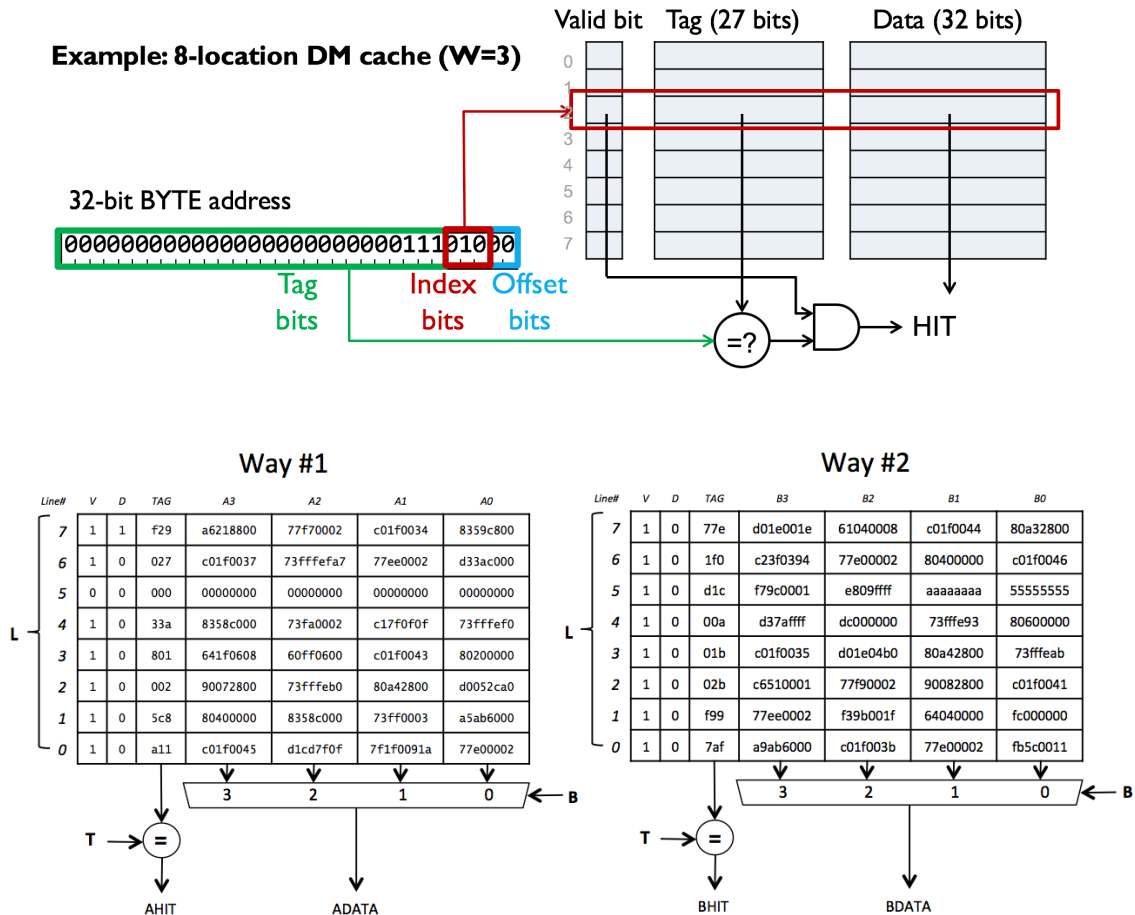# 6.004 Recitation Problems
L15 – Caches

Keep the most often-used data in a small, fast SRAM (often local to CPU chip). The reason this strategy works: LOCALITY.

– *Temporal locality: If a location has been accessed recently, it is likely to be accessed (reused) soon*
– *Spatial locality: If a location has been accessed recently,*
  it is likely that nearby locations will be accessed soon

AMAT(Average Memory Access Time) = HitTime + MissRatio * MissPenalty

Example: 8-location DM cache (W=3)

32-bit BYTE address

00000000000000000000000111 010 00

Tag bits   Index bits   Offset bits

Valid bit   Tag (27 bits)   Data (32 bits)

=?   HIT

Way #1

Line#	V	D	TAG	A3	A2	A1	A0
7	1	1	f29	a6218800	77f70002	c01f0034	8359c800
6	1	0	027	c01f0037	73fffefa7	77ee0002	d33ac000
5	0	0	000	00000000	00000000	00000000	00000000
4	1	0	33a	8358c000	73fa0002	c17f0f0f	73fffef0
3	1	0	801	641f0608	60ff0600	c01f0043	80200000
2	1	0	002	90072800	73fffeb0	80a42800	d0052ca0
1	1	0	5c8	80400000	8358c000	73ff0003	a5ab6000
0	1	0	a11	c01f0045	d1cd7f0f	7f1f0091a	77e00002

