In-network Monitoring and Control Policy for DVFS of CMP Networks-on-Chip and Last Level Caches

Xi Chen[1], Zheng Xu[1], Hyungjun Kim[1], Paul V. Gratz[1], Jiang Hu[1], Michael Kishinevsky[2] and Umit Ogras[2]

[1]*Computer Engineering and Systems Group, Department of ECE, Texas A&M University*

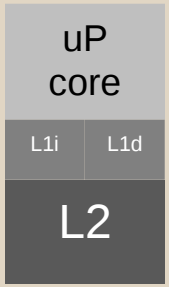
[2]*Strategic CAD Labs, Intel Corp.*

Introduction – The Power/Performance Challenge

TEXAS A&M UNIVERSITY

- VLSI Technology Trends
  - Continued transistor scaling
    - More transistors
  - Traditional VLSI gains stop
    - Power increasing and transistor performance stagnant
- Achieving performance in modern VLSI
  - Multi-core/CMP for performance
    - NoCs for communication
  - CMP power management to permit further performance gains and new challenges

Core Power Management

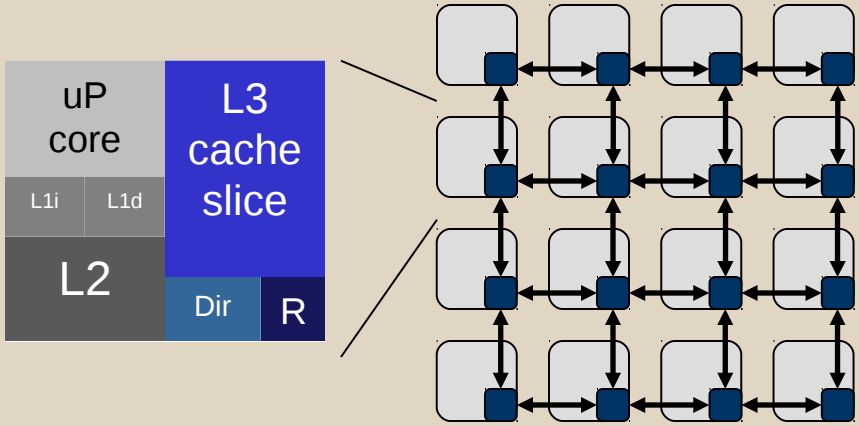


Typically power management covers only the core and lower-level caches

- Simpler problem (relatively speaking)
  - All performance information locally available
    - Instructions per cycle
    - Lower-level cache miss rates
    - Idle time
  - Each core can act independently
  - Performance scales approximately linearly with frequency
- Cores are only part of the problem
  - Power management in the uncore is a different domain…

Typical Chip-Multiprocessors

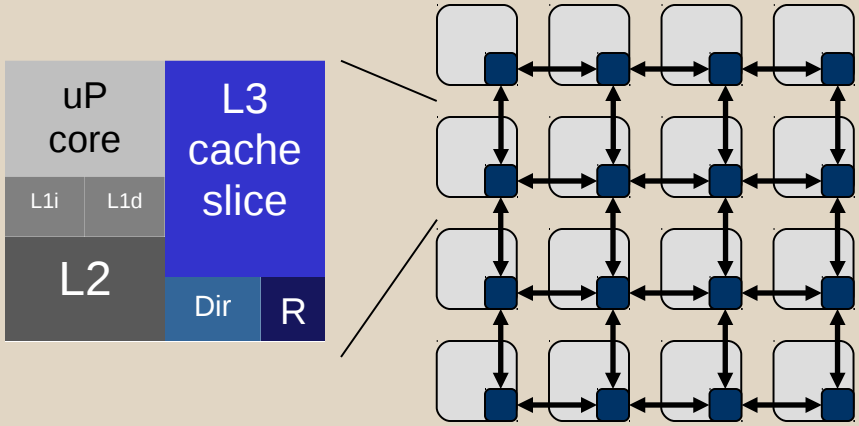
- Chip-multiprocessors (CMPs): Complexity moves from the cores up the memory system hierarchy.



- Multi-level hierarchies
  - Private lower levels
  - Shared last-level
- Networks-on-chip for:
  - Cache block transfers
  - Cache coherence

