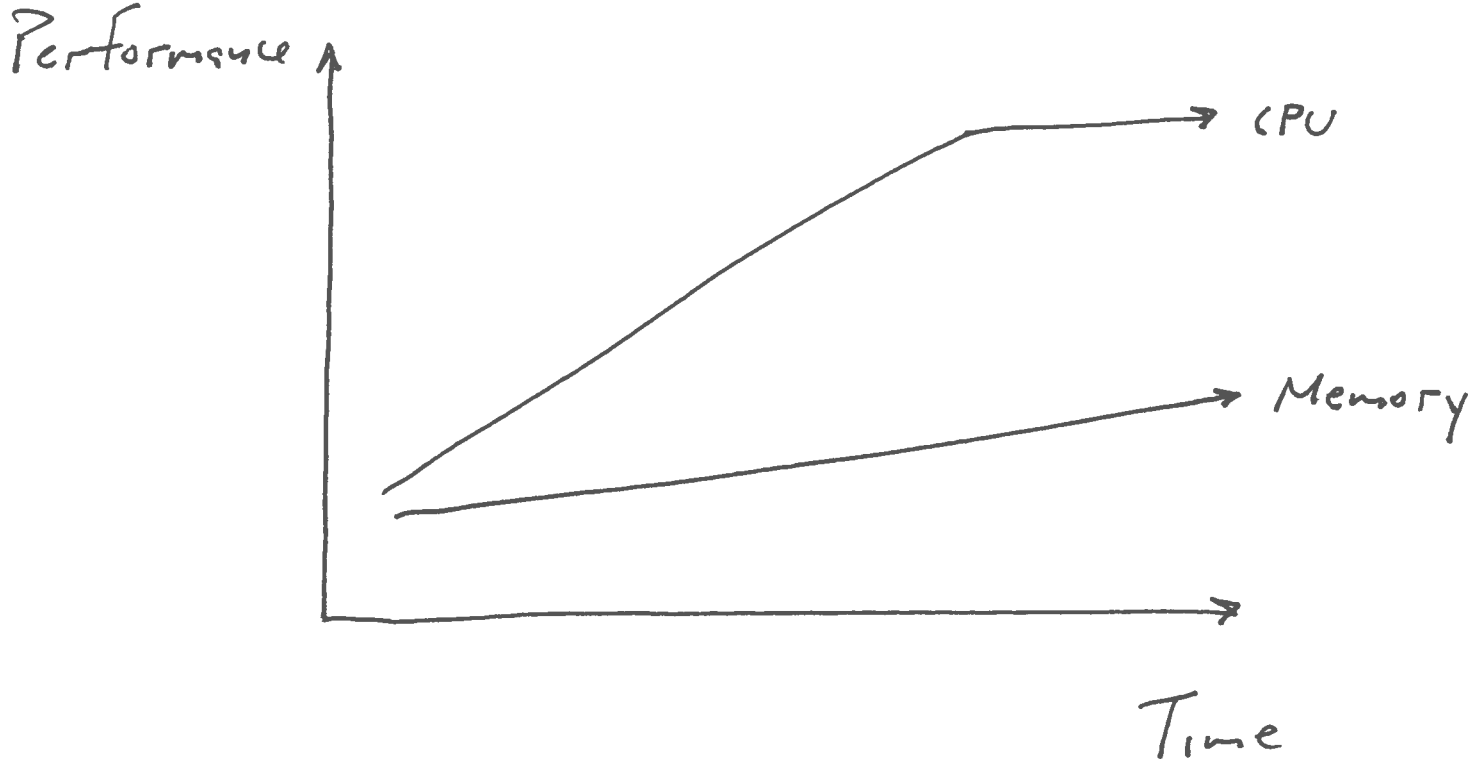
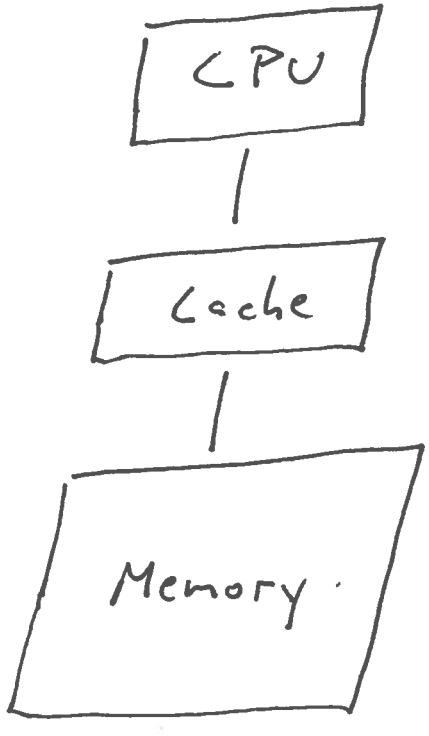
Memory Caches

