CS 354 - Machine Organization & Programming
Tuesday, October 22, 2019

Project p3 (6%): DUE at 10 pm on Monday, October 28th

Last Time

Memory Hierarchy
Locality
Bad Locality
Rethinking Addressing
Caching Basic Idea

Today

Caching Basic Idea (from last time)
Designing a Cache - Blocks
Designing a Cache - Sets and Tags
Basic Cache Lines
Basic Cache Operation
Basic Cache Practice
Direct Mapped Cache
Set Associative Cache

Next Time

Cache Performance and Coding Considerations
Read: B&O 6.4.5 - 6.4.7, 6.5 - 6.7

Designing a Cache - Blocks

❋ *The bits of an address* are used to look up if a block is in a cache.

How many bytes in an address space?

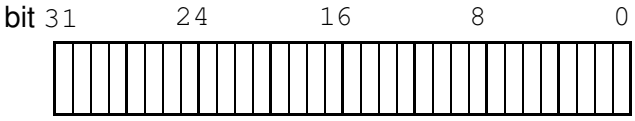
$M = 2^m$

$m = \log_2 M$

Let M be number of bytes in AS. IA-32 4GB.

Thus m is number of bits in Addr 32 bits.

32-bit Address Breakdown



How big is a block?

$B = 2^b$

$b = \log_2 B$

Let B be, number of byt per block 32 bytes. set by CPU Architecture.

b bits: bits in addr that identify which byte in block to access.

➢ What is the problem with using the most significant bits (left side) for the b bits?

❋ *Cache blocks must be big enough* to capture (nearby) spatial locality.
but small enough to minimize mem Latency.

How many 32-byte blocks of memory in a 32-bit address space?

$2^{(m-b)} = 2^{(32-5)} = 2^{27} = 128M$ Blocks.

❋ *The remaining bits of an address* are the blocks number but they're divided into 2 more parts used to access A. cache.

Designing a Cache - Sets & Tags

❋ *A cache must be searched* if an unrestrictive FP is used.

→ Problem? slow, worst case O(N) where N is number of locations.

Improvement? limited locs where block can be stored to a specific set.

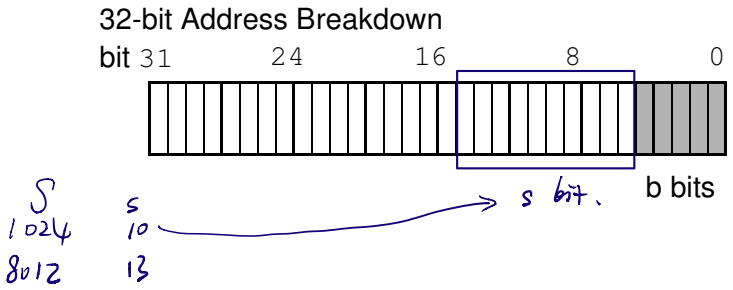
set: loc in cache where block is uniquely mapped.

How many sets in the cache?

$S = 2^s$

$s = \log_2 S$

Let S be number of sets in caches.



S | s
1024 | 10
8012 | 13

s bits: identify which set in cache to look in.

➢ What is the problem with using the most significant bits (left side) for the s bits?

→ How many blocks map to each set for a 32-bit AS and a cache with 1024 sets? 8192 sets?

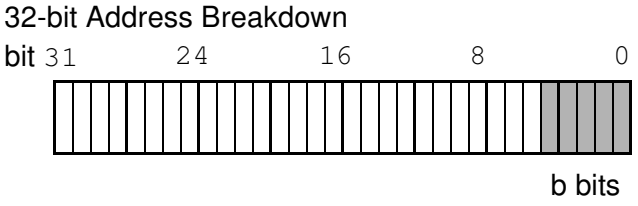
1024 $2^{(m-s-b)} = 2^{(32-10-5)} = 2^{17}$ = 128 K blocks

8192_2 16 k blocks

❋ *Different blocks of memory that map to same set* might result in collisions/ conflicts.

