

Multicore Computer Architecture - Storage and Interconnects

Lecture 7

Advanced Cache Optimization Techniques-I



NPTEL Online Course : Multi-Core Computer Architecture and Interconnects Dr. John Jose, IIT Guwahati

Dr. John Jose

Assistant Professor

**Department of Computer Science & Engineering
Indian Institute of Technology Guwahati, Assam.**

Accessing Cache Memory



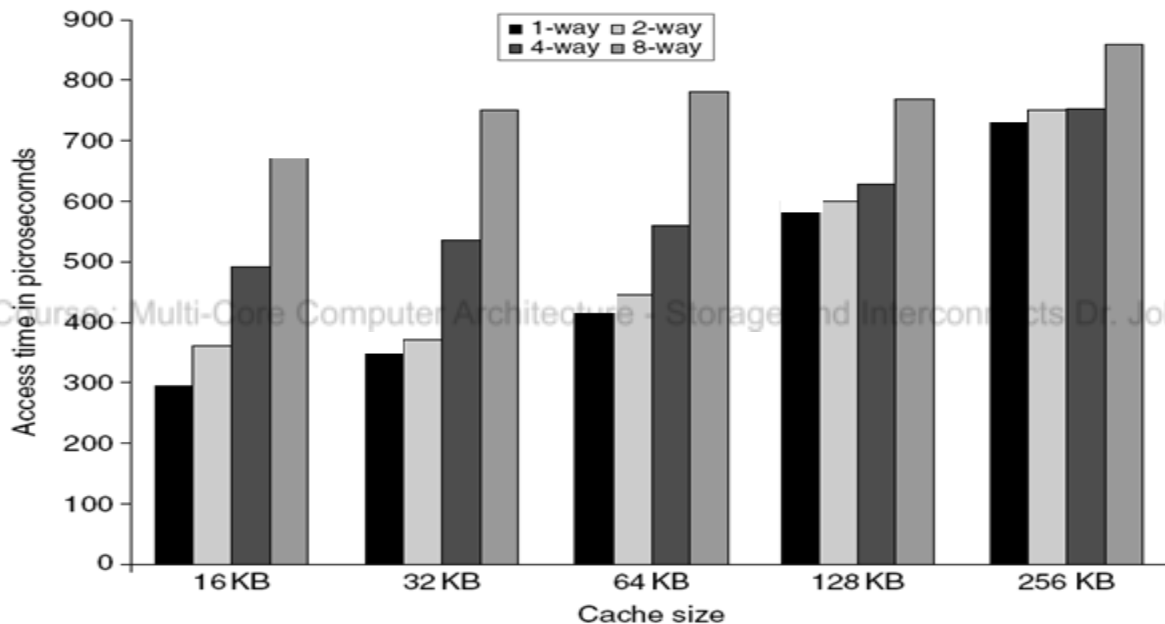
$$\text{Average memory access time} = \text{Hit time} + (\text{Miss rate} \times \text{Miss penalty})$$

- ❖ **Hit Time:** Time to find the block in the cache and return it to processor [*indexing, tag comparison, transfer*].
- ❖ **Miss Rate:** Fraction of cache access that result in a miss.
- ❖ **Miss Penalty:** Number of additional cycles required upon encountering a miss to fetch a block from the next level of memory hierarchy.

Small and simple first level caches

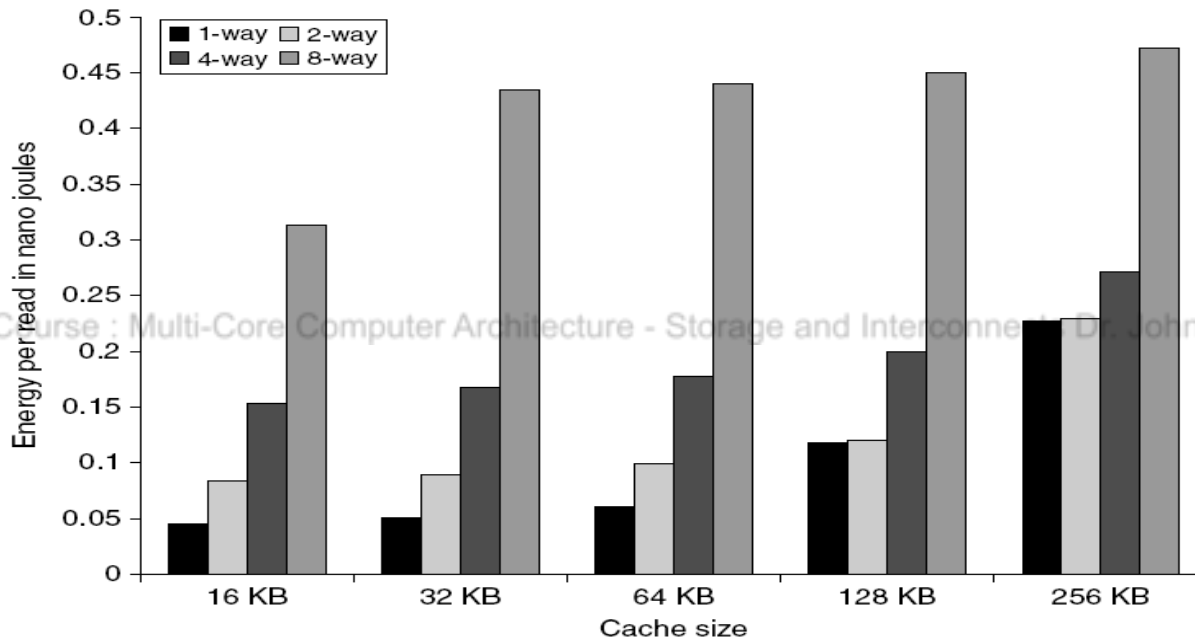
- ❖ **Small and simple first level caches to reduce hit time**
- ❖ Critical timing path:
 - ❖ addressing tag memory (indexing)
 - ❖ comparing tags (tag comparison)
 - ❖ selecting correct way (mux control selection)
- ❖ Direct-mapped caches can overlap tag compare and transmission of data
- ❖ Lower associativity reduces power because fewer cache lines are accessed

