CS M151B: Homework 8

Jonathan Woong

804205763

Spring 2017

Discussion 1A

Wednesday 31st May, 2017

Problem 1. Compute the effective access time, in seconds, of a memory system consisting of a cache and main memory with a processor running at 2 GHz. The cache hit time is one cycle. The miss penalty is 45 cycles. The hit rate is 93%.

$$\boxed{\text{AMAT} = \text{time}_{\text{hit}} + \text{ miss rate } \times \text{ miss penalty} = 1 + (1 - 0.93) \times 45 = 4.15 \text{ clock cycles}}$$

Problem 2. Compute the **total** number of bits of storage required to implement a cache that is similar to the cache shown in Figure 5.18 on page 408, except that the associativity of the cache is 16 instead of 4. All the other parameters (cache size, block size) are the same as in Figure 5.18. Note that the number you are being asked to compute is different from the "size of the cache," which refers only to the number of bits of data stored in the cache.

1024 blocks per set

2 sets

6 bit index

24 bit tag

1 bit verify

32 bit data

total # of bits = (verify + tag + data)×(# sets)×(# blocks per set) = (1+24+32)×2×1024 = $\boxed{116736 \text{ bits}}$

Problem 3. A CPU generates 50 bit addresses. The memory system is word-addressable. The cache contains 8192 block frames. The tag size is 34 bits. Based on this information, what is the block size? If a range of block sizes is possible, specify this range.

Direct-mapped:

$\log_2(8192) = 13$ bit index

$34 + 13 = 47$ bits for tag and index

$50 - 47 = 3$ bits $\Rightarrow 2^3 = 8$ words per block Fully associative:

$2^1 6$ words

Range: $\boxed{(8, 65536) \text{ words}}$

Problem 4. 5.2.2: For each of these references, identify the binary address, the tag, and the index given a direct-mapped chace with two-word blocks and a total size of 8 blocks. Also list if each reference is a hit or a miss, assuming the cache is initially empty. In addition, show the final contents of the cache. This means that, for each block frame that contains a valid block, show the block number, the word address

of the words in each block, and the tag of that block.

Address	**Binary Address**	**Tag**	**Index**	**Hit/Miss**
3_{10}	00000011_2	00000	001	M
180_{10}	10110100_2	10110	010	M
43_{10}	00101011_2	00101	010	M
2_{10}	00000010_2	00000	010	M
191_{10}	10111111_2	10111	111	M
88_{10}	01011000_2	01011	100	M
190_{10}	10111110_2	10111	111	H
14_{10}	00001110_2	00001	111	M
181_{10}	10110101_2	10110	010	H
44_{10}	00101100_2	00101	110	H
186_{10}	10111010_2	10111	101	M
253_{10}	11111101_2	11111	110	M

Block number	Word Addresses	Tag
126_{10}	$252_{10}, 253_{10}$	11111
93_{10}	$186_{10}, 187_{10}$	10111
22_{10}	$44_{10}, 45_{10}$	00101
90_{10}	$180_{10}, 181_{10}$	10110
7_{10}	$14_{10}, 15_{10}$	00001
95_{10}	$190_{10}, 191_{10}$	10111
44_{10}	$88_{10}, 89_{10}$	01011
1_{10}	$2_{10}, 3_{10}$	00000

Problem 5. 5.5.1: Assume a 64 KiB direct-mapped cache with a 32-byte block. What is the miss rate for the address stream above? How is this miss rate sensitive to the size of the cache or the working set? How would you categorize the misses this workload is experienceing, based on the 3C model?

$\frac{1}{8} = 0.125 \Rightarrow 12.5\%$ miss rate does not change with cache size or working set = cold misses

Problem 6. 5.5.2: Re-compute the miss rate when the cache block size is 16 bytes, 64 bytes, and 128 bytes. What

kind of locality is this workload exploiting?

Cache block size	miss rate
16 bytes	$\frac{1}{4} = 25\%$
64 bytes	$\frac{1}{16} = 6.25\%$
128 bytes	$\frac{1}{32} = 3.125\%$

Spacial locality

Problem 7. 5.5.3: "Prefetching" is a technique that leverages predictable address patterns to speculatively bring in additional cache blocks when a particular cache block is accessed. Assume a two-entry stream buffer and assume that the cache latency is such that a cache block can be loaded before the computation on the previous cache block is completed. What is the miss rate for the address stream above?

