Lecture-18 (Cache Optimizations)
CS422-Spring 2018

Biswa@CSE-IITK

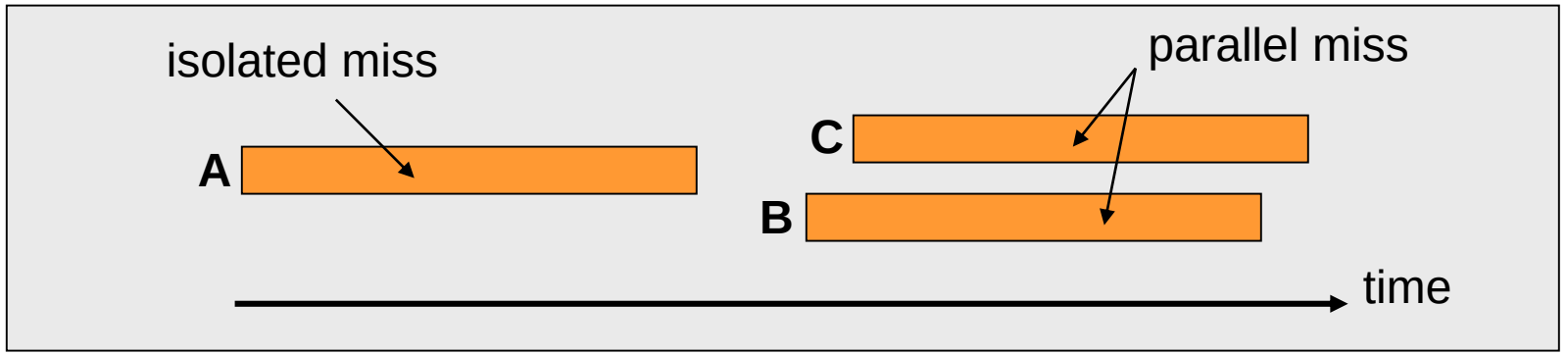
Compiler Optimizations

- Loop interchange
- Merging
- Loop fusion
- Blocking

Refer H&P: You need it for PA3 and PA4 too.

Non-blocking Cache

- Enable cache access when there is a pending miss
- Enable multiple misses in parallel
  - Memory-level parallelism (MLP)
    - generating and servicing multiple memory accesses in parallel
  - Why generate multiple misses?

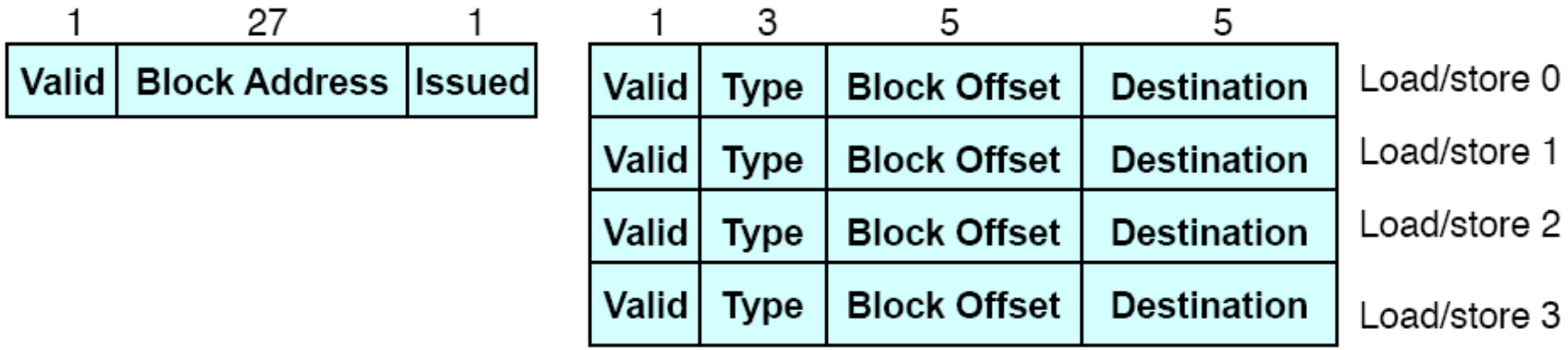


    - Enables latency tolerance: overlaps latency of different misses
  - How to generate multiple misses?
    - Out-of-order execution, multithreading, prefetching

