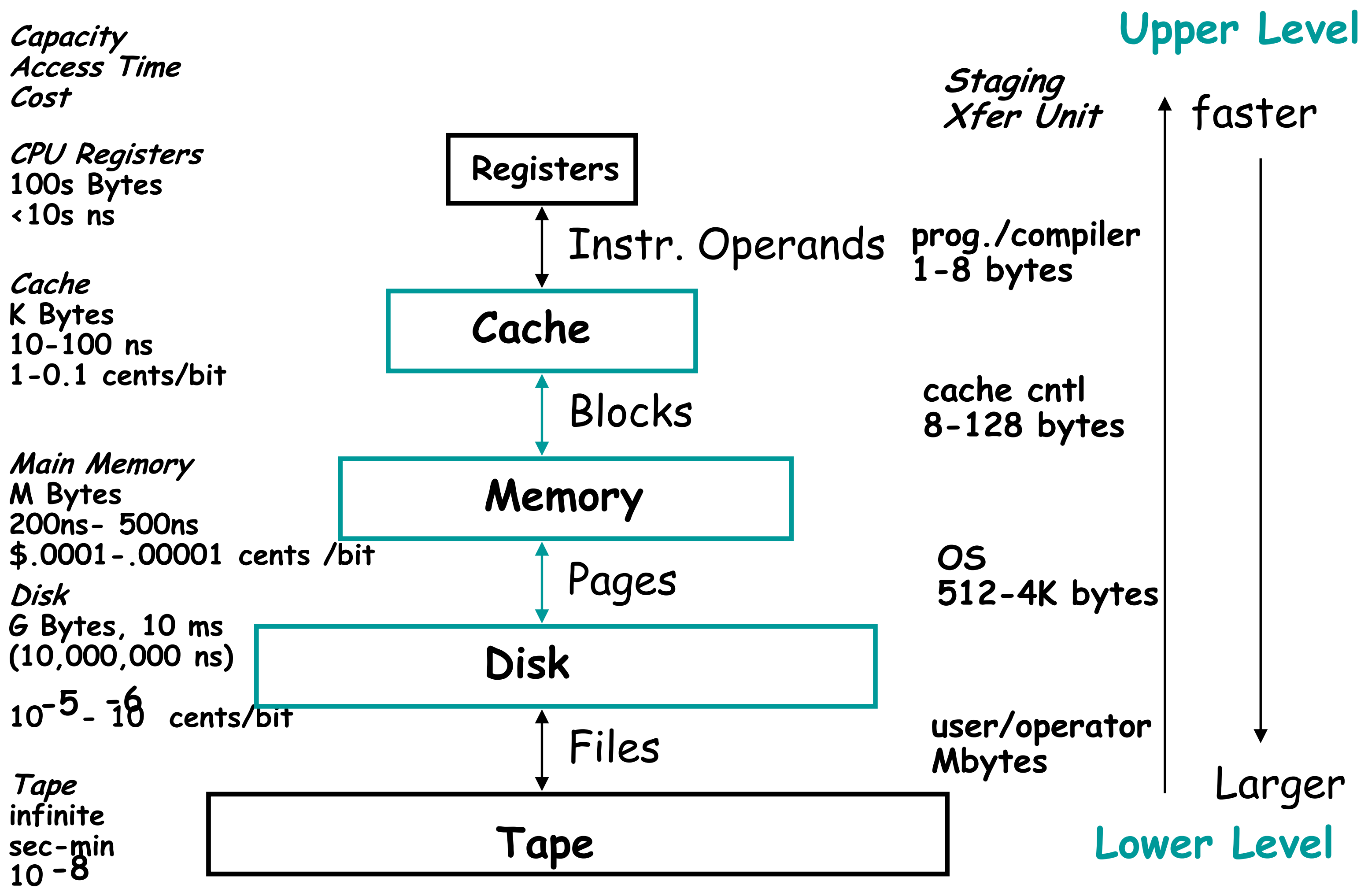
CMPE 180-90

Lecture - Caches

Haluk Katircioglu
Computer Engineering Department
SJSU

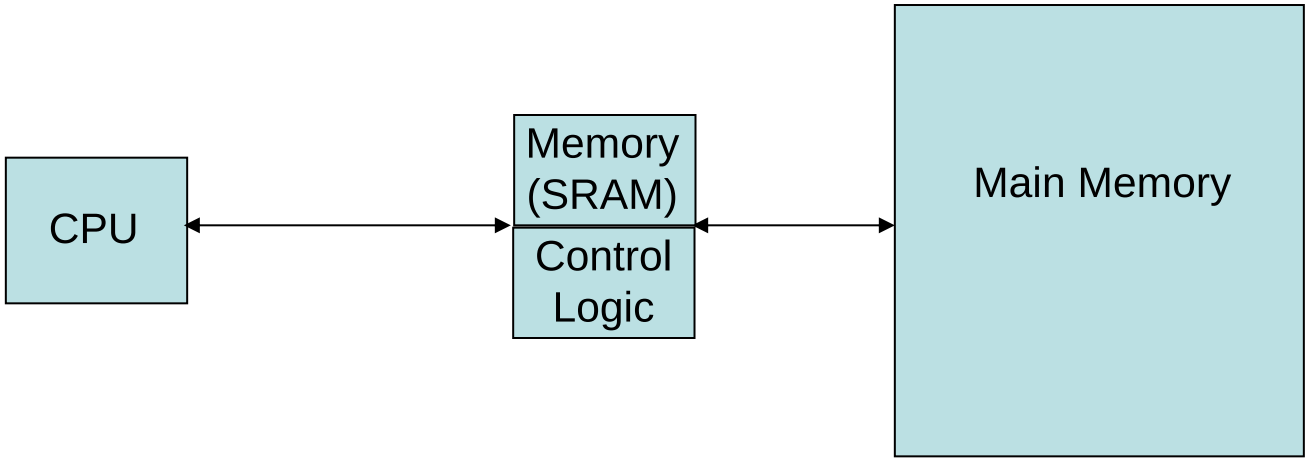
Levels of the Memory Hierarchy



from David Patterson

What is a Cache?

