CS2100 Computer Organization
Tutorial 11: Caching
(Week 13: 13 Nov – 17 Nov 2023)

1. A byte-addressed machine with a word size of 16 bits and address width of 32 bits has a direct-mapped cache with 16 blocks and a block size of 2 words.

   (a) Given a sequence of memory references as shown below, where each reference is given as a byte address in both decimal and hexadecimal forms, indicate whether the reference is a hit (H) or a miss (M). For a miss, also identify the type of miss it is (i.e., cold/compulsory, conflict or capacity.) Assume that the cache is initially empty.

Memory address		**Hit (H) or Miss (M)?**
(in decimal)	**(in hexadecimal)**	
4	4	**M (cold)**
92	5C	
7	7	
146	92	
30	1E	
95	5F	
176	B0	
93	5D	
145	91	
264	108	
6	6	

(b) Given the above sequence of memory references, fill in the final contents of the cache. Use the notation M[*i*] to denote the word at memory address *i*.

Index	**Tag value**	**Word 0**	**Word 1**
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			

(c) Repeat (a) and (b) if the cache is instead two-way set associative while having the same block and overall size.

(d) Repeat (a) and (b) if the cache is instead fully associative while having the same block and overall size.

2. A machine with byte addresses and a word size of 32 bits and address width of 32 bits has a direct-mapped data cache with 4 blocks each consisting of 2 words.

(a) Given the MIPS program below, and assuming that array `A` starts at memory hexadecimal location `0x1000` while array `B` starts at memory hexadecimal location `0x4010`. Fill in the first 10 address requests seen at the data cache and indicate whether the reference is a hit (H) or a miss (M). Assume that the cache is initially empty.

```
start:
      la   $s0, A                #PC=0x100
      la   $s1, B                #PC=0x104
      li   $t0, 1                #PC=0x108
loop:
      slt  $t1, $t0, 1000        #PC=0x10c
      beq  $t1, $zero, end_loop  #PC=0x110
      sll  $t2, $t0, 2           #PC=0x114
      add  $t3, $s0, $t2         #PC=0x118
      lw   $a0, 0($t3)           #PC=0x11c
      add  $t4, $s1, $t2         #PC=0x120
      lw   $a1, 0($t4)           #PC=0x124
      add  $v0, $a0, $a1         #PC=0x128
      sw   $v0, -4($t3)          #PC=0x12c
      addi $t0, $t0, 1           #PC=0x130
      j    loop                  #PC=0x134
end_loop:
      .    .    .
```

(b) Given the above program, fill in the final contents of the cache. Use the notation M[*i*] to denote the word at memory address *i*. (*i* may be in hexadecimal.)

(c) What is the total number of data cache memory references, hits and misses after the execution of the above MIPS program?

3. Suppose a (32 bit) MIPS processor has an *instruction* cache that is also direct mapped with 4 blocks each consisting of 2 words. Using the same program found in Question 2, answer the following:

   (a) Fill in the final contents of this cache assuming that **`start`** is at location 0x100 and each of the assembler pseudo-instructions (like **`la`** and **`li`**) also occupy 32 bits. You can use "PC=…" to indicate the word content of the cache.

   (b) What is the miss rate in this case?

4. Using your knowledge of Boolean circuits, produce a circuit (the simpler the better) that will check the Valid and Tag bits of a cache block to determine if it is a hit (1) or a miss (0). For simplicity, assume that tags are 20 bits long. Limit each gate to at most 4 inputs. Do not use adders. Suggestion: Use Logisim for easier replications.