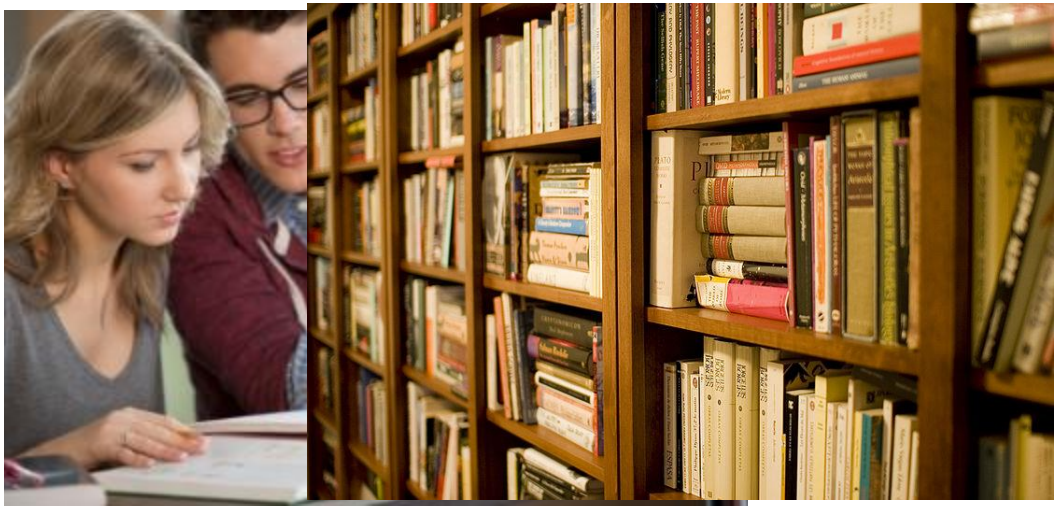
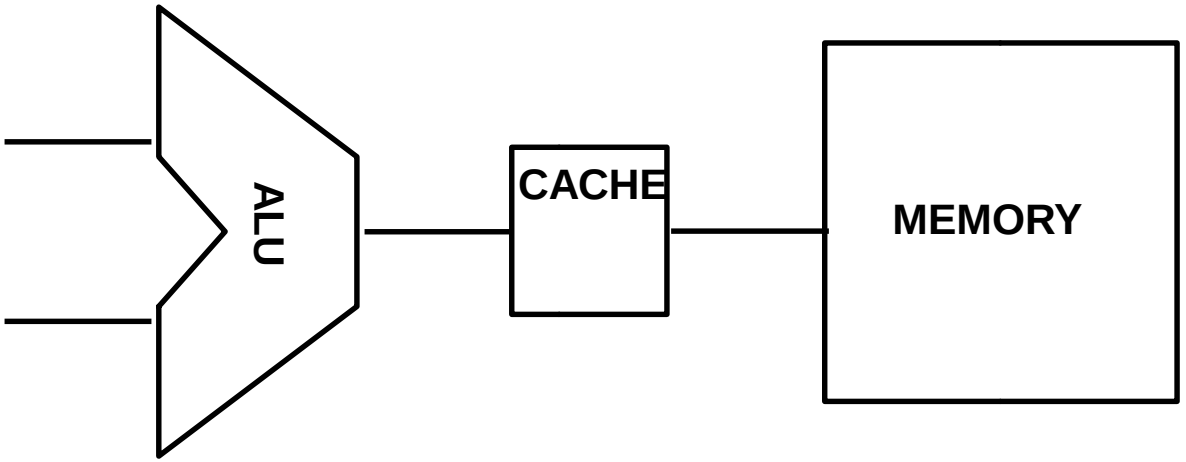
CHAPTER 7-3

Memory

By Pattama Longani
Collage of arts, media and Technology

Books in a library





http://www.gettyimages.com/detail/photo/college-students-studying-at-table-royalty-free-image/485207369
http://commons.wikimedia.org/wiki/File:Bookshelf.jpg

Reduce Cache Miss Scheme

- **direct mapped**
  - A block can go in exactly one place in the cache = mapping from any block address in memory to a single location in the upper level of the hierarchy
- **fully associative**
  - a block can be placed in any location in the cache.
  - To find a given block in a fully associative cache, all the entries in the cache must be searched

Reduce Cache Miss Scheme

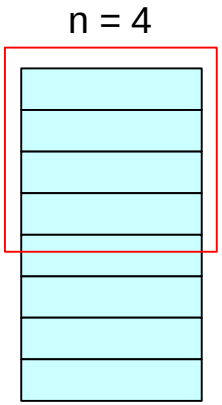
- To make the search practical, it is done in parallel with a ***comparator*** (significantly increase the hardware cost) associated with each cache entry.
- effectively only for caches with small numbers of blocks.

