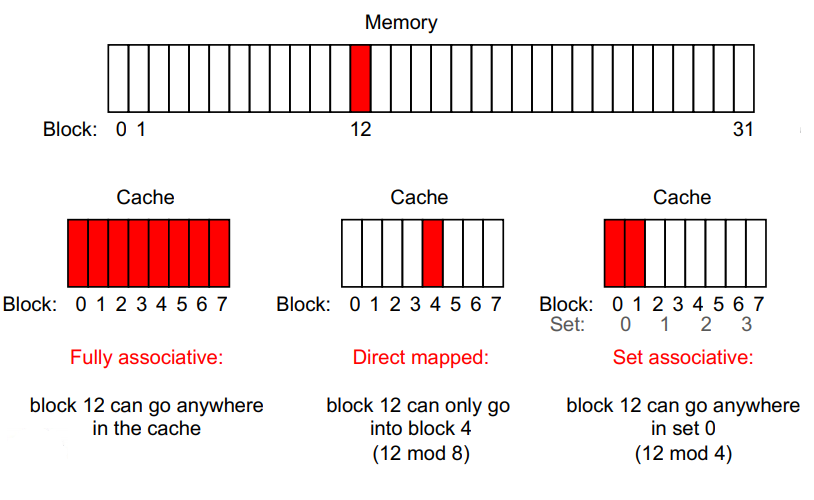
Computer Systems Lecture 17

Cache Associativity Options

* Fully Associative
  + The block can go into any location in the cache
  + Good: most flexible approach with the lowest miss rate
  + Bad: must search the whole cache to find the block causing speed and power usage to suffer
* Direct Mapped
  + The block can only go into one location in the cache
  + Good: very simple hardware it’s fast and low power
  + Bad: block mapping to the same location (thrashing) can occur, this ends up meaning it has increased miss rates
* Set Associative
  + Split the cache into groups (sets) of m blocks each ( m-way set associative)
  + A given block can only go into one set (based on block address), but within that set it can go anywhere
  + This is a good compromise between direct-mapped and fully associative caches
  + The typical degree of associativity is 2-16.



Example Problem

Given a 4KB, 4-way set-associative cache with 4-byte blocks and 32-bit addresses, how many tag, index and offset bits does the address decompose into?

If we have a 4 KB cache and use 4 bytes per block, then we have space for 1K blocks, but there are 4-ways per set giving 256 sets requiring 8-bits to select a set.

A 4 byte block will require a 2-bit offset (to index each byte)

This leaves the tag with 32-8-2 = 22 bits.

Writing to Caches: on a hit

* Write through – Write to both cache and memory
  + Good: memory and cache are always synchronized
  + Bad: writes are slow and require memory bandwidth
* Write back – write to cache only
  + Each cache block has a dirty bit, set if the block has been written to
  + When a dirty cache block is replaced, it is written to memory
  + Good: writes are fast and generate little memory traffic
  + Bad: memory can have stale data for some time

Writing to Caches: On a Miss

* Write allocate- Bring the block into the cache and write to it
  + Useful if locality exists
* Write no-allocate – Do not bring the block into the cache; modify data only in memory
  + Useful if no locality
  + Guarantees that cache and memory are synchronized.