CMP Power Management Challenge

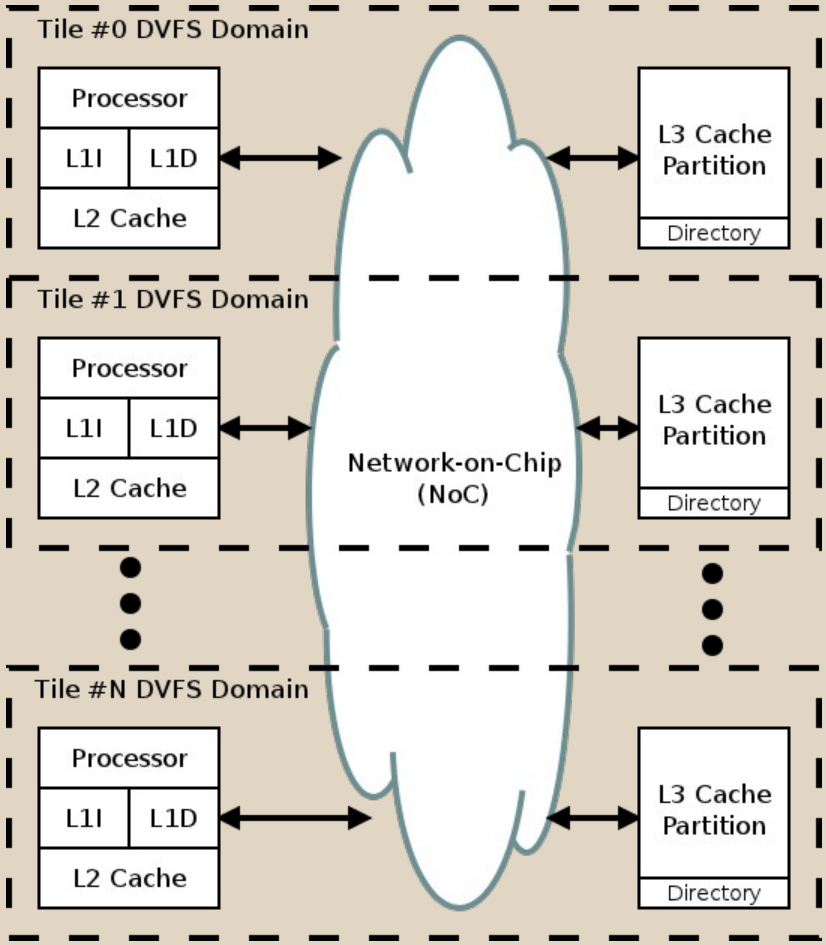
- Chip-multiprocessors (CMPs): Complexity moves from the cores up the memory system hierarchy.



- Multi-level hierarchies
  - Private lower levels
  - Shared last-level
- Networks-on-chip for:
  - Cache block transfers
  - Cache coherence

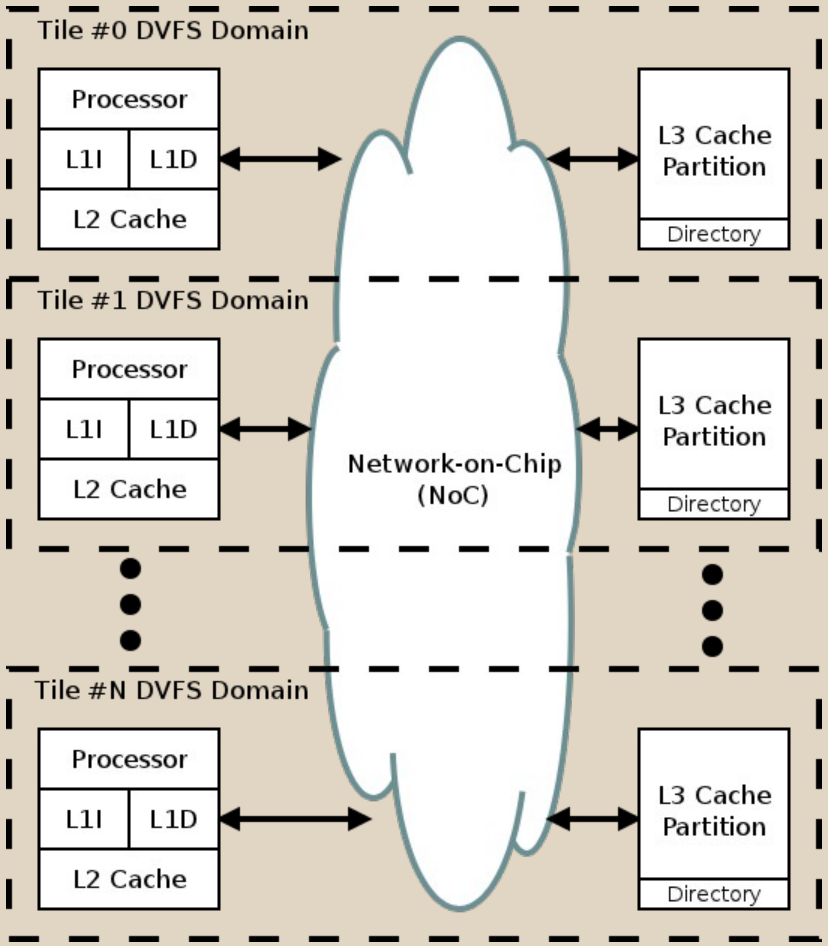
- Large fraction of the power outside of cores
  - LLC shared among many cores (distributed!)
  - Network-on-chip interconnects cores
    - 12 W on the Single Chip Cloud Computer!
- Indirect impact on system performance
  - Depends upon lower-level cache miss-rates

