A Comprehensive Analytical Performance Model of DRAM Caches

Authors:
Nagendra Gulur*, Mahesh Mehendale*, and R Govindarajan+
Presented by: Sreepathi Pai§
***Texas Instruments,**
+Indian Institute of Science
§University of Texas, Austin

6th ACM/SPEC International Conference on Performance Engineering, 2015

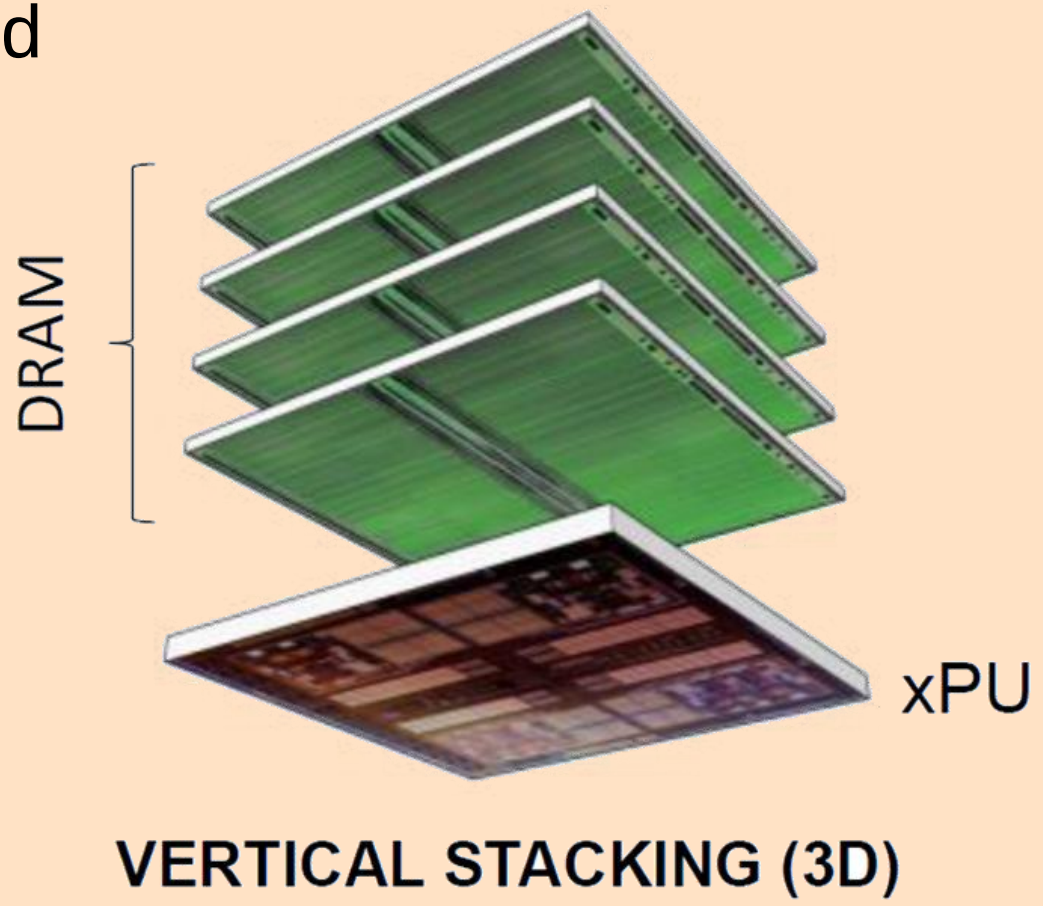
Talk Outline

- Introduction to stacked DRAM Caches
- Background (An overview of *ANATOMY*[§])
- *ANATOMY-Cache*: Modeling Stacked DRAM Cache Organizations
- Evaluation
- Insights
- Conclusions

[§]*ANATOMY: An Analytical Model of Memory System Performance (Published in the 2014 ACM international conference on Measurement and modeling of computer systems)*

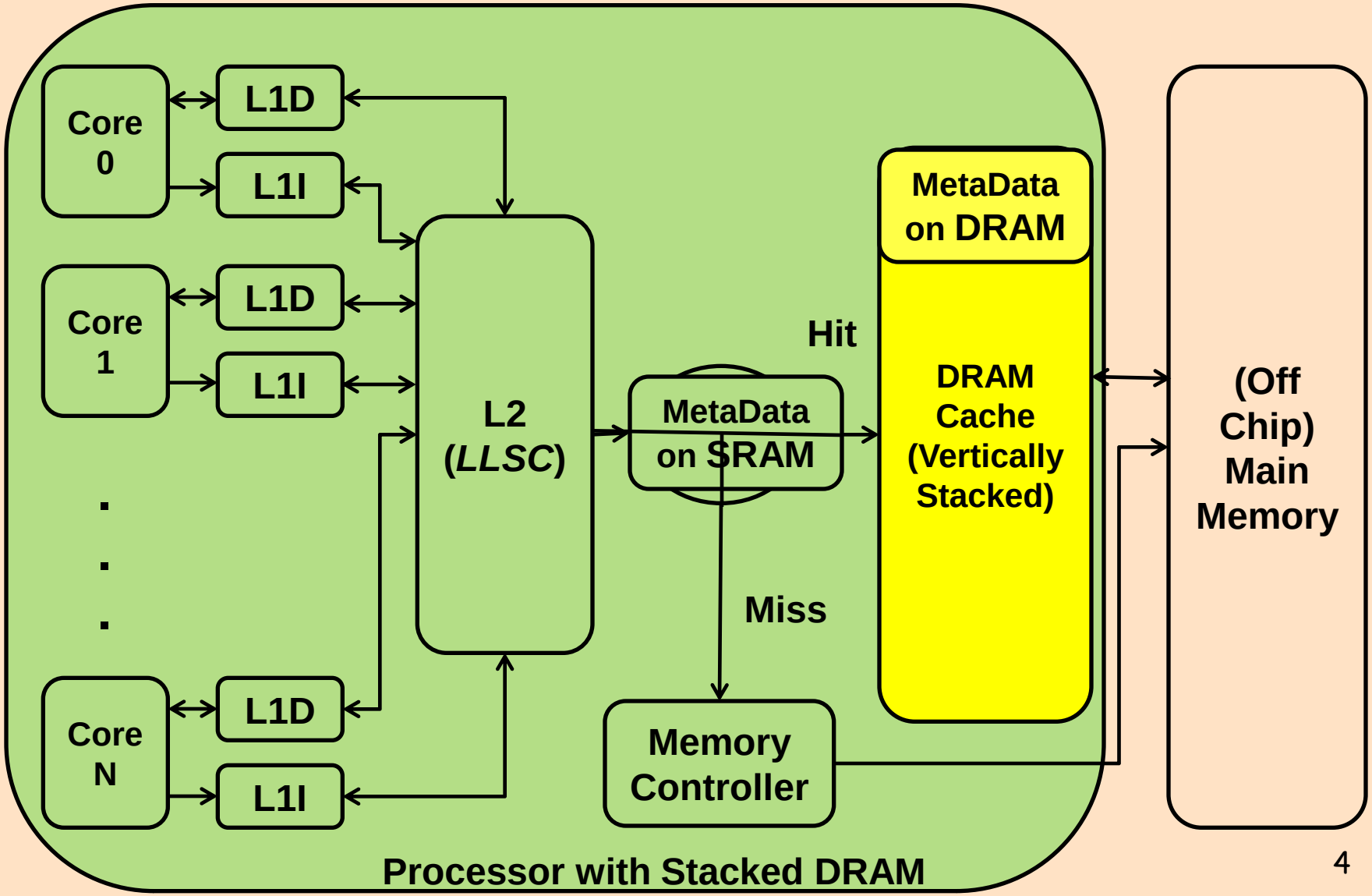
Stacked DRAM

- DRAM vertically stacked over the processor die.
- Stacked DRAMs offer
  - High bandwidth
  - High capacity
  - Moderately low latency.
- Several proposals to organize this large DRAM as a last-level cache.



Picture courtesy Bryan Black (From MICRO 2013 Keynote)

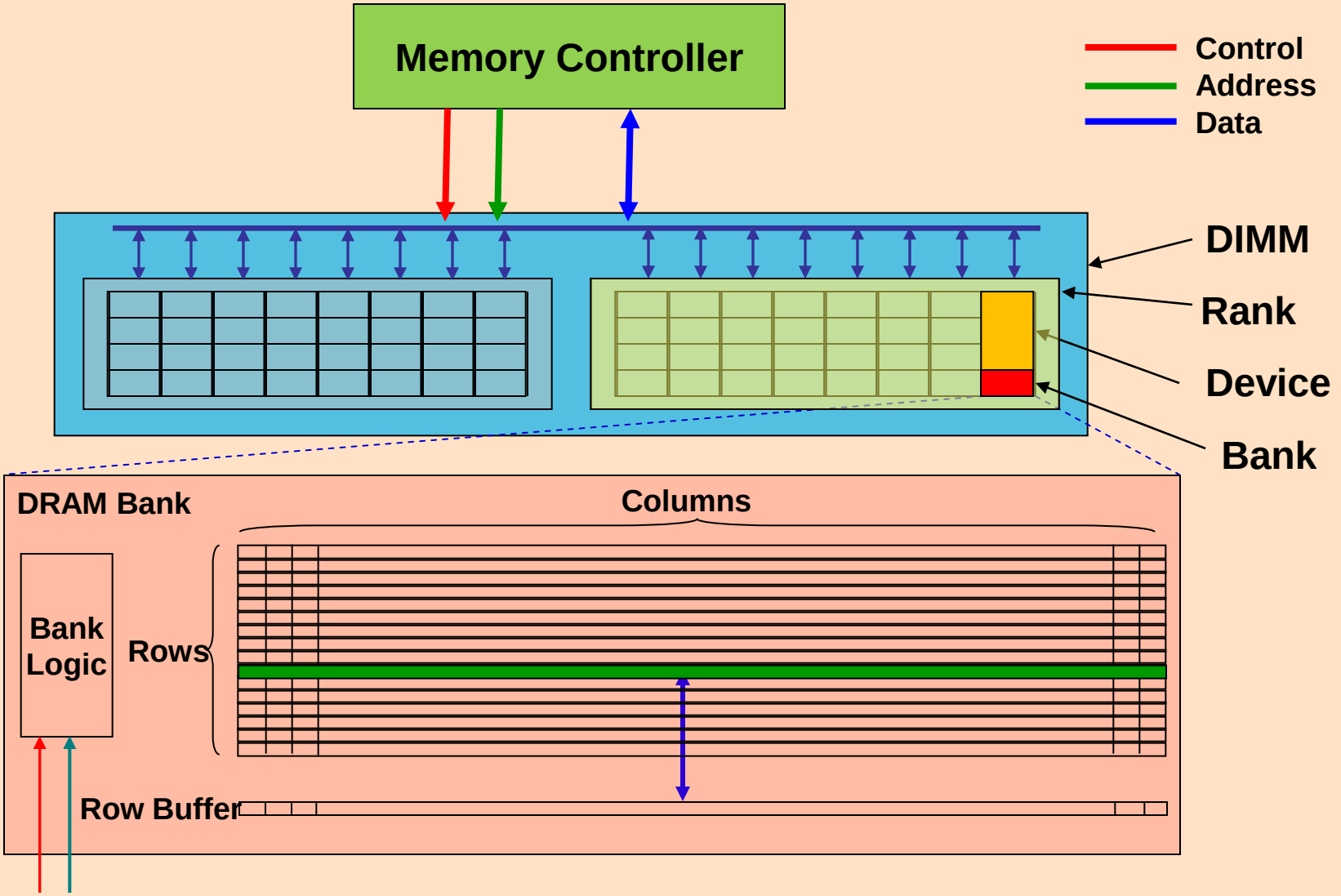
Processor Orgn. With DRAM Cache



Talk Outline

- Introduction to stacked DRAM Caches
- Background (An overview of *ANATOMY*)
- *ANATOMY-Cache*: Modeling Stacked DRAM Cache Organizations
- Evaluation
- Insights
- Conclusions