Miss-Status Holding Registers

- Also called “miss buffer”
- Keeps track of
  - Outstanding cache misses
  - Pending load/store accesses that refer to the missing cache block
- Fields of a single MSHR
  - Valid bit
  - Cache block address (to match incoming accesses)
  - Control/status bits (prefetch, issued to memory, which subblocks have arrived, etc)
  - Data for each subblock
  - For each pending load/store
    - Valid, type, data size, byte in block, destination register or store buffer entry address

MSHRs


1
27
1
Valid
Block Address
Issued
1
3
5
5
Valid
Type
Block Offset
Destination
Load/store 0
Load/store 1
Load/store 2
Load/store 3

MSHR in Action

- On a cache miss:
  - Search MSHR for a pending access to the same block
    - Found: Allocate a load/store entry in the same MSHR entry
    - Not found: Allocate a new MSHR
    - No free entry: stall

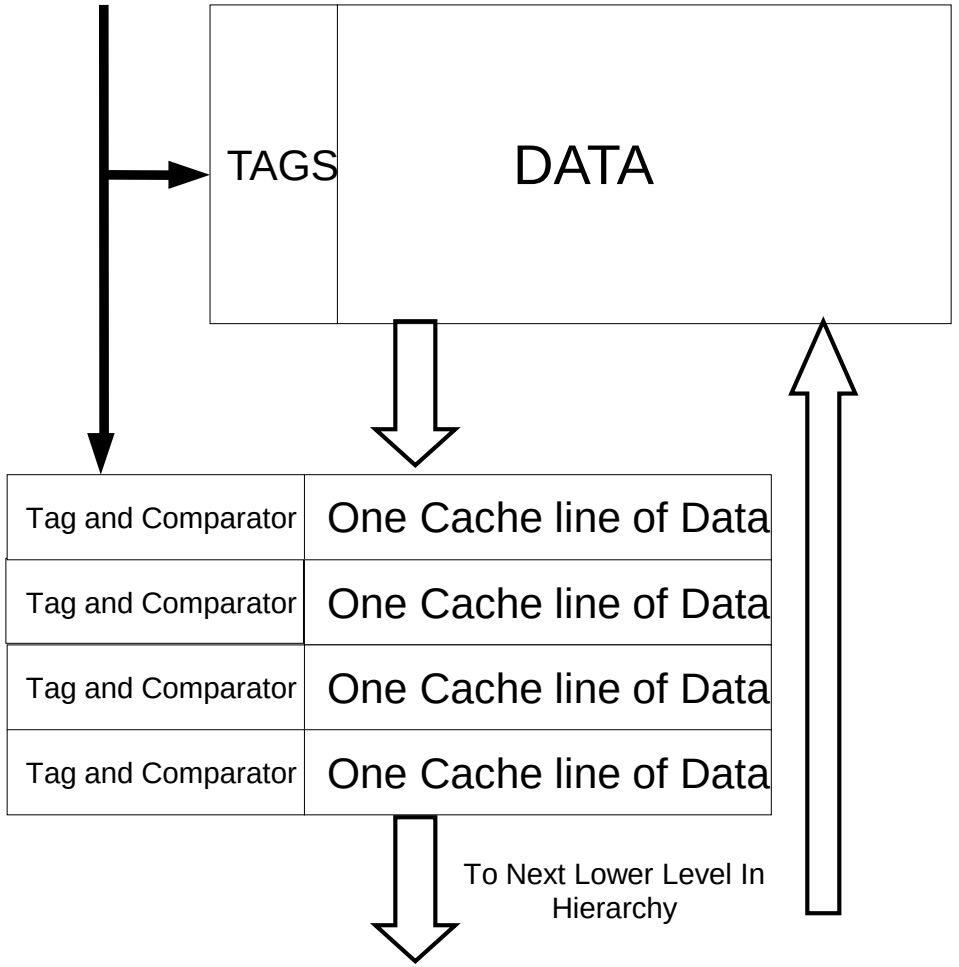
- When a subblock returns from the next level in memory
  - Check which loads/stores waiting for it
    - Forward data to the load/store unit
    - Deallocate load/store entry in the MSHR entry
  - Write subblock in cache or MSHR
  - If last subblock, dellaocate MSHR (after writing the block in cache)

