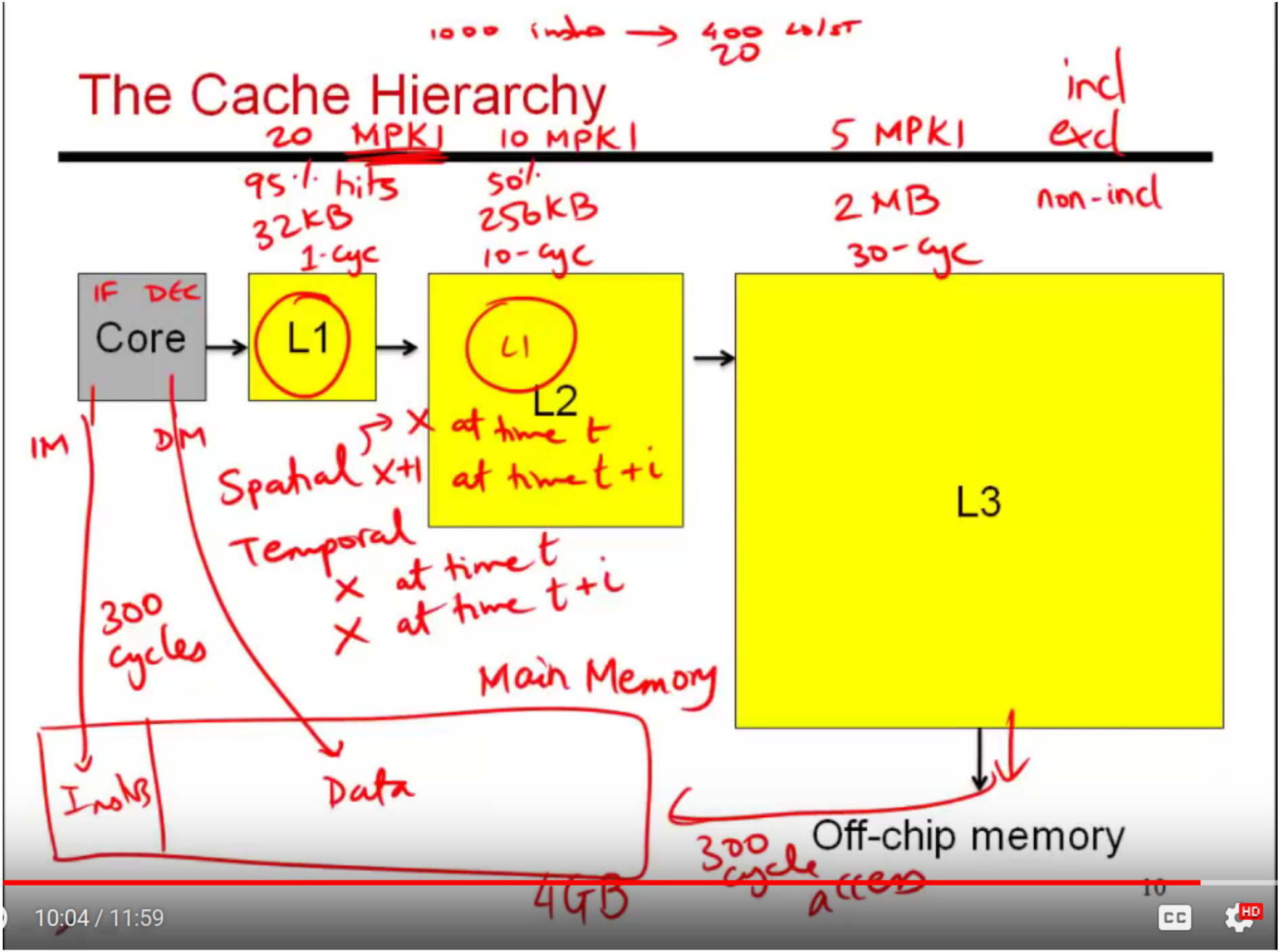