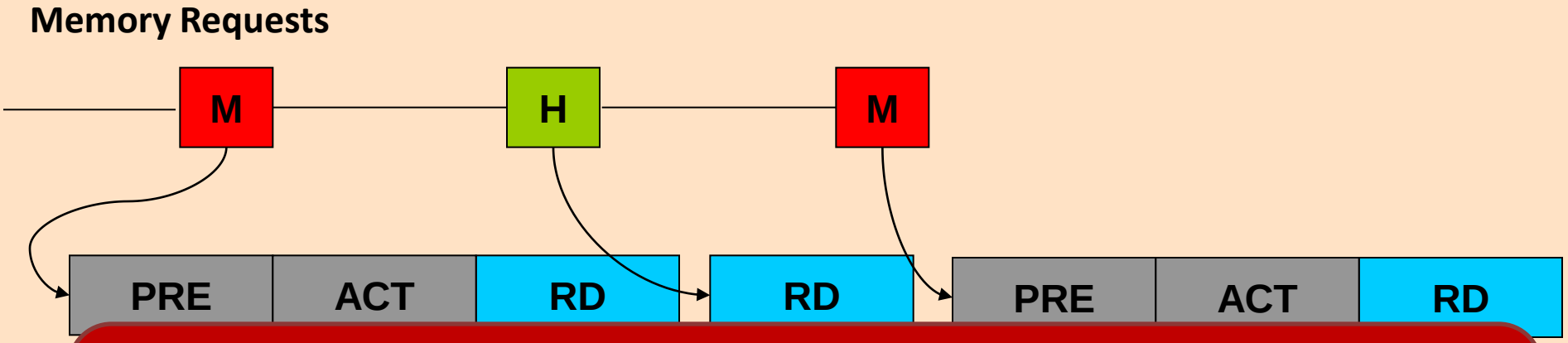
Overview of a DRAM based memory



Data Read & Write operations

Basic DRAM Operations

- ACTIVATE ➔ Bring data from DRAM core into the row-buffer
- READ/WRITE ➔ Perform read/write operations on the contents in the row-buffer
- PRECHARGE ➔ Store data back to DRAM core (ACTIVATE discharges capacitors), put cells back at neutral voltage



Bank Level Parallelism (BLP)

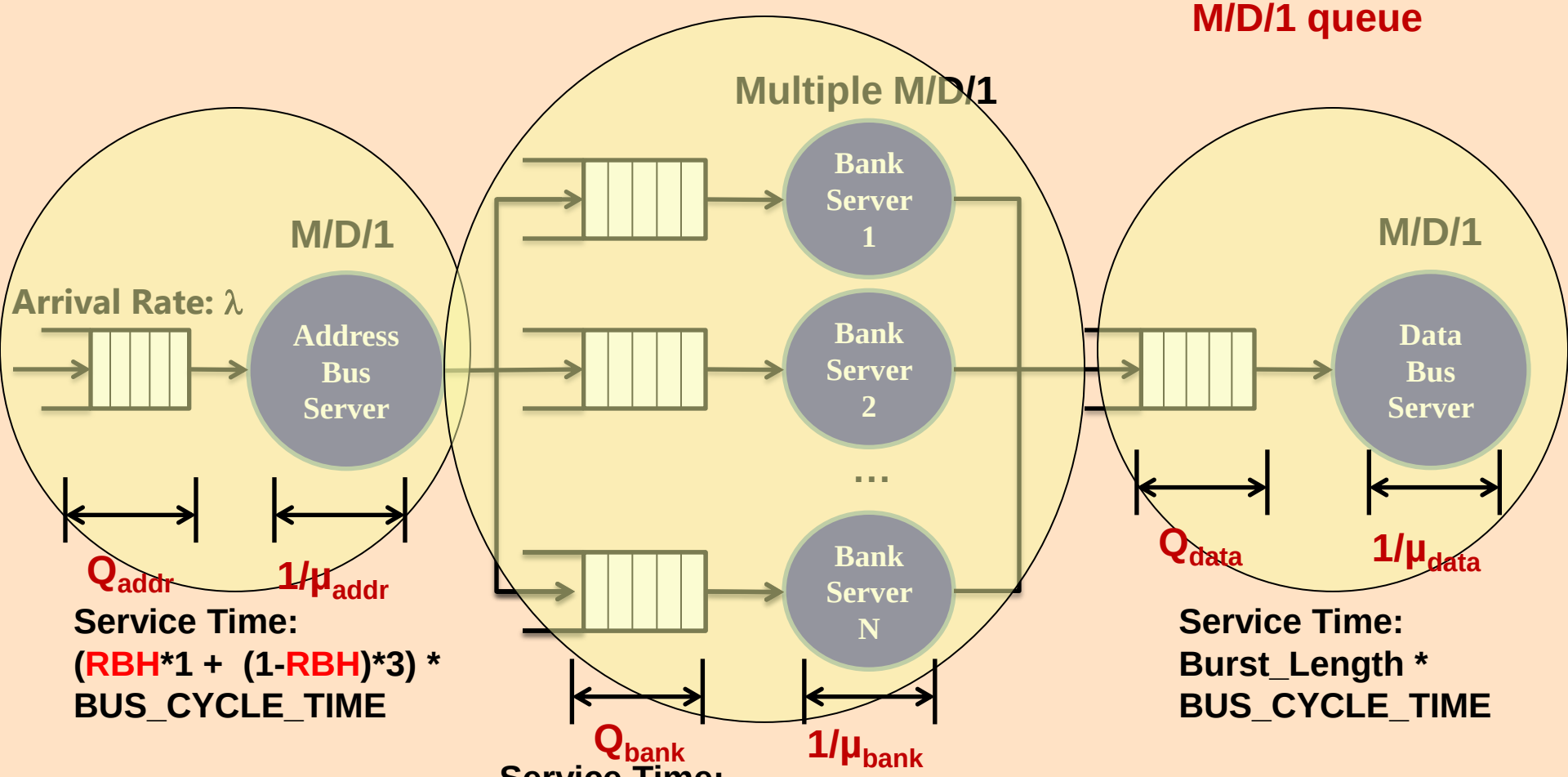
- Parallelism improves performance
- Some switching delays hurt performance

ANATOMY – Analytical Model of Memory

Two components

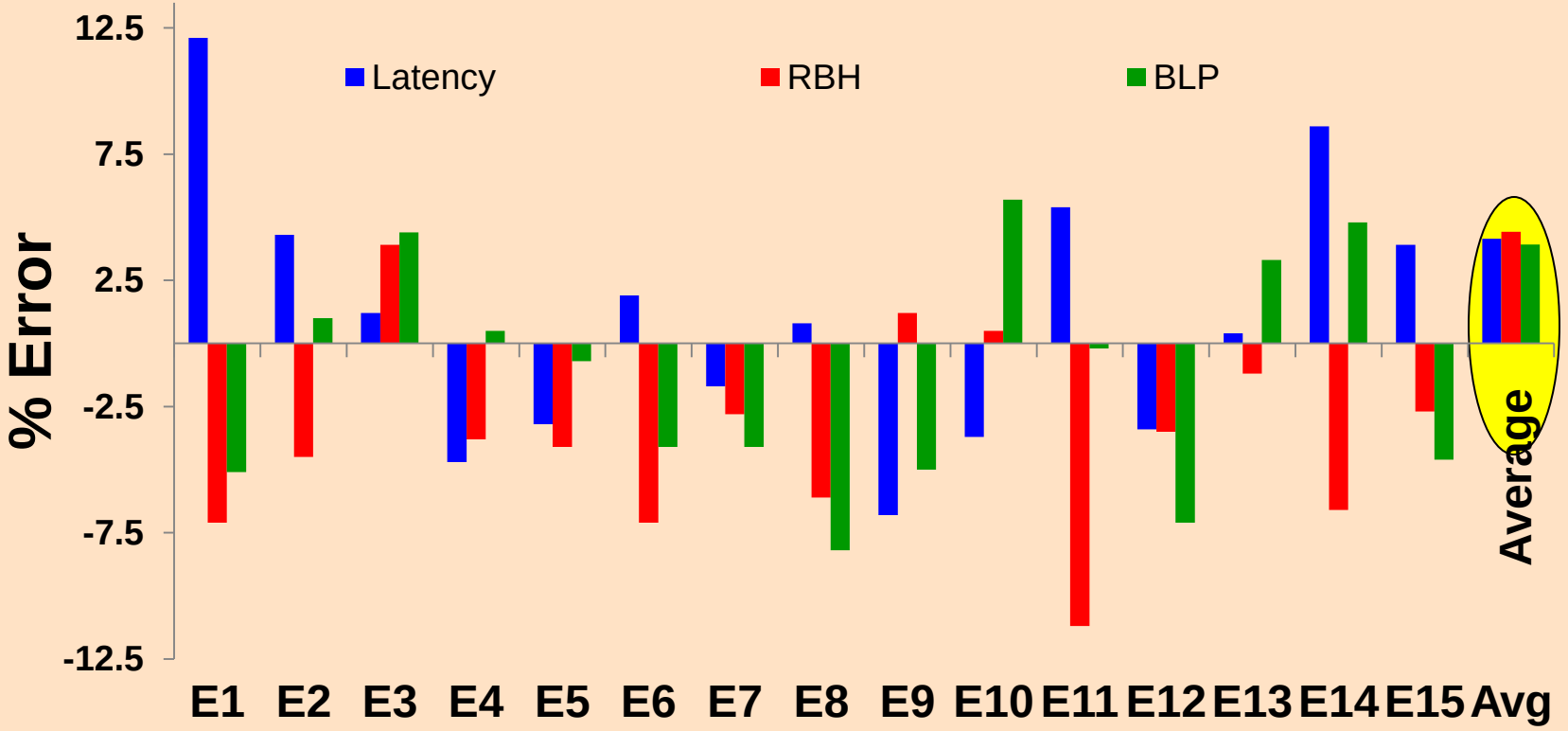
1) Queuing Model of Memory
   - Organizational and Technological characteristics
   - Workload characteristics used as input
2) Use of Workload Characteristics
   - Locality and Parallelism in workload's memory accesses

Analytical Model for Memory System Performance



$$\text{Latency} = Q_{addr} + Q_{bank} + Q_{data} + 1/\mu_{addr} + 1/\mu_{bank} + 1/\mu_{data}$$

Validation - Model Accuracy

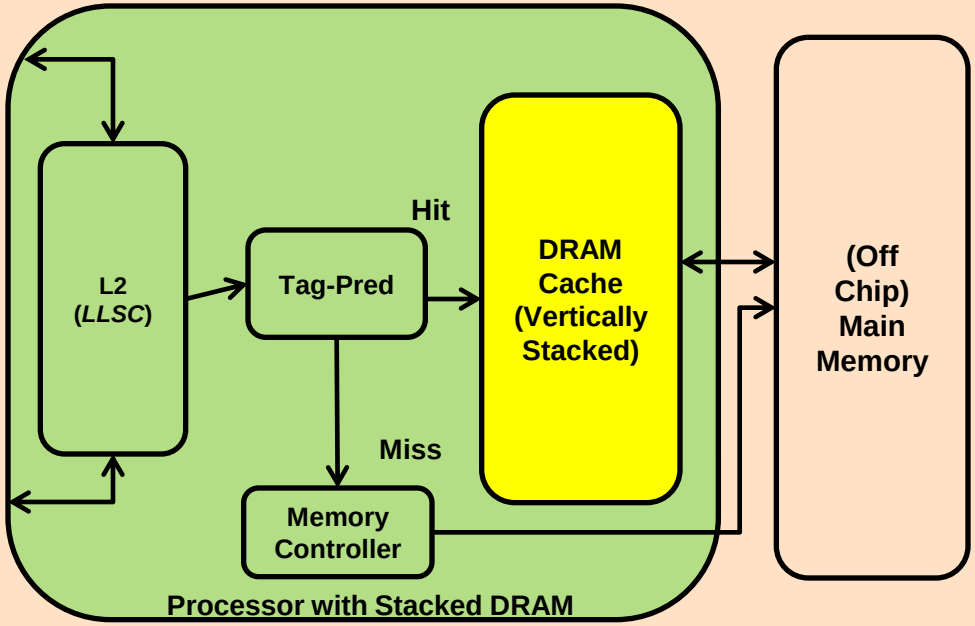


- Low Errors in RBH, BLP and Latency Estimation
  - Average error of 3.9%, 4.2% and 4%
- ANATOMY predicts trends accurately