Since different blocks map to the same set how do we identify which block is in a set?

use remaining bits of address as a unique tag.



t bits: bits of an address that identify which block from Address space.

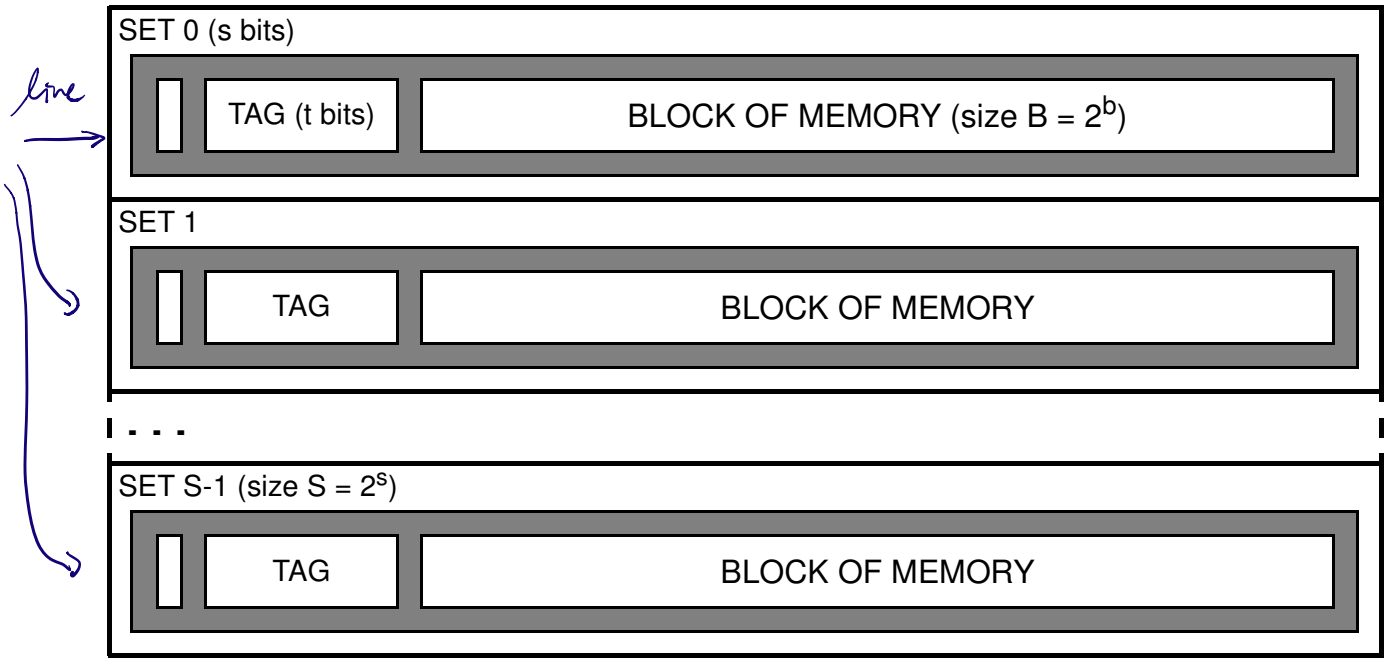
Basic Cache Lines

What? A *line* is

- one location in cache.
- composed of storage for one block and info needed for cache function.

❋ *In our basic cache each set* has only 1 line.

Basic Cache Diagram



→ How do you know if a line in the cache is used or not?

use a valid bit.