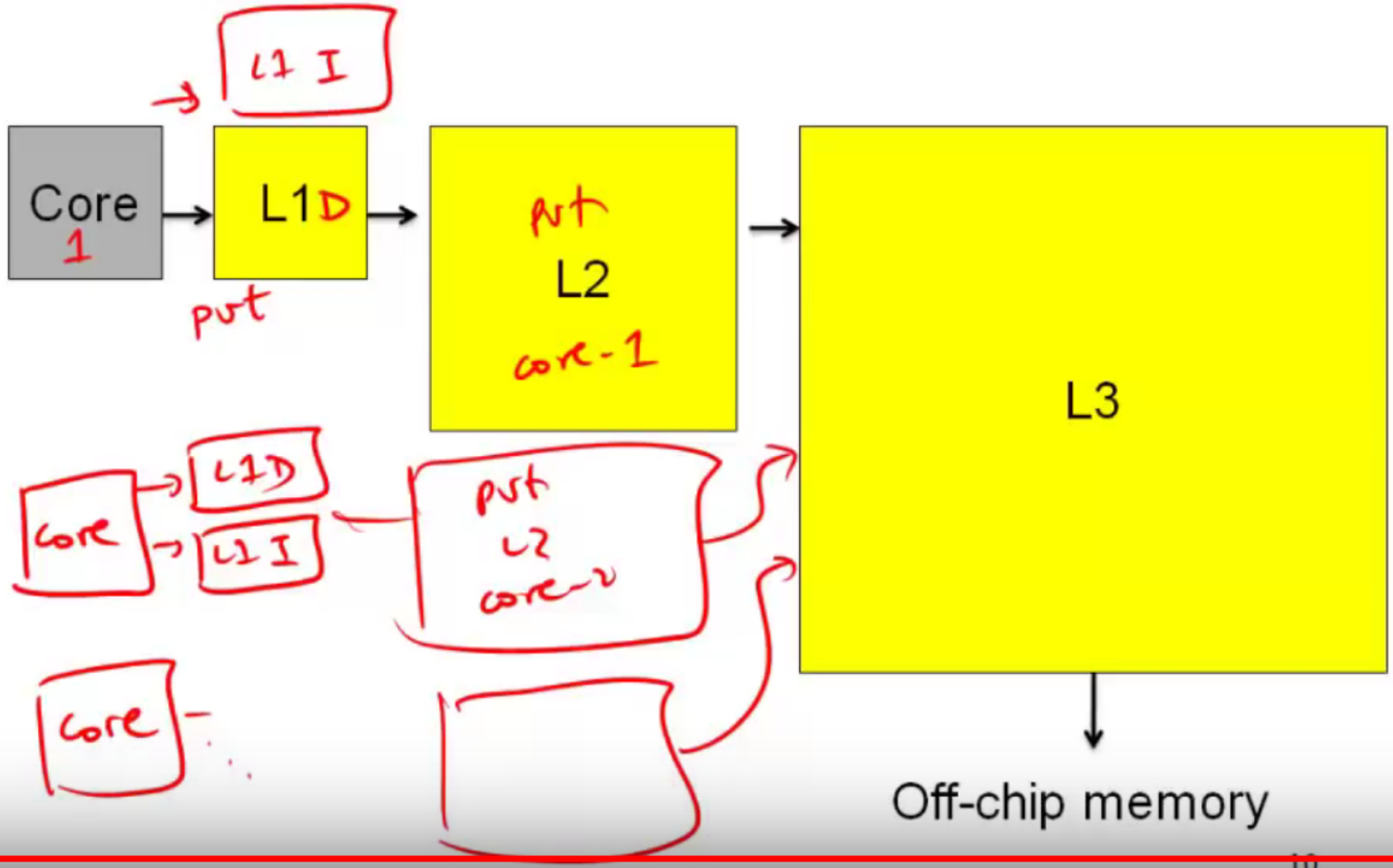