Talk Outline

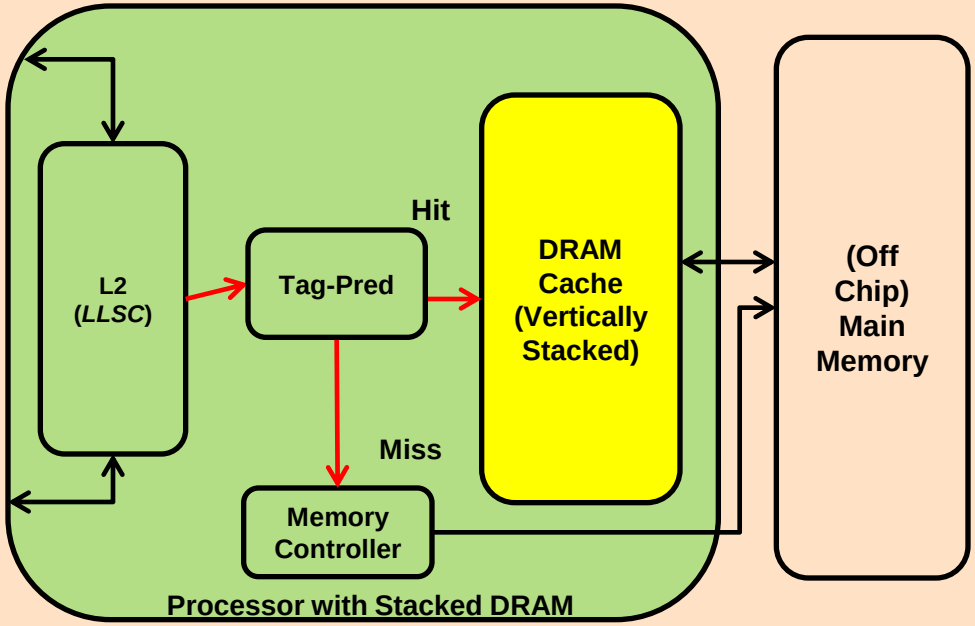
- Introduction to stacked DRAM Caches
- Background (An overview of *ANATOMY*)
- *ANATOMY-Cache*: Modeling Stacked DRAM Cache Organizations
- Evaluation
- Insights
- Conclusions

ANATOMY-Cache Model



Key Parameters that govern performance:

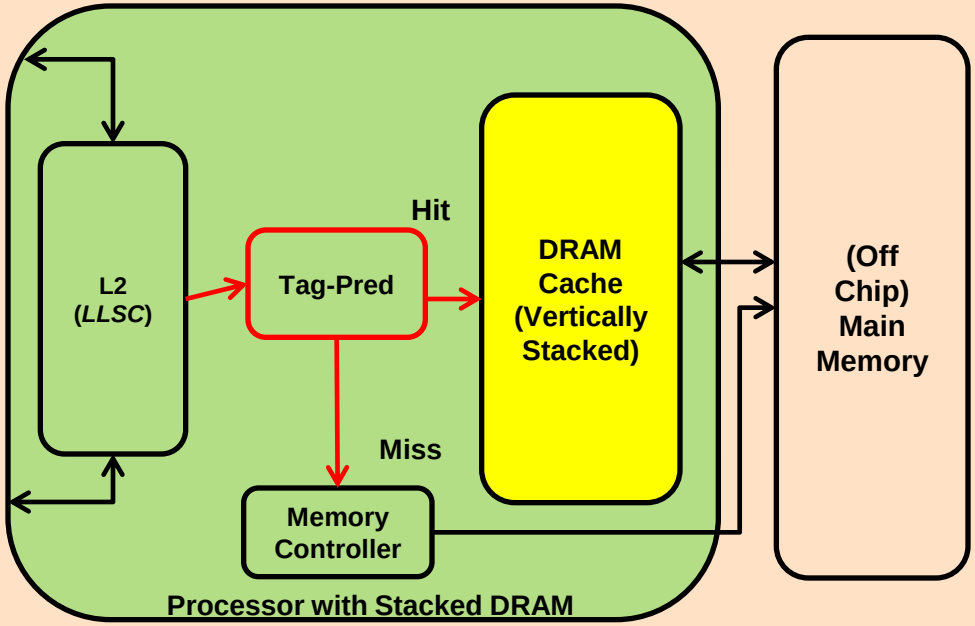
ANATOMY-Cache Model



Key Parameters that govern performance:

- Arrival Rate

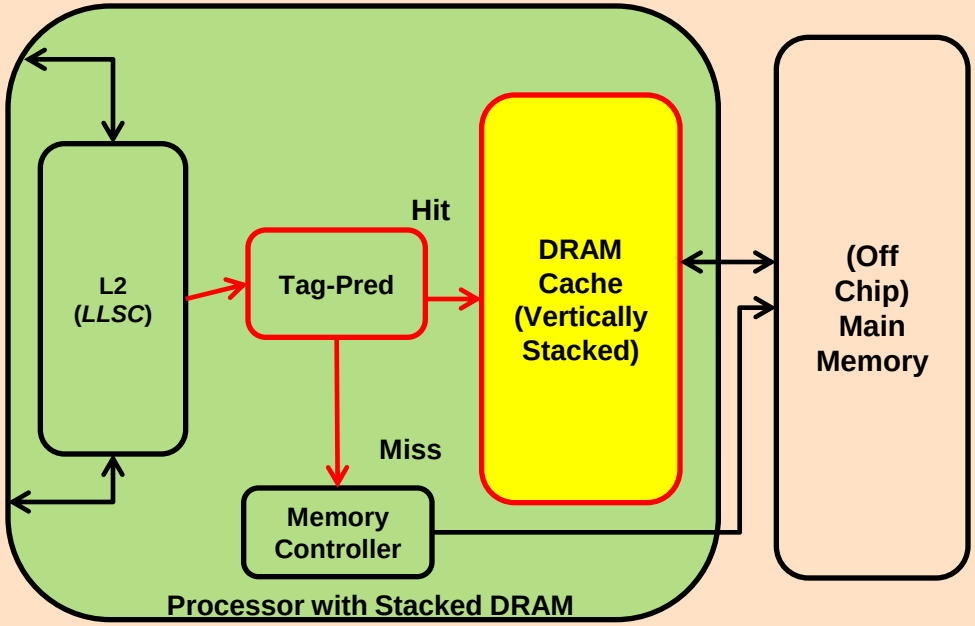
ANATOMY-Cache Model



Key Parameters that govern performance:

- Arrival Rate
- Tag access time

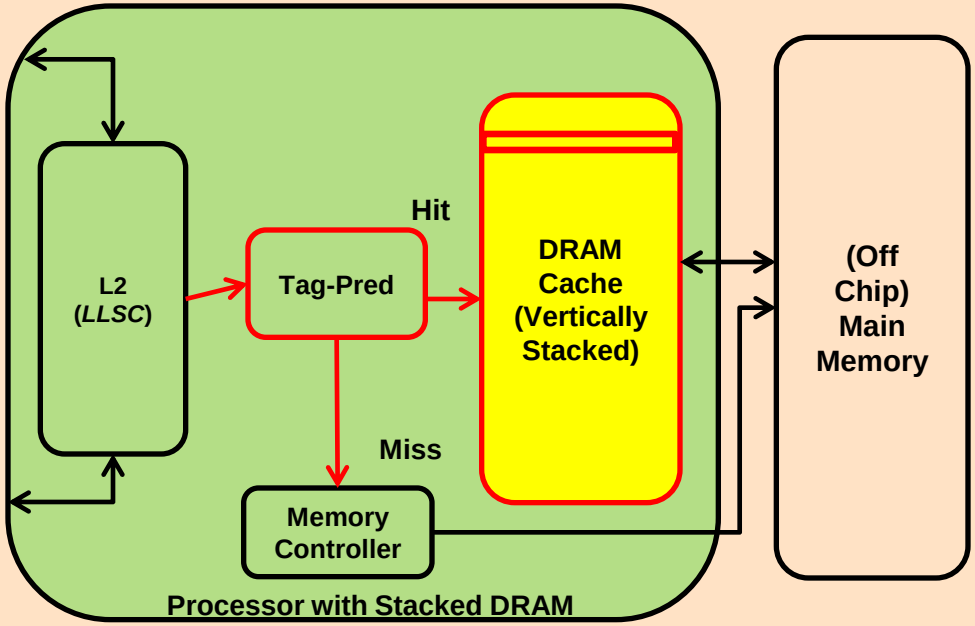
ANATOMY-Cache Model



Key Parameters that govern performance:

- Arrival Rate
- Tag access time
- Cache hit rate

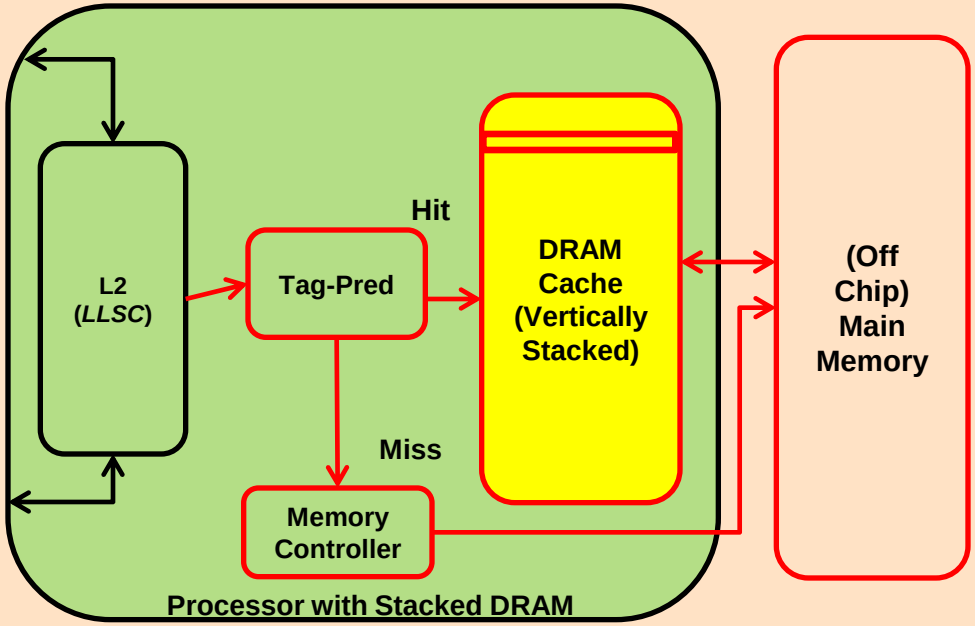
ANATOMY-Cache Model



Key Parameters that govern performance:

- Arrival Rate
- Tag access time
- Cache hit rate
- Cache RBH

ANATOMY-Cache Model

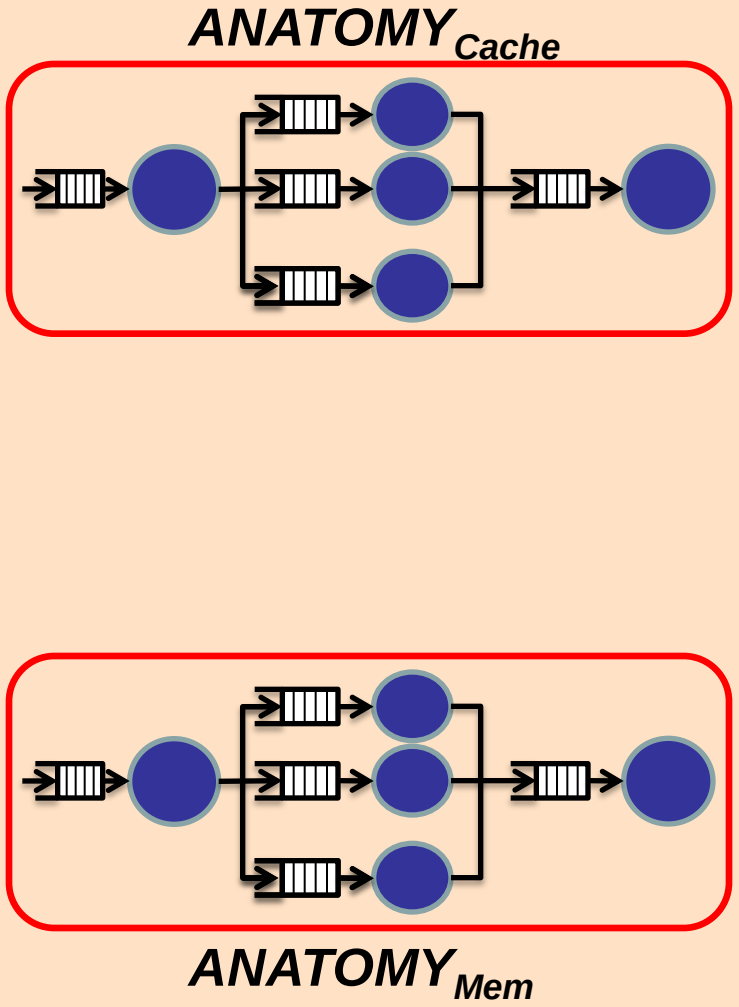


Key Parameters that govern performance:

- Arrival Rate
- Tag access time
- Cache hit rate
- Cache RBH
- Cache Miss Penalty

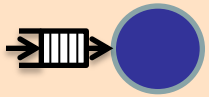
Extending *ANATOMY* to DRAM Caches

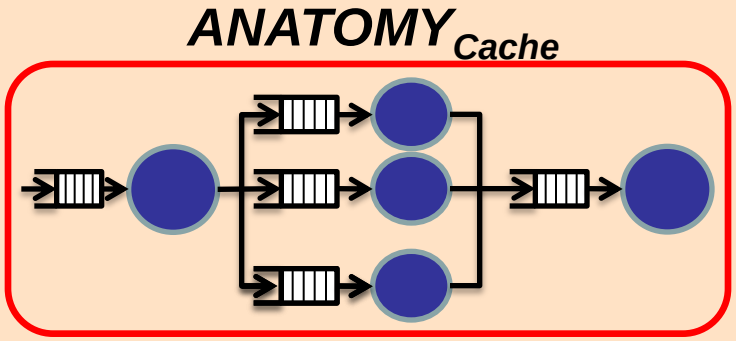
- Two *ANATOMY* instances - one for DRAM cache and one for main memory.

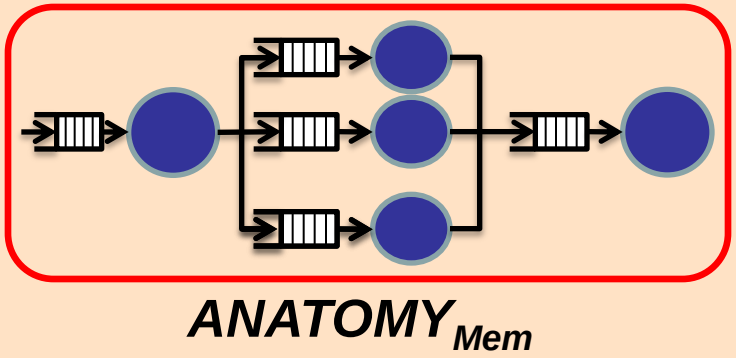


Extending *ANATOMY* to DRAM Caches

- Two *ANATOMY* instances - one for DRAM cache and one for main memory.
- The models are fed by the output of the tag server and each other's outputs.

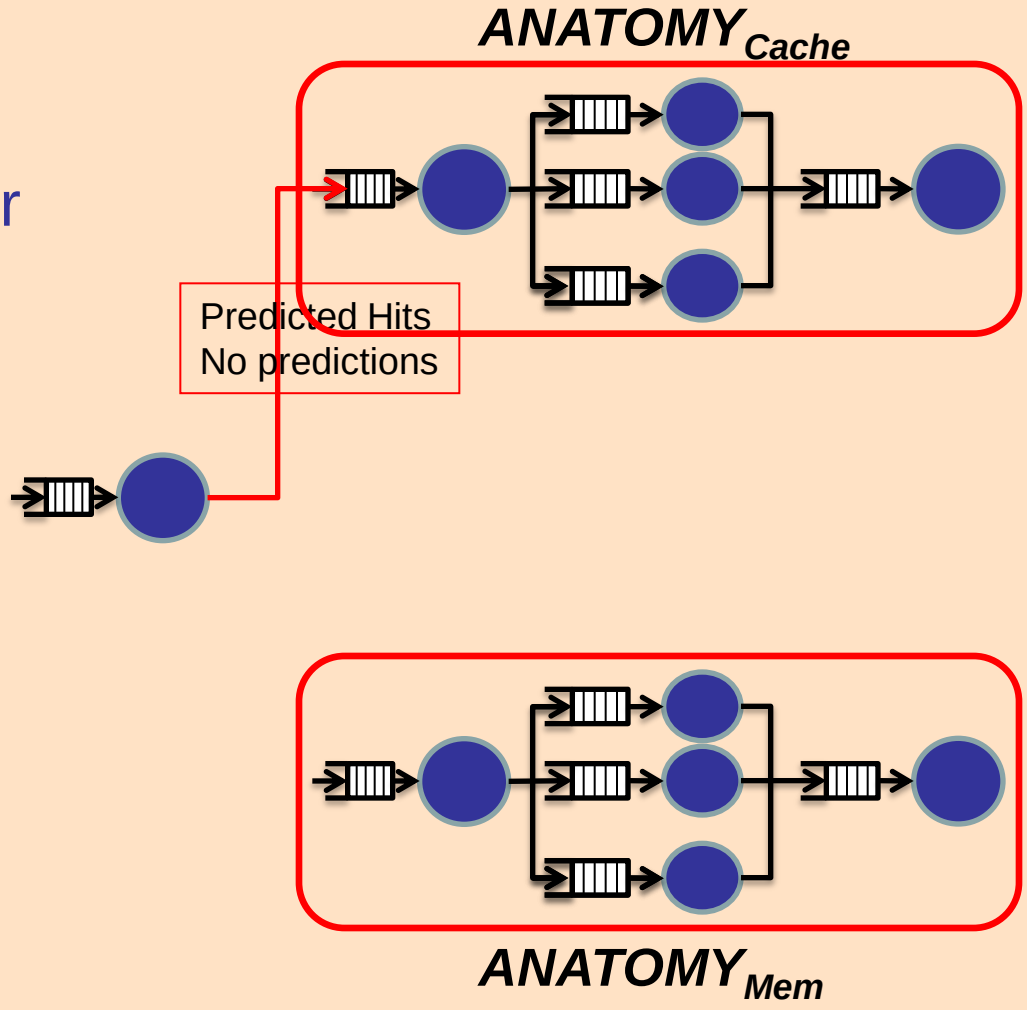






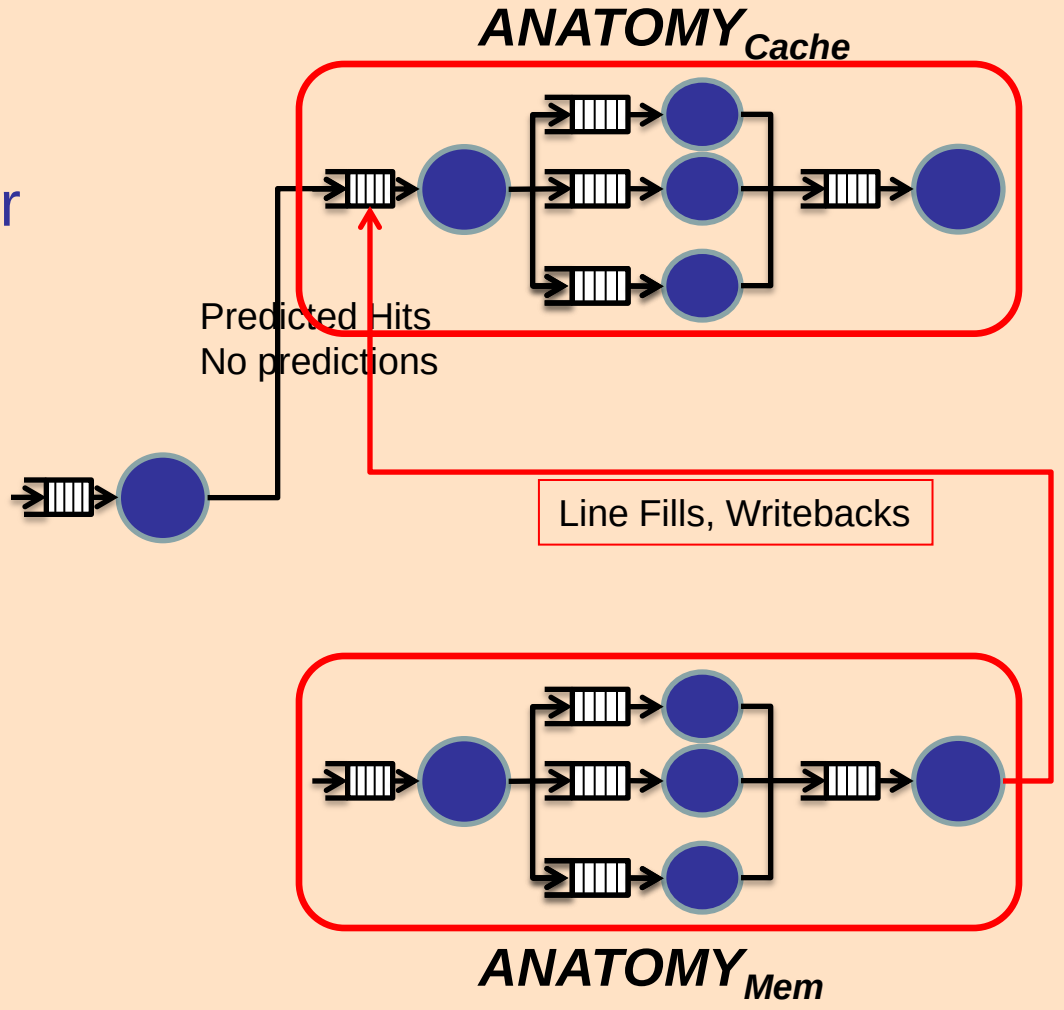
Extending *ANATOMY* to DRAM Caches

- Two *ANATOMY* instances - one for DRAM cache and one for main memory.
- The models are fed by the output of the tag server and each other's outputs.
  - Predicted Cache Hits
  - No Predictions