CMP DVFS Partitioning

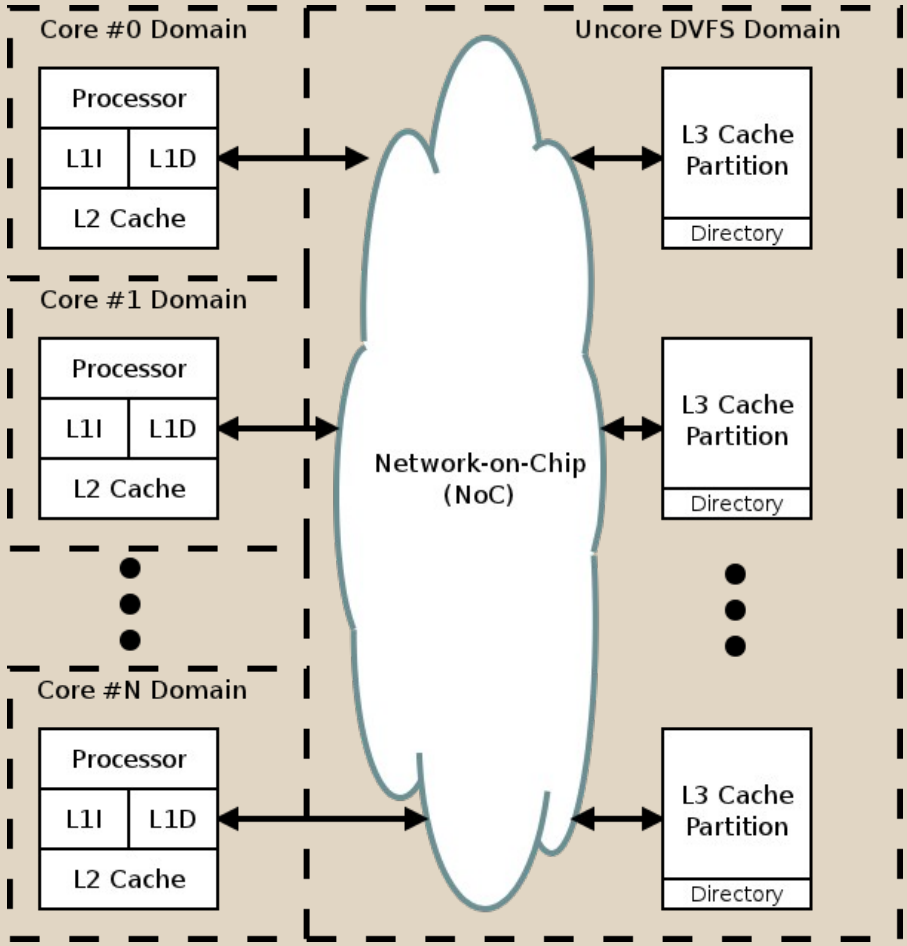


Domains per tile

CMP DVFS Partitioning



Domains per tile



Domains per core
Separate domain for uncore

Project Goals

Develop a power management policy for a CMP uncore.

- Maximum savings with minimal impact on performance (< 5% IPC loss).
  - What to monitor?
  - How to propagate information to the central controller?
  - What policy to implement?

Outline

- Introduction
- Design Description
  - Uncore Power Management
  - Metrics
  - Information Propagation
  - PID Control
- Evaluation
- Conclusions and Future Work

Uncore Power Management

- Effective uncore power management
  - Inputs:
    - Current performance demand
    - Current power state (DVFS level)
  - Outputs:
    - Next power state
- Classic control problem
  - Constraints
    - High speed decisions
    - Low hardware overhead
    - Low impact on system from management overheads

Design Outline

Three major components to uncore power management:

- Uncore performance metric
  - Average memory access time (AMAT)
- Status propagation
  - In-network, unused header portion
- Control policy
  - PID Control over a fixed time window

Performance Metrics

Uncore: LLC + NoC

- Which performance metric?
  - NoC Centric?
    - Credits
    - Free VCs
    - Per-hop latency
  - LLC Centric?
    - LLC Access rate
    - LLC Miss rate

Performance Metrics

Uncore: LLC + NoC

- Which performance metric?
  - NoC Centric?
    - Credits
    - Free VCs
    - Per-hop latency
  - LLC Centric?
    - LLC Access rate
    - LLC Miss rate

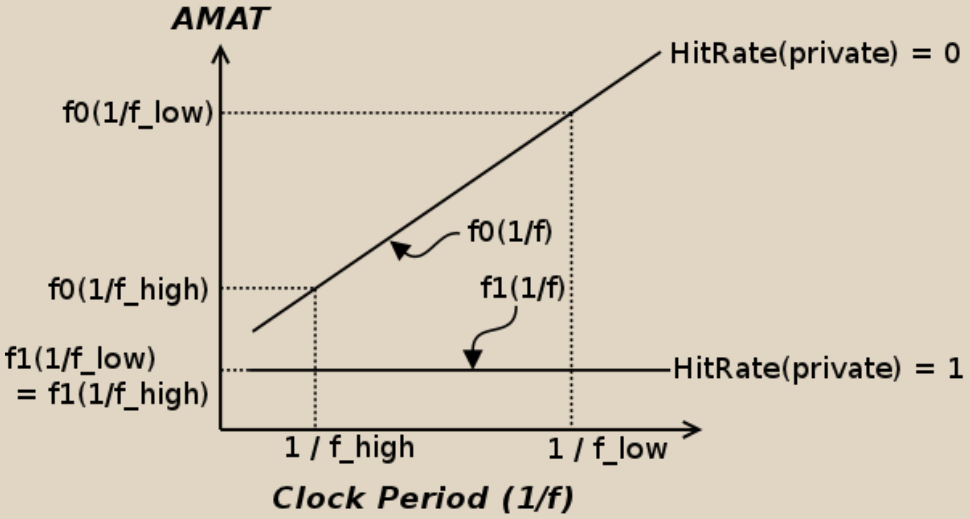
Ultimately who cares about uncore performance?

