CS 222 Computer Organization & Architecture

Lecture 28 [08.04.2019]

Cache Optimization Techniques-II

John Jose

Assistant Professor

Department of Computer Science & Engineering

Indian Institute of Technology Guwahati, Assam.

How to optimize cache ?

- Reduce Average Memory Access Time
- **AMAT= Hit Time + Miss Rate x Miss Penalty**
- Motives
  - **Reducing the miss rate**
  - **Reducing the miss penalty**
  - **Reducing the hit time**

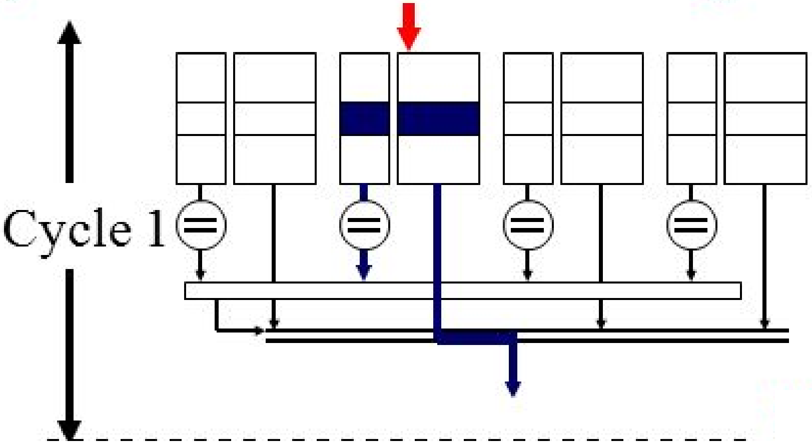
Prioritize read miss over writes

- **Prioritize read misses to reduce miss penalty**
- If a read miss has to evict a dirty memory block, the normal sequence is write the dirty block to memory and read the missed block
- This can be optimized by - copy the dirty block to a buffer, read from memory and then write the block - reduces CPU’s waiting time on read miss
- In write through caches, write buffers may hold the updated values of a block that encountered a read miss
- Either wait till write buffer is empty or search in write buffer before going to memory.

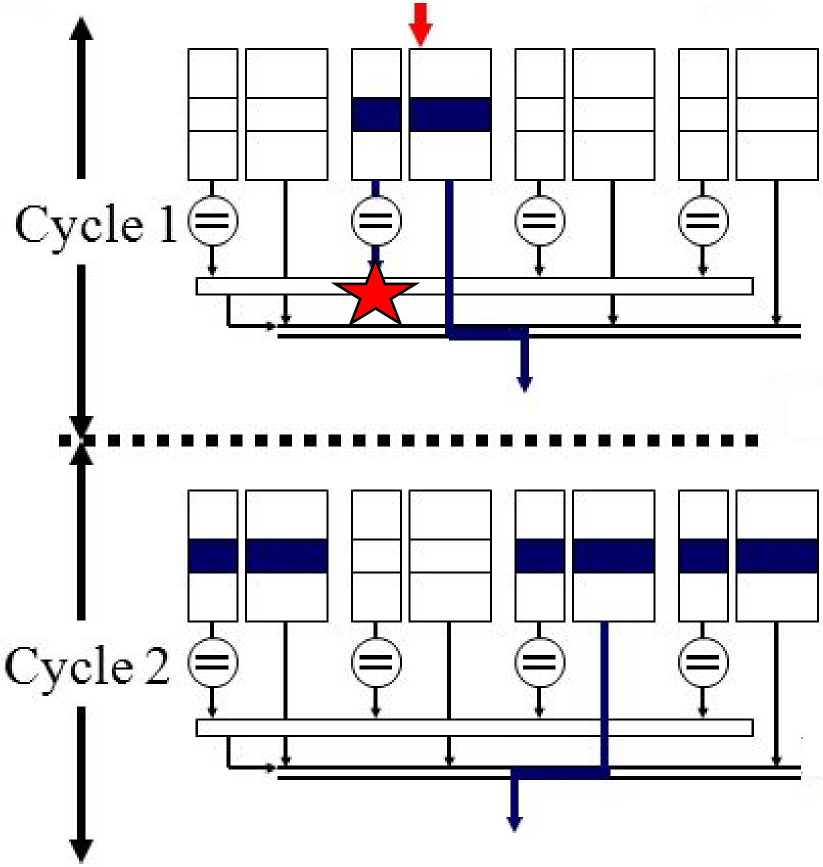
Way Prediction

- **Predict the way in a set to reduce hit time**
- To improve hit time, predict the way to pre-set the MUX.
- Extra bits are set to predict the block with in the set.
- Mis-prediction gives longer hit time (additional cycle)
- Performance depends on prediction accuracy.
  - I-cache has better accuracy than D-cache
- Using the prediction bits, power gating can be done on unused ways for reducing power.

