CASPER

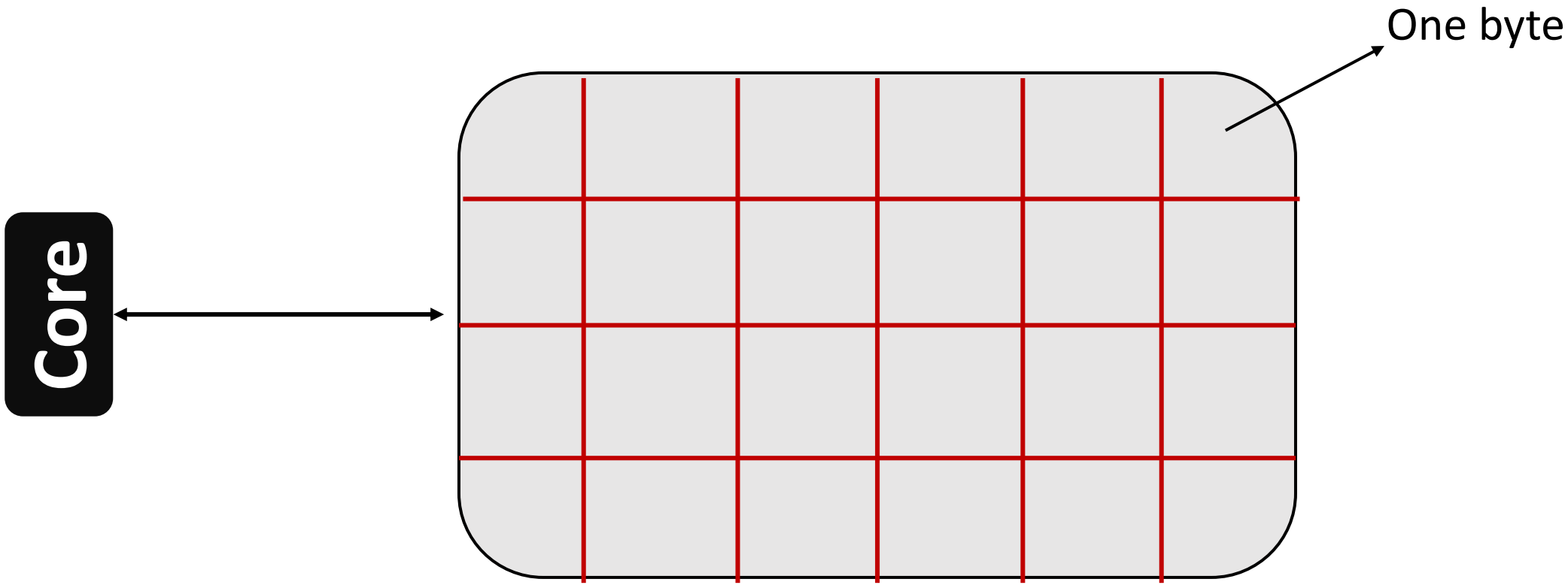
CS305: Computer Architecture

Caches-II

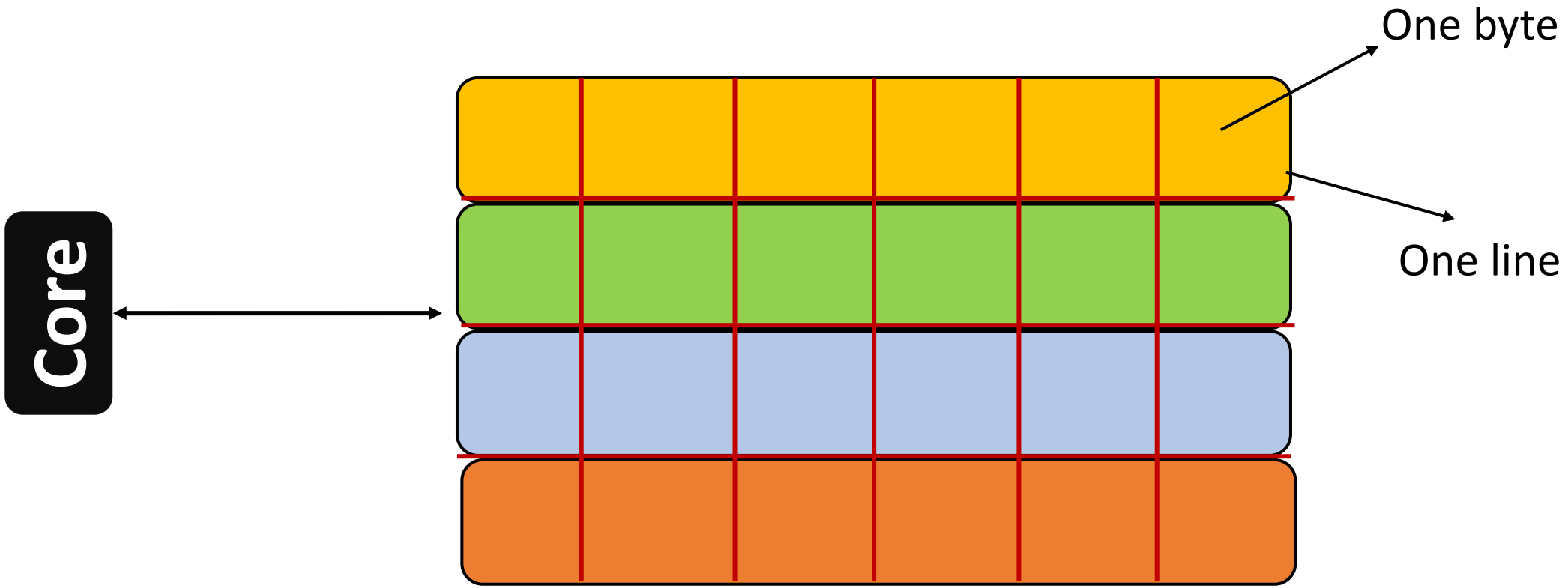
https://www.cse.iitb.ac.in/~biswa/courses/CS305/main.html