- Need a metric that quantifies the memory system's effect on system performance!
- Average memory access time (AMAT)

Average Memory Access Time

*AMAT = HitRateL1*AccTimeL1+(1-HitRateL1)* (HitRateL2*AccTimeL2+ ((1-HitRateL2) * LatencyUncore))*

- Direct measurement memory system performance
- AMAT increase X yields IPC loss of ~1/2X for small X
  - Experimentally determined

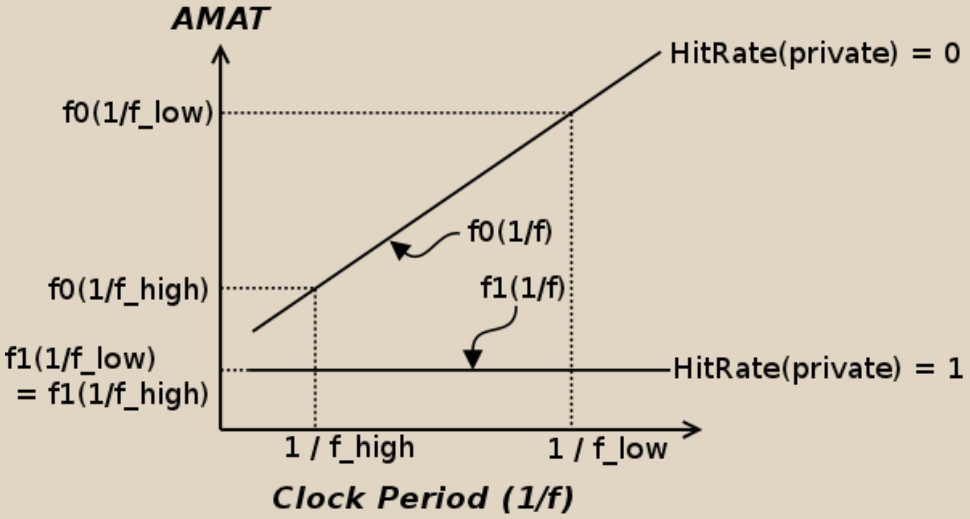


AMAT vs Uncore clock rate for two cases: f0 – no private hits; f1 – all private hits.

Average Memory Access Time

*AMAT = HitRateL1*AccTimeL1+(1-HitRateL1)* (HitRateL2*AccTimeL2+ ((1-HitRateL2) * LatencyUncore))*

- Direct measurement memory system performance
- AMAT increase X yields IPC loss of ~1/2X for small X
  - Experimentally determined



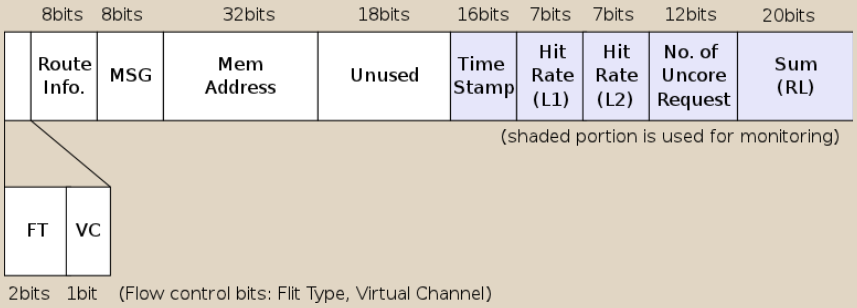
AMAT vs Uncore clock rate for two cases: f0 – no private hits; f1 – all private hits.

Note: HitRateL1, HitRateL2, and LatencyUncore require information from each core to calculate weighted averages!

Information Propagation

- In-network status packets too costly
  - Bursts of status would impact performance
  - Increased dynamic energy
- Dedicated status network would be overkill
  - Somewhat low data rate: ~8 bytes per core per 50000-cycle time window
  - Constant power drain

Information Propagation



- In-network status packets too costly
  - Bursts of status would impact performance
  - Increased dynamic energy
- Dedicated status network would be overkill
  - Somewhat low data rate: ~8 bytes per core per 50000-cycle time window
  - Constant power drain

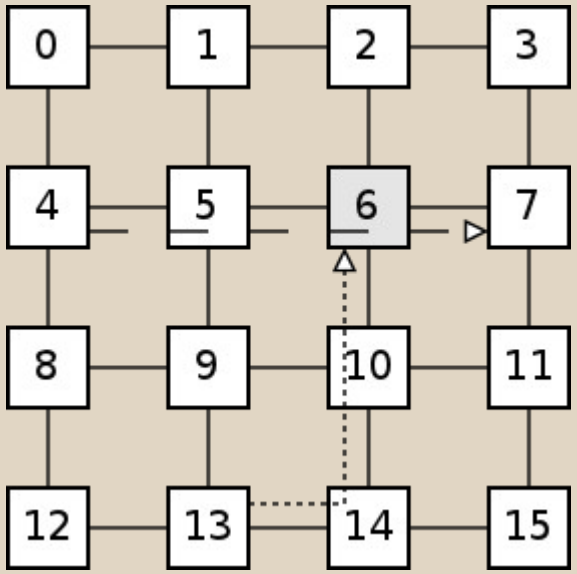
"Piggieback" info in packet headers

- Link width often an even divisor of cache line size – unused space in header
- No congestion or power impact

- Status info timeliness?

