CprE 381 Homework 10

[Note: The homework below focuses on the details of caching and cache design. Once you have completed this homework you should be able to motivate the need for caches using specific program examples. You should also be able to configure and hand-simulate a range of caches.]

1. Cache Configuration and Simulation

In this problem we will consider several cache designs for a processor implementing the MIPS ISA *[Note that this has implications needed to answer the below questions].* Assume that the block offset is four bits and the index is four bits.

a. What is the cache block size in bytes? words? double words?

16 bytes, 4 words, 2 double words

b. How many sets does this cache have? *[Hint: note that both direct-mapped and fully-associative caches can be considered to have sets.]*

2^4 sets. The amount of sets is based on the number of bits used for the index.

c. Record both the amount of data and meta-data (in bits) this cache holds if it is direct-mapped, two-way set associative, and four-way set associative.

Direct-mapped - Index bits:

24 x 16 (2^4) = 384 bits

24 x 16 x 2 = 768 bits

24 x 16 x 4 = 1536 bits

*There is no trade off between sets and ways because you have 4 bits for each i.e. 4 offset and 4 index

d. Simulate the direct-mapped and four-way set associative cache with respect to the following series of memory accesses. In the table below, indicate whether each memory access was a hit or a miss and provide the reason for each miss. Assume the caches have no valid entries to begin with and use a least-recently used (LRU) replacement policy.

Memory Access	Direct-Mapped	4-way Set Associative
0x1001FEA0	Miss	Miss
0x1001EFA4	Miss	Miss
0x1001FEA8	Miss	Miss
0x100100A0	Miss	Miss

0x100100B0	Miss	Miss
0x100100C0	Miss	Miss
0x10011FA1	Miss	Miss
0x1001EEA2	Miss	Miss
0x1001EFAF	Miss	Miss
0x100100A2	Miss	Miss

They are all misses because none of the tags match. We have 24 bit tags and none of the first 4 hex values match.

Write the valid entries in the final state of each cache using the format <set#, way#, tag>. What was the hit rate of each cache?
Both have a hit rate of 0%
Direct: <0, 0, 00C0>, <1, 0, 1FA1>, <2, 0, 00A2>, <4, 0, EFA4>, <9, 0, FEA8>, <15, 0, EFAF>

4-Way: <0, 0, FEA0>, <0, 1, 00A0>, <0, 2, 00B0>, <0, 3, 00C0>, <1, 0, 1FA1>, <2, 0, EEA2>, <2, 1, 00A2>, <4, 0, EFA4>, <9, 0, FEA8>, <15, 0, EFAF>

2. Cache Write Nuances
   a. ZyBooks (Textbook) 5.19.6 all (a-c)

   a)

Between L1 and L2 we certainly want a write buffer. This is because L1 has a write-through policy, so every time we change something in L1, we must change it in L2 and RAM. Write buffer improves performance.

Unsure

   b)

Unsure

   c)

Unsure

b. ZyBooks (Textbook) 5.19.7 all (a-b)

Unsure what bandwidth is?

3. Cache Hierarchy Performance

a. ZyBooks (Textbook) 5.19.10 all (a-g)

a) 1/.66 ns = 1.5 GHz

b) AMAT = hit time + missrate * miss penalty
P1 = .66 ns + .08 * 70 ns
P2 = .90 ns + .06 * 70 ns

c) CPI = base CPI + (Stall Cycles / IC)
Mem Access rate = 0.36
p1 miss rate = .08
Stall Cycles/ instruction = .08 *.36
CPI = 1.0 + 0.036 = 1.036
This is for both P1 and P2
They both have the same frequency and same CPI so they have the same speed

d) AMAT (with L2) = L1 Hit Time + L1 Miss Rate * (L2 Hit Time + L2 Miss Rate * Main Memory Access Time)
= 0.66 ns + .08(5.62 ns + .95 * 70ns)

e) CPI = base CPI + (Stall Cycles / IC)
Mem Access rate = 0.36
l2 miss rate = .95
Stall Cycles/ instruction = .95 *.36
CPI = 1.0 + (.95*.36)

f) Speed = IC * CPI/Clock rate
IC assumed to be the same
Clock rate is the same

CPI p1 = CPI p2
1.0 + (L2 miss rate * .36) = 1.0 + .08 * .36
L2 miss rate < .06

g)

CPI p1 = CPI p2
1.0 + (L2 miss rate * .36) = 1.0 + .06 * .36
L2 miss rate < .06