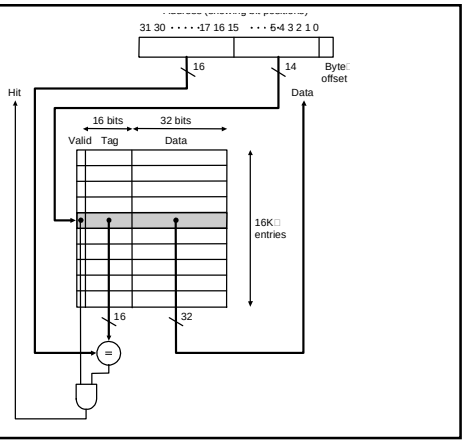
Caches II

Chapter 7

Example Cache – DecStation 3100

- Released 1989
- Used a MIPS R2000
- separate Instruction and Data caches
  - single word block size
  - each cache 16K words



DecStation 3100 cache

- Steps for Read:
  - send address to appropriate cache
  - If a hit then return value on data-lines
  - If a miss then get value from main memory then put in cache – re-perform original read
- For writes have to ensure cache and main memory remain consistent:
  - simplest approach is to write data to both main memory and cache

DecStation 3100 cache

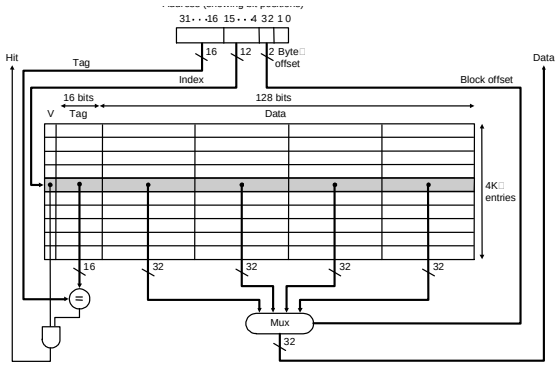
- Writes on the 3100
  - Index cache using bits 15-2 of the address
  - Write bits 31-16 of address to the tag field, write data & update valid field
  - Also write word to main memory
- CPI on 3100 is about 1.2 without cache misses
- adding 10 cycles for every write operation changes this to:
- 1.2 + 10*13% = 2.5

Write Buffer

- One solution is to use a write buffer (e.g. 3100 caches can buffer up to 4 words)
- Assumes that writes occur at a slower rate than the values can be written to memory
- Can overflows still occur?
- What should happen if an overflow is going to occur?

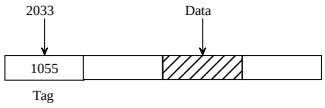
Taking advantage of Spatial Locality

- Need a block size bigger than a word
- When a miss (read or write) occurs then need to fetch the whole block



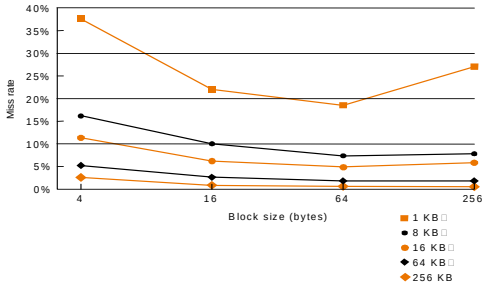
Multiword block size

- Reads are dealt with as before
- With writes not possible to just write tag and data

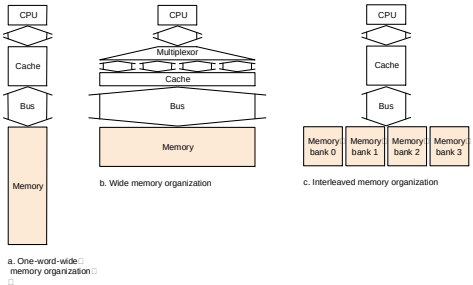


- For a write miss then have to fetch replacement block from memory and re-perform write

Miss rate verses Block Size

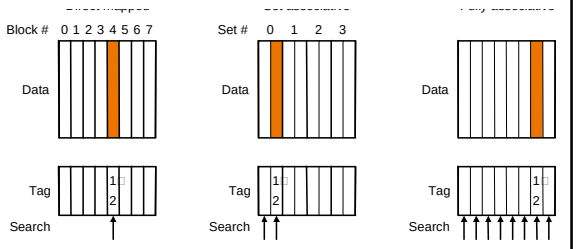


Designing Memory to Support Caches

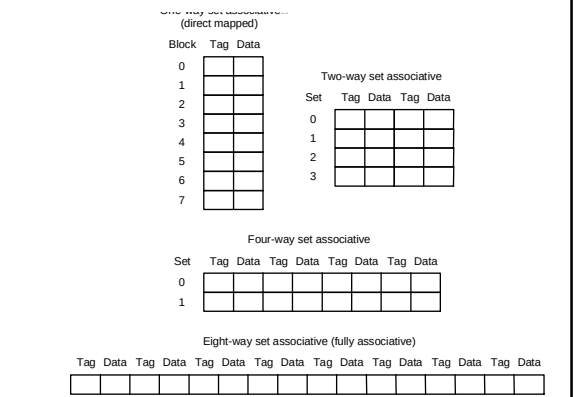


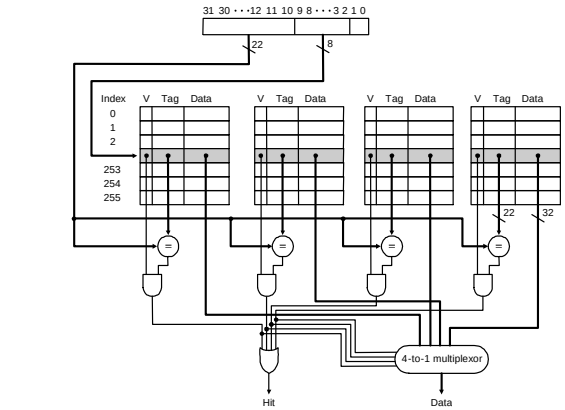
Reducing Cache misses by more flexible placement of blocks

- Fully associative caches allow blocks to be placed anywhere in the cache
  - Have to search every tag field for every memory access
- Set associative cache allows blocks to be placed in a fixed number of locations in the cache
  - an *n-way set associative cache* allows a block to be placed in one of *n* locations in the cache



- All caches can be considered as being set associative
- Increasing associatively tends to decrease the miss rate





Summary

- Store more than one word per block for spatial locality
  - Adds complexity to write-miss
- Provide more than one location to store a block in a cache to reduce miss rate
  - Associative caches
  - Need mechanism to determine which block to replace
  - Rate of return of additional associativity reduces fairly quickly compared with additional complexity in implementation