Hit Time vs L1 Size and Associativity



Access time vs. size and associativity

Energy / read vs L1 Size and Associativity

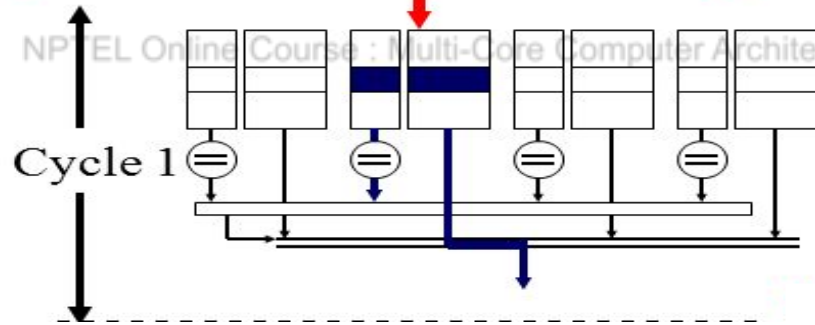


Energy per read vs. size and associativity

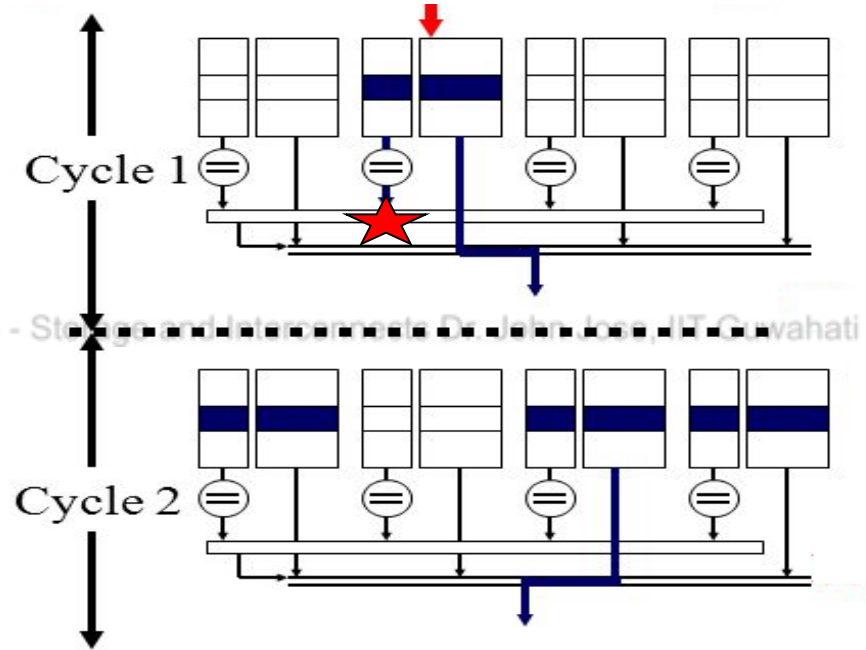
Way Prediction

- ❖ **Predict the way in a set to reduce hit time**
- ❖ To improve hit time, predict the way to pre-set multiplexer
- ❖ Extra bits are set to predict the block with in the set
- ❖ Remember the MRU way of a given set
- ❖ Using the prediction bits, power gating can be done on unused ways for reducing power.