Way Prediction



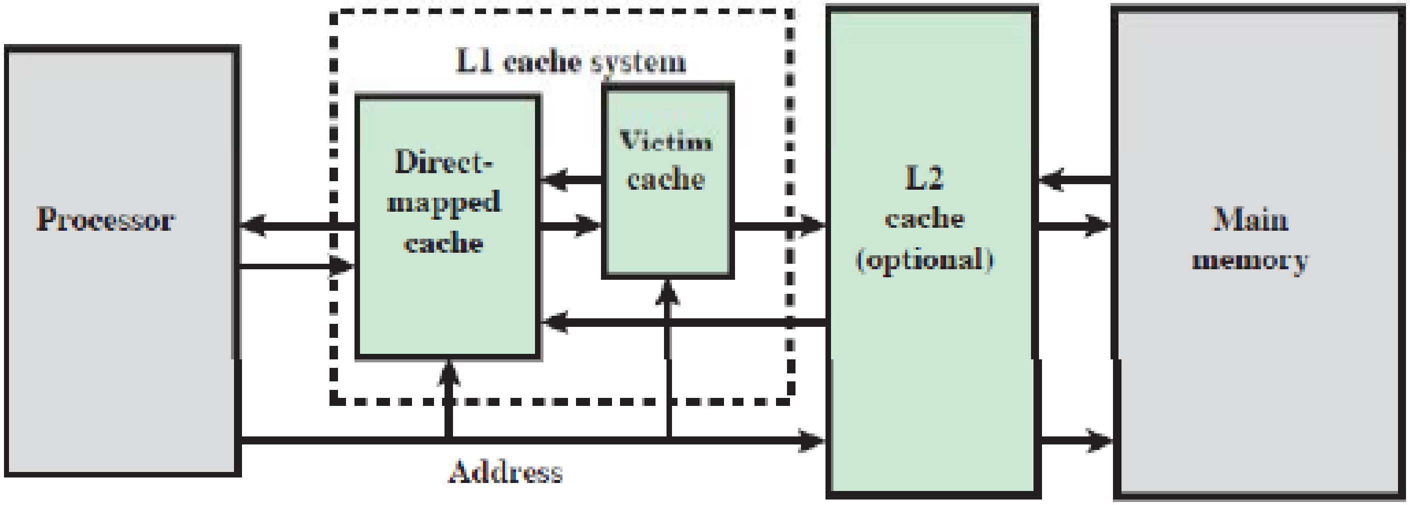
Way Prediction - Correct



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

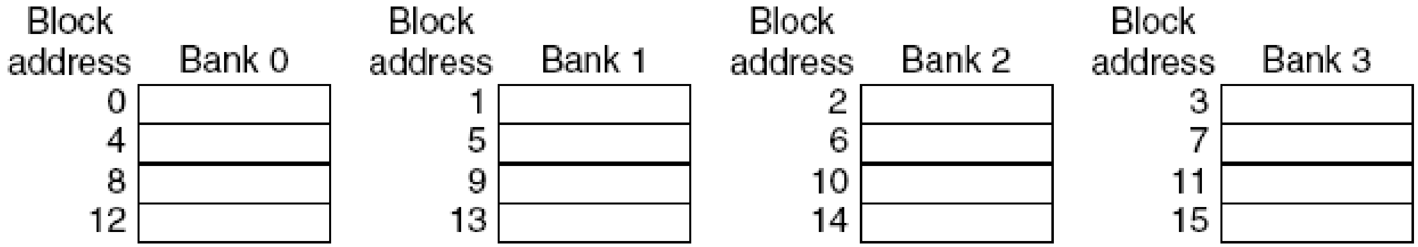


Pipelining Cache

- **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: 1 cycle
    - Pentium Pro – Pentium III: 2 cycles
    - Pentium 4 – Core i7: 4 cycles
- Increases branch mis-prediction penalty

Multi-banked Caches

- **Multi-banked caches to increase cache bandwidth**
- Rather a single monolithic unit, divide cache into many 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
- Interleave banks according to block address
- Sequential Interleaving