Nearly 0%

Problem 8. The minimum addressable unit of a memory system is a 32-bit word. The memory system includes a four-way set-associative cache. The block size is eight words. The **data** part of the cache is implemented using a single conventional 64 $K \times 32$ RAM chip. We will call this chip C1. The connections to C1 are shown on page 10.24 in the notes. You cannot modify C1 in any way. For each address generated by the CPU, the cache controller, using the directory part of the cache, maps that address to an address in C1.

A) The CPU generates the address 0x321. The access is a **hit** in the cache. Specify the address in C1 where the word is found. If more than one answer is possible for your assumed implementation of the cache controller and directory, you must specify all possible answers. Addresses must be specified in hex.

$$\frac{64K \text{ words}}{32 \text{ words per set}} = 2048 \text{ set frames} \Rightarrow 11 \text{ bit index}$$

Possible answer 1:

tag	**index**	**block offset**
18	11	3

Possible answer 2:

index	**set offset**	**block offset**
11	2	3

Possible answer 3:

set offset	**index**	**block offset**
2	11	3

Assume possibility 1:

tag	**index**	**block offset**
...	00001100100	001

Possible locations in C1:

binary	hex
0000110010000001	0x0C81
0000110010001001	0x0C89
0000110010010001	0x0C91
0000110010011001	0x0C99

B) Repeat part A for address 0x9876543.

tag	index	block offset
...	10010101000	011

Possible locations in C1:

binary	hex
1001010100000011	0x9503
1001010100001011	0x950B
1001010100010011	0x9513
1001010100011011	0x951B

Problem 9. Consider the multicycle MIPS implementation described in Figure 5.28 and 5.37 in Chapter 5, 3rd Ed. This processor executes a program will the following instruction mix:

loads	25%
stores	15%
R-format	40%
branches	18%
jumps	2%

Unlike Figure 5.28, the memory subsystem of the system you are considering consists of a **write-back, write-around** unified instruction/address cache and main memory. The access time of the cache is 1 cycle. The access time to main memory is 5 cycles. The block size is 8 bytes (2 words). The main memory word size is 32-bits. On average, 25% of the blocks are dirty. There is no write buffer. State every assumption and be sure that it is clear how you arrive at your answer.

A) Compute the CPI of the system assuming the cache hit rate is 100%.

$$\text{CPI} = 0.25(5) + 0.15(4) + 0.40(4) + 0.18(3) + 0.02(3) = \boxed{4.05}$$

B) For what cache hit rate will performance decrease (i.e., execution time increase) by a factor of 2. Assume the cache can hold both data and instructions.

$$\frac{\text{CPI}_{\text{miss}}}{\text{CPI}_{\text{100\% hit}}} = 2$$

$$\text{time}_{\text{copy block}} = 5 \text{ cycles/word} \times 2 \text{ words/block} = 10 \text{ cycles/block}$$

75% of misses have a clean victim block with 10 cycles to read the block from memory. 25% of misses have a dirty victim block with 10 cycles to write the victim block to memory and 10 cycles to read the block from memory.

$$\text{CPI}_{\text{I-miss}} = (1 - \text{cache}_{\text{hit}}) \times (0.65 \times 10 + 0.35 \times 20) = 13.5 \times (1 - \text{cache}_{\text{hit}})$$

$$\text{CPI}_{\text{load-miss}} = 0.25 \times (1 - \text{cache}_{\text{hit}}) \times (0.65 \times 10 + 0.35 \times 20) = 3.375 \times (1 - \text{cache}_{\text{hit}})$$

$$\text{CPI}_{\text{store-miss}} = 0.15 \times (1 - \text{cache}_{\text{hit}}) \times 5 = 0.75 \times (1 - \text{cache}_{\text{hit}})$$

$$\text{CPI}_{\text{miss}} = 4.05 + 13.5 \times (1 - \text{cache}_{\text{hit}}) + 3.375 \times (1 - \text{cache}_{\text{hit}}) + 0.75 \times (1 - \text{cache}_{\text{hit}}) = 21.675 - 17.625 \times \text{cache}_{\text{hit}}$$

$$\frac{\text{CPI}_{\text{miss}}}{\text{CPI}_{\text{100\% hit}}} = \frac{21.675 - 17.625 \times \text{cache}_{\text{hit}}}{4.05} = 2$$

$$\text{cache}_{\text{hit}} = 0.7702 \Rightarrow \boxed{77\%}$$