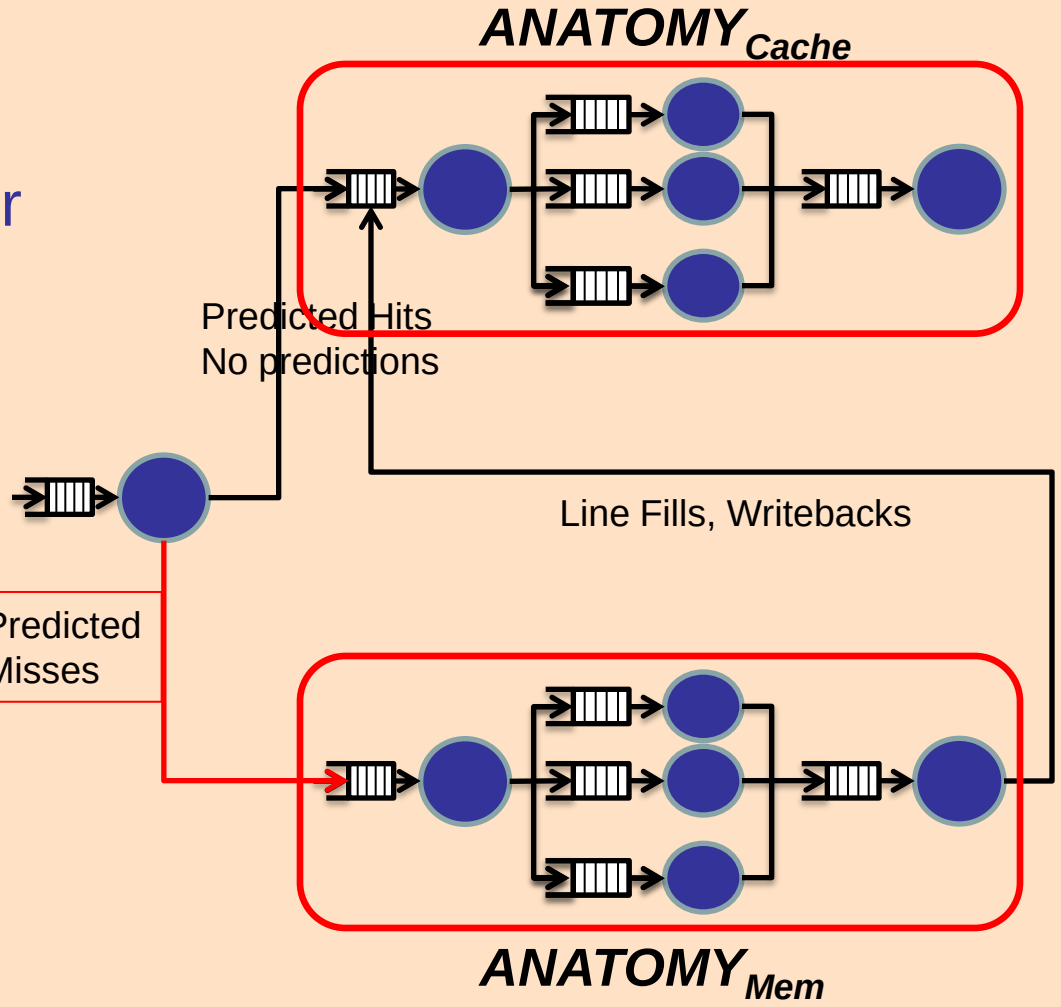
Extending *ANATOMY* to DRAM Caches

- Two *ANATOMY* instances - one for DRAM cache and one for main memory.
- The models are fed by the output of the tag server and each other's outputs.
  - Predicted Cache Hits
  - No Predictions
  - Line fills and write back requests from main memory



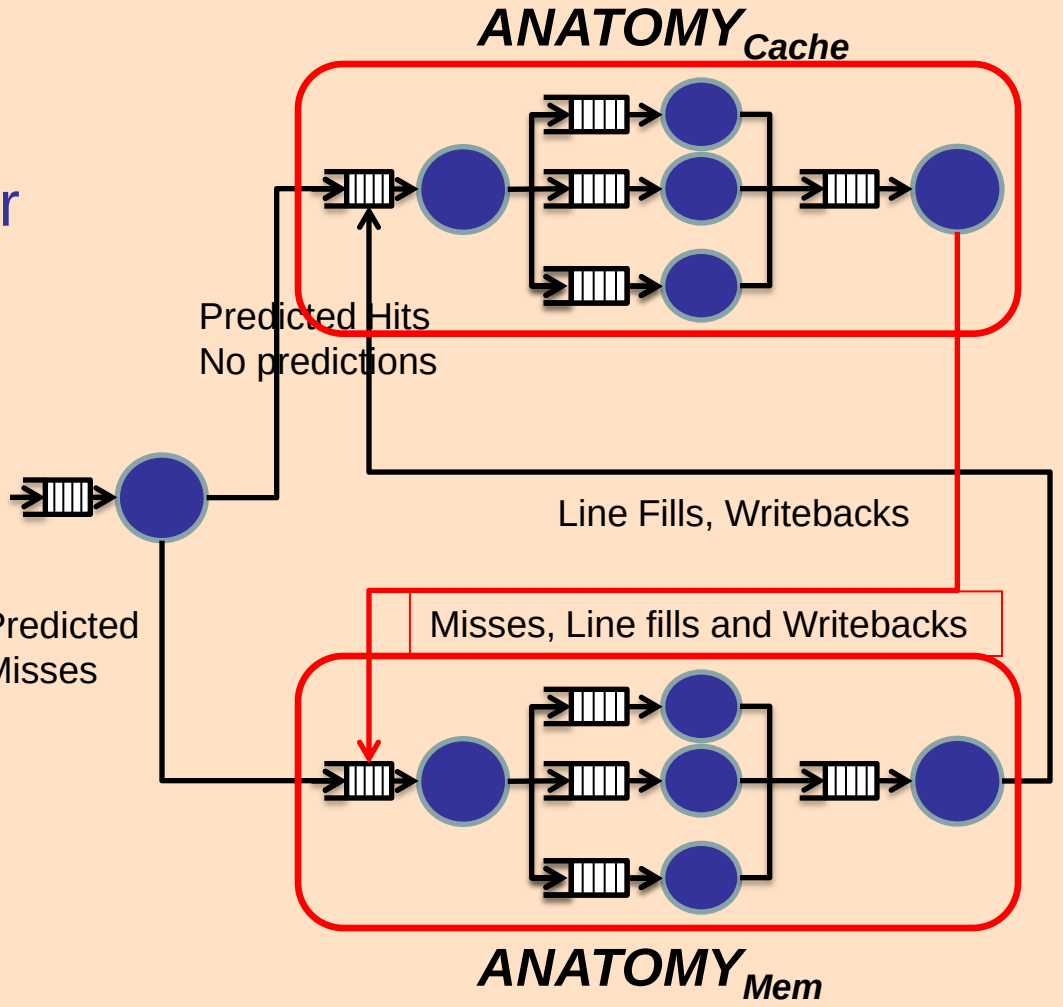
Extending *ANATOMY* to DRAM Caches

- Two *ANATOMY* instances - one for DRAM cache and one for main memory.
- The models are fed by the output of the tag server and each other's outputs.
  - Predicted Cache Hits
  - No Predictions
  - Line fills and write back requests from main memory
  - Predicted Misses



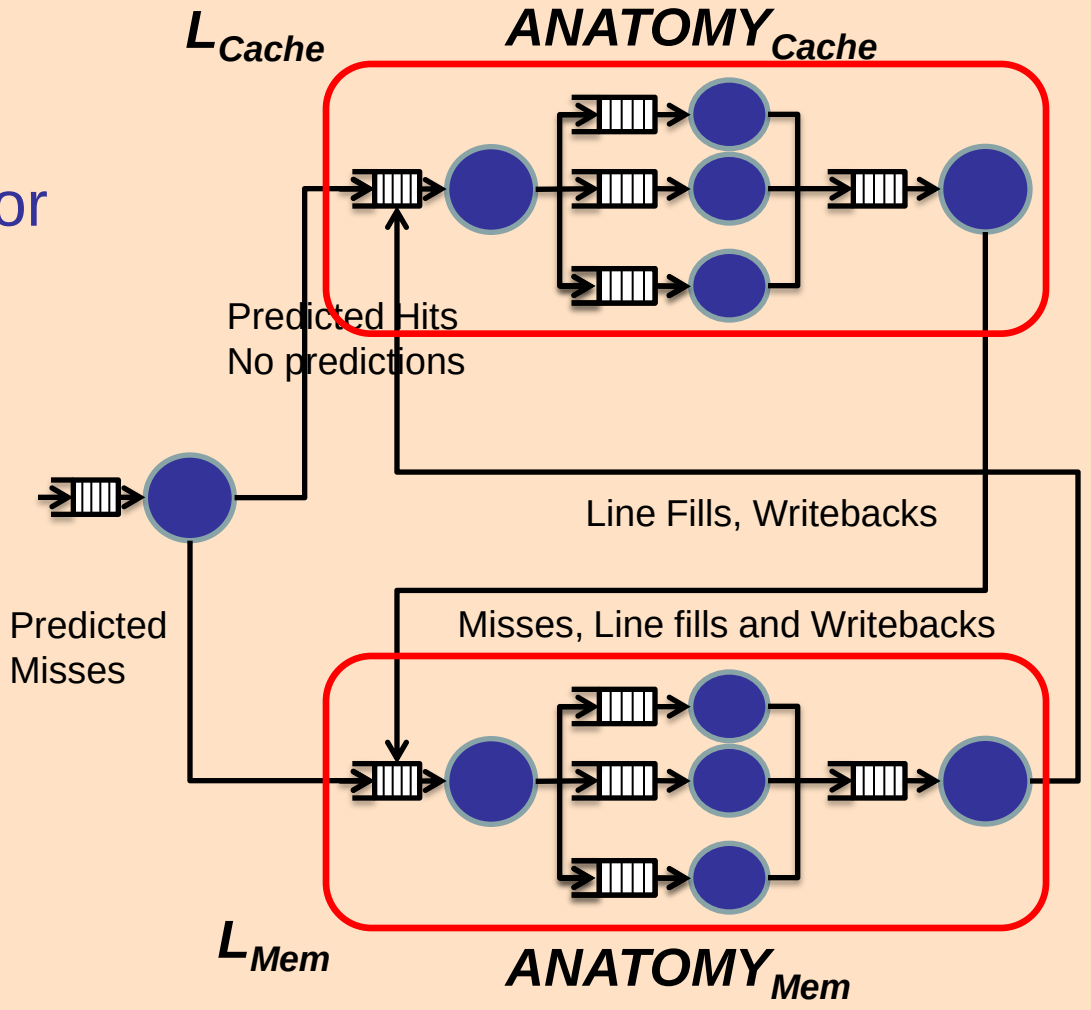
Extending *ANATOMY* to DRAM Caches

- Two *ANATOMY* instances - one for DRAM cache and one for main memory.
- The models are fed by the output of the tag server and each other's outputs.
  - Predicted Cache Hits
  - No Predictions
  - Line fills and write back requests from main memory
  - Predicted Misses
  - Requests from Cache



Extending *ANATOMY* to DRAM Caches

- Two *ANATOMY* instances - one for DRAM cache and one for main memory.
- The models are fed by the output of the tag server and each other's outputs.
- We compute the latencies at the cache and memory using *ANATOMY*.



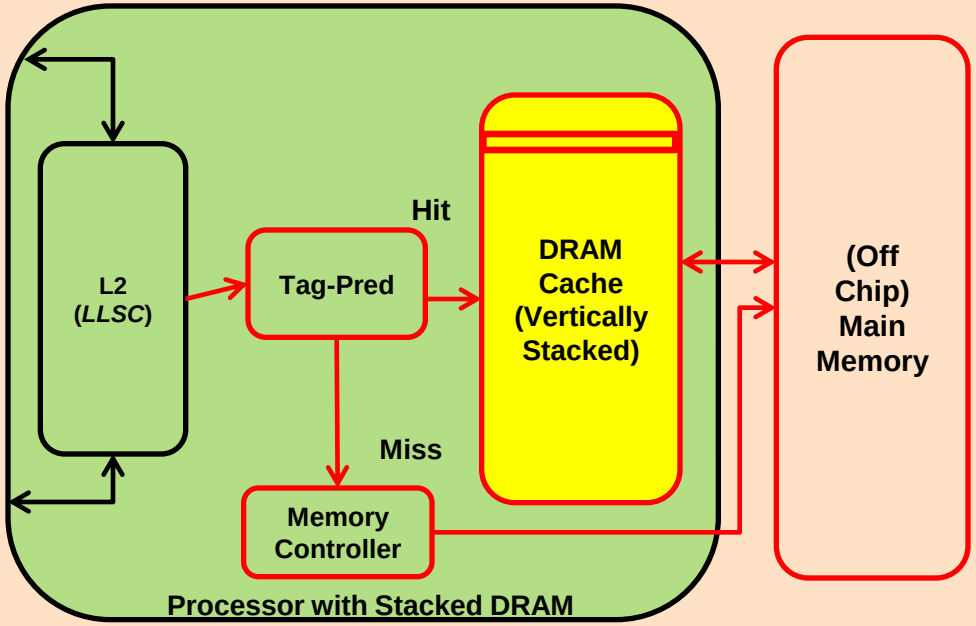
Obtaining the average LLSC miss penalty

- L_{cache} and L_{mem} are combined by to estimate the average LLSC miss penalty.