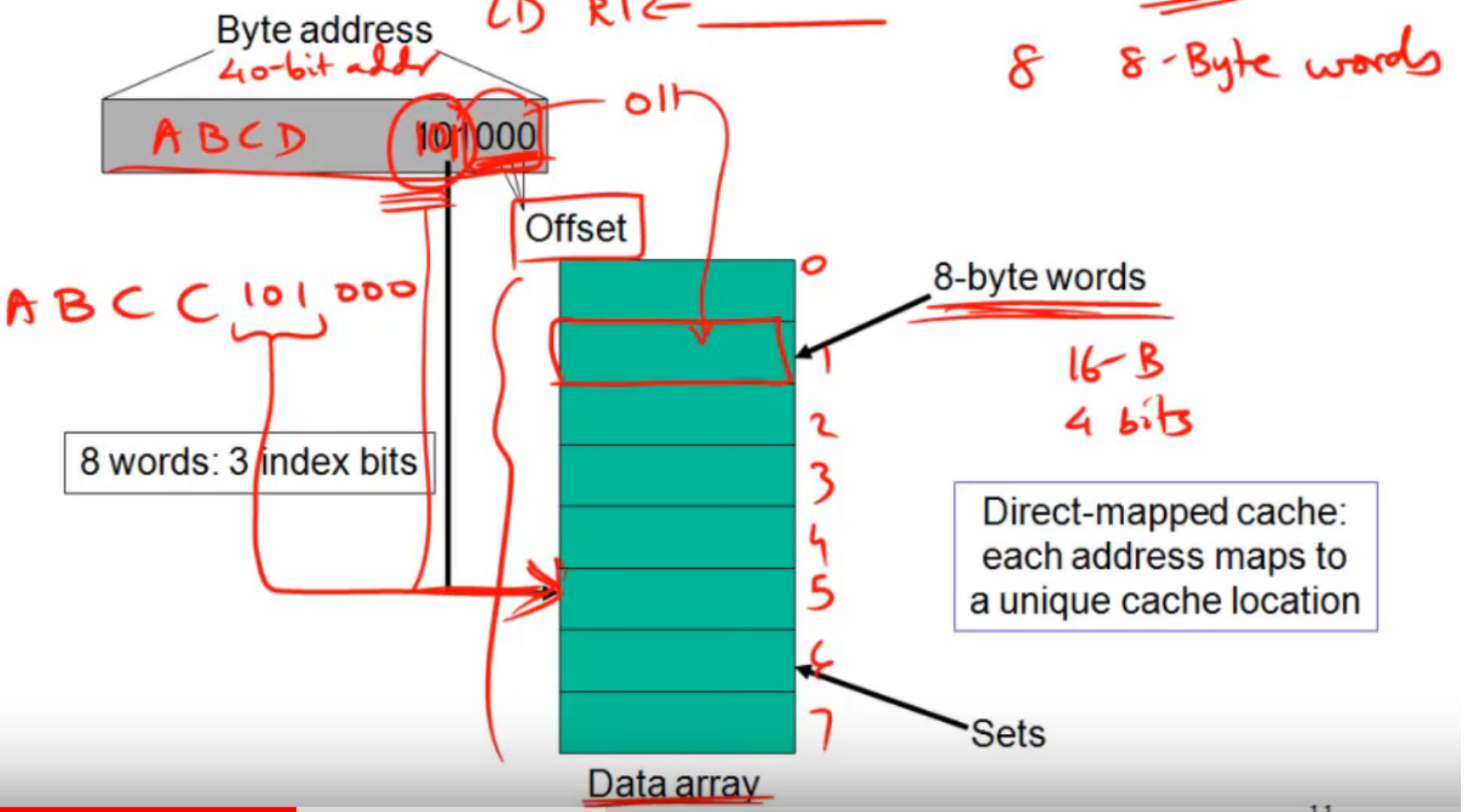