Early Restart and CWF

- **Don't wait for full block to be loaded before restarting CPU**
  - ***Early restart*—As soon as the requested word of the block arrives, send it to the CPU and let the CPU continue execution**
  - ***Critical Word First*—Request the missed word first from memory and send it to the CPU as soon as it arrives; let the CPU continue execution while filling the rest of the words in the block. Also called *wrapped fetch* and *requested word first***
- **Generally useful only in large blocks,**
- **Spatial locality a problem; tend to want next sequential word, so not clear if benefit by early restart**

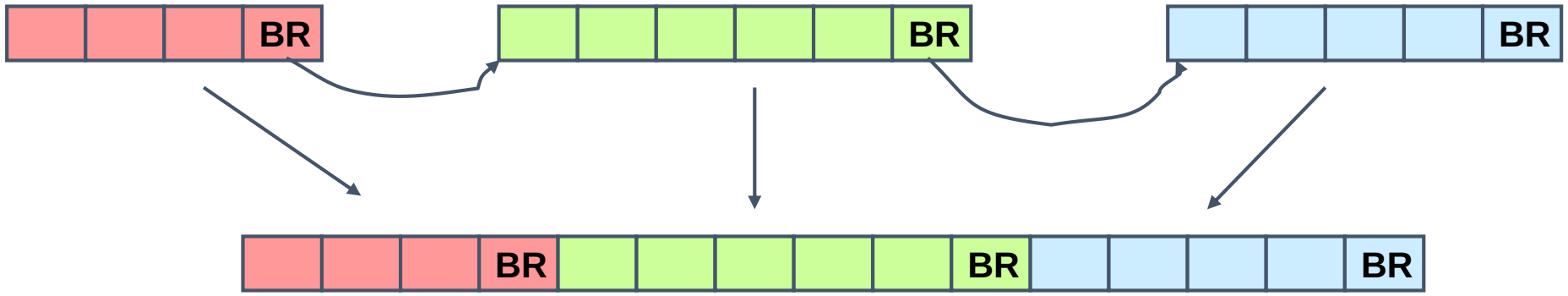
Victim Cache

- How to combine fast hit time of direct mapped yet still avoid conflict misses?
- Add buffer to place data discarded from cache
- Jouppi [1990]: 4-entry victim cache removed 20% to 95% of conflicts for a 4 KB direct mapped data cache
- Used in Alpha, HP machines



Trace Cache

Key Idea: Pack multiple non-contiguous basic blocks into one contiguous trace cache line



- **Single fetch brings in multiple basic blocks**
- **Trace cache indexed by start address *and* next *n* branch predictions**

Multi-Banked Cache

- **Rather than treat the cache as a single monolithic block, divide into independent banks that can support simultaneous accesses**
  - **E.g.,T1 ("Niagara") L2 has 4 banks**
- **Banking works best when accesses naturally spread themselves across banks ⇒ mapping of addresses to banks affects behavior of memory system**
- **Simple mapping that works well is "sequential interleaving"**
  - **Spread block addresses sequentially across banks**
  - **E,g, if there 4 banks, Bank 0 has all blocks whose address modulo 4 is 0; bank 1 has all blocks whose address modulo 4 is 1; …**

Way Prediction

- ***Way prediction*** helps select one block among those in a set, thus requiring only one tag comparison (if hit).
  - Preserves advantages of direct-mapping (why?);
  - In case of a miss, other block(s) are checked.
- ***Pseudoassociative*** (also called column associative) caches
  - Operate exactly as direct-mapping caches when hit, thus again preserving advantages of the direct-mapping;
  - In case of a miss, another block is checked (as if in set-associative caches), by simply inverting the most significant bit of the index field to find the other block in the "pseudoset".
  - real hit time < pseudo-hit time