COMPUTER ORGANIZATION (IS F242)

LECT 40: CACHE MEMORY

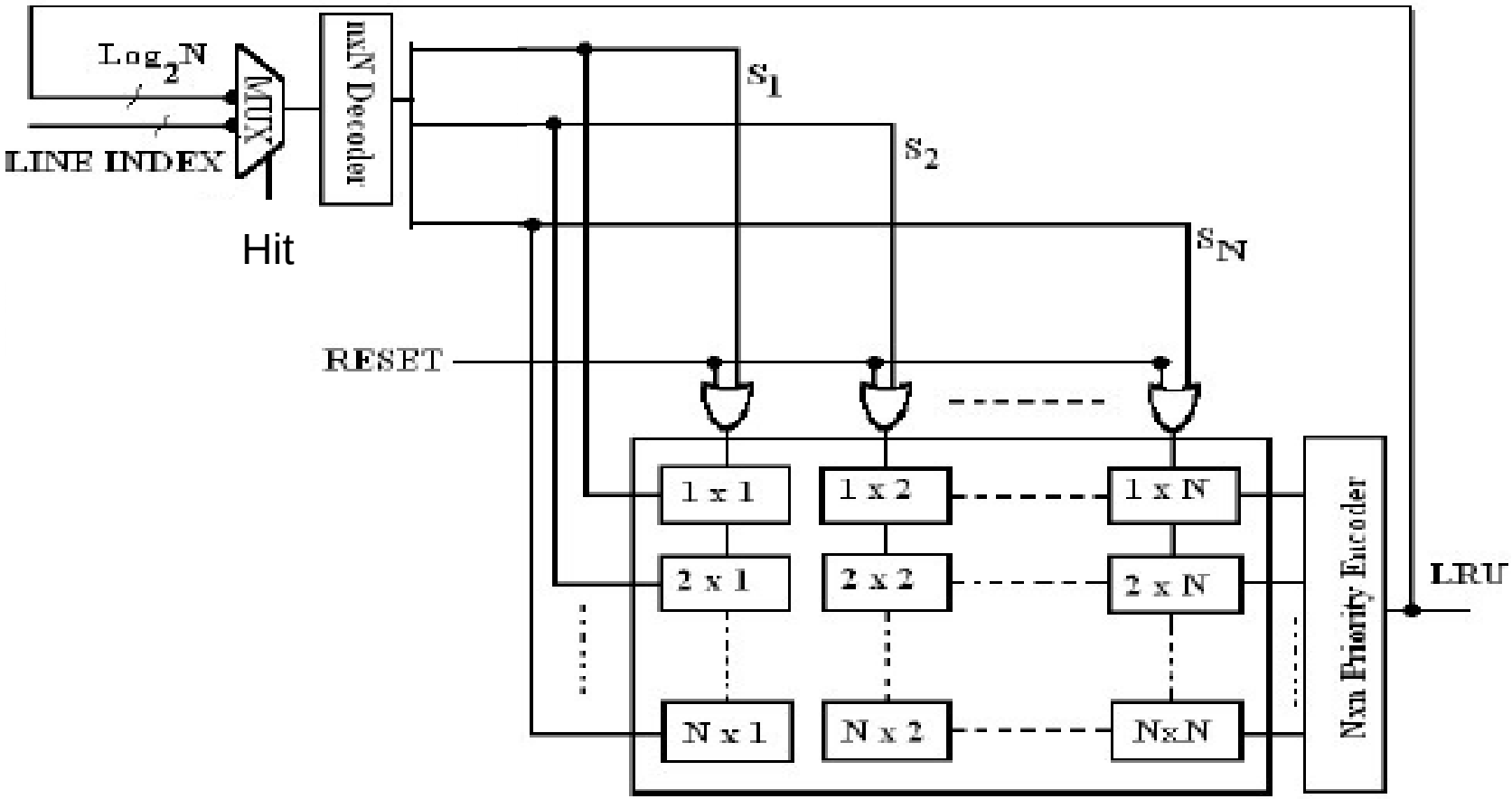
Cache Design

- Cache access
  - Look through, Look aside
- Block Placement
  - Direct, Fully Associative, Set-Associative
- Block Identification
  - TAG, INDEX, OFFSET
- Block Replacement
  - LRU, PLRU, LFU, OPT, FIFO, RANDOM
- Write Policies
  - Write Through, Write Back, Write Buffer
- Coherency
  - Snooping, MESI