1000 insts → 400 cost
20
The Cache Hierarchy
20 MPKI
10 MPKI
5 MPKI
incl
excl
non-incl
95% hits
50%
32KB
1-cyc
256KB
10-cyc
2MB
30-cyc
IF DEC
Core
L1
L1
L2
L3
IM
DM
X
X+1
Spatial
at time t
at time t+i
Temporal
X at time t
X at time t+i
300 cycles
Main Memory
Insts
Data
4GB
Off-chip memory
300 cycle access
10:04 / 11:59
CC
HD

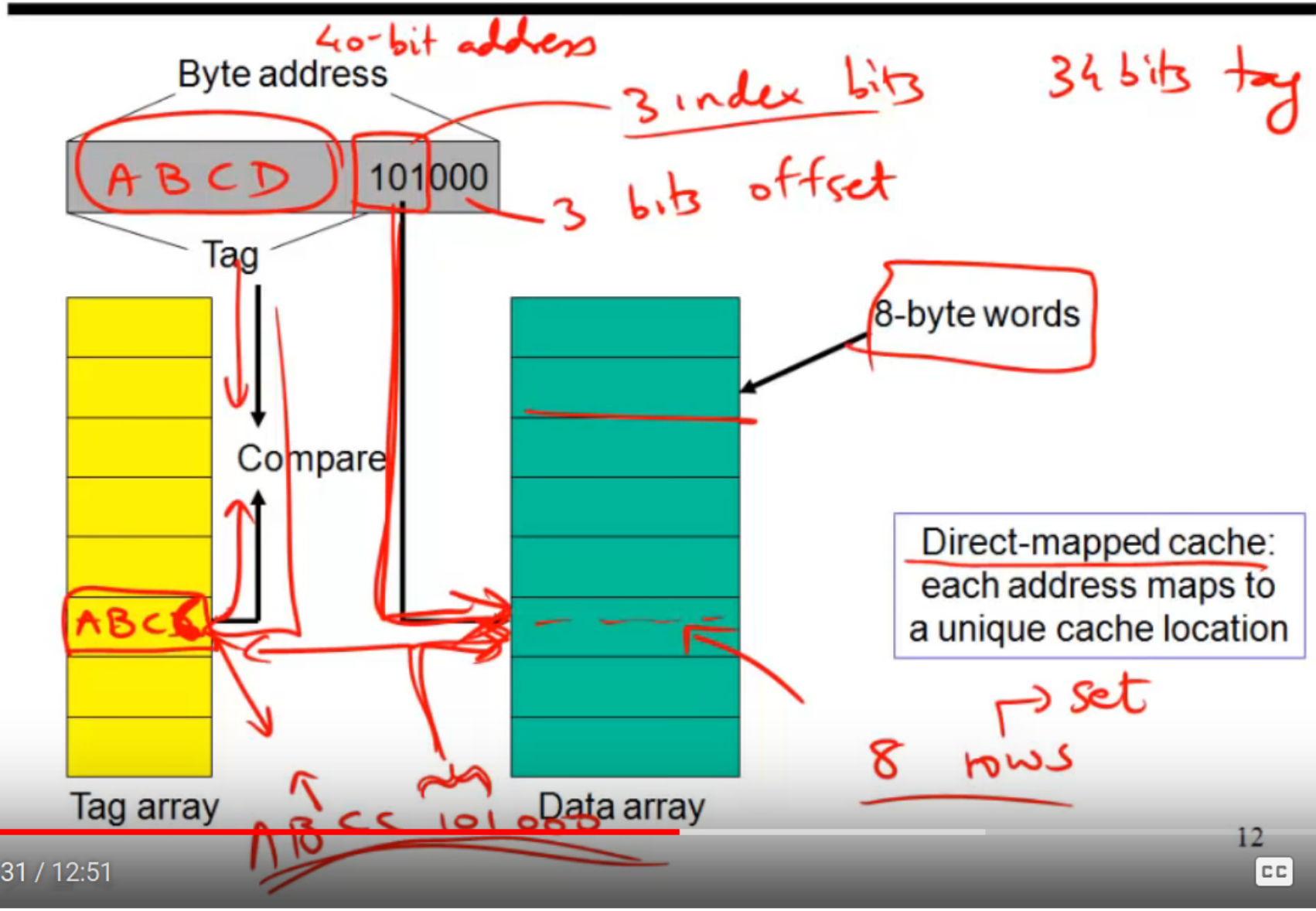
The Cache Hierarchy