CS520

Dept. of Computer Science
Univ. of New Hampshire


Performance
CPU
Memory
Time

Big Picture



locality

temporal

spatial

Details

cache is a sequence of sets

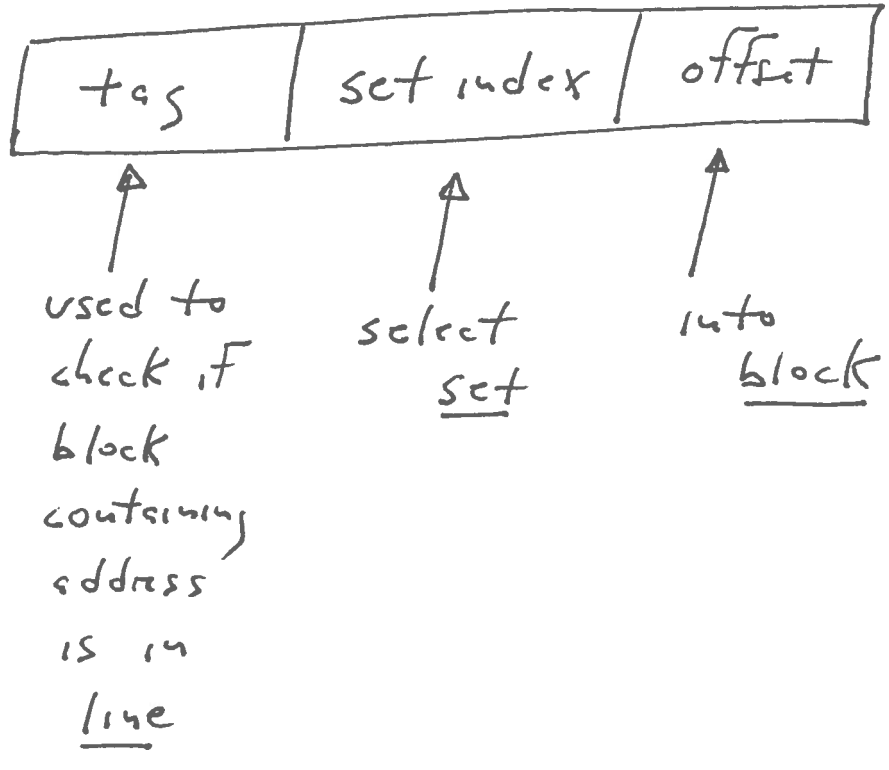
each set contains lines

each line is a block plus control information

each block contains contiguous bytes/words

From memory — spatial locality

address


tag
set index
offset
used to check if block containing address is in line
select set
into block

Control information in line

valid flag – does cache line contain a block?

tag – upper bits for addresses of words/bytes in the block in the line

dirty flag – has block been modified since it was loaded into cache? (used for "write-back" caches)