0 = invalid (Empty)

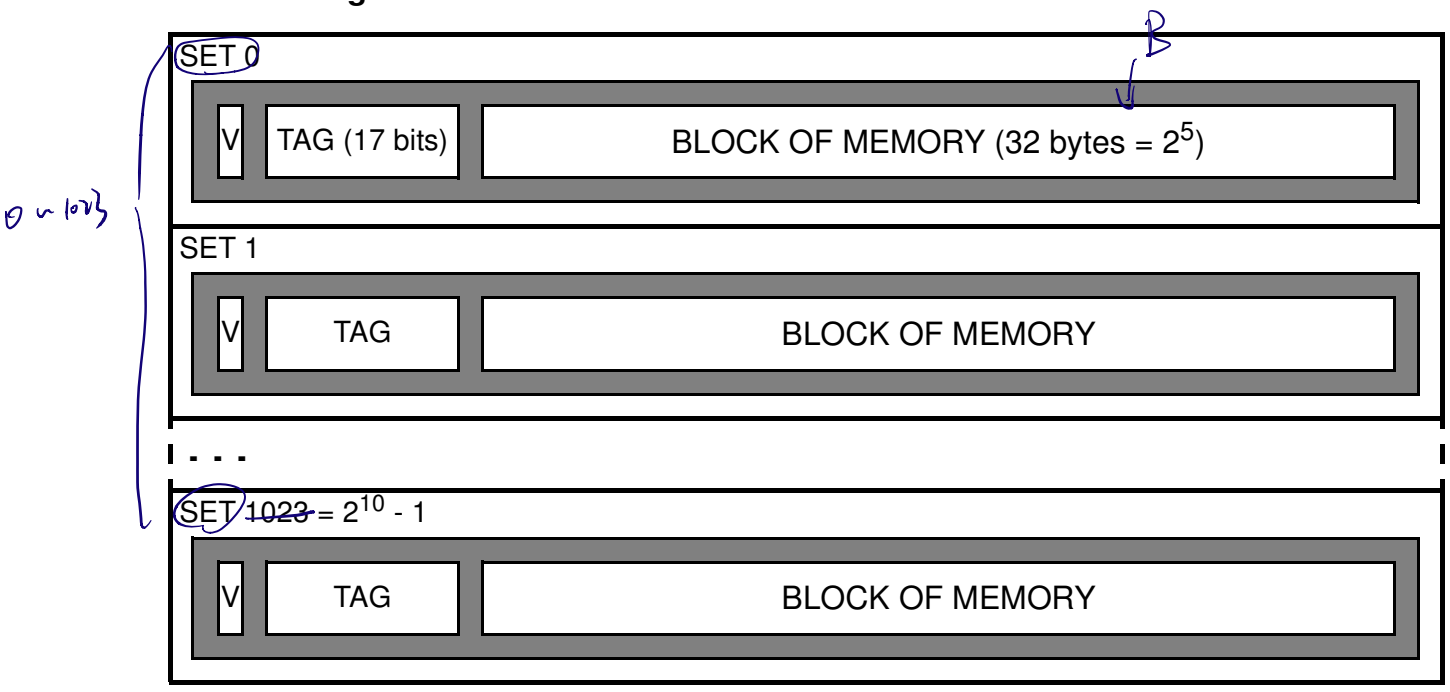
1 = valid (Full)

→ How big is a basic cache given S sets with blocks having B bytes?

S x B

Basic Cache Operation

Basic Cache Diagram



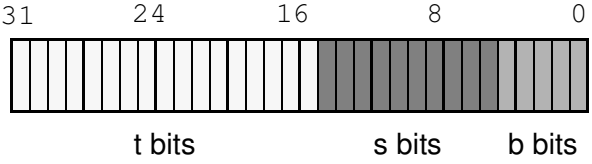
→ How big is this basic cache? 32 K Bytes.

How does a cache process a request for a word at a particular address?

1. Set Selection
   extract s bits to identify which set to check.
2. Line Matching
   extract t bits and compare with Lines' Tag
   if no match or valid bit is 0 Miss!
   Fetch from next lower level.

