Com S 321 Problem Set 3

1. A computer has a main memory of size 8M words and a cache size of 64K words.

(a) Give the address format for a direct mapped cache with a block size of 32 words.

(b) Give the address format for a fully associative cache with a block size of 2^m words.

(c) Give the address format for a set associative cache with a block size of 64 words and a set size of 4 (i.e., 4 block frames per set).

2. Suppose you have a word addressed memory hierarchy system with the following parameters:

Block size = 16 words
Main memory size = 64 blocks
Cache size = 8 blocks
Block placement policy: direct mapping

The tag values in the cache directory are:

Tag	Cache Block Frame #
000	0
101	1
100	2
010	3
101	4
011	5
100	6
110	7

(a) Give the address format for the memory system.

(b) Will main memory address 37A (hex) be a cache hit? Explain your answer.

(c) Will main memory address 22C (hex) be a cache hit? Explain your answer.

(d) Will main memory address 1B9 (hex) be a cache hit? Explain your answer.

3. Suppose your cache from Problem 2 is set associative with a set size of 2 (i.e., 2 block frames per set). Assume the following tag values:

Tag	Cache Block Frame #	Set #
0000	0	0
0100	1	0
1000	2	1
1001	3	1
1100	4	2
1000	5	2
0110	6	3
1101	7	3

Do parts (a) through (d) given in Problem 2.

4. Suppose you have a cached computer system with 1M words in main memory and a cache size of 4K words. Assume main memory is divided into blocks with each block containing 16 words. Assume word addressing.

(a) If the cache uses direct mapping, what cache frame does the physical (hex) address 0x949DA map into? What value needs to be in that cache frame's tag to get a cache hit?

(b) Assume a set associative cache with 64 sets of 4 block frames per set. What set does the physical (hex) address 0x949DA map into? How many bits are there in each cache tag? What is the cache tag (in hex) for this physical address?

(c) If the cache is fully associative, what is the cache tag for the physical (hex) address 0x949DA?

5. Suppose you have a cached computer system with a main memory size of 32K words and a cache size of 4K words. The cache is fully associative with a block size of 8 words.
Assume word addressing.

(a) How many hardware comparators are needed to support the fully associative cache?

(b) What is the size of the tag field?

(c) How many hardware comparators would be needed to support a direct mapped cache? What would be the size of the tag field with a direct mapped cache?

6. Suppose you have a word-addressable cache memory system with the following parameters:

8K **blocks** in main memory
512 **block frames** in cache memory
8 words in a **block**

(a) How many bits are there in a main memory address?

(b) Given the memory address 0x**CA49** (hex), give the 3-bit offset of the word within the block that is being referenced.

(c) In a direct mapped cache, give the cache block frame number to which the above referenced address is mapped to. Give your answer in decimal. Give the number and width of hardware comparators that will be needed to determine if the reference is a hit.

(d) In a set associative cache with four block frames per set, give the set number to which the above referenced address is mapped to. Give your answer in decimal. Give the number and width of hardware comparators that are needed.

(e) In a fully associative cache, how many bits are in each cache tag? Give the number and width of hardware comparators that are needed.

7. Consider a cached memory system with t_m = 300ns and t_c = 50ns. Suppose that, on the average, an instruction requires 1.3 memory accesses. The typical instruction execution time exclusive of memory accesses is 175ns.

(a) Calculate the average instruction execution time (i.e., execution + memory access) with no cache.

(b) Calculate the average instruction execution time with each of the following cache hit ratios (assume a read through policy, that is, upon a cache miss the data is directly supplied to the CPU as well as the cache) :

h = 0.90
h = 0.95
h = 0.98

8. Suppose you are the head of a computer architecture group that has produced a computer system whose main memory has an access time of 250ns. Careful statistical sampling has indicated that each instruction requires an average of 2 memory accesses and has an average execution time (exclusive of memory accesses) of 150ns. Three members of your group each present proposals to speed up this system and their proposals are given below. Rank these proposals assuming each costs the same. Justify your ranking.

(a) Use a main memory that is twice as fast.

(b) Add enough CPU registers to reduce memory accesses to an average of 1.5 memory accesses per instruction. Assume t_m = 250ns.

(c) Add a cache to the system with t_c = 50ns and a hit ratio h = 0.9. Assume that upon a cache miss, a word is first transferred from main memory to cache and then from cache to the CPU. Assume t_m = 250 ns and an average of 2 memory accesses per instruction.