https://www.cse.iitb.ac.in/~biswa/

Accessing a cache

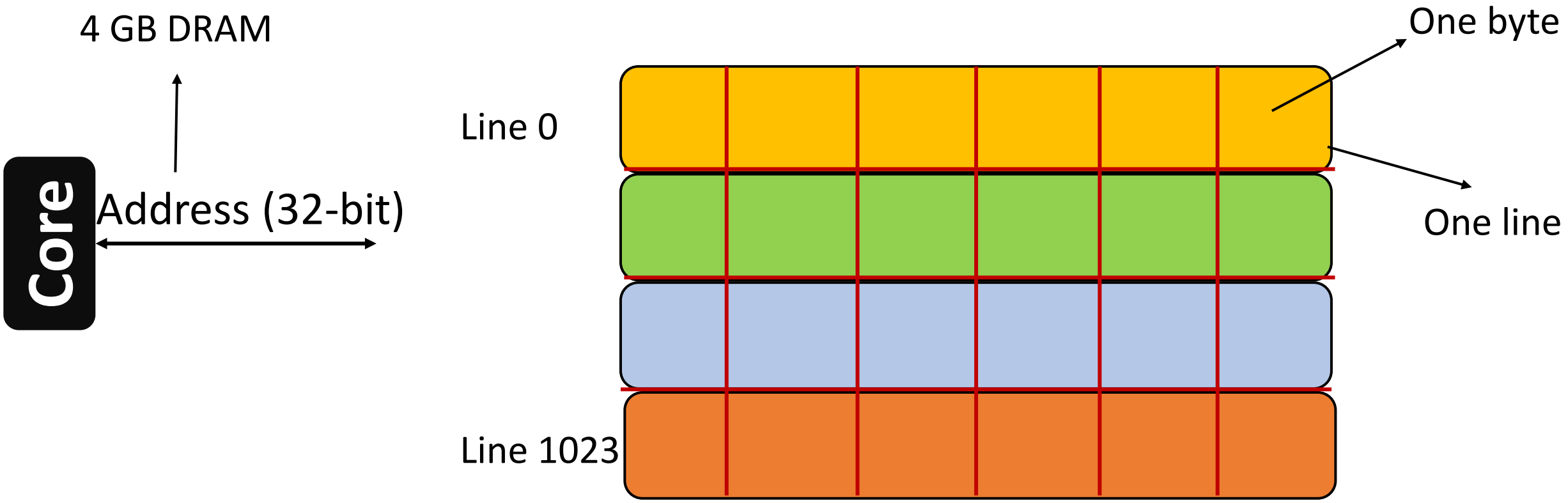


Bytes to blocks (lines)