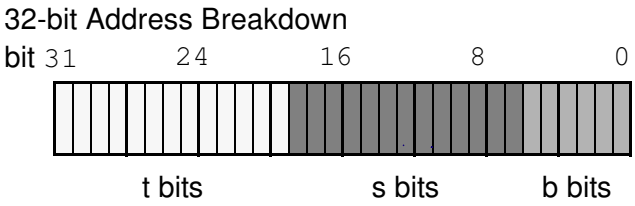
   if match and valid bit is 1 Hits Faster mem access
   For L1 cache only
3. word extraction.
   uses word offsets bits to identify which word in block is accessed.

32-bit Address Breakdown



Basic Cache Practice

You are given the following 32-bit address breakdown used by a cache:



→ How big are the blocks?

$B = 2^b = 64$

→ How many sets? $2^{13} = 8K$ sets.

→ How big is this basic cache?

$S \times B = 2^{13} \times 2^6 = 2^{19} = 512K$ bytes.

Assume the cache design above is given the following specific address: `0x07515E2B`

0 7 5 1 5 E 2 B

0000 0111 0101 0001 0101 1110 0010 1011

s b

word offset.

→ Which set should be checked given the address above?

Convert s bits to decimal

1400.

→ Which word in the block does the L1 cache access for the address?

Convert word offset bytes to decimal. 10.

➢ Which byte in the word does the address specify?

→ If a set had the following V bit and tag, does the address produce a hit or miss?

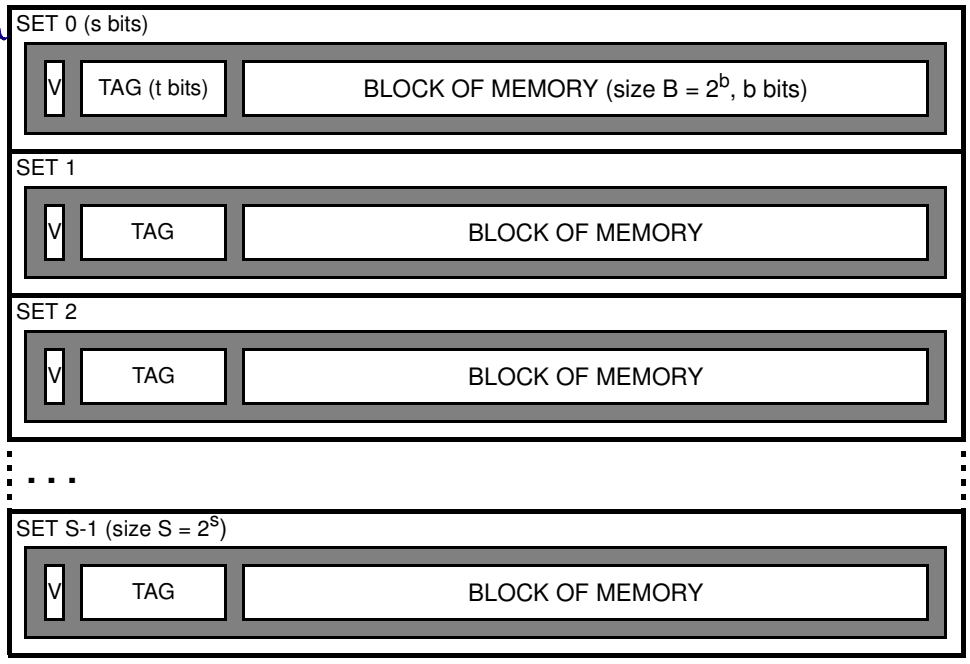
	V	tag	
1.)	1	0x0750	MISS
2.)	0	0x0750	MISS
3.)	1	0x00EA	HIT
4.)	0	0x00EA	MISS.

Direct Mapped Cache

Direct Mapped Cache
is a cache where each block maps to exactly 1 set and each set has exactly one line.

+ no need to search

+ Simple circuitry to match tag in line with t bits in addr.



→ What happens when two different memory blocks map to the same set?

– conflict misses and frequent since each set can store only one block.