Information Propagation

- One power controller node
  - Node 6 in figure
- Status opportunistically sent
- Info harvested as packet pass through controller node
- However, per-core info not received at the end of every window…

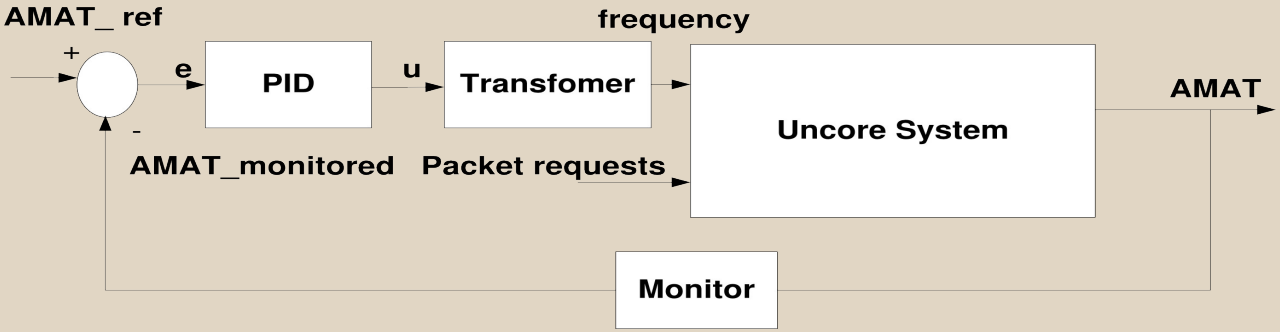


Uncore NoC, grey tile contains perf. monitor. Dashed arrows represent packet paths.

Extrapolation

- AMAT calculation requires information from all nodes at the end of each time window
- Opportunistic piggy-backing provides no guarantees on information timeliness
  - Naïvely using last-packet received leads to bias in weighted average of AMAT
- Extrapolate packet counts to the end of the time window
  - More accurate weights for AMAT calculation
  - Nodes for which no data is received are excluded from AMAT

Power Management Controller



- PID (Proportional-Integral-Derivative) Control
  - Computationally simpler than computer learning techniques
  - More readily and quickly adapts to many different workloads than rule based approaches
  - Theoretical grounds for stability
    - (proof in paper)

Outline

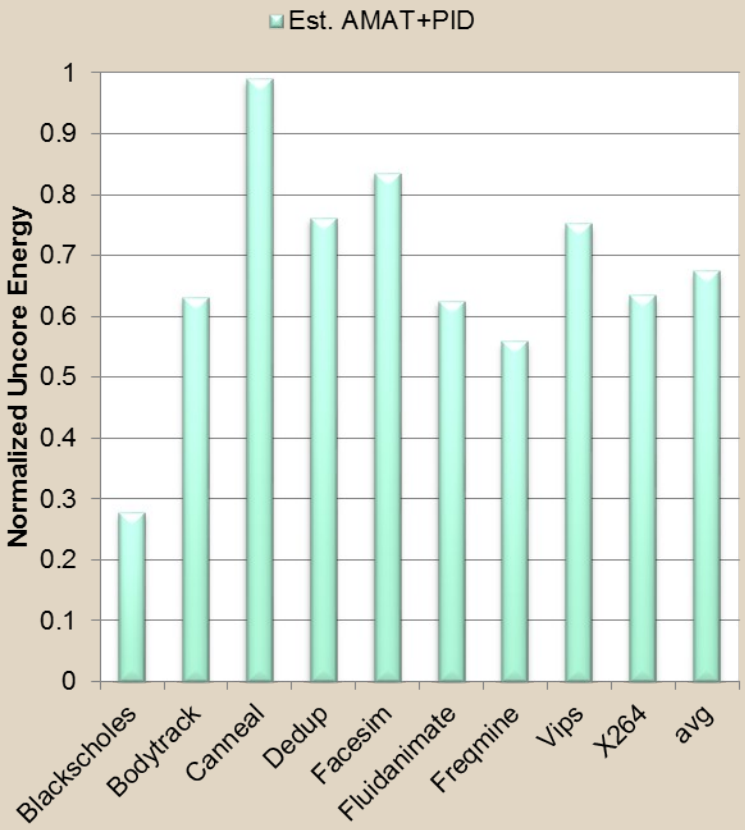
- Introduction
- Design Description
- Evaluation
  - Methodology
  - Power and Performance
    - Estimated AMAT + PID
    - Vs. Perfect AMAT + PID
    - Vs. Rule-based
  - Analysis
    - Tracking ideal DVFS ratio selection
- Conclusions and Future Work

