Derek Albosta

1. Do problem 5.2.1 from the book (5th edition).

16 one word blocks

Address	8-bit binary	tag	index block	Hit?
3	00000011	0000	0011	Miss
180	10110100	1011	0100	Miss
43	00101011	0010	1011	Miss
2	00000010	0000	0010	Miss
191	10111111	1011	1111	Miss
88	01011000	0101	1000	Miss
190	10111110	1011	1110	Miss
14	00001110	0000	1110	Miss
181	10110101	1011	0101	Miss
44	00101100	0010	1100	Miss
186	10111010	1011	1010	Miss
253	11111101	1111	1101	Miss

2. Do problem 5.2.2 from the book (5th edition).

Two word blocks of size 8 (only need 3 bits for index)

Address	8-bit binary	tag	index block	Hit?
3	00000011	0000	001	Miss
180	10110100	1011	010	Miss
43	00101011	0010	101	Miss
2	00000010	0000	001	Hit
191	10111111	1011	111	Miss
88	01011000	0101	100	Miss
190	10111110	1011	111	Hit
14	00001110	0000	111	Miss
181	10110101	1011	010	Hit
44	00101100	0010	110	Miss
186	10111010	1011	101	Miss
253	11111101	1111	110	Miss

3. We've got a benchmark program consisting of 30 billion (30x10^9) instruc
tions, and it runs at an average CPI of 3.0 on a processor that has
a clock rate of 1.5GHz. Loads and stores make up 35% of the instructions
executed.

10.5 billion lw/sw instructions
19.5 billion r-type instructions

a. How long will the benchmark program take to run if we assume a perfect c
ache? (That is, we assume all memory accesses are cache hits and
therefore experience no memory-related delays.)

```
        CPU time = (#inst * CPI) / clock rate
        CPU time = (30*10^9 inst * 3.0 CPI) / 1.5 GHz
        CPU time = 60*10^9 ns = 60 sec
```

b. If the system had no cache, and the penalty on each memory access was 12
0 cycles, how long would the program take to run?

```
        CPU time = (((#lw/sw * (240+CPI)) + (#r-type * (120+CPI))) / clock rate
        CPU time = ((10.5*10^9 * 243) + (19.5*10^9 * 123)) / 1.5GHZ
        CPU time = 3,300 sec
```

c. Assume we've got a cache with a miss rate of 2% for instructions and 5%
for loads and stores. Now how long would the program take to execute?
(Memory accesses that aren't satisfied by the cache still take 120 cycle
s, as before.)

```
        inst miss = 30*10^9 * .02 = 525*10^6 misses
        data miss = 10.5*10^9 * .05  = 525*10^6 misses

        CPU time = ((#inst * CPI) + (instmiss * 120) + (datamiss * 120)) / clock
 rate
        CPU time = ((30*10^9 * 3) + (525*10^6 * 120) + (525*10^6 * 120)) / 1.5GH
Z
        CPU time = 144 sec


d.      The engineering team reports that they can get the miss rates down if th
ey make the cache a bit bigger, but that doing so will cause even the
        cache hits to take a little longer. Specifically, they can get the miss
rates down to 1.5% for instructions and 4% on data, but the CPI will increase
        to an average of 4 cycles/instruction when we take into account the incr
eased delays on cache hits. Is this new design better than the one from part c,
        above, or not? Justify your answer.

        inst miss = 30*10^9 * .015 = 450*10^6 misses
        data miss = 10.5*10^9 * .04 = 420*10^6 misses

        CPU time = ((#inst * CPI) + (instmiss * 120) + (datamiss * 120)) / clock
 rate
        CPU time = ((30*10^9 * 4) + (450*10^6 * 120) + (420*10^6 * 120)) / 1.5GH
Z
        CPU time = 149.6 sec

        It is worse than c as d is slower
```