- *fully associative* and *direct mapped* are extreme schemes

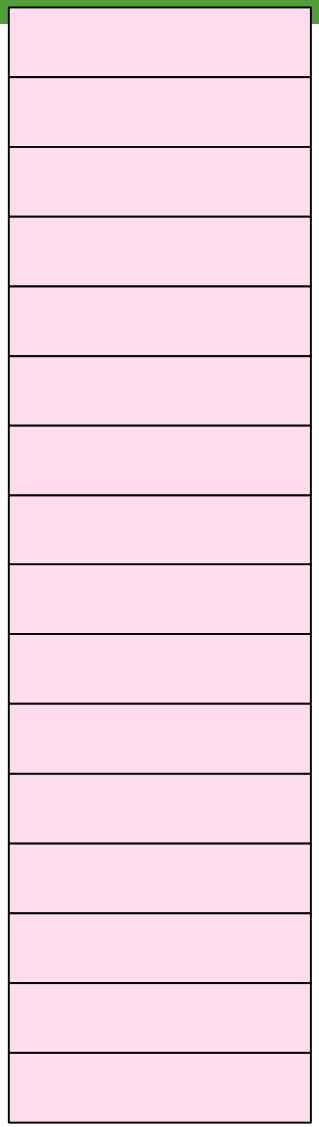
Reduce Cache Miss Scheme

- **set associative**
  - there are a fixed number of locations where each block can be placed.
  - A set-associative cache with n locations for a block is called an n-way set-associative cache.
  - Each block in the memory maps to a unique set in the cache given by the index field, and a block can be placed in any element of that set.
  - combines direct-mapped placement and fully associative placement

Set Associative



Set in cache



Block of Memory

Example

The location of a memory block whose address is 12 in a cache with eight blocks varies for direct mapped, two way set-associative, and fully associative placement.

One-way set associative
(direct mapped)

Block	Tag	Data
0		
1		
2		
3		
4		
5		
6		
7		

Two-way set associative

Set	Tag	Data	Tag	Data
0				
1				
2				
3				

Four-way set associative

Set	Tag	Data	Tag	Data	Tag	Data	Tag	Data
0								
1								

Eight-way set associative (fully associative)

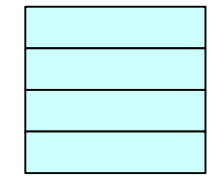
Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data

Cache scheme/Search

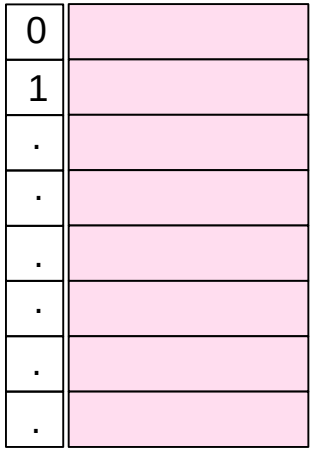
- direct mapped
  - (Block number) modulo (Number of blocks in the cache)
  - Specific location for search
- set-associative
  - (Block number) modulo (Number of sets in the cache)
  - All elements of in the set of cache location must be search
- fully associative
  - All the block in the cache must be search
- Increasing the degree of associativity -> decreases the miss rate (advantage), but increase in the hit time (disadvantage)

Example

Assume there are three small caches, each consisting of four one word blocks. One cache is fully associative, a second is twoway set-associative, and the third is directmapped. Find the number of misses for each cache organiza tion given the following sequence of block addresses: 0, 8, 0, 6, and 8.