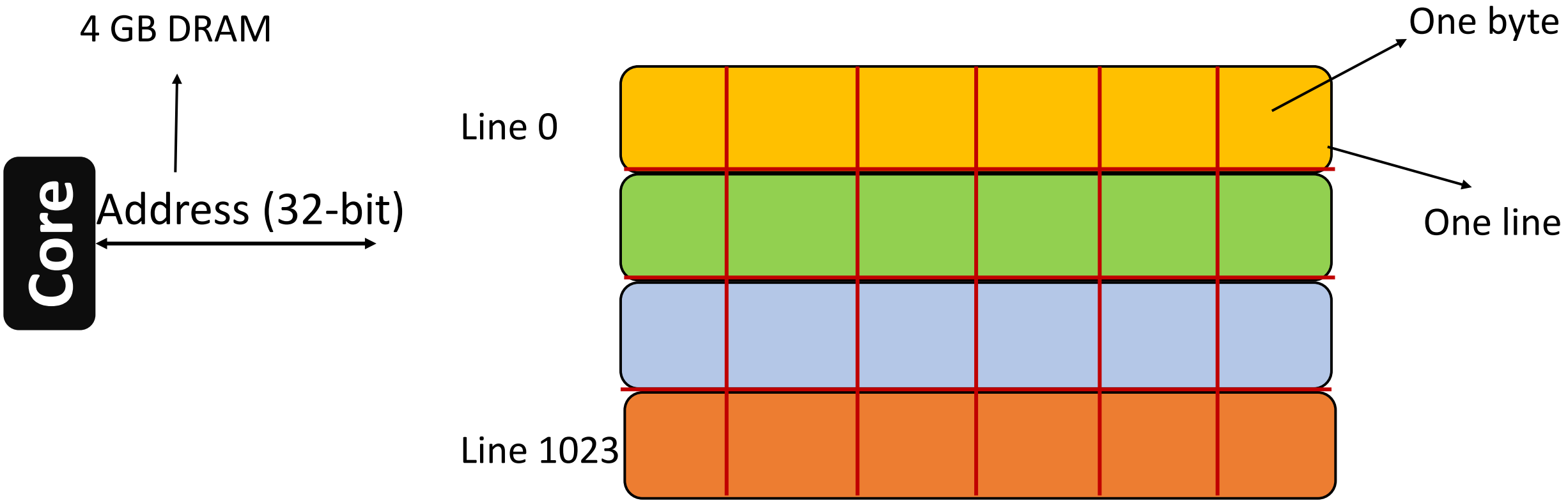


Typical line size: 64 to 128 Bytes

A bit deeper: 1024 lines each of 32B



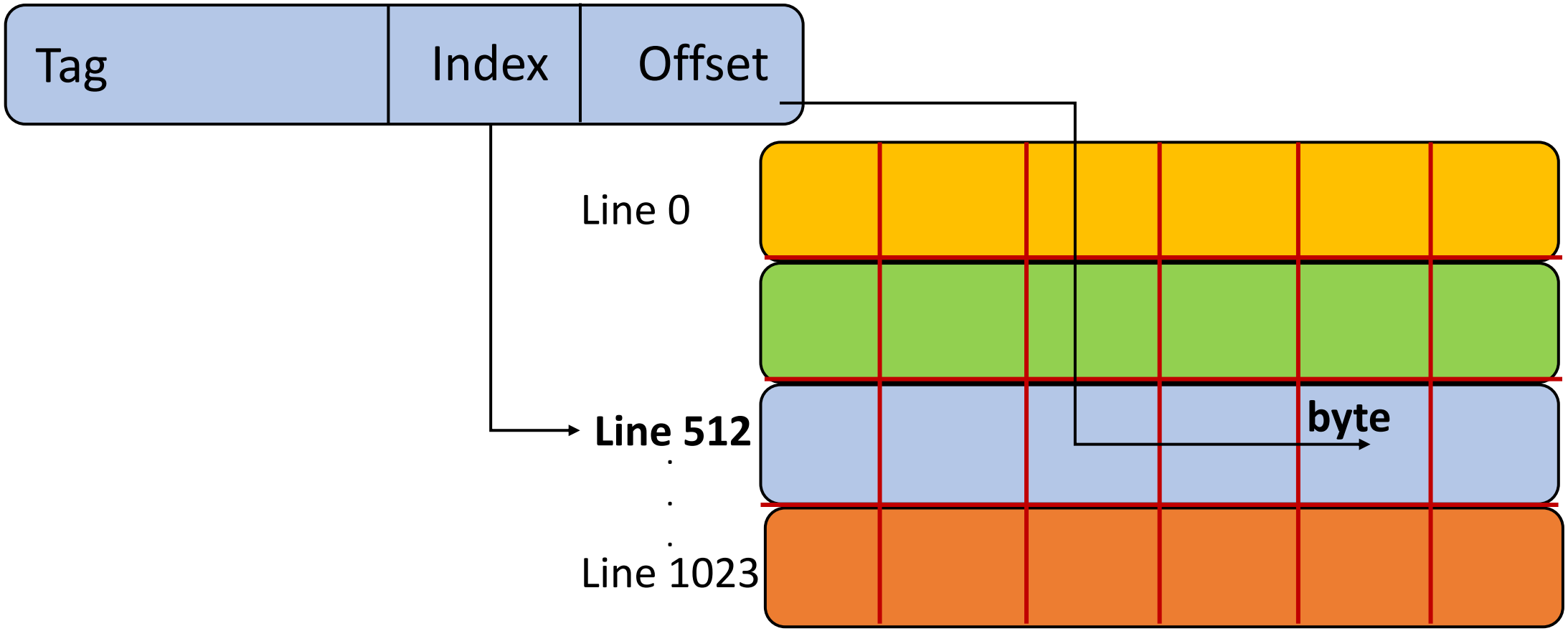
A bit deeper: 1024 lines each of 32B