LRU stamp – used to implement/approximate Least Recently Used replacement policy

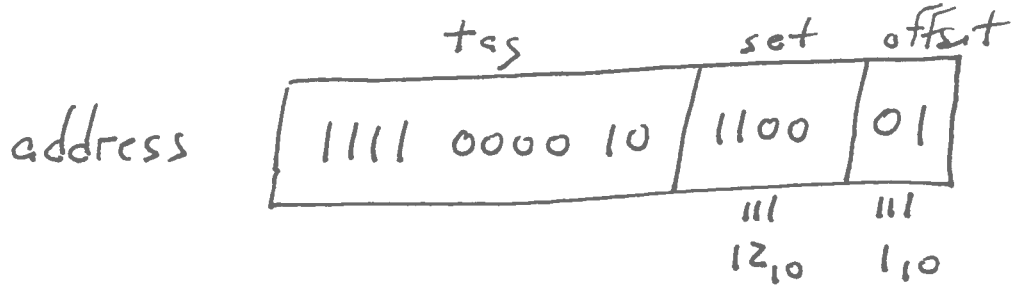
temporal

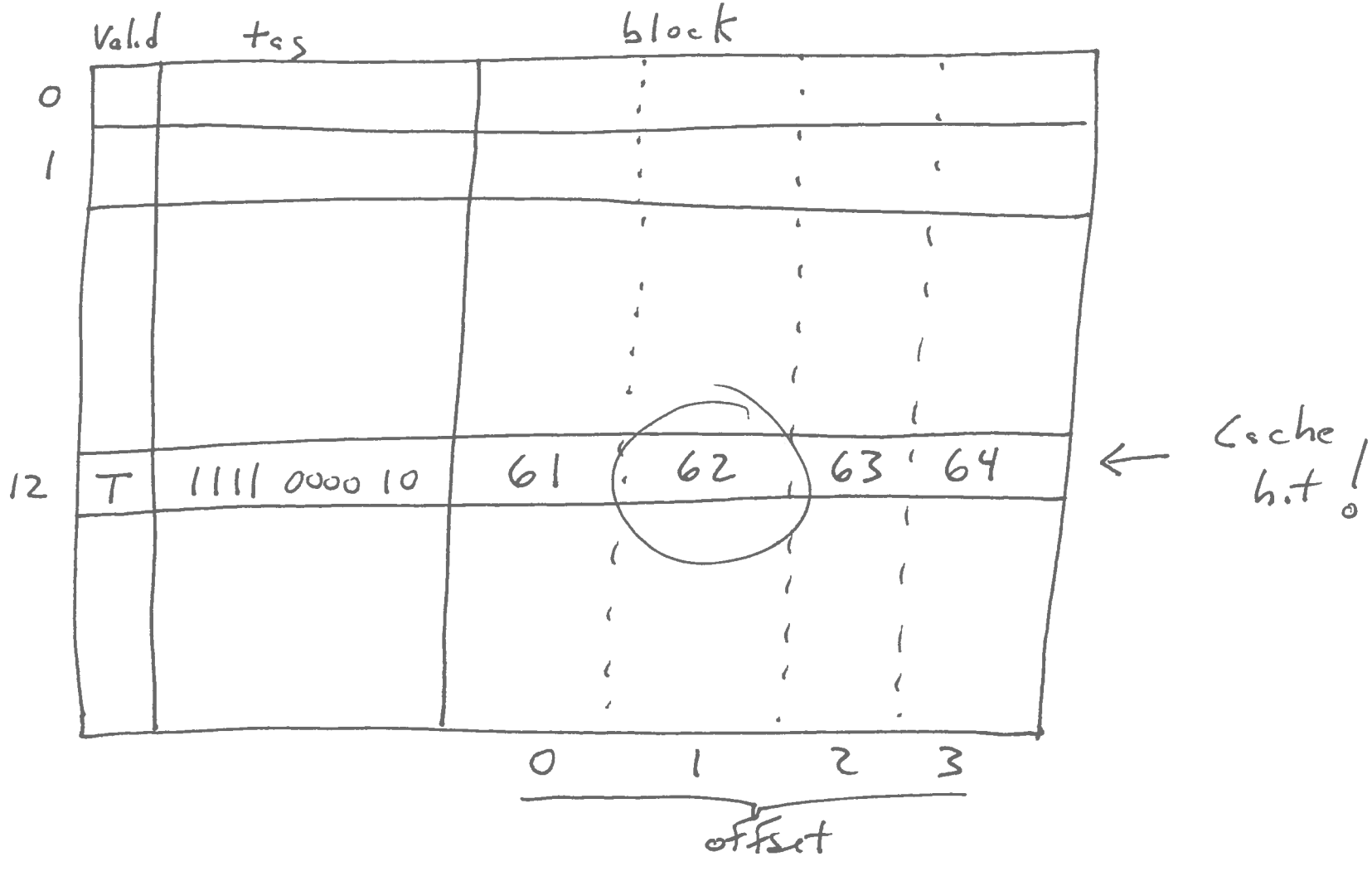
16-bit address

block size = 4 bytes

16 sets

1 line per set





note

memory is much larger then cache
so multiple blocks will map to the same set
use tag to distinguish

caches with one line per set are
known as direct-mapped

set-associative caches

multiple lines per set

must search lines in set to find matching tag

provides more flexibility when placing blocks into cache

Fully-associative caches

only 1 set

so blocks can be placed anywhere in cache

most flexible

search for tag can now be expensive

replacement policy

when bringing new block into cache,
will need to choose block to
be replaced if set is full

usually LRU or approximation of LRU
- timestamp
- or flag that is set when block is accessed and periodically cleared in all blocks in set

Writes

if not in cache already, do memory read first

then modify block in cache

write-through:

modify cache and write change to memory (need a buffer for efficiency)

write-back:

accumulate changes in block in cache

utilize "dirty" flag

at replacement time write block to memory if "dirty" flag set

What if multiple CPUs sharing memory?

with each CPU having its own cache

need a method to ensure consistency
across caches

e.g. write to a block in one cache
may cause invalidate of a copy
of the block in another cache