The University of Alabama in Huntsville
ECE Department
Homework Assignments
CPE 431/531 01/91/92
Fall 2015
Homework #6 Solution

5.2.2(10), 5.2.3(20), 5.6.3(5), 5.6.5(5), 5.6.6(10), 5.7.1(10), 5.7.3(15), 5.7.5(10) Total = 85 points

5.2 Caches are important to providing a high-performance memory hierarchy to processors. Below is a list of 32-bit memory addresses, given as word addresses. 3, 180, 43, 2, 191, 88, 190, 14, 181, 44, 186, 253.

5.2.2 For each of these references, identify the binary address, the tag, and the index given a direct-mapped cache with two-word blocks and a total size of 8 blocks. Also list if each reference is a hit or a miss, assuming the cache is initially empty.

8 blocks x 1 set /1 block = 8 sets

	Index Block Offset	
3	0000 001 1	miss
180	1011 010 0	miss
43	0010 101 1	miss
2	0000 001 0	hit
191	1011 111 1	miss
88	0101 100 0	miss
190	1011 111 0	hit
14	0000 111 0	miss
181	1011 010 1	hit
44	0010 110 0	miss
186	1011 101 0	miss
253	1111 110 1	miss

Set	Tag*	Data
0		
1	0x0000 000	M[2..3]
2	0x0000 00B	M[180..181]
3		
4	0x0000 005	M[88..89]
5	0x0000 00B	M[186..187]
6	0x0000 002, 0x0000 00F	M[44..45], M[252..253]
7	0x0000 00B, 0x0000 000	M[190..191], M[14..15]

5.2.3 You are asked to optimize a cache design for the given references. There are three direct mapped cache designs possible, all with a total of 8 word of data: C1 has 1-word blocks, C2 has 2-word blocks,and C3 has 4-word blocks. In terms of miss rate, which cache design Is the best? If the miss stall time is 25 cycles, and C1 has an access time of 2 cycles, C2 takes 3 cycles, and C3 takes 5 cycles, which is the best cache design?

C1: 8 words * 1block/1word* 1 set/1 block = 8 sets, Index = 3 bits, Block Offset = 0 bits

	Index	
3	0000 0 011	miss
180	1011 0 100	miss
43	0010 1 011	miss
2	0000 0 010	miss
191	1011 1 111	miss
88	0101 1 000	miss
190	1011 1 110	miss
14	0000 1 110	miss
181	1011 0 101	miss
44	0010 1 100	miss
186	1011 1 010	miss
253	1111 1 101	miss

Set	Tag	Data
0	0x0000 0051	M[88]
1		
2	~~0x0000 0000~~, 0x0000 00B1	~~M[2]~~, M[186]
3	~~0x0000 0000~~, 0x0000 0021	~~M[3],~~ M[43]
4	~~0x0000 00B0~~, 0x0000 0021	~~M[180]~~, M[44]
5	~~0x0000 00B0~~, 0x0000 00F1	~~M[181],~~ M[253]
6	~~0x0000 00B1~~, 0x0000 0001	~~M[190]~~, M[14]
7	0x0000 00B1	M[191]

C2: 8 words * 1block/2words* 1 set/1 block = 4 sets, Index = 2 bits, Block Offset = 1 bit

	Index
3	0000 0 01 1 miss
180	1011 0 10 0 miss
43	0010 1 01 1 miss
2	0000 0 01 0 miss
191	1011 1 11 1 miss
88	0101 1 00 0 miss
190	1011 1 11 0 hit
14	0000 1 11 0 miss
181	1011 0 10 1 hit
44	0010 1 10 0 miss
186	1011 1 01 0 miss
253	1111 1 10 1 miss

Set	Tag	Data
0	0x0000 0051	M[88:89]
1	~~0x0000 0000~~, ~~0x0000 0021,~~ ~~0x0000 0000,~~ 0x0000 00B1	~~M[2:3],~~ ~~M[42:43],~~ ~~M[2:3],~~ M[186:187]
2	~~0x0000 00B0~~, ~~0x0000 0021~~, 0x0000 00F1	~~M[180:181],~~ ~~M[44:45],~~ M[252:253]
3	~~0x0000 00B1,~~ 0x0000 0001	~~M[190:191].~~ M[14:15]

C3: 8 words * 1block/4 words* 1 set/1 block = 2 sets, Index = 1 bit, Block Offset = 2 bits

	Index
3	0000 0 0 11 miss
180	1011 0 1 00 miss
43	0010 1 0 11 miss
2	0000 0 0 10 miss
191	1011 1 1 11 miss
88	0101 1 0 00 miss
190	1011 1 1 10 hit
14	0000 1 1 10 miss
181	1011 0 1 01 miss
44	0010 1 1 00 miss
186	1011 1 0 10 miss
253	1111 1 1 01 miss

Set	Tag	Data
0	~~0x0000 0000~~, ~~0x0000 0021~~, ~~0x0000 0000~~, ~~0x0000 0051~~, 0x0000 00B1	~~M[0:3],~~ ~~M[40:43],~~ ~~M[0:3]~~ ~~M[88:91],~~ M[184:187]
1	~~0x0000 00B0~~ ~~0x0000 00B1,~~ ~~0x0000 0001~~, ~~0x0000 00B0~~, ~~0x0000 0021,~~ 0x0000 00F1	~~M[180:183],~~ ~~M[188:191].~~ ~~M[12:15],~~ ~~M[180:183],~~ ~~M[44:47~~], M[252:255]

By miss rate, the best cache design is C2.
Total time: C1 12*(25+2) = 324, C2 12*3 + 10*25 = 36 + 250 = 286, C3 = 12*5 + 11*25 = 60 + 275 = 335
C2 is also best by this metric.