CACHE: Small, fast memory and logic which improves average memory response time



System performance benefits:

- Read Performance: Read Policy, Cache Size, Cache Tags, Associativity, Line Size, cache update policy
- Write Performance: Write policy (write-through, Buffered (posted) write-through, Write back)
- Bus Utilization: Single vs dual memory bus architecture, Line fill techniques, line buffer caching, Bursting of data, data bus width

The Principle of Locality – why does a cache work?

- The Principle of Locality:
  - Program accesses a relatively small portion of the address space at any instant of time.
- Two Different Types of Locality:
  - Temporal Locality (Locality in Time): If an item is referenced, it will tend to be referenced again soon (e.g., loops, reuse)
  - Spatial Locality (Locality in Space): If an item is referenced, items whose addresses are close by tend to be referenced soon (e.g., straightline code, array access)
- Last 15 years, HW relied on locality for speed

It is a property of programs which is exploited in machine design.

from David Patterson

The Principle of Locality cont'd

- By taking advantage of the principle of locality:
  - Present the user with as much memory as is available in the cheapest technology
  - Provide access at the speed offered by the fastest technology
- DRAM is slow, but cheap and dense:
  - Good choice for presenting the user with a big memory system
- SRAM is fast, but expensive and not very dense
  - Good choice for providing the user fast access time

Cache Architecture

- Cache architecture means
  - Very fast local memory
  - Cache reduces average memory access time
- Levels of caching
  - First level cache
  - Second level cache
- Performance
  - Hit rate
  - Read policy
  - Write policy
  - Cache tag and tag array sizes
  - Associativity
  - Line size
  - Cache update policy

$$\text{Hit Rate } \% = \frac{\text{Cache Hits}}{\text{Total Memory Requests}} \times 100\ \%$$

What is "HIT"?
What is "MISS"?

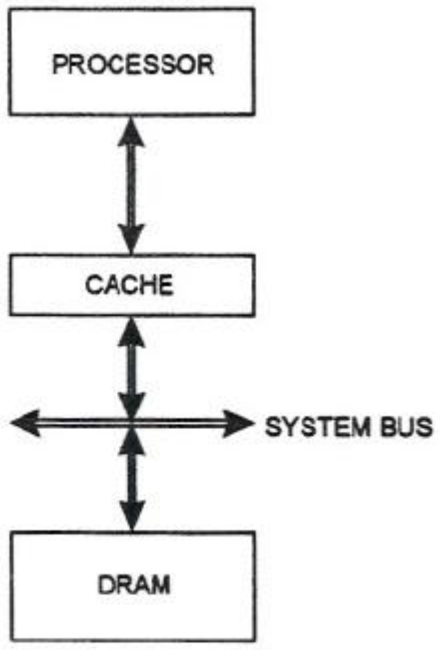
Cache Architecture cont'd

- Bus Utilization
  - Bus concurrency
  - Line fill techniques
  - Line Buffer Caching
  - Bursting
  - Wide data busses

Cache Architecture – Read Performance

Read Policy

- Look-Through (serial) Cache – Lookup Penalty
- Look-aside (parallel) cache – high bus utilization



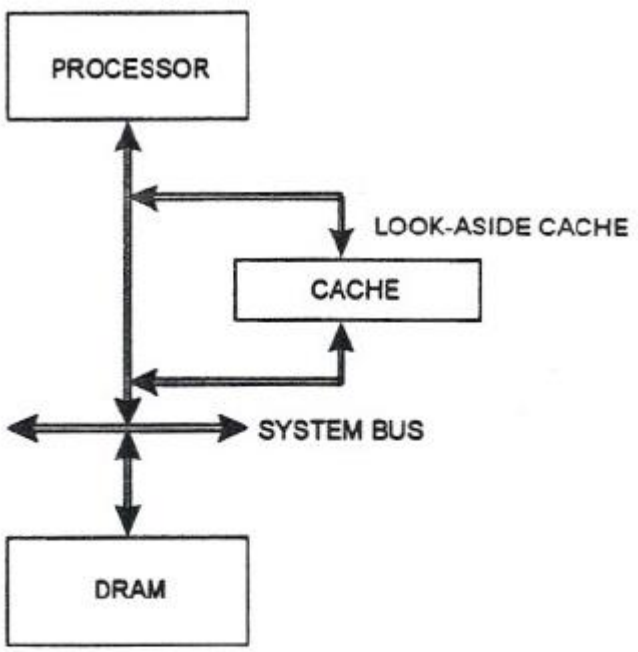
LOOK-THROUGH CACHE SYSTEM

Look Through Cache:
+ reduces the number of memory requests on the system bus
- memory requests are delayed to main memory
- two port cache design

Cache Architecture – Read Performance cont;d

Read Policy cont'd

- Look-Through (serial) Cache – Lookup Penalty
- Look-aside (parallel) cache – high bus utilization



LOOK-ASIDE CACHE SYSTEM

Look Aside Cache:
+ optionality: can be removed without the need to redesign the system
+ faster main memory response: main memory lookup starts immediately (no cache lookup delay)
+ simpler as it needs to support only one port
- memory always look up (cache miss or hit)
- high system bus utilization
- bus concurrency reduced (bus busy if another processor accessing memory)