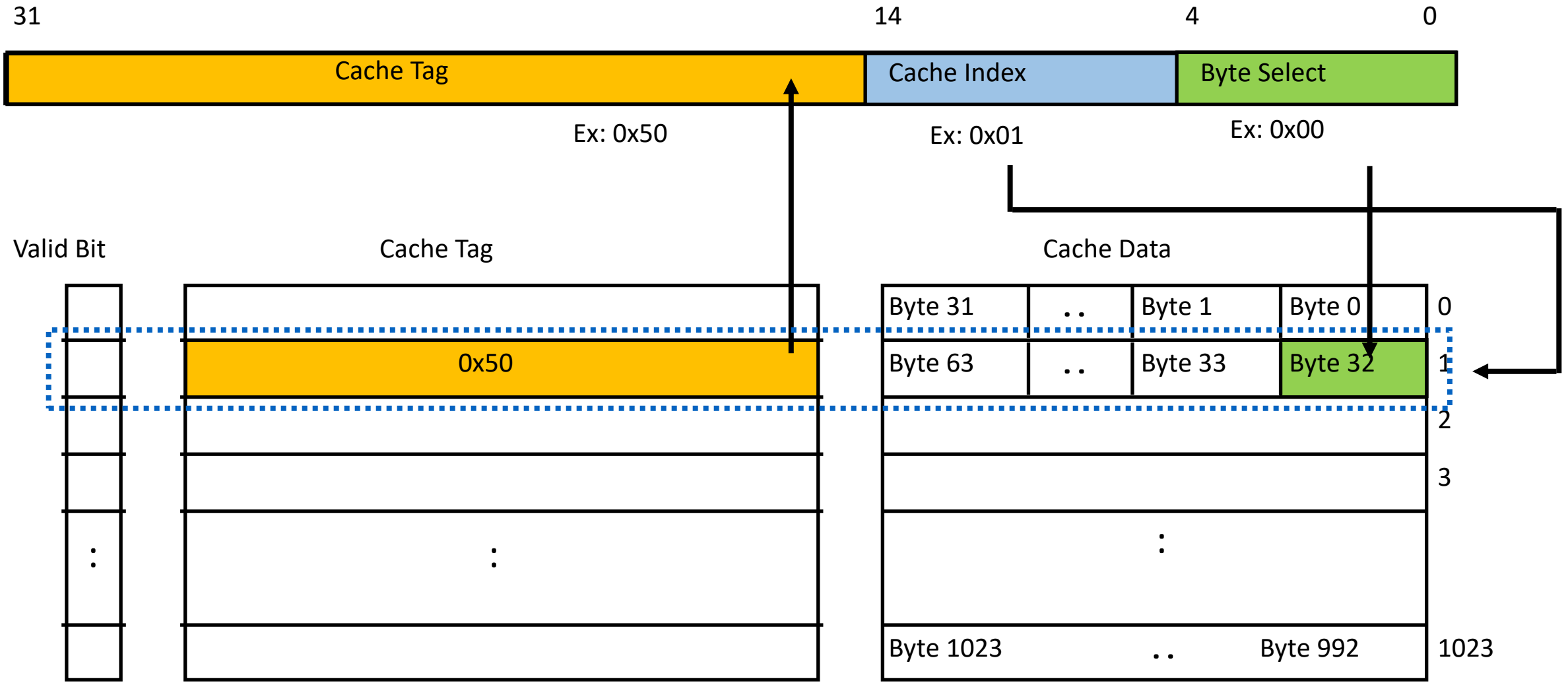
Line number (index): 10 bits

Byte offset (offset): 5 bits

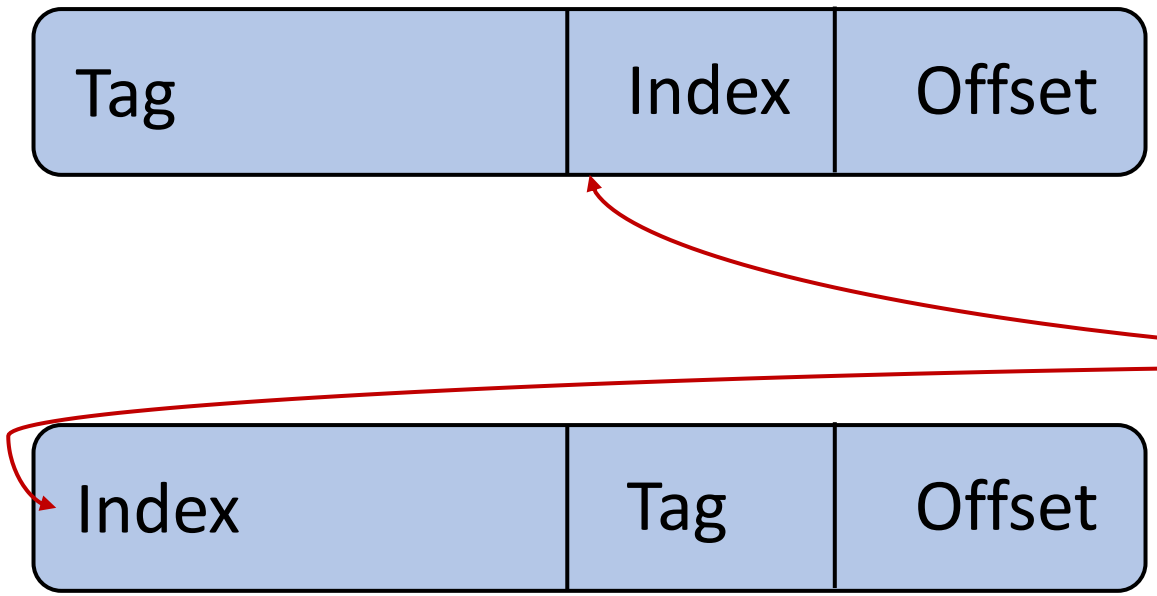
Direct Mapped Cache