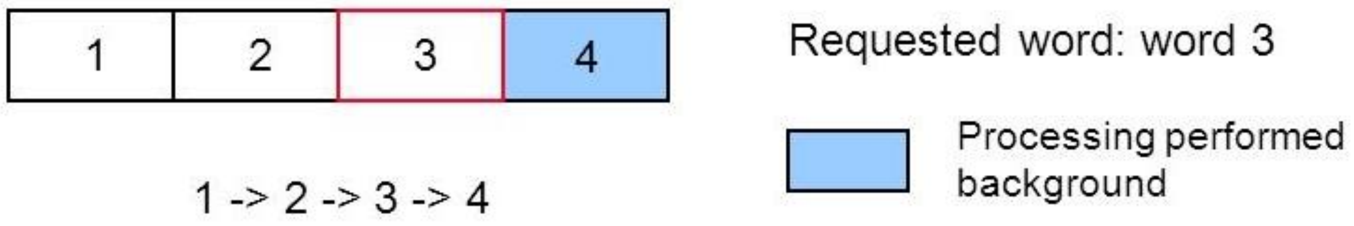


Block address	Bank 0	Block address	Bank 1	Block address	Bank 2	Block address	Bank 3
0		1		2		3	
4		5		6		7	
8		9		10		11	
12		13		14		15	

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**
- CPU do not wait for entire block to be loaded
- **Early restart**
  - Request words in normal order
  - Missed word to the processor as soon as it arrives
  - Generally useful in large blocks
  - L2 controller is not involved in this technique
    - **Early restart**



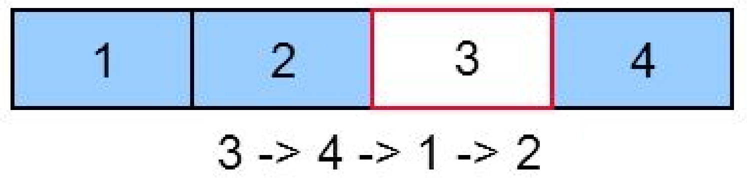
Requested word: word 3

🟦 Processing performed background

1 -> 2 -> 3 -> 4

Critical Word First

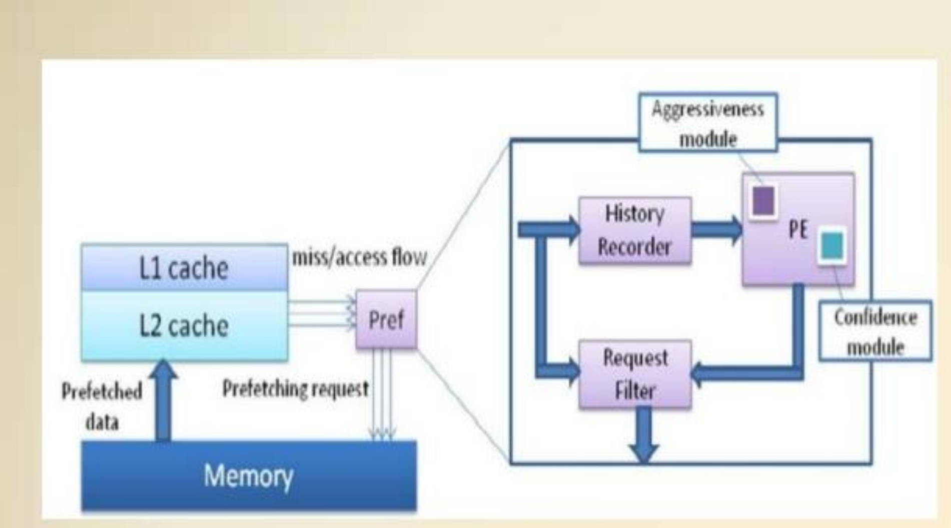
- ❖ Critical word first to reduce miss penalty
- ❖ **Critical word first**
  - ❖ Request missed word from memory first
  - ❖ Send it to the processor as soon as it arrives
  - ❖ Processor resume while rest of the block is filled in cache
  - ❖ L2 cache controller send words out of order.
  - ❖ L1 cache controller should re-arrange words in block
- **Critical word first**



As soon as 3 is brought it is forwarded to CPU
In background 4, 1 and 2 is brought

Hardware Prefetching

- **Pre-fetching to reduce miss rate and miss penalty.**
- Pre-fetch items before processor request them.
- Fetch more blocks on miss -include next sequential block
- Requested block is kept in I-cache and next in stream buffer.
- If a missed block is in stream buffer, cache miss is cancelled



johnjose@iitg.ac.in

http://www.iitg.ac.in/johnjose/