Cache Architecture – Read Performance cont;d

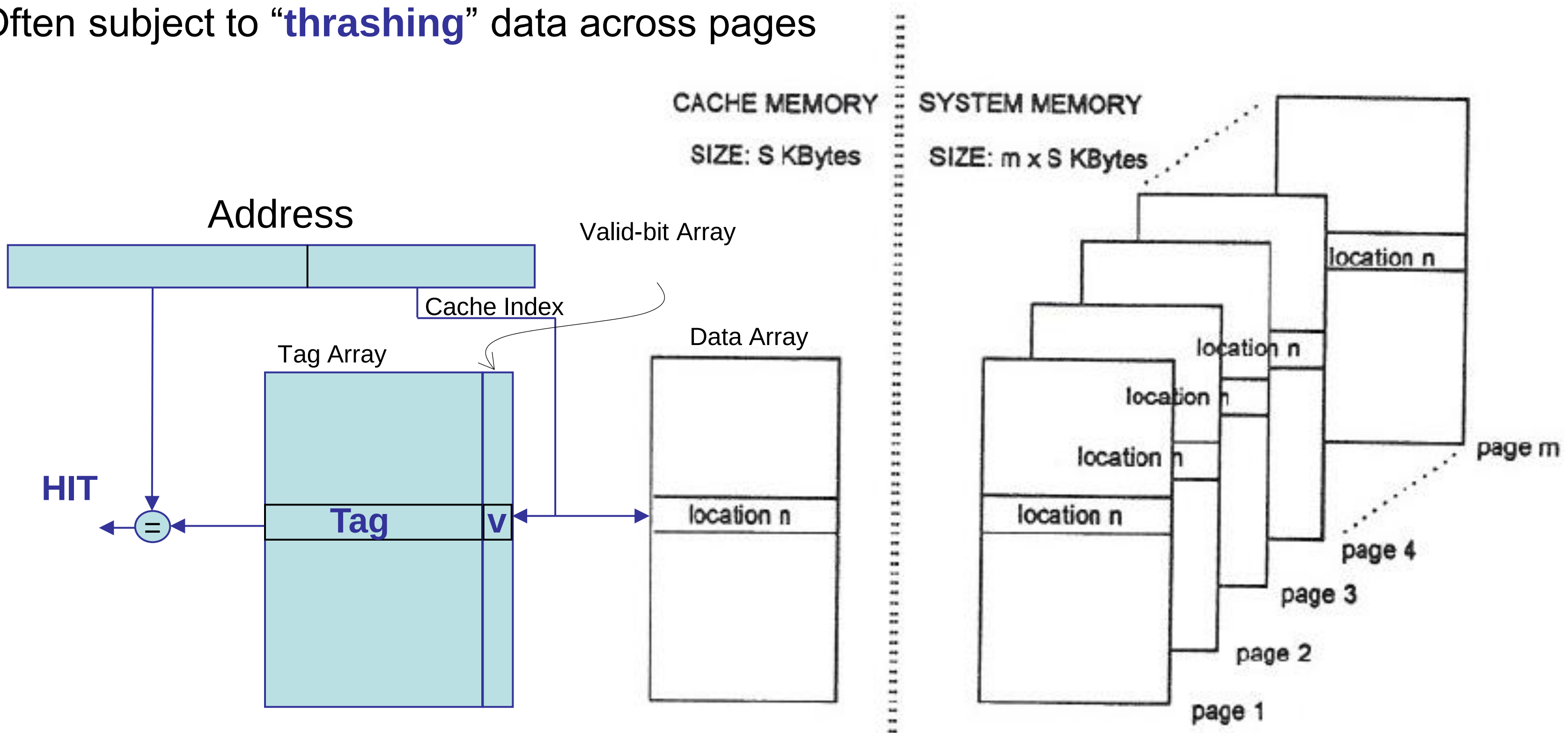
Cache size, tags, associativity, line size, update policy

- Cache Organizations
  - Direct mapped cache
  - Set associative caches
  - Fully associative caches

Caches

Cache Organizations

- Direct Mapped Cache Organization
  - Simple to Implement
  - Entire Cache as one bank (page) of memory
  - Often subject to "**thrashing**" data across pages