Methodology

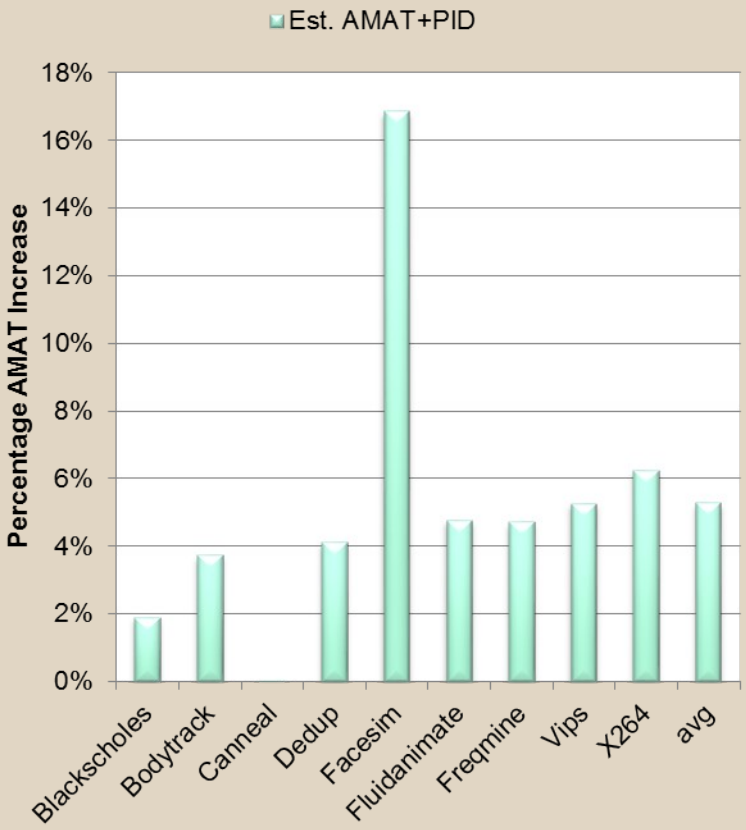
- Memory system traces
  - PARSEC applications
  - M5 trace generation
  - First 250M memory operations
- Custom Simulator:
  - L1 + L2 + NoC + LLC+ Directory
- Energy savings calculated based on dynamic power
  - Some benefit to static power as well, future work

Parameter	Values
#processing cores	16
L1 data cache	2-way 32Kb, 1 core cycle latency
L2 cache	8-way 256Kb, 13 core cycle latency
L3 cache (LLC)	16-way, 2MB/bank, 32MB/total, 15 uncore cycle latency
Directory cache	MESI, 4 uncore cycle latency
Memory access latency	100 core cycles
NoC	4×4 2D mesh, X-Y DOR, 2VCs/port 4flits deep
Voltage/Frequency	10 levels, voltage: 0.5V–1V, frequency: 250MHz–1GHz

