Lecture 19: Cache refresher

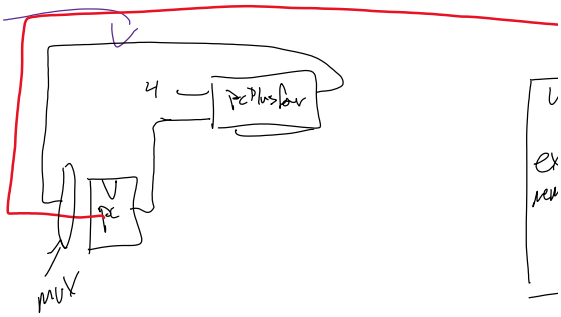
Friday, February 22, 2019 9:19 AM

Outline

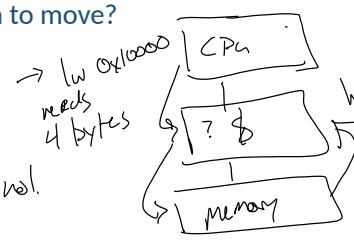
- Granularity of data movement
- Set associativity
- Cache examples

Chisel

pc := pPlusfour.io.result



How much data to move?



→ lw 0x10000 needs 4 bytes

Just address
↳ 1 byte val.

pc := ex_mem.next_pc

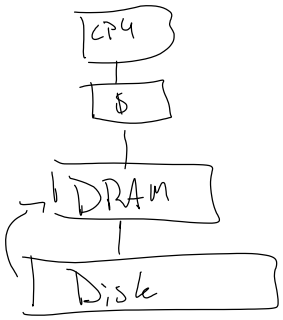
Minimum to move from memory is 4 bytes for lw

ld → 8 bytes

Might want more because spatial locality
64 bytes? → 1 miss + 15 hits
↑ fast

[Block / line size]

Block size << Cache size
power of two makes things easier



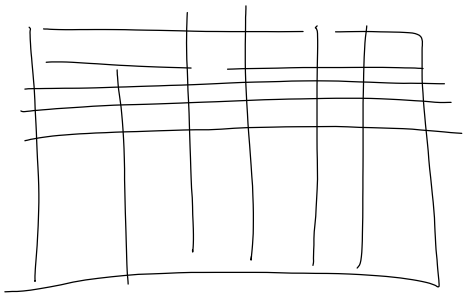
Depends on bus characteristics
↳ high latency bus → bigger block size
↳ high spatial locality

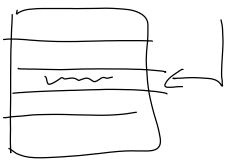
Small block sizes: fast transfers

Low spatial locality (tree or graph) → small block size
B-tree - 4kB

Block size depends on
Application characteristics → high or low spatial locality
Bus & technology → slow vs fast high vs low bw & granularity of access

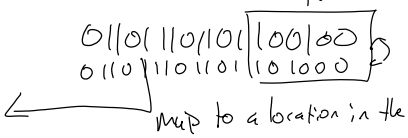
Where to put data?