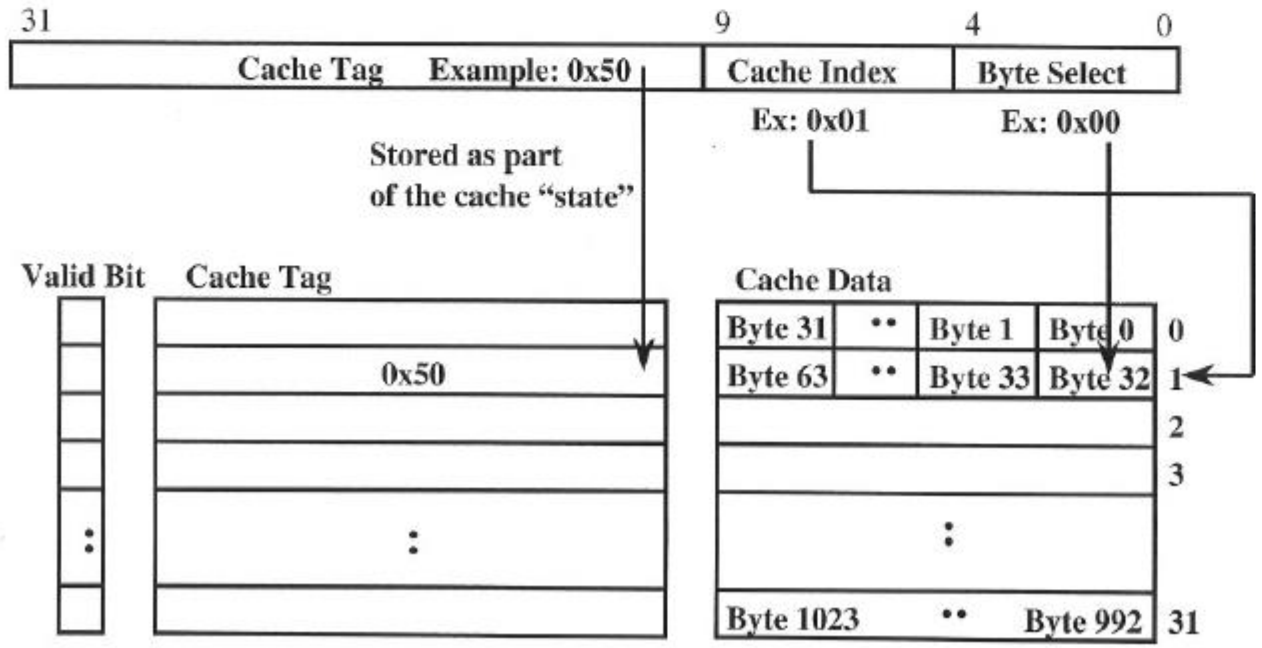


Direct Mapped Cache

Cache Organizations cont'd

Example: 1 KB Direct Mapped Cache with 32 B Blocks

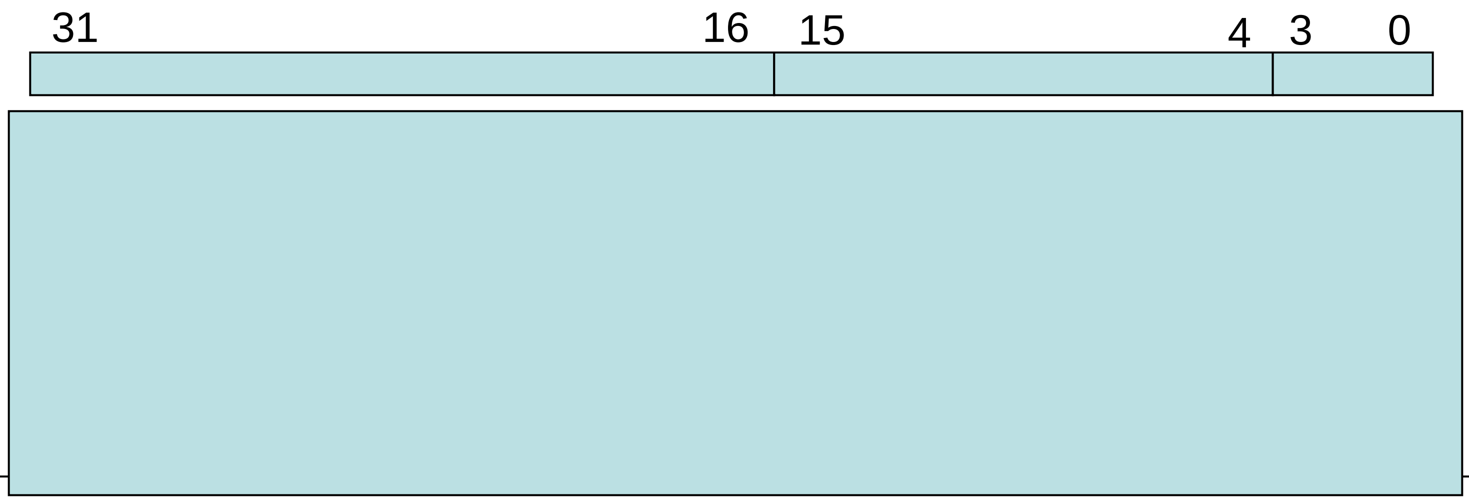
- For a 2 ** N byte cache:
  - The uppermost (32 - N) bits are always the Cache Tag
  - The lowest M bits are the Byte Select (Block Size = 2 ** M)



dapted from Lecture Notes: Computer Organization & Design: The Hardware/Software Interface, David A. Patterson and John L. Hennessy.
al Katircioglu SJSU Cmpe 140 Spring 1999