Replacement Algorithms

- Square Matrix Implementation
  - N^2 bits per set to store the LRU information
  - The cache line corresponding to the row with all zeros is the victim cache line for replacement
  - If cache hit, all the bits in corresponding row is set to 1 and all the bits in corresponding column is set to 0.
  - If cache miss, priority encoder selects the cache line corresponding to the row with all zeros for replacement
  - Used when associativity is less
  - Figure shows the Square matrix implementation

Square Matrix Hardware Implementation

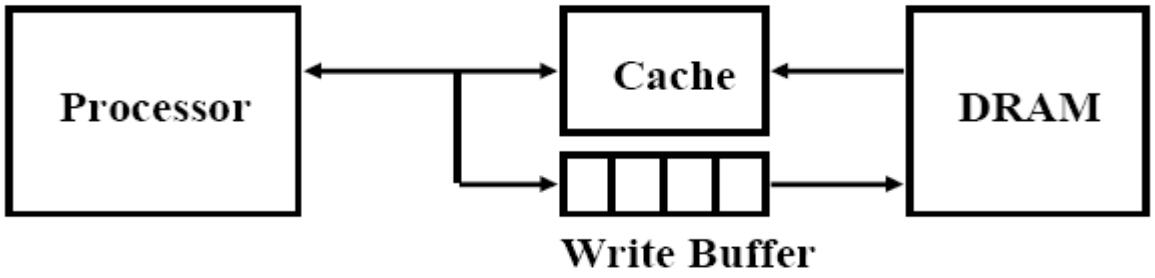


Write Policy

- Why is a need of Write Policy?
  - Must not overwrite a cache block unless main memory is up to date
  - Multiple CPUs may have individual caches
  - I/O may address main memory directly
- 15% memory references are writes
  - Write Through
    - Top hierarchy memory (L1 cache)
    - The information is written to both the block in the cache and to the block in the lower-level memory.
    - Lots of traffic and Slows down writes

Write Policy

- Pseudo Write Through (Write Buffer)
  - Processor writes data into the cache and the write buffer
  - Memory controller writes contents of the buffer to memory
  - FIFO (typical number of entries 4)
- Write Back
  - Lower hierarchy memory (main memory)
  - The information is written only to the block in the cache. The modified cache block is written to main memory only when it is replaced. Needs a dirty bit
  - Greatly reduces the memory bandwidth requirement
  - I/O must access main memory through cache



Write Miss

- Write allocate
  - The block is allocated on a cache miss, followed by write to that block (Write back or Write through)
  - Write misses act like read misses
  - Write Back is preferable

- Write no allocate ***OR*** No – Write Allocate
  - Write misses do not affect cache
  - Block is modified only in lower level memory
  - Write Through policy is only possible

Performance

- Configurations
  - Cache Size
  - Cache Line Size
  - Associativity
- Performance measures
  - Cache hit rate
  - Cache hit access time
  - Cache Miss penalty
  - Latency
  - Bandwidth
  - Energy Requirement

Improving Cache Performance

- Offset = $\log_2$ (cache line size)
- Index = $\log_2$ (Cache size/(cache line size * associativity))
- Tag = # of address lines – offset – index

- CPU exe.Time=(CPU clock cycles+Memory stall cycles)*Clock cycle time

- Memory stall cycles = # of Misses * Miss penalty
- Memory stall cycles = IC * (Misses/Instruction) * Miss penalty

- Misses/Instruction = Miss rate *(Memory accesses/ Instruction)

- Avg. memory Access Time = Hit time + Miss rate * Miss penalty

  $\mathbf{T_{access} = T_{Hit} + P_{Miss} * Penalty_{Miss}}$

- Avg. memory Access Time = Hit time of L1 + Miss rate of L1 *
  (Hit time of L2 + Miss rate of L2 * Miss Penalty of L2)