Direct Mapped in Action



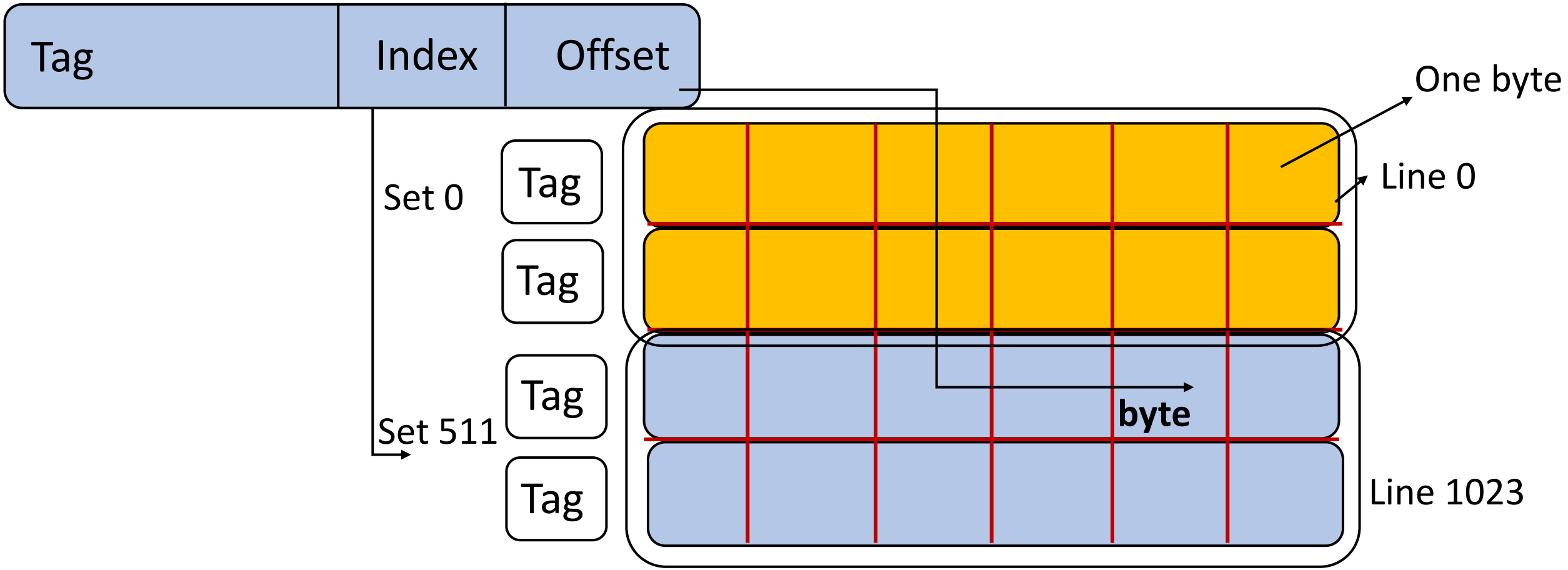
Why not this?



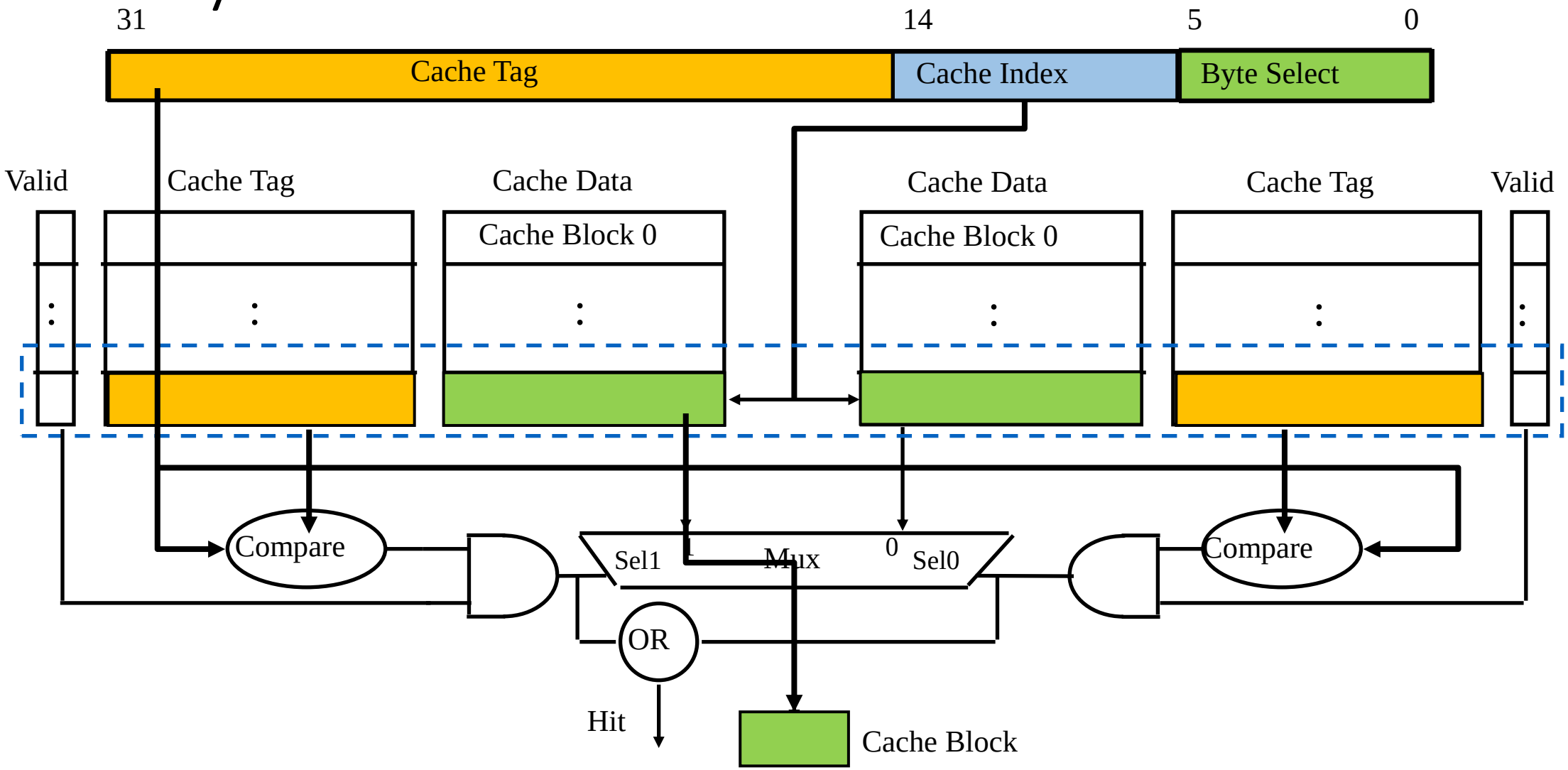

```
sum = 0;
for (i = 0; i < n; i++)
  sum += a[i];
return sum;
```