Set in cache



Block of Memory

direct mapped

Block address	Cache block
0	(0 modulo 4) = 0
6	(6 modulo 4) = 2
8	(8 modulo 4) = 0

Address of memory block accessed	Hit or miss	Contents of cache blocks after reference			
		0	1	2	3
0	miss	Memory[0]			
8	miss	Memory[8]			
0	miss	Memory[0]			
6	miss	Memory[0]		Memory[6]	
8	miss	Memory[8]		Memory[6]	

5 misses

set-associative

Block address	Cache set
0	(0 modulo 2) = 0
6	(6 modulo 2) = 0
8	(8 modulo 2) = 0

Address of memory block accessed	Hit or miss	Contents of cache blocks after reference			
		Set 0	Set 0	Set 1	Set 1
0	miss	Memory[0]			
8	miss	Memory[0]	Memory[8]		
0	hit	Memory[0]	Memory[8]		
6	miss	Memory[0]	Memory[6]		
8	miss	Memory[8]	Memory[6]		

4 misses

fully associative

Address of memory block accessed	Hit or miss	Contents of cache blocks after reference			
		Block 0	Block 1	Block 2	Block 3
0	miss	Memory[0]			
8	miss	Memory[0]	Memory[8]		
0	hit	Memory[0]	Memory[8]		
6	miss	Memory[0]	Memory[8]	Memory[6]	
8	hit	Memory[0]	Memory[8]	Memory[6]	

3 misses

- if we had eight blocks in the cache, there would be no replacements in the two way set associative cache
- if we had 16 blocks, all 3 caches would have the same number of misses.
- cache size and associativity are dependent in determining cache performance.

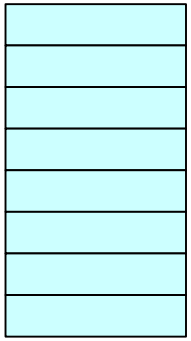
Locating a Block in the Cache

- direct mapped cache
  - only a single comparator is needed
- set-associative cache
  - **tag** : checked if it matches the block address from
  - **index value** : used to select the cache set that contain the address of the memory location that the processor request
  - all the tags in the selected set are searched in ***parallel***.

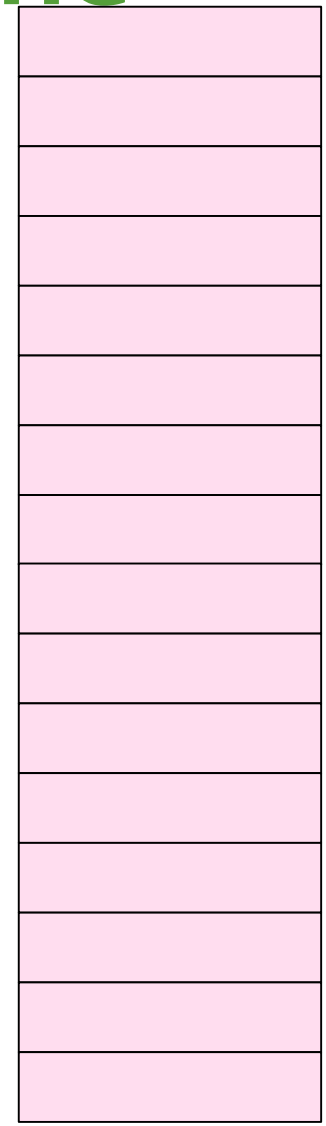
Locating a Block in the Cache

- If the total cache size is kept the same,
  - increasing the associativity (cache) increases the number of **blocks** (memory) per **set** (cache) & the data in set are simultaneous compares to perform the search in parallel
  - If the number of block (memory) per set (cache) increase by a factor of 2, it will decrease the number of set by 2.
    - decreases the size of the index by 1 bit
    - increases the size of the tag by 1 bit