Cache

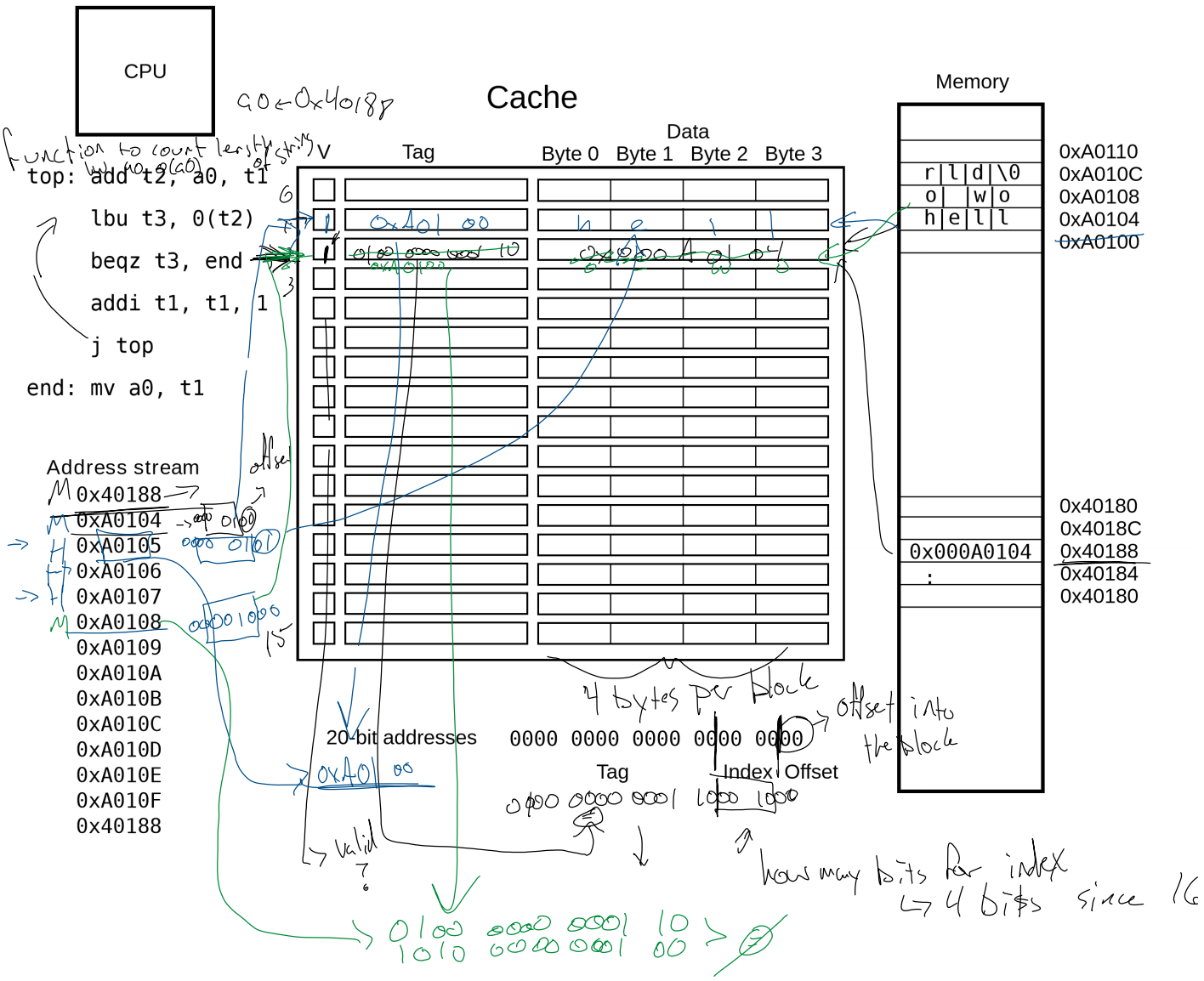
low-order bits

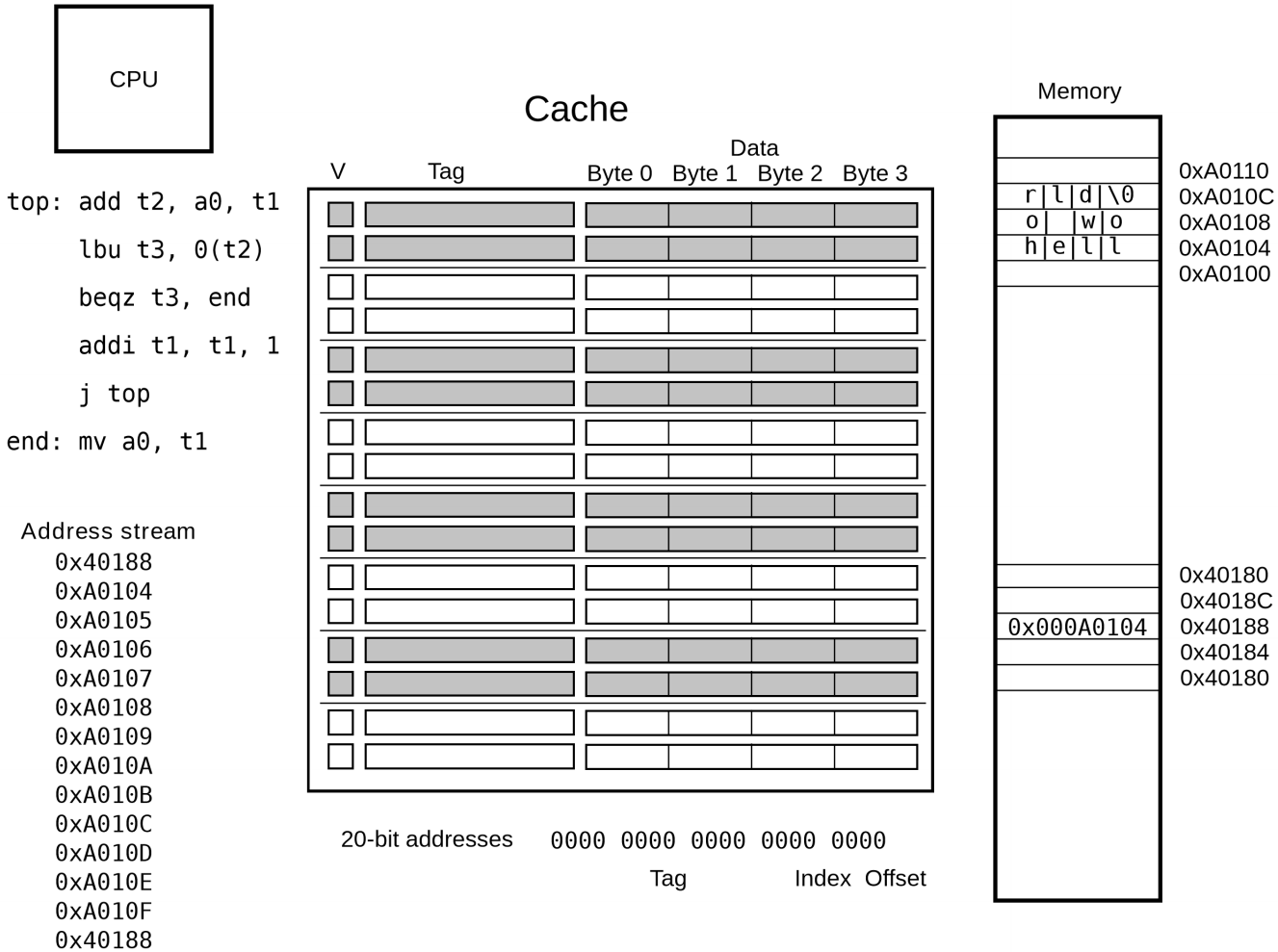


map to a location in the cache

"hash" (index)
↳ just use last few bits
least significant
↳ very simple

Cache Examples




CPU
Cache
Memory
V
Tag
Data
Byte 0
Byte 1
Byte 2
Byte 3
top: add t2, a0, t1
lbu t3, 0(t2)
beqz t3, end
addi t1, t1, 1
j top
end: mv a0, t1
Address stream
0x40188
0xA0104
0xA0105
0xA0106
0xA0107
0xA0108
0xA0109
0xA010A
0xA010B
0xA010C
0xA010D
0xA010E
0xA010F
0x40188
r|l|d|\0
o| |w|o
h|e|l|l
0x000A0104
0xA0110
0xA010C
0xA0108
0xA0104
0xA0100
0x40180
0x4018C
0x40188
0x40184
0x40180
20-bit addresses
0000 0000 0000 0000 0000
Tag
Index
Offset