Computer Architecture 2018 Fall Homework 5

5.5 For a direct-mapped cache design with a 64-bit address, the following bits of the address are used to access the cache.

Tag	Index	Offset
63–10	9–5	4–0

5.5.1 [5] <§5.3> What is the cache block size (in words)?
5.5.2 [5] <§5.3> How many blocks does the cache have?
5.5.3 [5] <§5.3> What is the ratio between total bits required for such a cache implementation over the data storage bits?
Beginning from power on, the following byte-addressed cache references are recorded.

Address												
Hex	00	04	10	84	E8	A0	400	1E	8C	C1C	B4	884
Dec	0	4	16	132	232	160	1024	30	140	3100	180	2180

5.5.4 [20] <§5.3> For each reference, list (1) its tag, index, and offset, (2) whether it is a hit or a miss, and (3) which bytes were replaced (if any).
5.5.5 [5] <§5.3> What is the hit ratio?
5.5.6 [5] <§5.3> List the final state of the cache, with each valid

entry represented as a record of <index, tag, data>. For example,

```
<0, 3, Mem[0xC00]-Mem[0xC1F]>
```

5.10 In this exercise, we will look at the different ways capacity affects overall performance. In general, cache access time is

proportional to capacity. Assume that main memory accesses take 70 ns and that 36% of all instructions access data memory. The following table shows data for L1 caches attached to each of two processors, P1 and P2.

	L1 Size	L1 Miss Rate	L1 Hit Time
P1	2 KiB	8.0%	0.66 ns
P2	4 KiB	6.0%	0.90 ns

5.10.1 [5] <§5.4> Assuming that the L1 hit time determines the cycle times for P1 and P2, what are their respective clock rates?

5.10.2 [10] <§5.4> What is the Average Memory Access Time for P1 and P2 (in cycles)?

5.10.3 [5] <§5.4> Assuming a base CPI of 1.0 without any memory stalls, what is the total CPI for P1 and P2? Which processor is faster? (When we say a "base CPI of 1.0", we mean that instructions complete in one cycle, unless either the instruction access or the data access causes a cache miss.)

For the next three problems, we will consider the addition of an L2 cache to P1 (to presumably make up for its limited L1 cache capacity). Use the L1 cache capacities and hit times from the previous table when solving these problems. The L2 miss rate indicated is its local miss rate.

L2 Size	L2 Miss Rate	L2 Hit Time
1 MiB	95%	5.62 ns

5.10.4 [10] <§5.4> What is the AMAT for P1 with the addition of an L2 cache? Is the AMAT better or worse with the L2 cache?

5.10.5 [5] <§5.4> Assuming a base CPI of 1.0 without any memory stalls, what is the total CPI for P1 with the addition of an L2 cache?

5.10.6 [10] <§5.4> What would the L2 miss rate need to be in order for P1 with an L2 cache to be faster than P1 without an L2 cache?

5.10.7 [15] <§5.4> What would the L2 miss rate need to be in order for P1 with an L2 cache to be faster than P2 without an L2 cache?

5.16 As described in Section 5.7, virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. This exercise shows how this table must be updated as addresses are accessed. The following data constitute a stream of virtual byte addresses as seen on a system. Assume 4 KiB pages, a four-entry fully associative TLB, and true LRU replacement. If pages must be brought in from disk, increment the next largest page number.

Decimal	4669	2227	13916	34587	48870	12608	49225
hex	0x123d	0x08b3	0x365c	0x871b	0xbee6	0x3140	0xc049

TLB

Valid	Tag	Physical Page Number	Time Since Last Access
1	0xb	12	4
1	0x7	4	1
1	0x3	6	3
0	0x4	9	7

Page table

Index	Valid	Physical Page or in Disk
0	1	5
1	0	Disk
2	0	Disk
3	1	6
4	1	9
5	1	11
6	0	Disk
7	1	4
8	0	Disk
9	0	Disk
a	1	3
b	1	12

5.16.1 [10] <§5.7> For each access shown above, list

- whether the access is a hit or miss in the TLB,
- whether the access is a hit or miss in the page table,
- whether the access is a page fault,
- the updated state of the TLB.

5.16.2 [15] <§5.7> Repeat Exercise 5.16.1, but this time use 16 KiB pages instead of 4 KiB pages. What would be some of the advantages of having a larger page size? What are some of the disadvantages?

5.16.3 [15] <§5.7> Repeat Exercise 5.16.1, but this time use 4 KiB pages and a two-way set associative TLB.

5.16.4 [15] <§5.7> Repeat Exercise 5.16.1, but this time use 4 KiB pages and a direct mapped TLB.

5.16.5 [10] <§§5.4, 5.7> Discuss why a CPU must have a TLB for high performance. How would virtual memory accesses be handled if there were no TLB?