Way Prediction



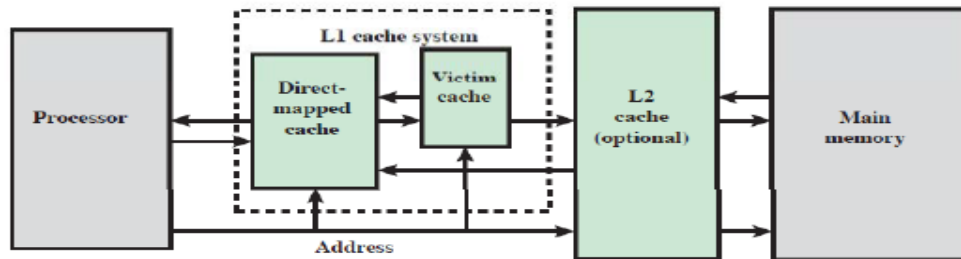
Way Prediction - Correct



Way Prediction Miss

Victim Caches

- ❖ **Introduce victim caches to reduce mis- penalty**
- ❖ Additional storage near L1 for MR evicted blocks
- ❖ Efficient for thrashing problem in L1 cache
- ❖ Look up Victim Caches before L2
- ❖ L1 and Victim caches are exclusive



Pipelining Cache

- ❖ **Pipelined cache for faster clock cycle time.**
- ❖ Split cache memory access into several sub-stages
- ❖ Ex: Indexing, Tag Read, Hit/Miss Check, Data Transfer
- ❖ Pipeline cache access to improve bandwidth
 - ❖ Examples: Pentium 4 – Core i7: 4 cycles
- ❖ But slow in hits.
- ❖ Increases branch mis-prediction penalty
- ❖ Makes it easier to increase associativity

Non-blocking Caches

- ❖ **Non blocking caches to increase cache bandwidth**
- ❖ Caches can serve hits under multiple cache miss in progress
 - ❖ *(a) Hit under miss (b) Hit under multiple miss*



- ❖ Must needed for OOO superscalar processor for IPC increase
- ❖ L2 must support it with L1-MSHR (Miss Status Holding Reg.)
- ❖ On an L1 miss allocate MSHR entry, clear upon L2 respond with cache block reply.

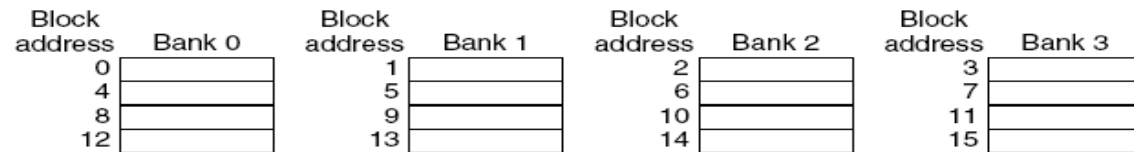
Non-blocking Caches

- ❖ **Non blocking caches to increase cache bandwidth**
- ❖ Processors can hide L1 miss penalty but not L2 miss penalty
- ❖ Reduces the effective miss penalty by overlapping miss latencies
- ❖ Significantly increases the complexity of the cache controller as there can be multiple outstanding memory accesses
- ❖ Requires pipelined or banked memory system

NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects Dr. John Jose, IIT Guwahati

Multi-banked Caches

- ❖ **Multi-banked caches to increase cache bandwidth**
- ❖ Rather having a single monolithic block, divide cache into independent banks that can support simultaneous accesses.
 - ❖ ARM Cortex-A8 supports 1-4 banks for L2
 - ❖ Intel i7 supports 4 banks for L1 and 8 banks for L2
- ❖ Sequential Interleaving



Four-way interleaved cache banks using block addressing. Assuming 64 bytes per blocks, each of these addresses would be multiplied by 64 to get byte addressing.

Early Restart

- ❖ **Early restart to reduce miss penalty**
- ❖ Do not wait for entire block to be loaded for restarting CPU
- ❖ **Early restart**
 - ❖ Request words in normal order
 - ❖ Send missed work to the processor as soon as it arrives
 - ❖ L2 controller is not involved in this technique

- **Early restart**



1 -> 2 -> 3 -> 4

Requested word: word 3



Processing performed
background

Critical Word First

- ❖ **Critical word first to reduce miss penalty**

- ❖ **Critical word first**

- ❖ Request missed word from memory first

- ❖ Send it to the processor as soon as it arrives

- ❖ Processor resume while rest of the block is filled in cache

- ❖ L2 cache controller forward words of a block out of order.

- ❖ L1 cache controller should re-arrange words in block

- **Critical word first**



3 -> 4 -> 1 -> 2

As soon as 3 is brought it is forwarded to CPU

In background 4, 1 and 2 is brought

**THANK
YOU!**

**FOR
YOUR
PRESENCE**

NPTEL Online Course : Multi-Core Computer Architecture - Storage and Interconnects Dr. John Jose, IIT Guwahati

johnjose@iitg.ac.in
<http://www.iitg.ac.in/johnjose/>