Power and Performance



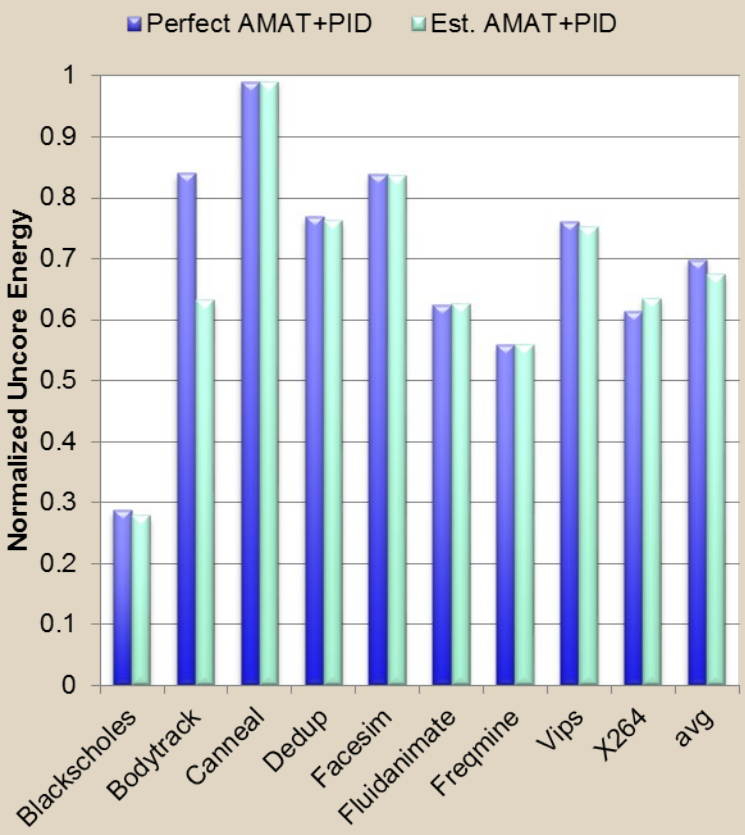
Normalized dynamic energy consumption



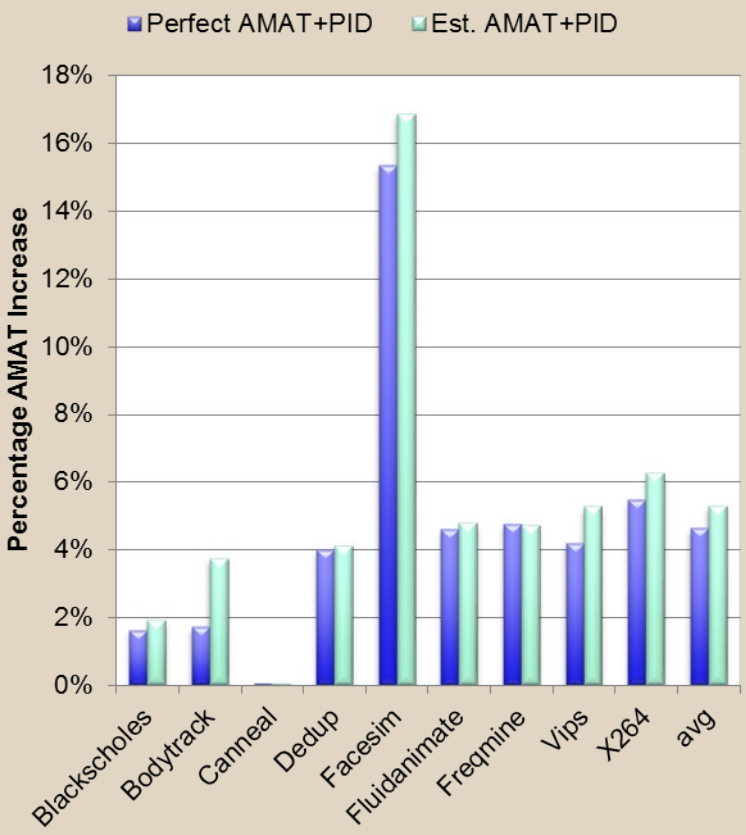
Normalized performance loss

- Average of 33% energy savings versus baseline
- Average of ~5% AMAT loss (<2.5% IPC)