9. Suppose you have a word addressable cached memory system with the following parameters:

t_m = 1000ns	t_c = 100ns
h = 0.90	Block size = 8 words

(a) Calculate the effective memory access time, if upon a cache miss the word is directly read from main memory (i.e., there is a read through policy implemented).

(b) Calculate the effective memory access time with no read through policy. Upon a cache miss, a block is first written to main memory (one word at a time), then the desired block is read from main memory into the cache (one word at a time), and finally the word is read from the cache.

(c) Consider the following options to improve the effective memory access time in part (b):

- (1) Make main memory twice as fast
- (2) Maintain t_m = 1000ns, but make the main memory to cache connection 4 words wide

Which of these two options is better, assuming each costs the same? Justify your answer.

10. Suppose you have a word-addressable cached memory system with the following parameters:

Hit ratio = h	Block size = B words
Main memory cycle time = t_m	Cache cycle time = t_c

Main memory to cache connection = c words
Fraction of Write accesses = W
No read through policy
Write through policy for write hits
No-write allocate for write misses
Upon a read miss, a block is read from memory to cache and the word is accessed from the cache by the CPU.

The average memory access time, T_{avg}, can be expressed as follows:

$T_{avg} = h$ [Total ReadHitTime + Total WriteHitTime] +
    $(1 - h)$ [Total ReadMissPenalty + Total WriteMissPenalty]

Give general expressions for each of the quantities in square brackets.

(a) Total ReadHitTime
(b) Total WriteHitTime
(c) Total ReadMissPenalty
(d) Total WriteMissPenalty

11. Suppose we have a cached system with the following parameters:

$t_c = 100$ ns $t_m = 1000$ ns
$h = 0.90$ Block size = 8 words
Main memory to cache connection = 2 words

(a) Calculate the effective memory access time if a read through policy is used.

(b) Calculate the effective memory access time if no read through policy is used. Assume that, upon a cache miss, the replaced cache block is first transferred to main memory, the requested block is then loaded into the cache, and then the selected word is accessed from the cache.

(c) Same as part (b) except that a *no write allocate* policy is used on a write miss. That is, the word is modified in main memory but the block is not loaded into the cache. Assume 30% of all accesses are writes. Recall that *no write allocate* does not imply anything about policies on write hits – all it says is what happens upon a cache miss.

12. Suppose we have a cached system with the following parameters:

$t_c = 100$ ns $t_m = 1000$ ns
$h = 0.95$ Block size = 8 words
Main memory to cache connection = 4 words

(a) Calculate the effective memory access time if a read through policy is used.

(b) Calculate the effective memory access time if no read through policy is used. Assume that, upon a cache miss, the replaced cache block is first transferred to main memory, the requested block is then loaded into the cache, and then the selected word is accessed from the cache.

(c) Same as part (b) except that a *no write allocate* policy is used on a write miss. That is, the block is modified in main memory but not loaded into the cache. Assume 25% of all accesses are writes. Recall that *no write allocate* does not imply anything about policies on write hits – all it says is what happens on a cache write miss.

(d) In this case a write through policy is also implemented for write hits. That is, the cache implements write through with *no write allocate* as in part (c). Because of the write-through policy, upon a cache read miss a block does not need to be written to main memory. The requested block is just loaded from main memory into the cache and the addressed word is then accessed by the CPU from the cache (assume no read through).

13. Consider a cache with four block frames and a fully associative mapping policy. Find the hit ratio for the following **Block** reference string for replacement policies of FIFO (First-In-First-Out), LRU (Least Recently Used), and MRU (Most Recently Used). In each case assume the cache is initially empty.

Block Reference String: 0, 1, 2, 3, 4, 0, 1, 5, 0, 1, 2, 3, 4, 5, 4

14. Suppose you have the following word addressable cached memory system:

Block size = 4 words
Main memory size = 1024 blocks
Cache memory size = 4 block frames
Time to read a block from main memory to cache = 400 ns
t_c = 50 ns
No read-through policy
In each case below, assume the cache is initially empty

The observed string of **WORD** addresses for instruction fetches for a certain program is:

Word Reference String: 2, 6, 11, 14, 1, 17, 5, 9, 22, 19, 12, 3, 21, 4, 18, 15

First convert the above Word reference string to a block reference string.

(a) Consider a direct mapped cache. Compute the total access time for instruction fetches for the above program (ignore data reads and writes). Because there is no read-through policy, all 16 references have to be accessed from the cache, each with time t_c.