- But first we discuss the estimation of the key parameters that govern L_{Cache} and L_{Mem}.

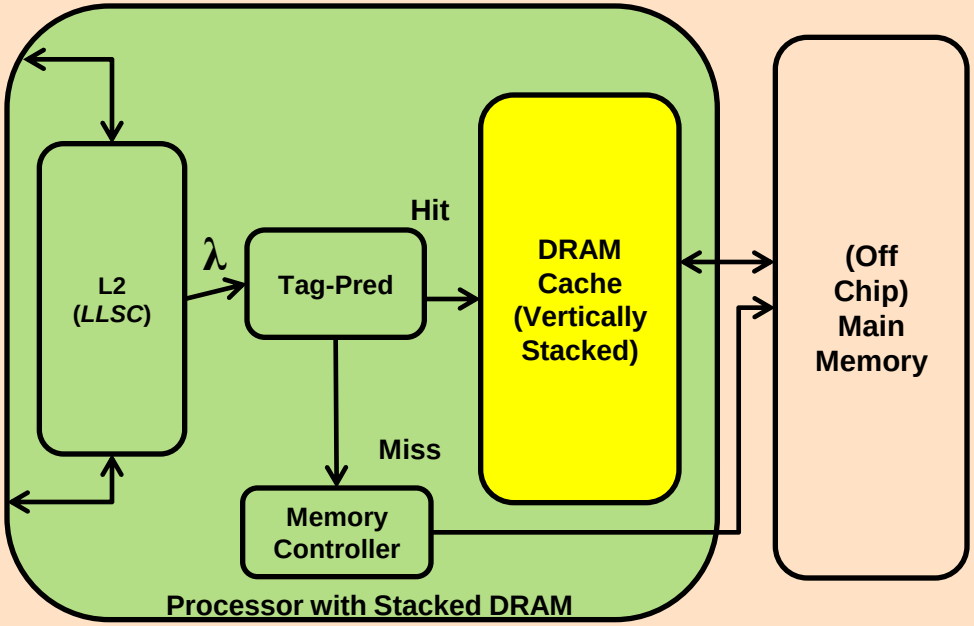
Estimating Key Parameters …

- Arrival Rate
- Tag access time
- Cache hit rate
- Cache RBH
- Cache Miss Penalty



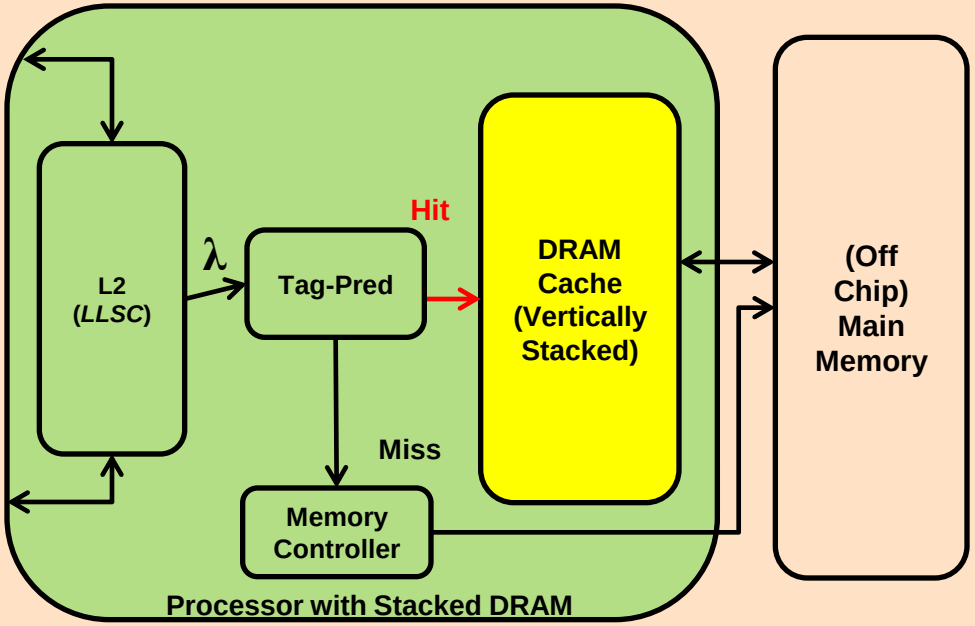
Estimating the Cache Arrival Rate

- Arrival Rate at the Cache is a sum of several streams of accesses.



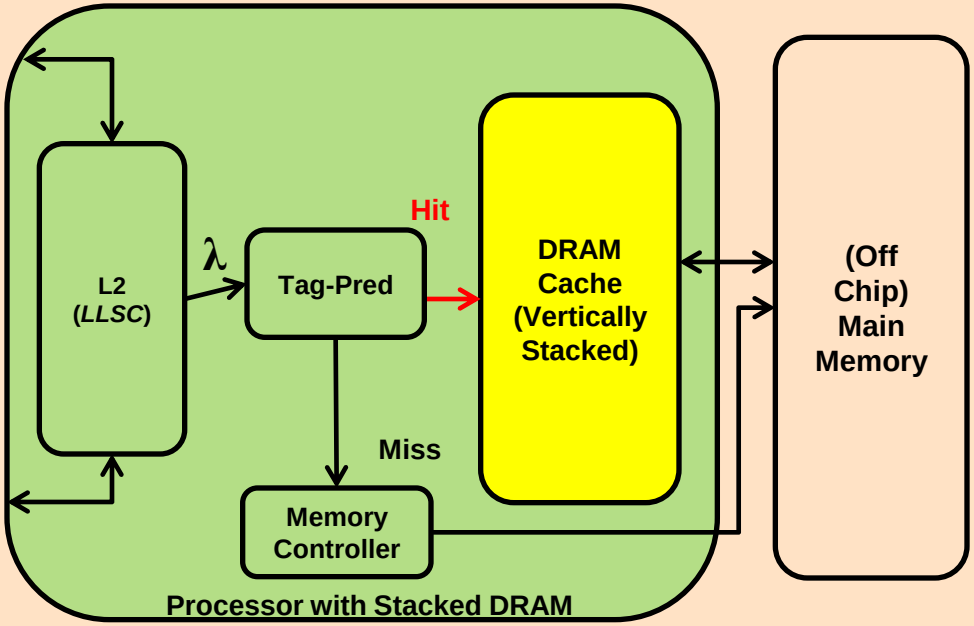
Estimating the Cache Arrival Rate

- Arrival Rate at the Cache is a sum of several streams of accesses.
- Predicted Hits



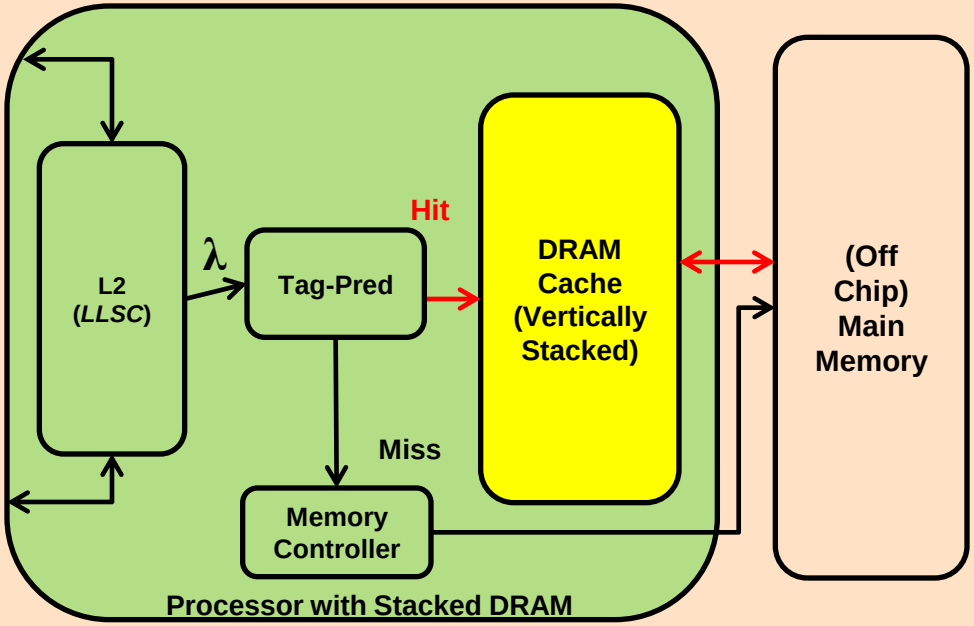
Estimating the Cache Arrival Rate

- Arrival Rate at the Cache is a sum of several streams of accesses.
- Predicted Hits
- No predictions



Estimating the Cache Arrival Rate

- Arrival Rate at the Cache is a sum of several streams of accesses.
- Predicted Hits
- No predictions
- Line fills and writebacks



Summarizing the Cache Arrival Rate

Request Stream	Rate	Notes
Predicted Hits	$\lambda * h_{pred} * h_{cache}$	
No predictions	$\lambda * (1-h_{pred})$	They are sent to the cache for tag look-up
Line Fills	$\lambda * (1-h_{cache}) * B_s$	B_s is the cache block size
Writebacks	$\lambda * (1-h_{cache}) * w$	w is the fraction of misses that cause write-backs

$$\lambda_{cache} = \lambda * h_{pred} * h_{cache} + \lambda * (1-h_{pred}) + \lambda * (1-h_{cache}) * B_s + \lambda * (1-h_{cache}) * w$$

Estimating Tag Predictor Hit Rate and Access Time

Tags-on-SRAM

- All tags on SRAM.
- Hit Rate = 100%

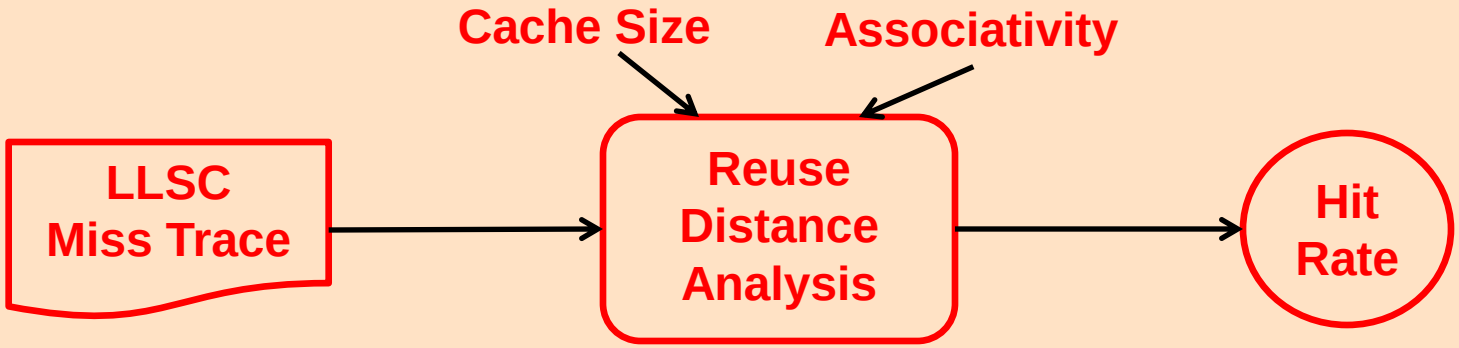
Tags-on-DRAM

- A small set-associative cache.
- Hit Rate determined by running an access trace through the cache model.

Predictor access time depends on its size.
An estimate is obtained using CACTII.