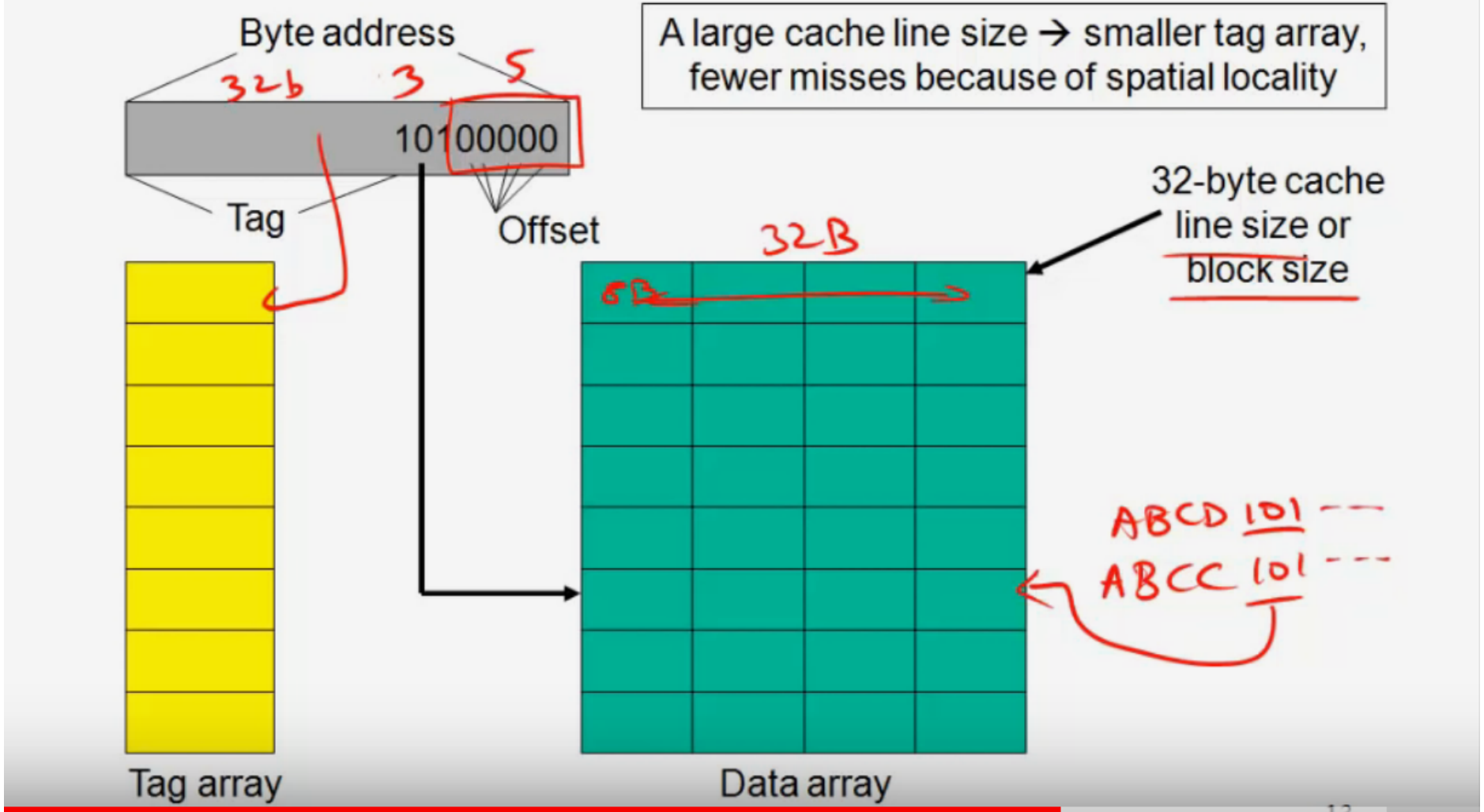
Accessing the Cache



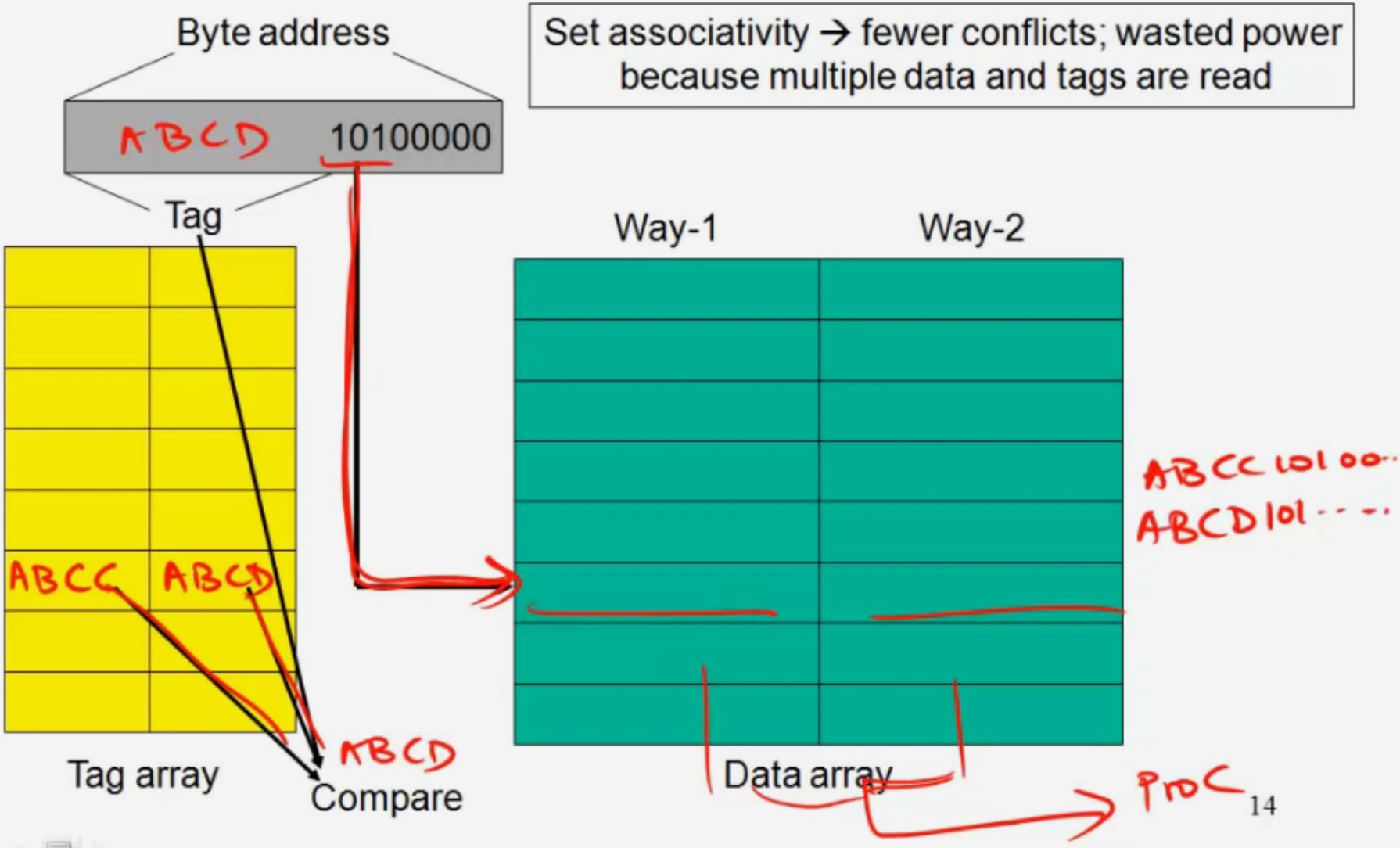
The Tag Array



Increasing Line Size

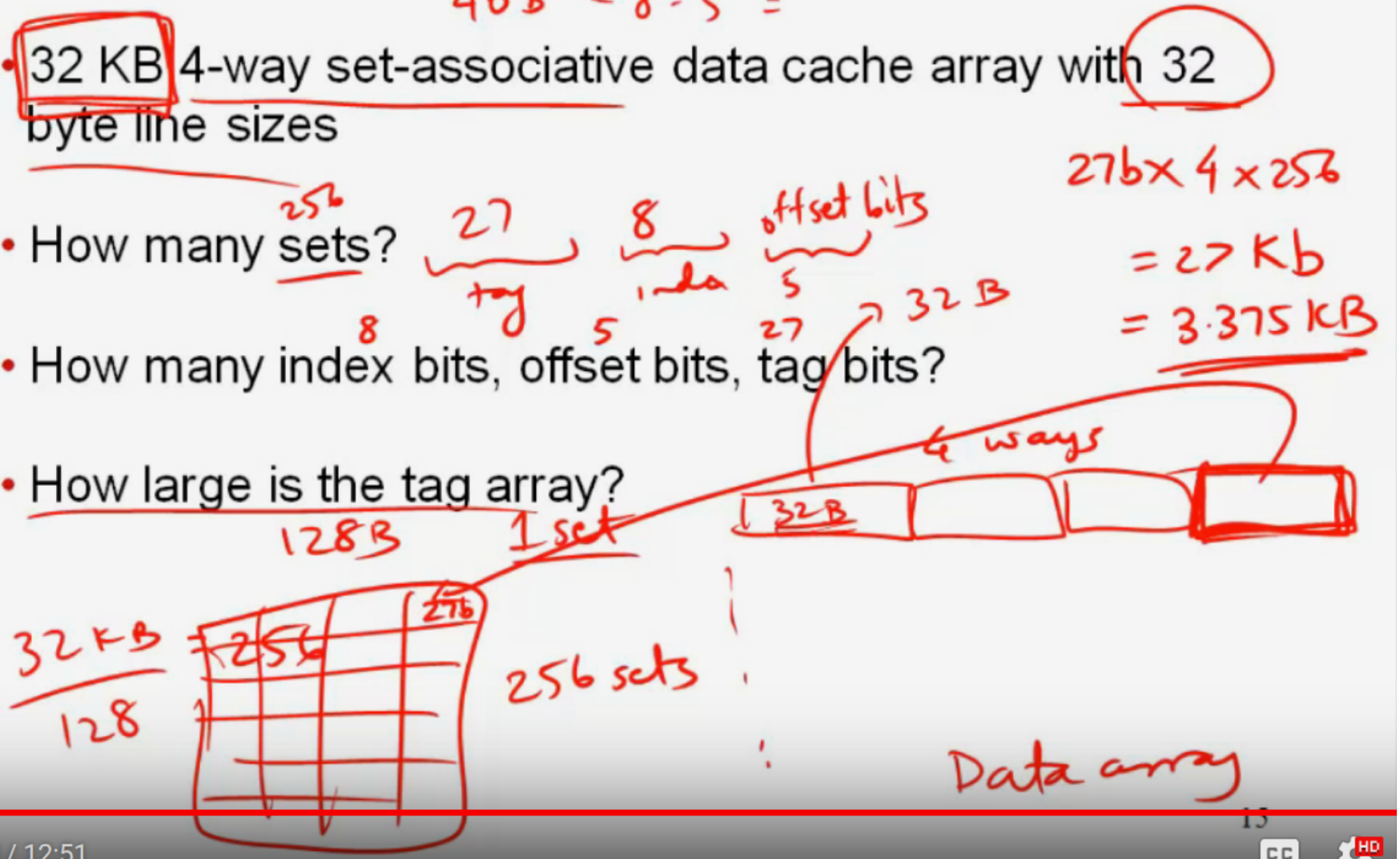


Associativity



Example

- 32 KB 4-way set-associative data cache array with 32 byte line sizes
- How many sets?
- How many index bits, offset bits, tag bits?
- How large is the tag array?