→ Improvements?

enable each set to have more than one line to resolve conflict

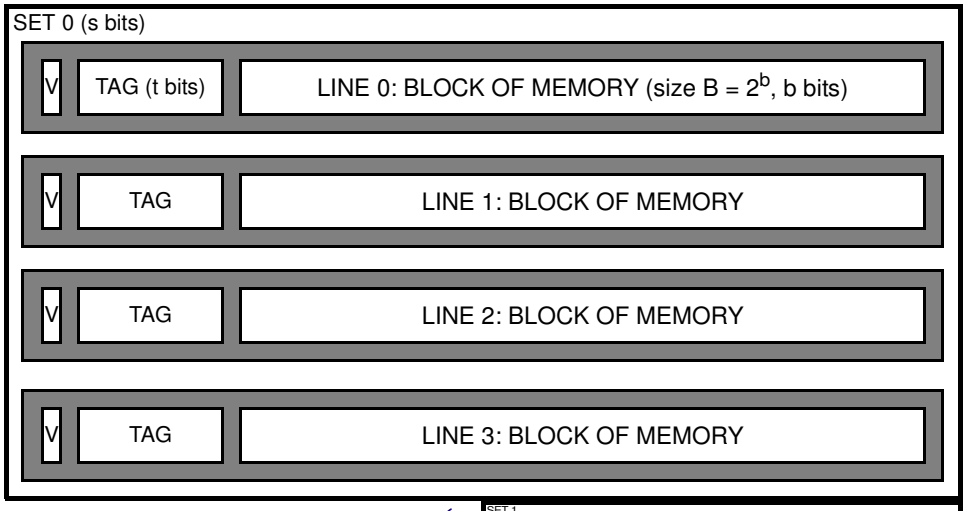
Set Associative Cache

Set Associative Cache

is a cache where each blocks maps to exactly one set.
and each set has 2 or more lines.

+ Reduces conflict misses.

- Requires complex to match tags with t bits.

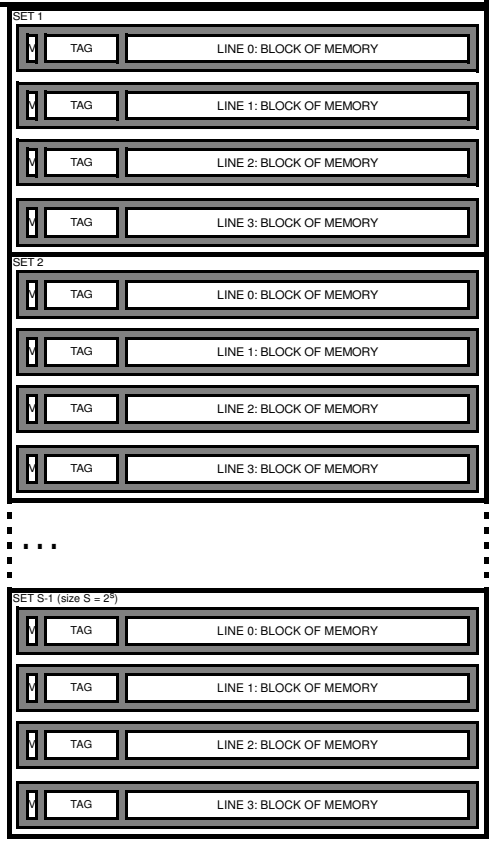


same 4 lines

Let E be the number of lines per set, which is the associativity of the cache.

E = 4 is a four-way set associative cache.

E = 1 is direct mapped cache.



Operation

1. Set Selection as usual
2. Line Matching Checks all tags in set simultaneously.

✵ ***C = (S, E, B, m)*** characterizes a cache design

Let C be the cache size in bytes. $C = S \times E \times B$

→ How big is a cache given (1024, 4, 32, 32)?

$1024 \times 4 \times 32 = 128$ KB.

1K