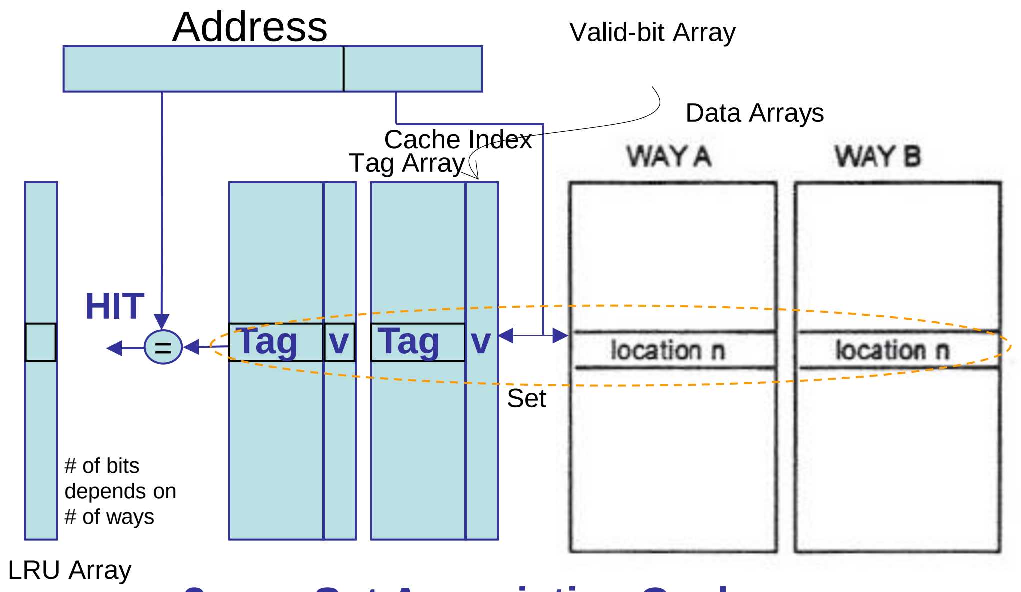
Examples

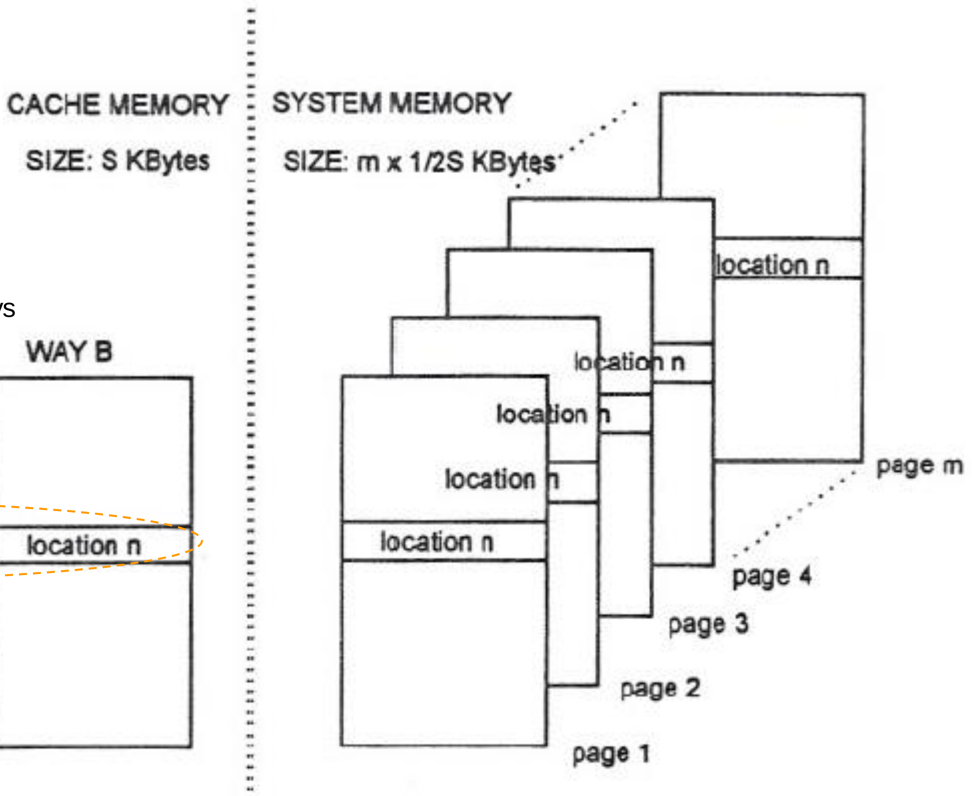
- Example 1
- In a direct-mapped cache, there are 4K lines and 16 bytes in a line Assume 32-bit addressing environment
  - How many address bits are required for byte select?
  - How many bits are required for cache index (line select)?
  - How many bits are there in each tag entry?
  - How many pages does this cache split the main memory into?



Cache Organizations cont'd

- Set Associative Cache Organization
  - Divides the cache into banks of memory
  - More complex design than direct mapped cache
  - One tag comparison for each way
  - LRU for performance



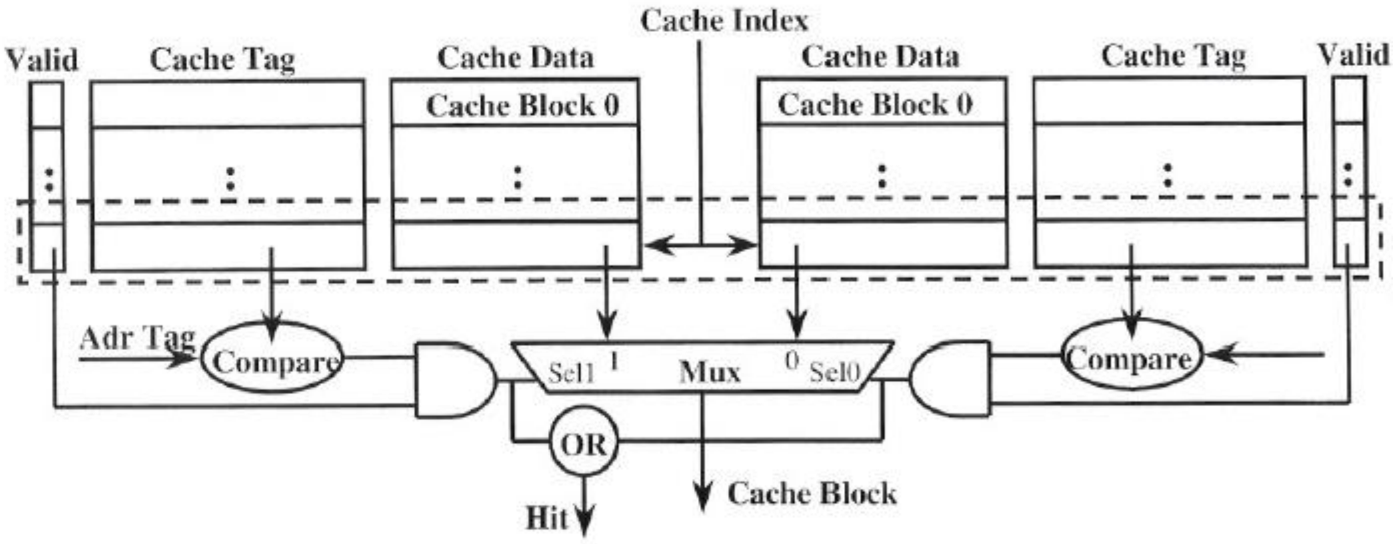


2-way Set Associative Cache

Cache Organizations cont'd

A Two-way Set Associative Cache

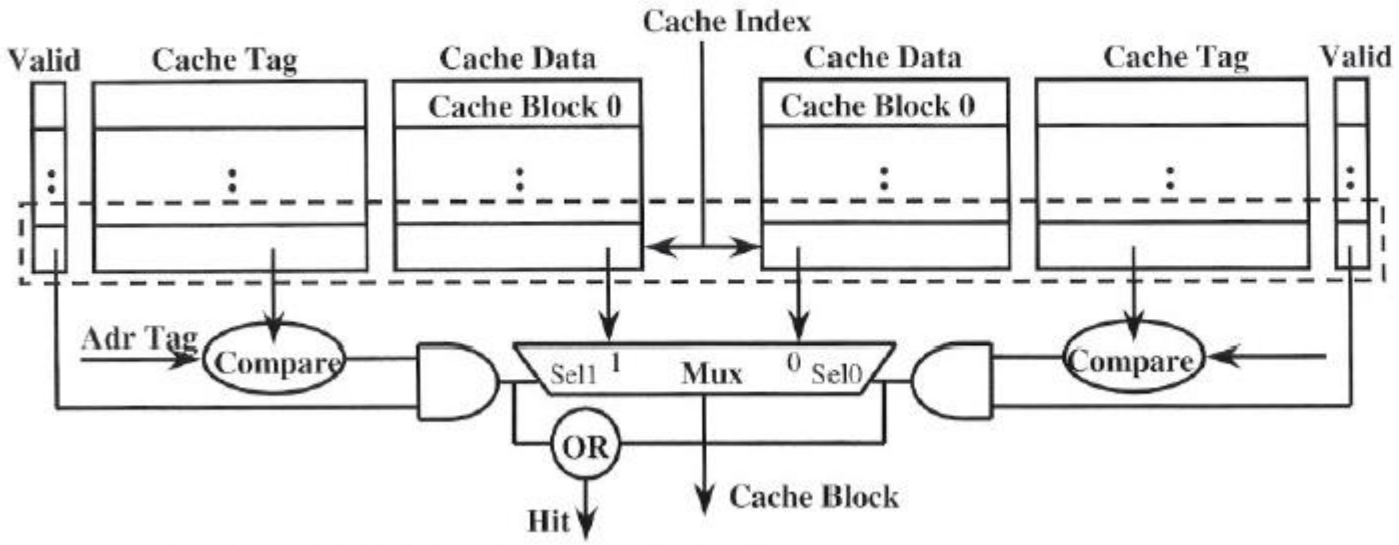
- N-way set associative: N entries for each Cache Index
  - N direct mapped caches operates in parallel
- Example: Two-way set associative cache
  - Cache Index selects a "set" from the cache
  - The two tags in the set are compared in parallel
  - Data is selected based on the tag result



