster/datacenter, integrated with memory DVFS, CPU DVFS, etc.