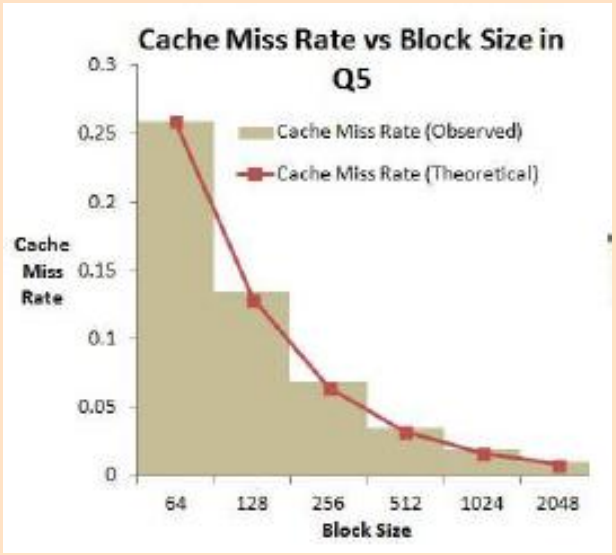
Estimating Cache Hit Rate

- Cache Hit Rate depends on 3 key parameters:
  - Cache Size
  - Set Associativity
  - Block Size
- Well-studied problem
  - A trace-based model and reuse distance analysis.
- We use a trace of accesses from the LLSC.



Estimating Cache Hit Rate with Block Size

- Larger block sizes can capture spatial locality.
- Bandwidth-neutral model: *Cache miss rate halves with doubling of cache block size.*
  - » If this holds, then measuring hit rate at smallest block size via trace based analysis is sufficient.
  - » For larger block sizes, estimate via:

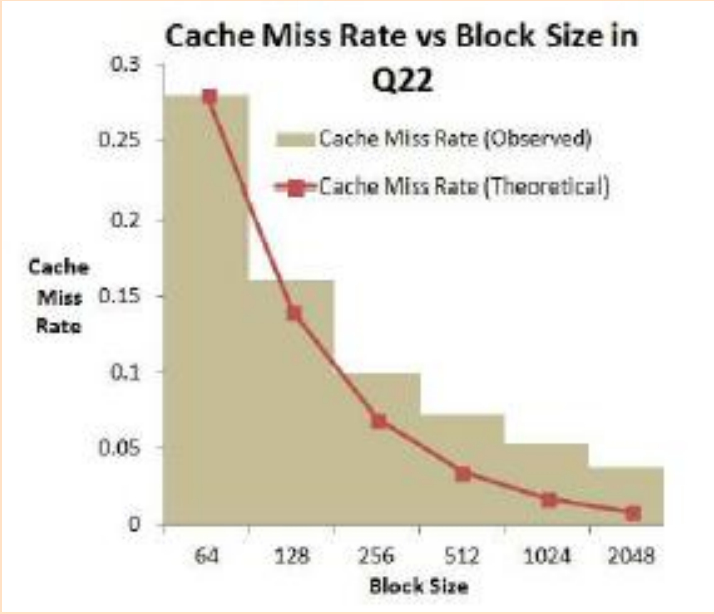


$$h_{b_s} = 1 - \frac{(1 - h_1)}{B_s}$$

Workload Q5 is bandwidth-neutral

Not all workloads are bandwidth-neutral

- For such workloads, bandwidth-neutral model leads to lower miss rate prediction.
- Use trace-based cache simulations in such cases.



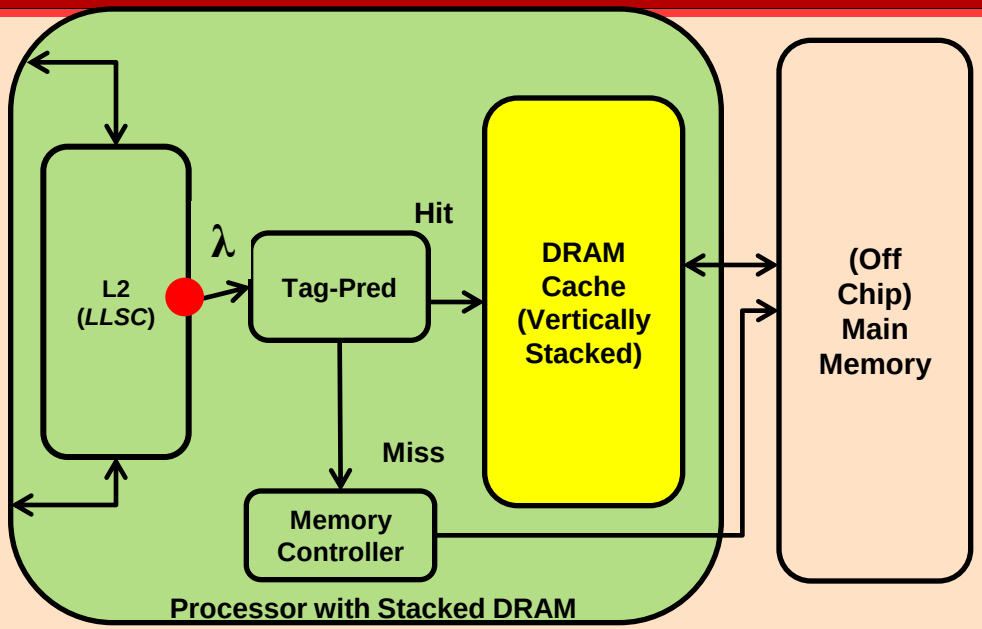
Workload Q22 is NOT bandwidth-neutral

Estimating DRAM Cache Row-Buffer Hit Rate

- Row-Buffer Hit rate (RBH) of the DRAM cache depends on the access pattern and the data organization on the DRAM.
- We estimate RBH using the Reuse-Distance framework similar to *ANATOMY*.
- Details are in the paper.

Putting them together …

- LLSC Miss Penalty from: L_{cache} and L_{mem}



Event	Latency
Predicted Cache Hit	A: $h_{pred}*h_{cache}*L_{cache}$
Predicted Cache Miss	B: $h_{pred}*(1-h_{cache})*L_{mem}$
No Prediction and Cache Hit	C: $(1-h_{pred})*h_{cache}*L_{cache}$
No Prediction and Cache Miss	D: $(1-h_{pred})*(1-h_{cache})*[L_{cache}+L_{mem}]$
L_{Avg}	A+B+C+D

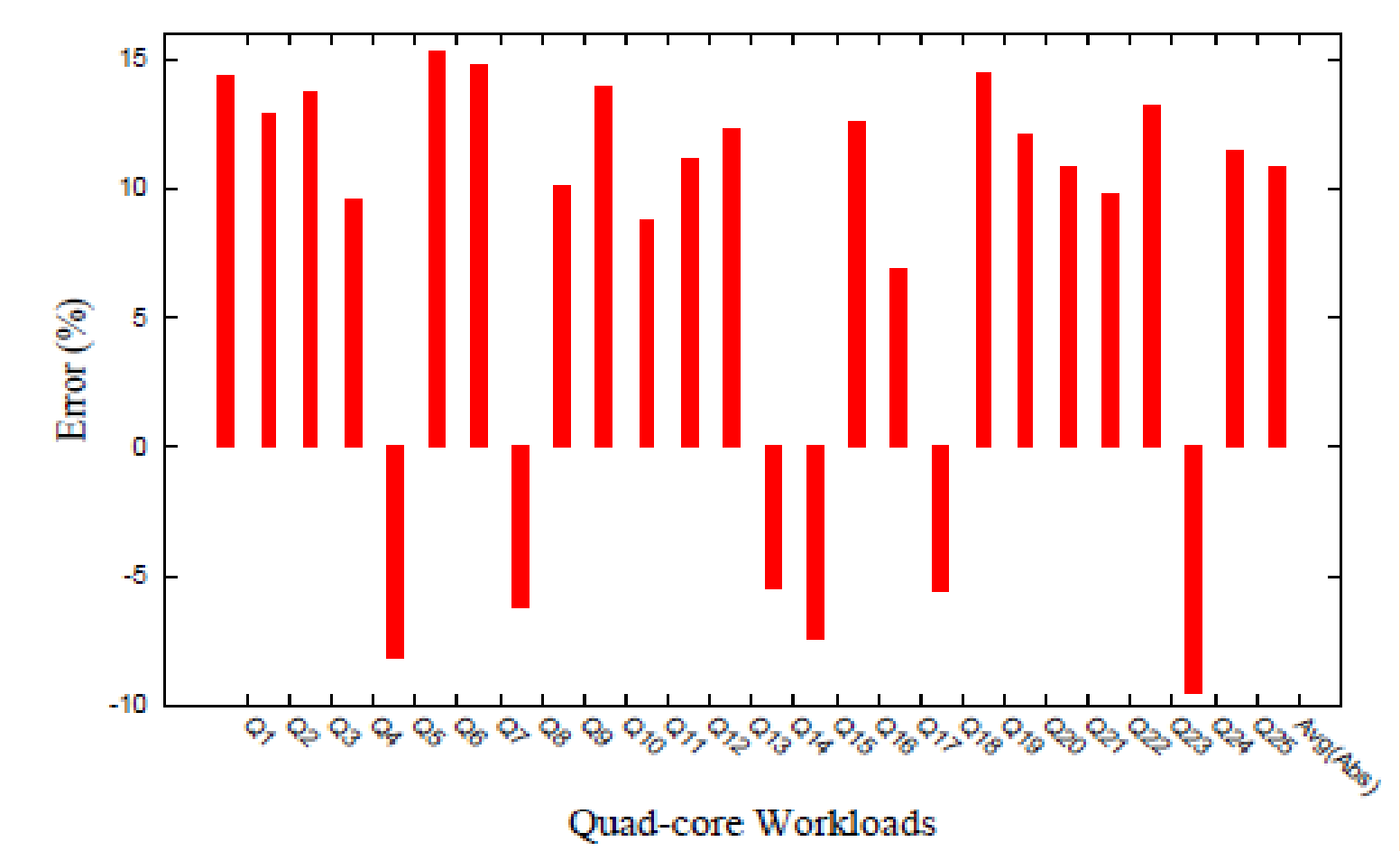
Talk Outline

- Introduction to stacked DRAM Caches
- Background (An overview of *ANATOMY*)
- *ANATOMY-Cache*: Modeling Stacked DRAM Cache Organizations
- Evaluation
- Insights
- Conclusions