Cache Organizations cont'd

Disadvantage of Set Associative Cache

- N-way Set Associative Cache versus Direct Mapped Cache:
  - N comparators vs. 1
  - Extra MUX delay for the data
  - Data comes AFTER Hit/Miss decision and set selection
- In a direct mapped cache, Cache Block is available BEFORE Hit/Miss:
  - Possible to assume a hit and continue. Recover later if miss.

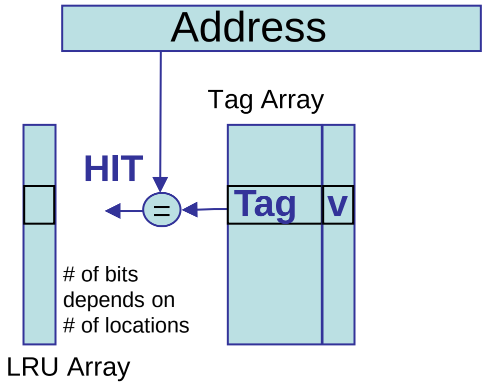


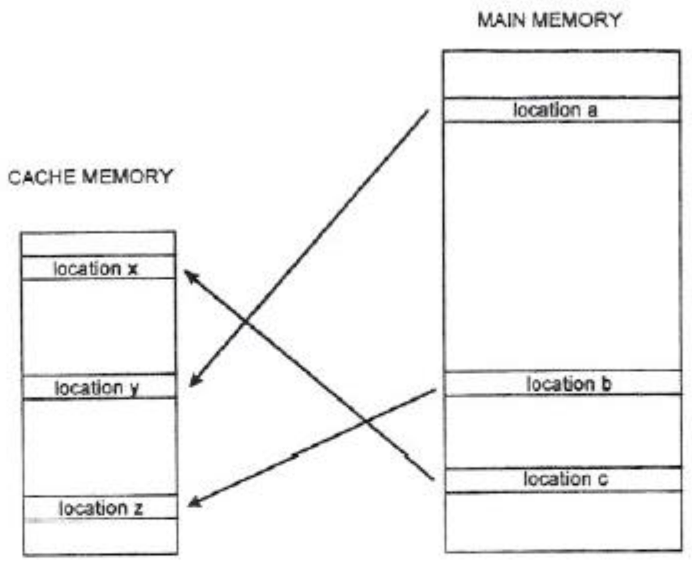
Examples

- Example 3
- In a 4-way set-associative cache, there are 1K sets and 16 bytes in a line
  Assume 32-bit addressing environment
  - How many address bits are required for byte select?
  - How many bits are required for cache index (line select)?
  - How many bits are there in each tag entry?
  - How many pages does this cache split the main memory into?

Cache Organizations cont’d

- Fully Associative Cache Organization
  - No relationship between cache and memory
  - Each location can hold information from any location in memory
  - Very complex and expensive comparison logic – one for each tag location
  - LRU for performance



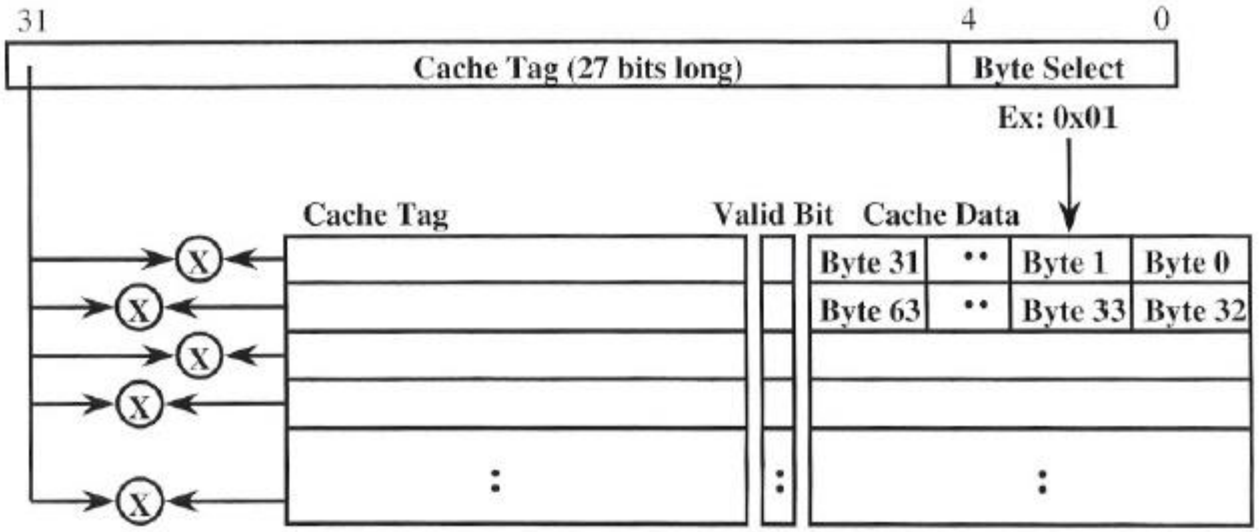


Fully Associative Cache

Cache Organizations cont'd

Another Extreme Example: Fully Associative

- Fully Associative Cache
  - Forget about the Cache Index
  - Compare the Cache Tags of all cache entries in parallel
  - Example: Block Size = 2 B blocks, we need N 27-bit comparators
- By definition: Conflict Miss = 0 for a fully associative cache