3 2 1 0 ← B

T → =

AHIT   ADATA

Way #2

Line#	V	D	TAG	B3	B2	B1	B0
7	1	0	77e	d01e001e	61040008	c01f0044	80a32800
6	1	0	1f0	c23f0394	77e00002	80400000	c01f0046
5	1	0	d1c	f79c0001	e809ffff	aaaaaaaa	55555555
4	1	0	00a	d37affff	dc000000	73fffe93	80600000
3	1	0	01b	c01f0035	d01e04b0	80a42800	73fffeab
2	1	0	02b	c6510001	77f90002	90082800	c01f0041
1	1	0	f99	77ee0002	f39b001f	64040000	fc000000
0	1	0	7af	a9ab6000	c01f003b	77e00002	fb5c0011

3 2 1 0 ← B

T → =

BHIT   BDATA

Replacement strategy choices:

- Oracle, OPT (optimal)
- RANDOM
    o evict a random way
- FIFO (first-in, first-out)
    o give every way equal residency
- LRU (least-recently used)
   o The one used least recently will get evicted first

Write-policy choices:

- Write-through (write data go to cache and memory)
  – Update the main memory as well as the cache on every write
  – Replacing a cache entry is simple (just overwrite new block)
  – Memory write causes significant delay

- Write-back (write data only goes to the cache)
  – Only the cache entry is updated on each cache write so main memory and the cache data are inconsistent.
  – Add "dirty" bit to the cache entry to indicate whether the data in the cache entry must be committed to memory.
  – Replacing a cache entry requires writing the data back to memory before replacing the entry if it is "dirty".

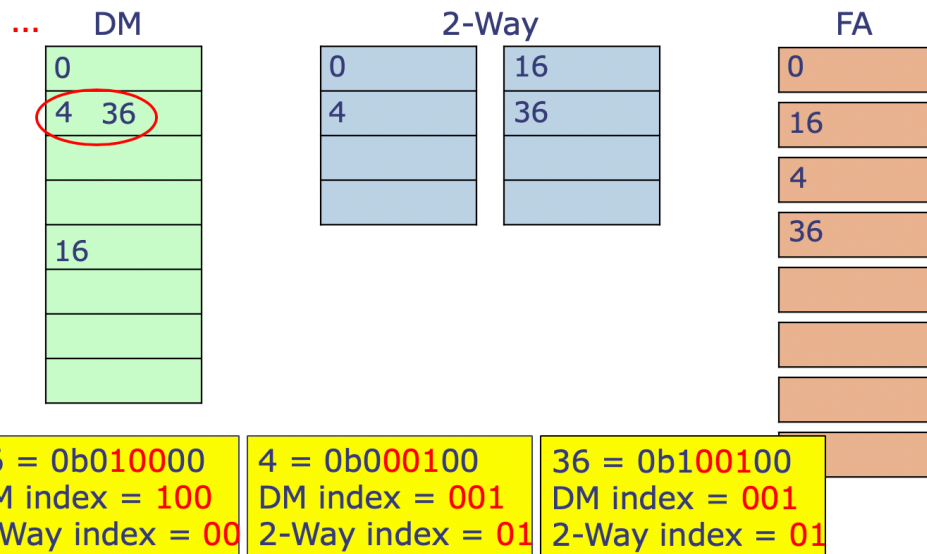(credit: Stanford EE108b lecture slides)

Example: Comparing Hit Rates

- 3 Caches: DM, 2-Way, FA: each has 8 (4 byte) words
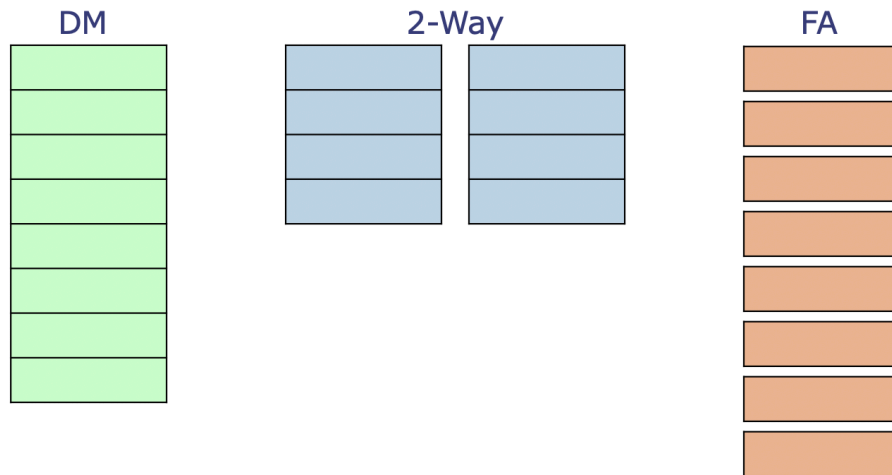
DM	2-Way	FA