Locating a Block in the Cache



Set in cache



Block of Memory

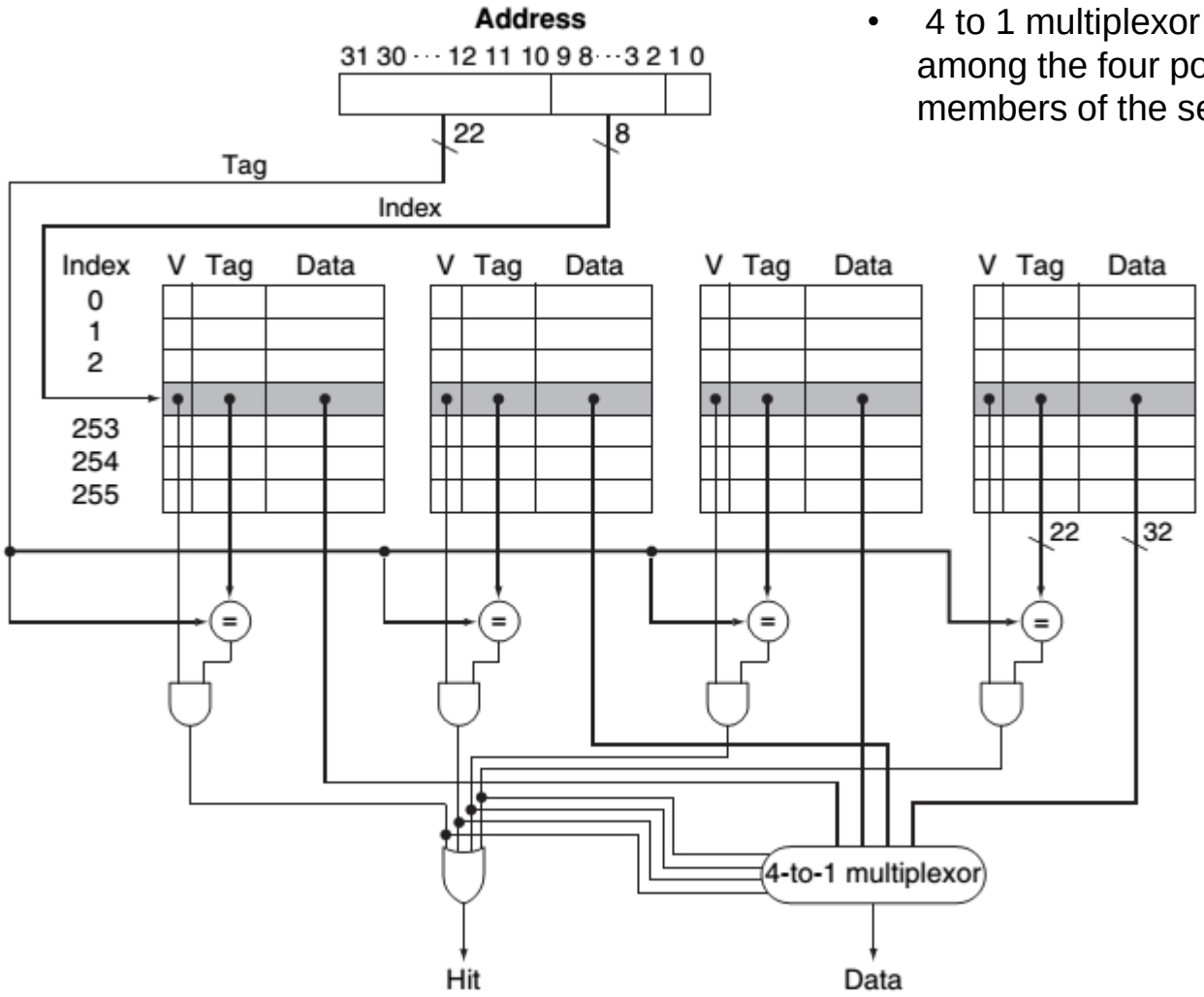
If n = 2 (set of 2 cache block),
there are 4 index field to tell which set of cache that a memory location will be match to
,and because the memory will be divided into 4 group there are 4 tag to indexed the memory location

If n = 4 (set of 4 cache block),
there are 2 index field to tell which set of cache that a memory location will be match to
,and because the memory will be divided into 2 group there are 8 tag to indexed the memory location

Locating a Block in the Cache

Four way set associative cache

- four comparators are needed (tag)
- 4 to 1 multiplexor to choose (index) among the four potential members of the selected set.



Locating a Block in the Cache

- The cache access consists of indexing the appropriate set and then searching the tags of the set
- The costs of an associative cache are the extra comparators and any delay imposed by having to do the compare and select from among the elements of the set.
- The choice among direct mapped, set-associative, or fully associative mapping in any memory hierarchy will depend on the cost of a miss versus the cost of implementing associativity, both in time and in extra hardware

Choosing Which Block to Replace

- direct mapped
  - the requested block can go in exactly one position, and the block occupying that position must be replaced.
- set-associative
  - we have a choice of where to place the requested block, and hence a choice of which block to replace.
- fully associative
  - all blocks are candidates for replacement.

Choosing Which Block to Replace

- The most commonly used scheme is **least recently used (LRU)**
  - the block replaced is the one that has been *unused* for the *longest time*.
  - It keeping track of when each element in a set was used relative to the other elements in the set.