Experimental Evaluation

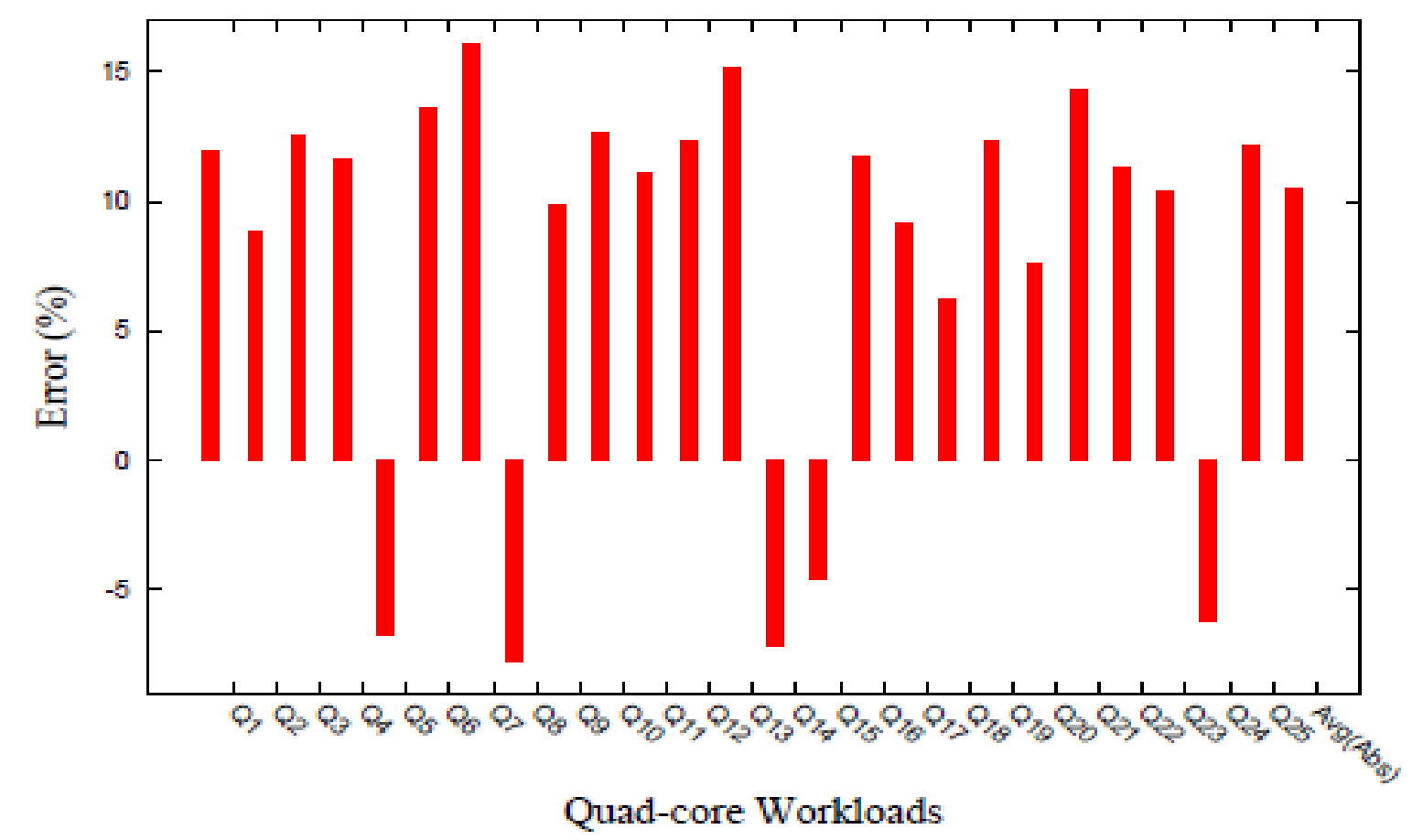
- Validation using GEM5 Simulation (with detailed Memory model)
- Use of Multiprogrammed workloads
- Workloads comprising of SPEC2000/SPEC2006 benchmarks
- Architecture Configurations
  - 4 core and 8 core
  - 128MB (4 core) and 256MB (8 core) DRAM caches
  - Cache Memory: 1.6GHz DRAM, 2KB page, 128-bit bus
  - DRAM Main Memory: 3.2GHz DRAM, 64-bit bus
  - Tags-on-DRAM:
    - Direct Mapped 64B block size
    - Tags and Data on the same DRAM rows
    - Tag Predictor: 2-way set associative tag cache
  - Tags-on-SRAM:
    - Block Size: 1024B
    - 2 way set associative

Validation of the Tags-on-DRAM Model



- Low errors in estimation of Avg. LLSC Miss Penalty (10.9% in 4-core and 9.3% in 8-core workloads)

Validation of the Tags-on-SRAM Model



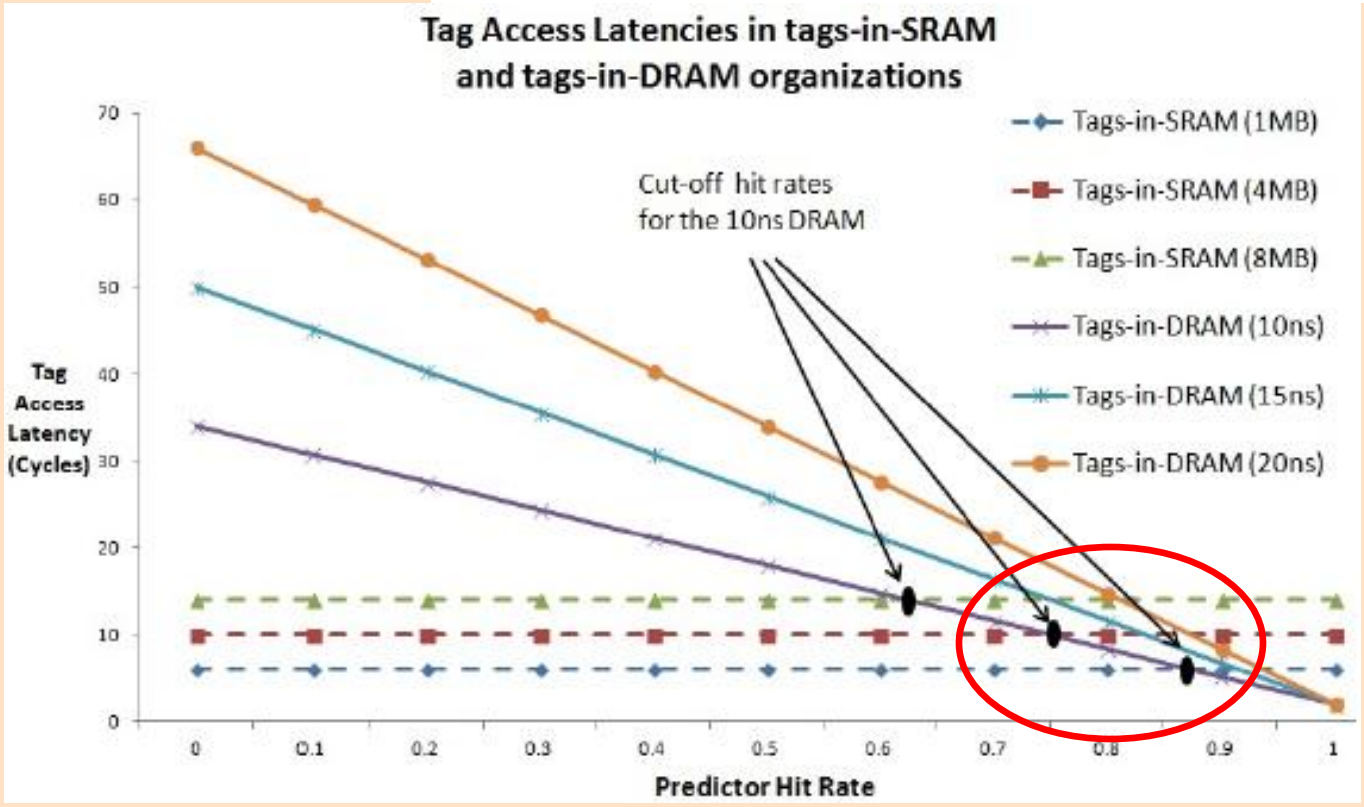
- Low errors in estimation of Avg. LLSC Miss Penalty (10.5% in 4-core and 8.2% in 8-core workloads)

Talk Outline

- Introduction to stacked DRAM Caches
- Background (An overview of *ANATOMY*)
- *ANATOMY-Cache*: Modeling Stacked DRAM Cache Organizations
- Evaluation
- Insights
- Conclusions

Insight 1: It is hard to out-perform Tags-on-SRAM designs

Tag Access Time: $t_{tag}^{tags-in-dram} = h_{pred} * t_{pred} + (1 - h_{pred}) * t_{dram\_cache}$



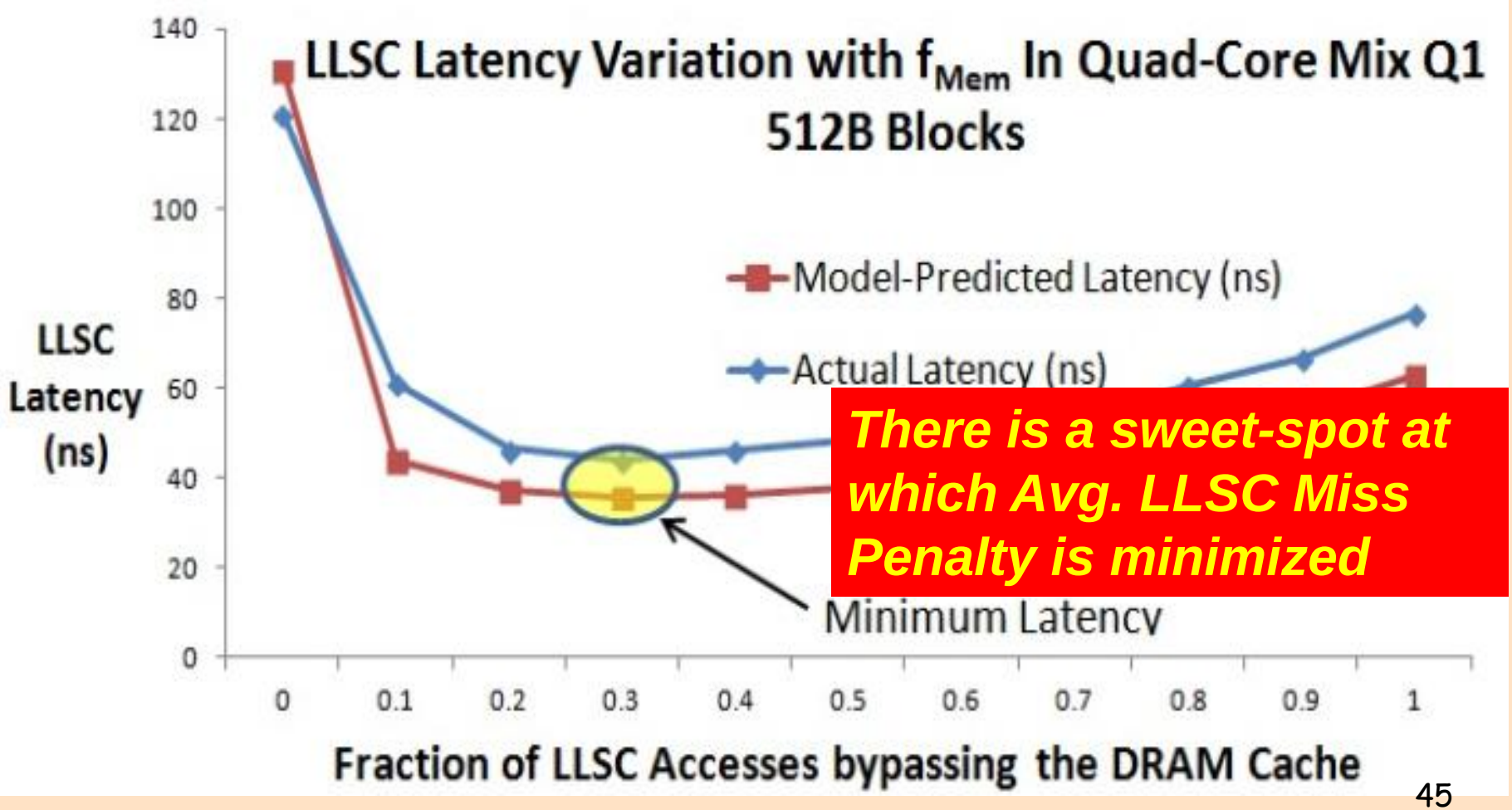
Requires High Predictor Hit Rate to beat Tags-on-SRAM Latency for Tag Lookup

Insight 2 - Motivation

- The DRAM Cache gets a very high cache hit rate.
- The Main Memory remains mostly idle!
- Cache is congested and memory is free!
- So we consider if bypassing some cache hits to main memory would get an overall latency benefit …
- We extend *ANATOMY-Cache* model by accounting for a fraction of requests that bypass the cache (details in the paper).

Insight 2: Cache Bypass/Offload Helps!

Congested Workload: Misses Are Expensive!



Talk Outline

- Introduction to stacked DRAM Caches
- Background (An overview of *ANATOMY*)
- *ANATOMY-Cache*: Modeling Stacked DRAM Cache Organizations
- Evaluation
- Insights
- Conclusions

ANATOMY-Cache

- First Analytical Model of Stacked DRAM Caches
- Covers Both Tags-on-DRAM and Tags-on-SRAM organizations
- We investigated two insights with the help of the model

Thank You !!