Go through Figure 5.9 of P&H and walk-through the example

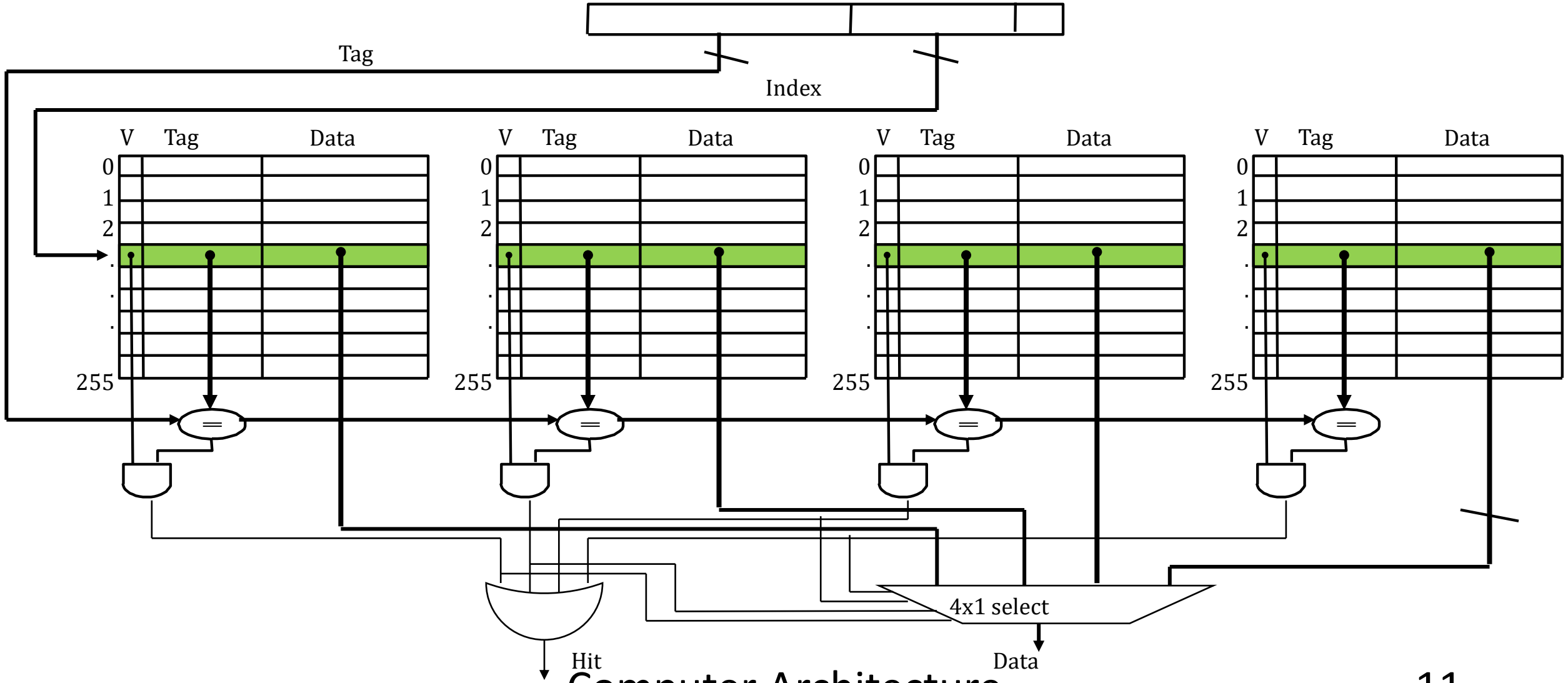
What if we have multiple ways?