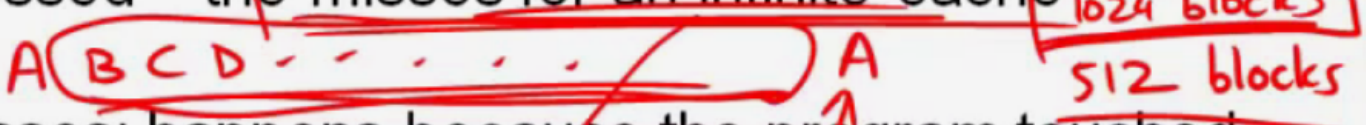


Types of Cache Misses

3 C's

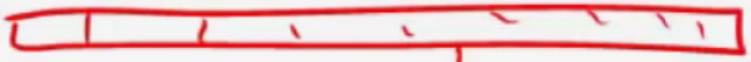
- Compulsory misses: happens the first time a memory word is accessed – the misses for an infinite cache

800 diff blocks

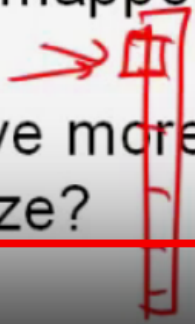


- Capacity misses: happens because the program touched many other words before re-touching the same word – the misses for a fully-associative cache

Miss



- Conflict misses: happens because two words map to the same location in the cache – the misses generated while moving from a fully-associative to a direct-mapped cache



- Sidenote: can a fully-associative cache have more misses than a direct-mapped cache of the same size?

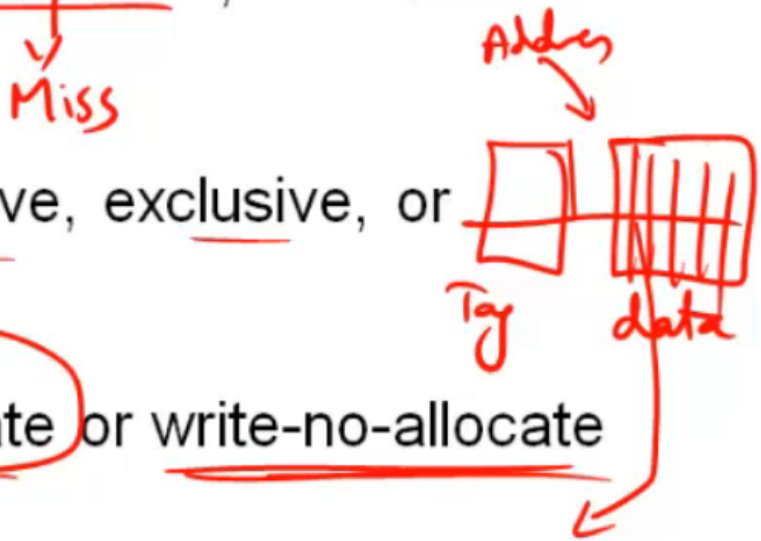
Reducing Miss Rate

- Large block size – reduces compulsory misses, reduces miss penalty in case of spatial locality – increases traffic between different levels, space waste, and conflict misses
- Large cache – reduces capacity/conflict misses – access time penalty
- High associativity – reduces conflict misses – rule of thumb: 2-way cache of capacity N/2 has the same miss rate as 1-way cache of capacity N – more energy

4-way 32KB 2-way 64KB 1-way

More Cache Basics

- L1 caches are split as instruction and data; L2 and L3 are unified



- The L1/L2 hierarchy can be inclusive, exclusive, or non-inclusive
- On a write, you can do write-allocate or write-no-allocate
- On a write, you can do writeback or write-through; write-back reduces traffic, write-through simplifies coherence
- Reads get higher priority; writes are usually buffered
- L1 does parallel tag/data access; L2/L3 does serial tag/data

Victim Caches

- A direct-mapped cache suffers from misses because multiple pieces of data map to the same location
- The processor often tries to access data that it recently discarded – all discards are placed in a small victim cache (4 or 8 entries) – the victim cache is checked before going to L2
- Can be viewed as additional associativity for a few sets that tend to have the most conflicts