Example: Comparing Hit Rates

Access following addresses repeatedly: 0, 16, 4, 36, …

DM	2-Way		FA
0	0	16	0
4 36	4	36	16
			4
			36
16			

16 = 0b010000	4 = 0b000100	36 = 0b100100
DM index = 100	DM index = 001	DM index = 001
2-Way index = 00	2-Way index = 01	2-Way index = 01

Example: Comparing Hit Rates

Access following addresses repeatedly: 0, 16, 4, 36,
…

DM		2-Way			FA
0		0		16	0
4, 36		4		36	16
					4
					36
16					

DM: 50% hit rate
2-Way: 100% hit rate
FA: 100% hit rate

Example 2: Comparing Hit Rates

▪ Access: 0, 4, 8, 12, 16, 20, 24, 28, 32, …

DM		2-Way			FA	
0, 32		0, 32, 16		16, 0, 32	0	32
4		4		20	4	0
8		8		24	8	4
12		12		28	12	8
16					16	12
20					20	16
24					24	20
28					28	24

DM: Hit rate = 7/9   2-Way: Hit rate = 6/9   FA: Hit rate = 0%

Example 3: Comparing Hit Rates

- Access: 0, 4, 8, 12, 32, 36, 40, 44, 16, …

DM	2-Way		FA
0, 32	0, 16, 32	32, 0, 16	0 16
4, 36	4	36	4 0
8, 40	8	40	8 4
12, 44	12	44	12 8
16			32 12
			36 32
			40 36
			44 40

DM: Hit rate = 1/9   2-Way: Hit rate = 6/9   FA: Hit rate = 0%

Problem 1.

(A) The timing for a particular cache is as follows: checking the cache takes 1 cycle. If there's a hit the data is returned to the CPU at the end of the first cycle. If there's a miss, it takes 10 *additional* cycles to retrieve the word from main memory, store it in the cache, and return it to the CPU. If we want an average memory access time of 1.4 cycles, what is the minimum possible value for the cache's hit ratio?

Minimum possible value of hit ratio: _____

(B) If the cache block size, i.e., words/cache line, is doubled but the total number of data words in the cache is unchanged, how will the following cache parameters change? Please circle the best answer.

# of offset bits: UNCHANGED … +1 … -1 … 2x … 0.5x … CAN'T TELL

# of tag bits: UNCHANGED … +1 … -1 … 2x … 0.5x … CAN'T TELL

# of cache lines: UNCHANGED … +1 … -1 … 2x … 0.5x … CAN'T TELL

Consider a direct-mapped cache with 64 total data words with 1 word/cache line, which uses a LRU replacement strategy and a write-back write strategy. This cache architecture is used for parts (C) through (F).

(C) If cache line number 5 is valid and its tag field has the value 0x1234, what is the address in main memory of the data word currently residing in cache line 5?

Main memory address of data word in cache line 5: 0x_____

The program shown on the right repeatedly executes an inner loop that sums the 16 elements of an array that is stored starting in location 0x310.

The program is executed for many iterations, then a measurement of the cache statistics is made during one iteration through all the code, i.e., starting with the execution of the instruction labeled `outer_loop:` until just before the next time that instruction is executed.

```
    . = 0                // tell assembler to start at
                         // address 0
outer_loop:
  addi x4, x0, 16   // initialize loop index J
  mv x1, x0         // x1 holds sum, initially 0

loop:               // add up elements in array
  subi x4, x4, 1    // decrement index
  slli x2, x4, 2    // convert to byte offset
  lw x3, 0x310(x2)  // load value from A[J]
  add x1, x1, x3    // add to sum
  bne x4, x0, loop  // loop until all words are summed

  j outer_loop      // perform test again!
```

(D) In total, how many instruction fetches occur during one complete iteration of the outer loop? How many data reads?

Number of instruction fetches: _____

Number of data reads: _____

(E) How many instruction fetch misses occur during one complete iteration of the outer loop? How many data read misses? Hint: remember that the array starts at address 0x310.