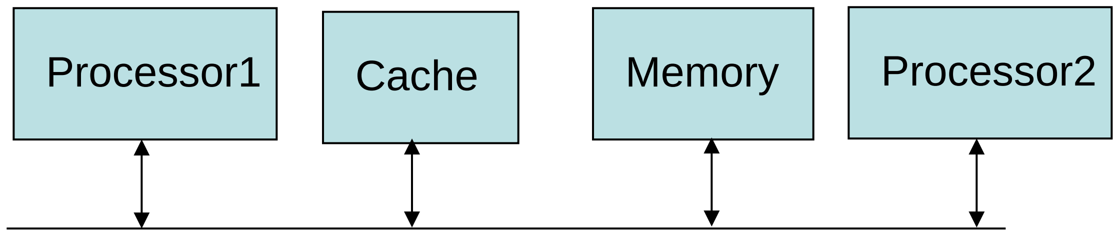


Cache Architecture – Read Performance cont;d

- Line Size:
  - The string of bytes of code or data copied from main memory into the cache
  - In general line size increases hit rate
- Replacement (update) Policy:
  - Always update – for Direct Mapped Cache
  - Least recently Used (LRU)
  - Random – Set Associative or Fully Associative Caches

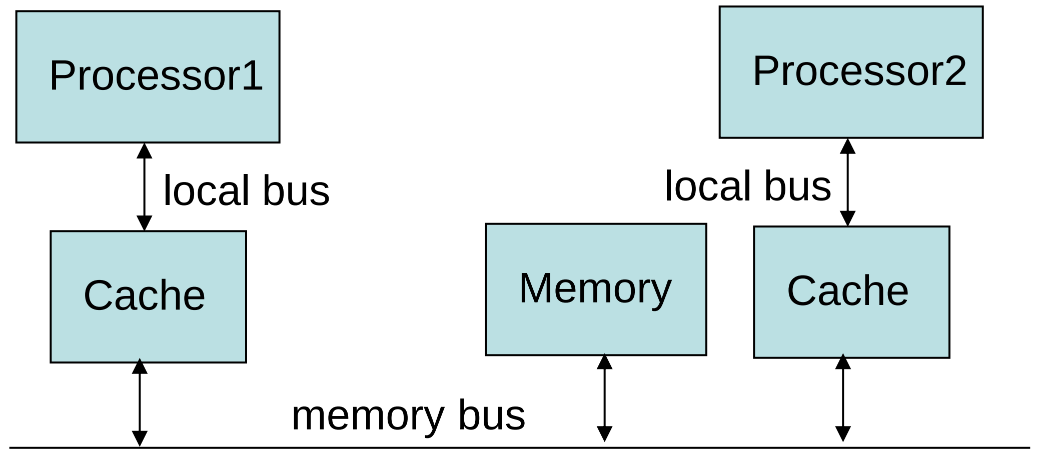
Cache Architecture – Bus Utilization cont'd

- Bus Concurrency
  - Single vs Dual memory bus architecture



High memory bus utilization

How do you handle cache consistency? What kind of cache do you think is best?



Reduced memory bus utilization

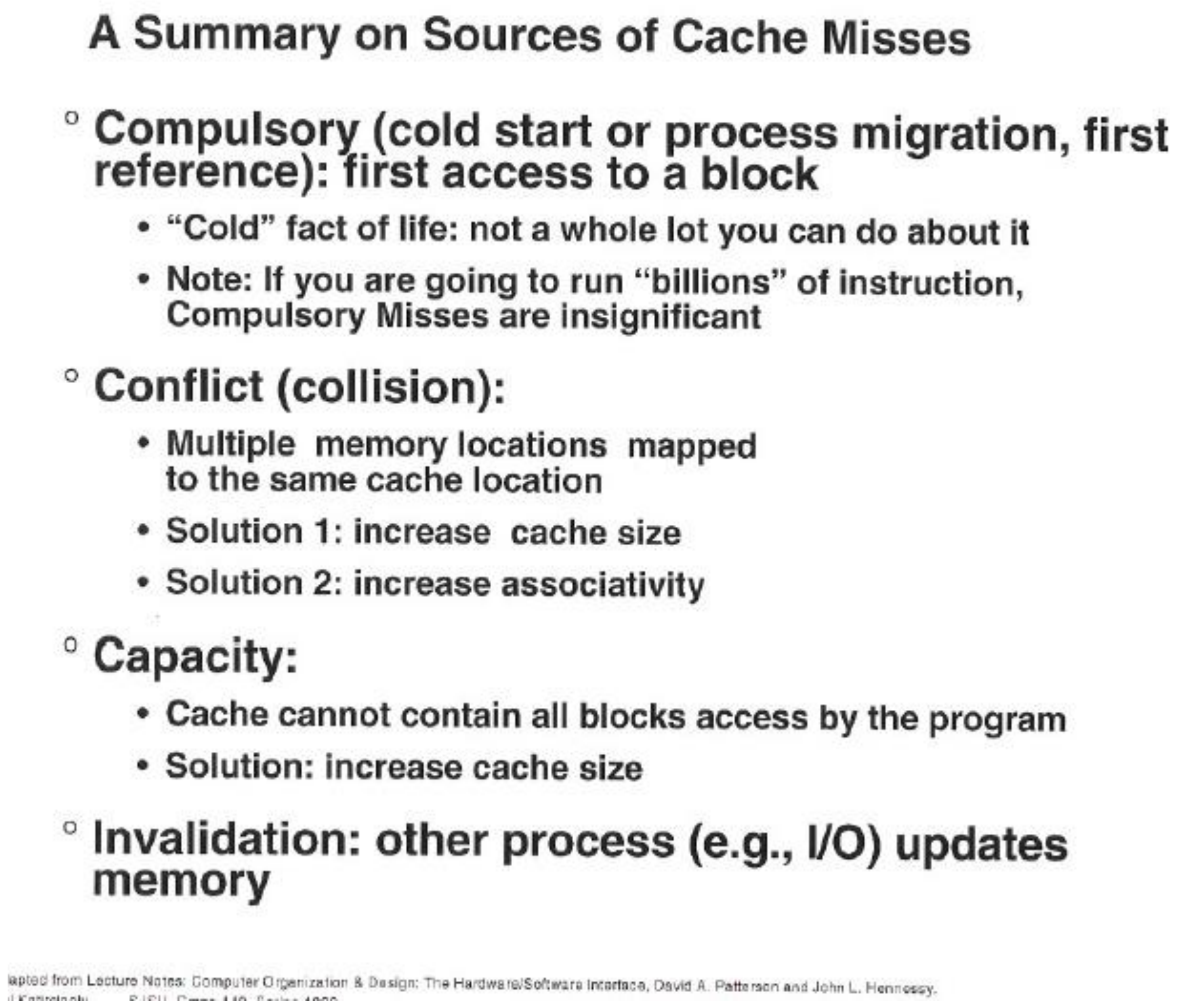
How do you handle cache consistency? What kind of cache do you think is best