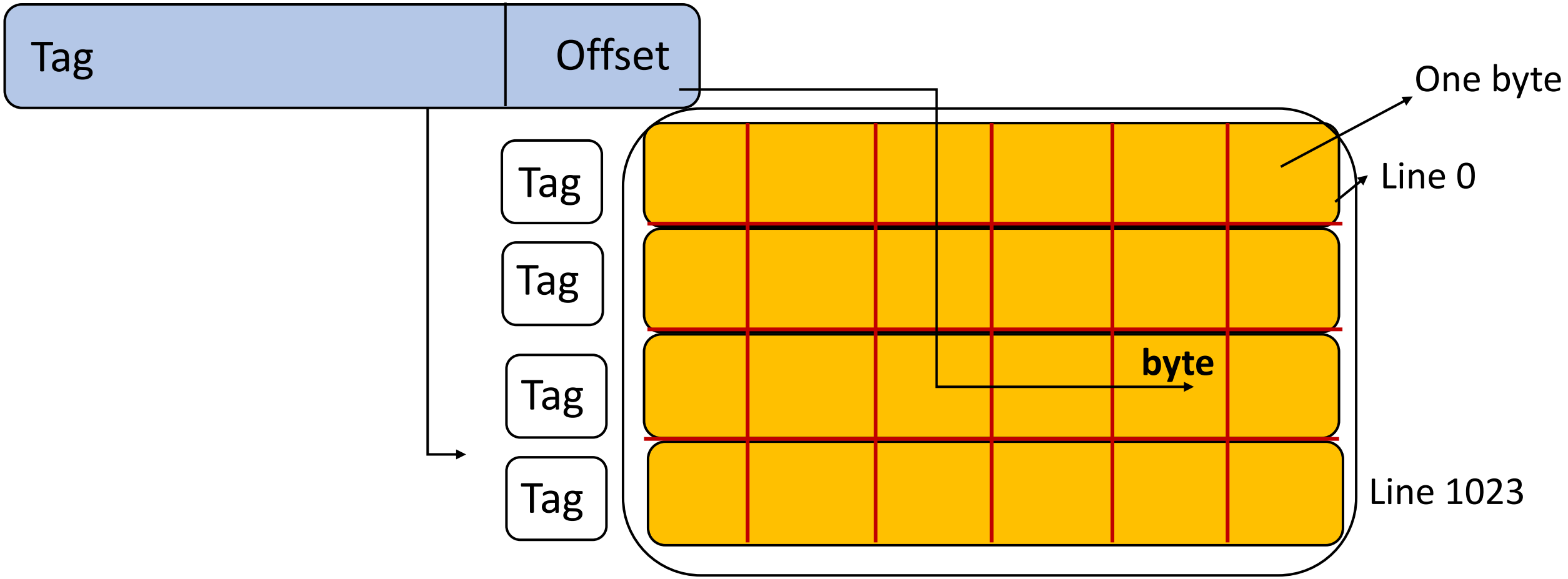
2-way associative in action



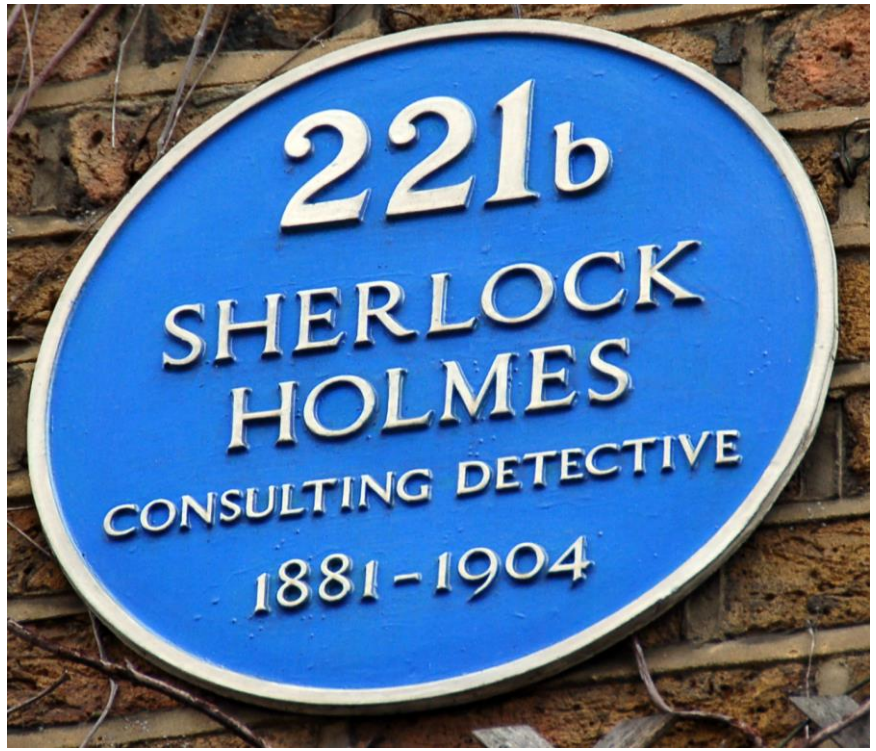
4-way associative: Just a better picture



Extreme: One cache, one set



A bit different way



Baker Street: Cache Index ☺

221b: Tag bits ☺

Sherlock Holmes: Byte offset ☺ ☺

Knobs of interest

Line size, associativity, cache size

Tradeoff: latency, complexity, energy/power

Tips: Think about the extremes:

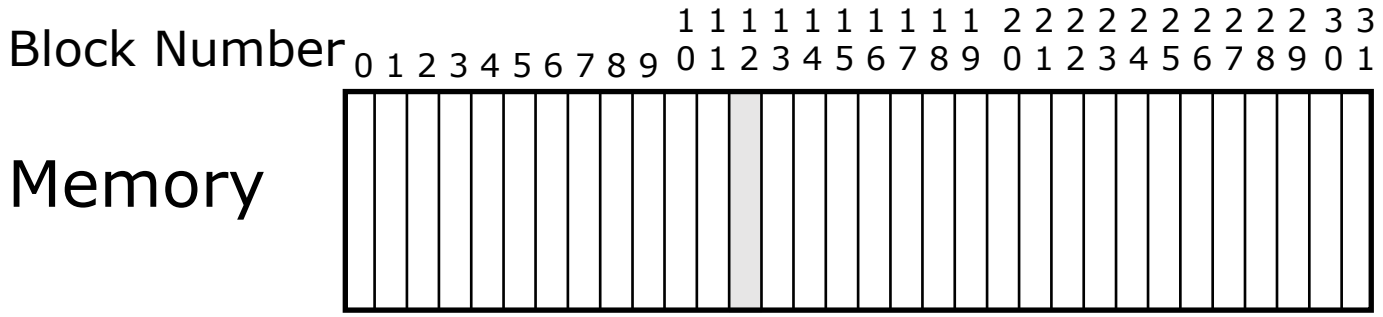
Line size = one byte or cache size

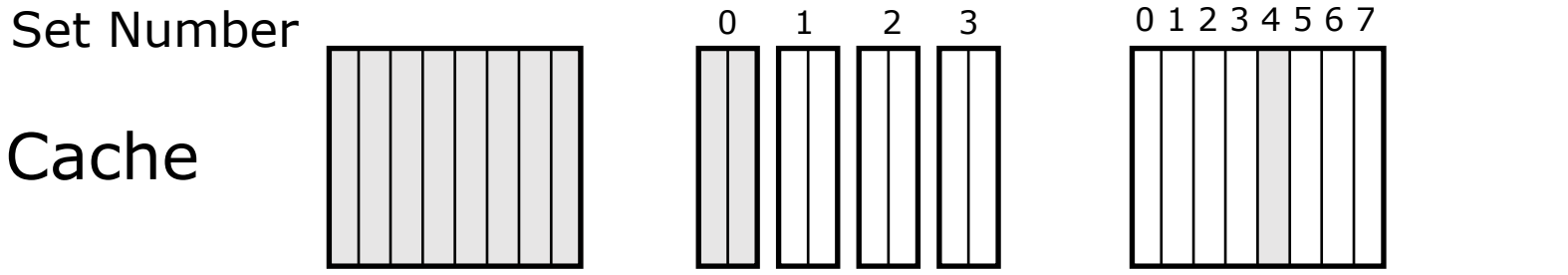
Associativity = one or #lines

Cache size = Goal oriented: latency/bandwidth or capacity

https://github.com/HewlettPackard/cacti/

Summary





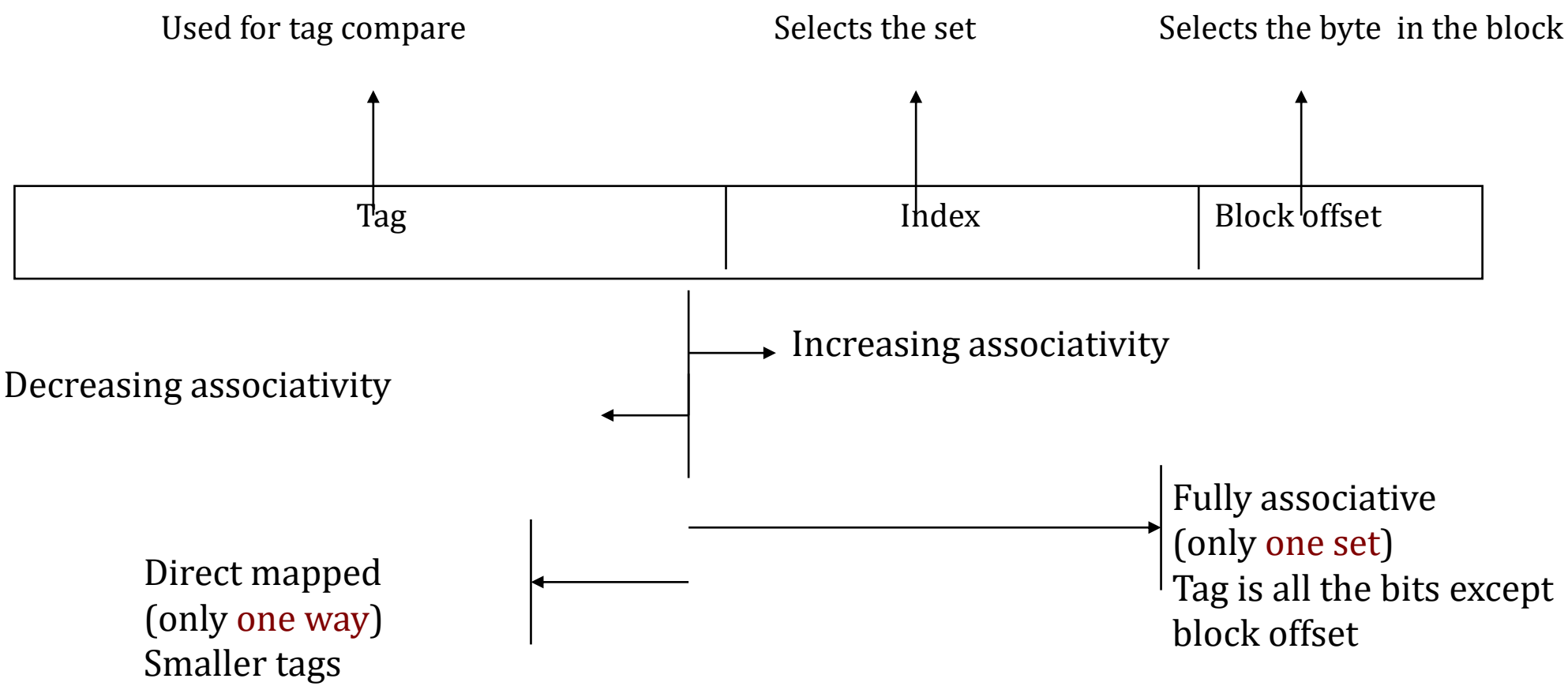
Fully Associative
anywhere

(2-way) Set Associative
anywhere in set 0
(12 mod 4)

Direct Mapped
only into block 4
(12 mod 8)

block 12 can be placed

Summary


Used for tag compare
Selects the set
Selects the byte in the block
Tag
Index
Block offset
Increasing associativity
Decreasing associativity
Fully associative
(only one set)
Tag is all the bits except
block offset
Direct mapped
(only one way)
Smaller tags

Kösz