Number of instruction fetch misses: _____

Number of data read misses: _____

(F)   What is the hit ratio measured after one complete iteration of the outer loop?

Hit ratio: _____

Problem 2.

The RISC-V Engineering Team is working on the design of a cache. They've decided that the cache will have **a total of $2^{10} = 1024$ data words**, but are still thinking about the other aspects of the cache architecture.

First assume the team chooses to build a direct-mapped write-back cache with a block size of 4 words.

(A) Please answer the following questions:

Number of lines in the cache: _____

Number of bits in the tag field for each cache entry: _____

(B) This cache takes *2 clock cycles* to determine if a memory access is a hit or a miss and, if it's a hit, return data to the processor. If the access is a miss, the cache takes *20 additional clock cycles* to fill the cache line and return the requested word to the processor. If the hit rate is 90%, what is the processor's average memory access time in clock cycles?

Average memory access time assuming 90% hit rate (clock cycles): _____

Now assume the team chooses to build a 2-way set-associative write-back cache with a block size of 4 words. *The total number of data words in the entire cache is still 1024.* The cache uses a LRU replacement strategy.

(C) Please answer the following questions:

Address bits used as offset (including byte offset):   A[_____ : _____]

Address bits used as cache line index: A[_____ : _____]

Address bits used for tag comparison: A[_____ : _____]

(D) To implement the LRU replacement strategy this cache requires some additional state for each set. How many state bits are required for each set?

Number of state bits needed for each set for LRU: _____

To test this set-associative cache, the team runs the benchmark code shown on the right. The code sums the elements of a 16-element array. The first instruction of the code is at location 0x0 and the first element of the array is at location 0x10000. Assume that the cache is empty when execution starts and remember *the cache has a block size of 4 words*.

```
. = 0x0
   mv x3, x0 // index
   mv x1, x0 // sum
   // x4 = 0x10000
   lui x4, 0x10

L: add x5, x4, x3
   lw x2, 0(x5)
   add x1, x1, x2
   addi x3, x3, 4
   slti x2, x3, 64
   bnez x2, L
   unimp    // halt

. = 0x10000
A: .word 0x1
   .word 0x2
   …
   .word 0xF
   .word 0x10
```

(E) How many instruction misses will occur when running the benchmark?

Number of instruction misses when running the benchmark: _____

(F) How many data misses (i.e., misses caused by the memory access from the LD instruction) will occur when running the benchmark?

Number of data misses when running the benchmark: _____

(g) What's the exact hit rate when the complete benchmark is executed?

Benchmark hit rate: _____

Problem 3.

Assume, the program shown on the right is being run on a RISC-V processor with a cache with the following parameters:

- **2-way set-associative**
- **block size of 2**, i.e., 2 data words are stored in each cache line
- total number of data words in the cache is **32**
- **LRU** replacement strategy

(A) The cache will divide the 32-bit address supplied by the processor into three fields: B bits of block offset (including byte offset bits), L bits of cache line index, and T bits of tag field.   Based on the cache parameters given above, what are the appropriate values for B, L, and T?

<div align="center">

value for B: _____

value for L: _____

value for T: _____

</div>

```
. = 0x240           // start of program
test:
  addi x4, x0, 16 // initialize loop index J
                  // to size of array
  mv x1, x0       // x1: sum

loop:               // add up elements in array
  subi x4, x4, 1  // decrement index
  slli x2, x4, 2  // convert to byte offset
  lw x3, 0x420(x2)// load value from A[J]
  add x1, x1, x3  // add to sum
  bnez x4, loop   // loop N times

  j test          // perform test again!

// allocate space to hold array
. = 0x420
A: .word A[0]
   .word A[1]
   …
```

(B) If the SLLI instruction is resident in a cache line, what will be its cache line index? the value of the tag field for the cache?

<div align="center">

Cache line index for SLLI when resident in cache: _____

Tag field for SLLI when resident in cache: 0x_____

</div>

(C) Given that the code begins at address 0x240 and the array begins at address 0x420, and that there are 16 elements in the array as shown in the code above, list *all* the values j (0 ≤ j < 16) where the location holding the value A[j] will map to the same cache line index as the SLLI instruction in the program.

<div align="center">

List all j where A[j] have the same cache line index as SLLI: _____

</div>

(D) If the outer loop is run many times, give the steady-state hit ratio for the cache, i.e., assume that the number of compulsory misses as the cache is first filled are insignificant compared to the number of hits and misses during execution.

<div align="center">

Steady-state hit ratio (%): _____

</div>