

Submit answers by committing a file into the hw4 sub-directory of your CNETID-cs154-aut-19 svn repository. Do not create this directory yourself! An “svn update” at the top level of your checkout will create it. Your answer file should be either hw4.txt or hw4.pdf for answers in plain ASCII text, or PDF, respectively. PDFs of scanned hand-written pages must not exceed 6 megabytes. No other file formats, or filenames are acceptable, and no files besides hw4.txt or hw4.pdf will be graded. Not following directions will result in losing points.

(1) (20 points) Suppose we have a system with the following properties:

- o The memory is byte addressable.
- o Memory accesses are to **1-byte words** (not to 4-byte words).
- o Addresses are 12 bits wide.
- o The cache has four sets ($S = 4$). Each set consists of two lines (two-way set associative, $E = 2$). Each line is four bytes ($B = 4$).
- o LRU replacement is used for the cache.

The contents of the cache are as follows, with all addresses, tags, and values given in hex:

Set index	Valid	Tag	Byte 0	Byte 1	Byte 2	Byte 3
0	1	00	40	41	42	43
	1	83	FE	97	CC	D0
1	1	00	44	45	46	47
	0	83	–	–	–	–
2	1	00	48	49	4A	4B
	0	40	–	–	–	–
3	1	FF	9A	C0	03	FF
	0	54	–	–	–	–

A. Each of the bits in the 12-bit address can have a role in caching. Assume the bits are ordered from left to right in decreasing significance. Identify how each of the bits is used in caching by writing a sequence of 12 letters, each one indicating the purpose of the corresponding bit, with these possibilities:

- o “s”: the bit is part of the Set index
- o “b”: the bit is part of the Block offset
- o “t”: the bit is part of the Tag
- o “n”: the bit is not used for cache indexing

One possible (incorrect) answer is “tssnnttssbb” where the first “t” describes the most significant bit of the address, and the last “b” describes the least significant bit.

B. For each of the following memory accesses, indicate two things: will it be a cache hit or miss **when carried out in sequence** as listed, and, for **reads**, give the value if it can be inferred from the information given. The cache and memory are affected by nothing except the operations listed. If the read value is not inferrable, give “unknown” for Value. For all writes, give “n/a” for Value.

Operation	Address	Hit/Miss?	Value
Read	0x833		
Read	0x00A		
Read	0x006		
Read	0xFFE		
Read	0x54D		
Write	0x54E		
Write	0x007		
Read	0x44C		
Read	0xFFE		

(2) (20 points) Consider the following bit of C code:

```

1  int x[2][128];
2  int i, sum=0;
3  for (i=0; i<128; i++) {
4      sum += x[0][i] * x[1][i];
5  }
```

and assume the following conditions:

- o `sizeof(int) == 4`
- o Array `x` begins at memory address `0x0` and is stored in row-major order.
- o In each case below, the cache is initially empty.
- o The only memory accesses are to the entries of the array `x`. All other variables are stored in registers.

Given these assumptions, estimate the miss rates for the following cases and answer the associated questions. You do not need more than 50 words per part.

- A. Case 1: Assume the cache is 512 bytes, direct-mapped, with 16-byte cache blocks. What is the miss rate?
- B. Case 2: What is the miss rate if we double the cache size to 1024 bytes?
- C. Case 3: Now assume the cache is 512 bytes, two-way set associative with an LRU replacement policy, with 16-byte cache blocks. What is the cache miss rate?
- D. For Case 3, will a larger cache size help to reduce the miss rate? Why or why not?
- E. For Case 3, will a larger block size help to reduce the miss rate? Why or why not?

(3) (15 points) Consider a combination of software and hardware that is quantified in terms of the following variables. All reported latencies are average latencies.

- o Latency of ALU operations: $L_{AL} = 1.1$ cycles.
- o Latency of branches/jumps: $L_{BJ} = 3.0$ cycles.
- o Cache hit rate: $R = 60\%$.
- o A cache hit takes $H = 1$ cycle and a cache miss takes $M = 120$ cycles.
- o The breakdown of instruction types is $F_{LD} = 22\%$ moves from memory to register (loads), $F_{ST} = 12\%$ moves from register to memory (stores, which are similar with loads), $F_{BJ} = 20\%$ branches/jumps, and the rest are ALU operations.

A. In terms of the variables L_{AL} , L_{BJ} , R , H , M , F_{LD} , F_{ST} , and F_{BJ} , what is a formula for computing the average latency?

B. With the values given, what is the expected latency (a numeric value, to 2 significant digits)?