Time for instruction fetches can be computed as: 16 * t_c + (No. of block misses * 400)

(b) Consider a fully associative cache. Compute the total access time for instruction fetches using a FIFO replacement policy to replace a cache block frame (ignore data reads and writes).

(c) Consider a fully associative cache. Compute the total access time for instruction fetches using a LRU replacement policy to replace a cache block frame (ignore data reads and writes).

(d) Consider a fully associative cache. Compute the total access time for instruction fetches using a MRU replacement policy to replace a cache block frame (ignore data reads and writes).

15. Consider the following reference string given as WORD addresses:

1, 4, 8, 5, 20, 17, 19, 56, 9, 11, 4, 43, 5, 6, 9, 17.

Assume a Direct-Mapped cache with 16 one-word block frames that is initially empty. When this reference string is processed, label each reference as a Hit or Miss, and pictorially show the state of the cache as each word address is processed. The state of the cache at the end should be the final state after the last word is processed.

16. Using the same reference string as in Problem 15, indicate the Hits and Misses and show the state of the cache after each address including the final contents for a Direct-Mapped cache with 4-word block frames and a total cache size of 16 words. Assume the cache is initially empty.

17. Using the same reference string as in Problem 15, indicate the Hits and Misses and show the state of the cache after each address including the final contents for a Set Associative cache with a total size of 16 words, a Block frame size of one-word and a Set size of 2 (that is, 2 block frames per set). Within a Set, assume an LRU (Least Recently Used) policy for block replacement. Assume the cache is initially empty.

18. Repeat Problem 17 with a FIFO (First In First Out) policy for block replacement.

19. Repeat Problem 17 with an MRU (Most Recently Used) policy for block replacement.

20. A certain program consists of two nested loops, a large outer loop that executes 10 times, and a small inner loop that executes 20 times for each iteration of the outer loop. The general structure of the program is as follows:

17, . . . 23, . . . 165, . . . 239, . . . 1200, . . . 1464

The decimal memory addresses shown delineate the various sections of the program and contain instructions to be executed in straight-line sequencing. The words 165-239 represent the inner loop, and words 23-1200 represent the outer loop.

The memory parameters are: main memory size of 64K words, cache size of 1K words, and a block size of 128 words. The cache has a cycle time of *t*, and main memory has a cycle time of 10*t*. Memory is word addressable. Assume no read-through policy and a direct mapped cache.

(a) Specify the number of bits in the tag, block, and word fields in the interpretation of a main memory address.

(b) Ignoring the reads and writes associated with operand and result fetching and storing, compute the total amount of time needed for instruction fetching in the program. Assume that memory is not interleaved, so the time to load a block into the cache is 10*wt* where *w* is the number of words in a block.

(c) Suppose the cache were set associative with a set size of 2. Find the total time for instruction fetches for this program with an LRU replacement policy.

(d) Same as part (c) but with an MRU replacement policy.

21. A computer system has a word-addressable main memory consisting of 256K 16-bit words. It also has a 4K-word cache organized in a set associative manner with 4 block frames per set and 64 words per block. The cache is 10 times faster than main memory. Assume the cache is initially empty. Suppose the CPU fetches 4352 words from locations 0, 1, 2, 3, . . . , 4351 in order. It then repeats this fetch sequence 14 more times. Assume no interleaved memory and no read-through policy.

(a) Specify the number of bits in the tag, set, and word fields in the interpretation of a main memory address.

(b) Assuming the LRU algorithm is used for block replacement, estimate the speedup (ratio of time without cache to time with cache) resulting from use of the cache.

(c) Repeat part (b) assuming a most recently used (MRU) policy.

22. Consider a computer with the following features:

• 90% of all memory accesses are found in the cache;
• The block size is 2 words and the whole block is read on any miss;
• The CPU sends references to the cache at the rate of 10^7 words per second;
• 25% of the above references are writes;
• The bus can support 10^7 words per second, reads or writes;
• The bus reads or writes a single word at a time;
• Assume at any one time, 30% of the block frames in the cache have been modified;
• The cache uses write allocate on a write miss.

You are considering adding a peripheral to the bus, and you want to know how much of the bus bandwidth is already used. Calculate the percentage of bus bandwidth used on the average in the two cases below. This percentage is called the *traffic ratio*. State any further assumptions you need, if any.

(a) The cache is write through.

(b) The cache is write back.