Cache Measures:

- *Hit rate*: fraction found in that level
  - So high that usually talk about *Miss rate*
  - Miss rate fallacy: as MIPS to CPU performance,
    miss rate to average memory access time in memory
- Average memory-access time
  = Hit time x Hit Rate + Miss rate x Miss penalty
  (ns or clocks)
- *Miss penalty*: time to replace a block from lower level, including time to replace in CPU
  - *access time*: time to lower level = f(latency to lower level)
  - *transfer time*: time to transfer block =f(**BW** between upper & lower levels)

from David Patterson

Cache Measures cont'd



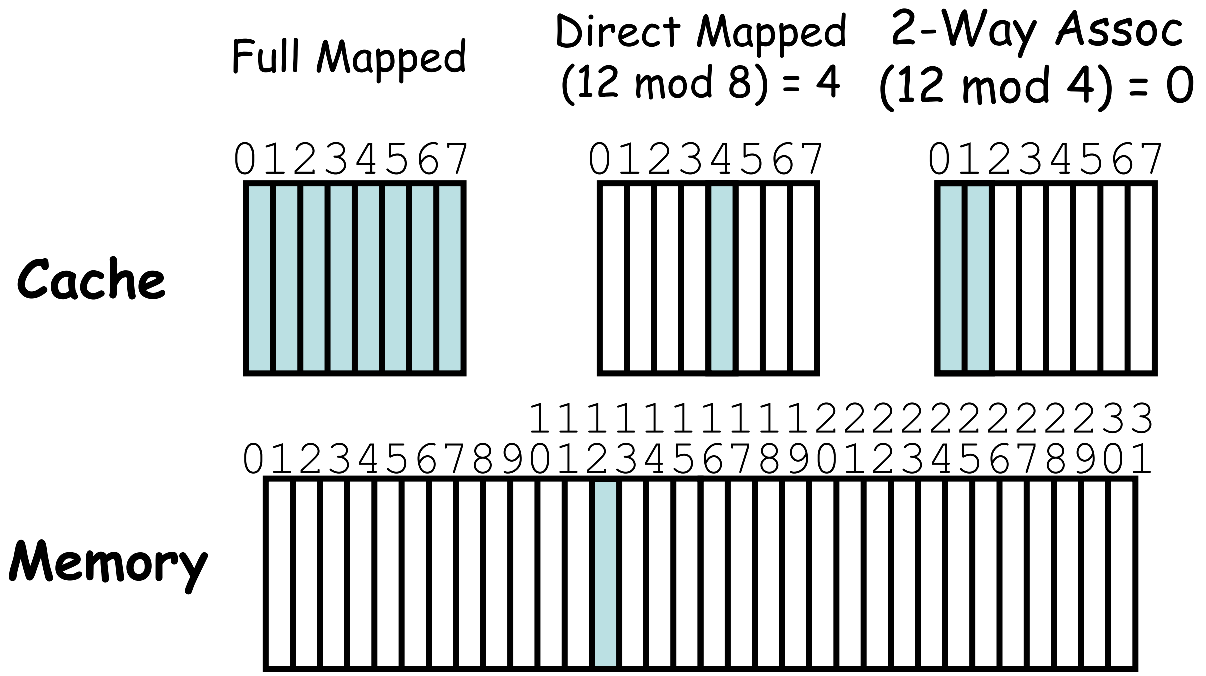
A Summary on Sources of Cache Misses

- Compulsory (cold start or process migration, first reference): first access to a block
  - "Cold" fact of life: not a whole lot you can do about it
  - Note: If you are going to run "billions" of instruction, Compulsory Misses are insignificant
- Conflict (collision):
  - Multiple memory locations mapped to the same cache location
  - Solution 1: increase cache size
  - Solution 2: increase associativity
- Capacity:
  - Cache cannot contain all blocks access by the program
  - Solution: increase cache size
- Invalidation: other process (e.g., I/O) updates memory

apted from Lecture Notes: Computer Organization & Design: The Hardware/Software Interface, David A. Patterson and John L. Hennessy.
[illegible]

Q1: Where can a block be placed in the upper level?

- Block 12 placed in 8 block cache:
  - Fully associative, direct mapped, 2-way set associative
  - S.A. Mapping = Block Number Modulo Number Sets



from David Patterson

Cache Architecture – EXAMPLE

- HIT/MISS Example
  - The following is a direct mapped cache. The lower 4 hex digits in the address are decoded as an index to cache lines. The upper 4 hex digits are decoded as TAG. No allocation is allowed for write misses.
    1. Mark each access as MISS or HIT appropriately while program is being executed
    2. What is the number of lines in this cache

Index	TAG
FFFF	0100
:	
F000	1000
:	
0E00	0100
:	
0300	FFFF
:	
0200	0100
:	
0100	0000
:	
0000	0100

```
Read  0000  0100
Read  0022  0E00
Write  1000  F000
Read  FFFF  0300
Read  0100  0E00
Read  0300  0200
Read  0200  0100
Write   0022  0E00
```

HIT/MISS