Comparison vs. Perfect AMAT



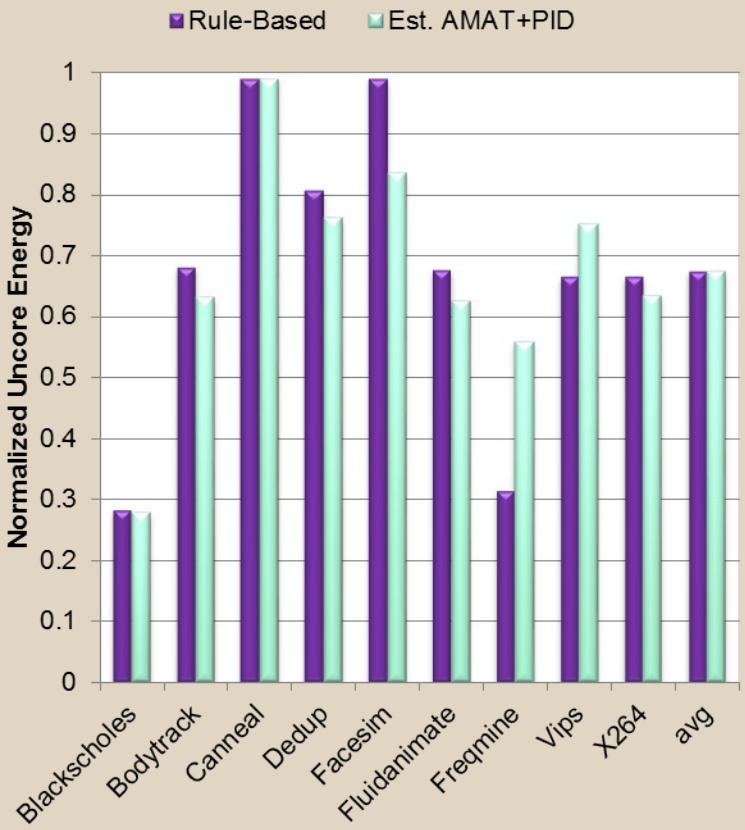
Normalized dynamic energy consumption



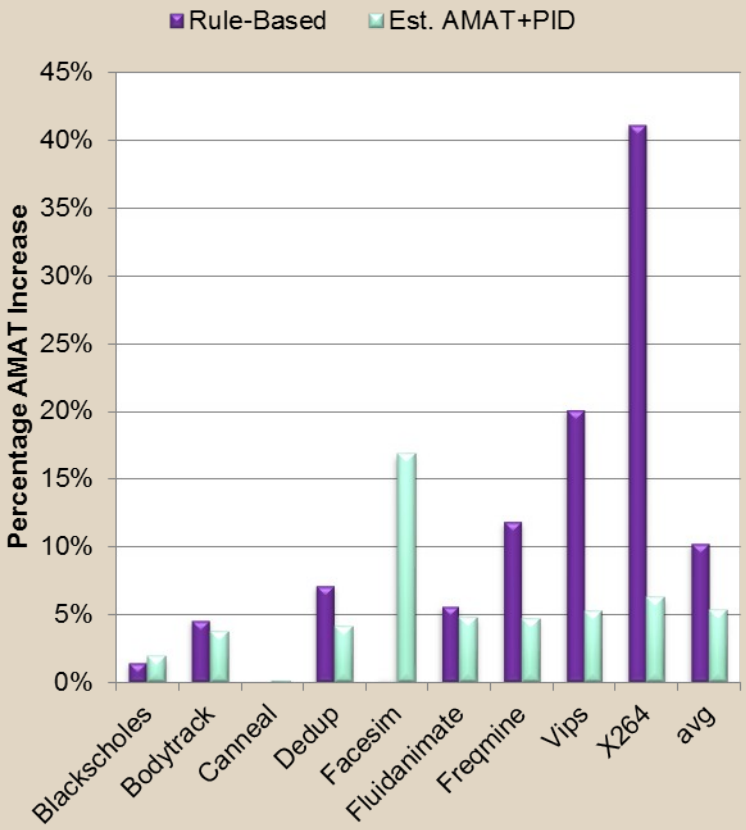
Normalized performance loss

- Virtually identical power savings vs. perfect AMAT
- Slight loss in performance vs. perfect AMAT

Comparison vs. Rule-Based



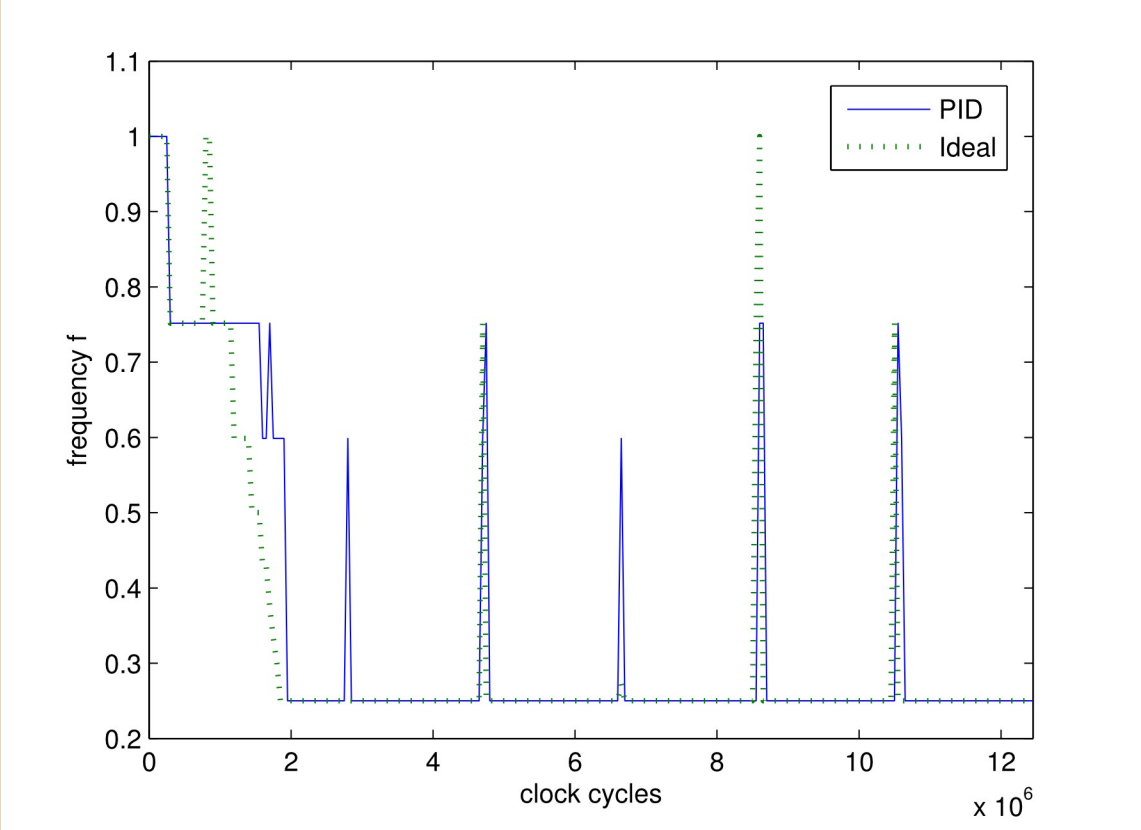
Normalized dynamic energy consumption



Normalized performance loss

- Virtually identical power savings vs. Rule-Based
- 50% less performance loss

Analysis: PID tracking vs. ideal



- Generally PID is slightly conservative
- Reacts quickly and accurately to spikes in need

Conclusions and Future Work

- We introduce a power management system for the CMP Uncore
  - Performance metric: estimated AMAT
  - Information propagation: In-network, piggy-backed
  - Control Algorithm: PID
- 33% energy savings with insignificant performance loss
  - Near ideal AMAT estimation
  - Outperforms rule-based techniques

Conclusions and Future Work

- Just scratched the surface here
  - Dynamic cache footprint analysis for LLC power gating
  - Are cycles of uncore utilization predictable?
    - Neural net approaches to control
    - Other predictive techniques
  - Not all misses are equally important
    - Load criticality analysis to improve control

Backup

DVFS Background

$$P = \alpha \cdot C \cdot V^2 \cdot f$$

- Reduce frequency to allow voltage reduction
  - Dynamic power reduces exponentially
  - Static power reduction as well
  - Obvious performance impacts
- Power management algorithm:
  - Choose best power-performance tradeoff