5.6 In this exercise, we will look at the different ways capacity affects ovaerall performance. In general, cache access time is proportional to capacity. Assume that main memory accesses take 70 ns and that memory accesses are 36% of all instructions. 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

5.6.3 Assuming a base CPI of 1.0 without any memory stalls, what is the total CPI for P1 and P2? Which processor is faster?

Hit time is included in base CPI.
CPI_{P1} = 1.0 + 1.36(0.08*70ns)/0.66 ns = 12.54
CPI_{P2} = 1.0 + 1.36(0.06*70ns)/0.9 ns = 7.35

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

5.6.5 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?

$CPI_{total} = CPI_{base} + 1.36(L2_{search} * L2_{hit\_time} + Main\_memory_{search} * Main\_memory_{hit\_time})$/cycle time
$CPI_{total} = 1.0 + 1.36(L1_{miss} * L2_{hit\_time} + L1_{miss} * L2_{miss} * Main\_memory_{hit\_time})$/cycle time
CPI_{total} = 1.0 + 1.36(0.08 * 5.62 ns + 0.08 * 0.95 * 70 ns)/0.66 ns
CPI_{total} = 1.0 + 1.36(0.4496 ns + 5.32 ns)/0.66 ns
CPI_{total} = 1.0 + 1.36 * 8.74 = 12.88

5.6.6 Which processor is faster, now that P1 has an L2 cache? If P1 is faster, what miss rate would P2 need in its L1 cache to match P1's performance? If P2 is faster, what miss rate would P1 need in its L1 cache to match P2's performance?

$$\frac{P_{P1}}{P_{P2}} = \frac{ET_{P2}}{ET_{P1}} = \frac{IC_{P2} \times CPI_{P2} \times CT_{P2}}{IC_{P1} \times CPI_{P1} \times CT_{P1}} = \frac{CPI_{P2} \times CT_{P2}}{CPI_{P1} \times CT_{P1}} = \frac{7.35 \times 0.9ns}{12.88 \times .0.66ns} = 0.778$$

P2 is faster.

x * 0.66 ns = 7.35 * 0.9, x = 10.02

10.02 = 1.0 + 1.36(y * 5.62 + y * 0.95 * 70)/0.66

9.02 = 1.36(72.12 * y)/0.66

5.95 = 98.08 * y, y = 0.06

y =6%

5.7 This exercise examines the impact of different cache designs, specifically comparing associative caches to the direct-mapped caches from section 5.4. For this exercise, use the address stream shown in Exercise 5.2

5.7.1 Using the sequence of addresses given, show the final cache contents for a three-way set associative cache with two-word blocks and a total size of 24 words. Use LRU replacement. For each reference identify the index bits, the tag bits, the block offset bits, and if it is a hit or miss.

$$24words \times \frac{1block}{2words} \times \frac{1set}{3blocks} = 4sets \text{ , Index} - 2 \text{ bits, Block Offset} = 1 \text{ bit.}$$

	Index Block Offset
3	0000 0 01 1 miss
180	1011 0 10 0 miss
43	0010 1 01 1 miss
2	0000 0 01 0 hit
191	1011 1 11 1 miss
88	0101 1 00 0 miss
190	1011 1 11 0 hit
14	0000 1 11 0 miss
181	1011 0 10 1 hit
44	0010 1 10 0 miss
186	1011 1 01 0 miss
253	1111 1 10 1 miss

*Last hex digit is actually a bit

Set	Tag*	Data	Tag*	Data	Tag*	Data
0					0x0000 005 1	M[88..89]
1	0x0000 002 1	M[42..43]	0x0000 000 0	M[2..3]	0x0000 00B 1	M[186..187]
2	0x0000 002 1	M[44..45]	0x0000 00F 1	M[252..253]	0x0000 00B 0	M[180..181]
3	0x0000 00B 1	M[190..191]	0x0000 000 1	M[14..15]		

5.7.3 Using the references given, what Is the miss rate for a fully associative cache with two-word blocks and a total size of 8 words, using LRU replacement? What is the miss rate for MRU (most recently used) replacement? Finally, what is the best possible miss rate for this cache, given any replacement policy?

8 words x 1 block/2 words = 4 blocks in one set

LRU

3 0000 001 1 miss
180 1011 010 0 miss
43 0010 101 1 miss
2 0000 001 0 hit
191 1011 111 1 miss
88 0101 100 0 miss
190 1011 111 0 hit
14 0000 111 0 miss
181 1011 010 1 miss
44 0010 110 0 miss
186 1011 101 0 miss
253 1111 110 1 miss

***The last digit of the tag is three bits**

Tag*	Data	Tag*	Data	Tag*	Data	Tag*	Data
0x0000 00B2, 0x0000 0054, 0x0000 0026	M[180..181], M[88..89], M[44..45]	0x0000 00B7, 0x0000 00B5	M[190..191], M[186..187]	0x0000 0001, 0x0000 00B2	M[2..3], M[180..181]	0x0000 0025, 0x0000 0007	M[42..43], M[14..15], M[252..253]

Miss rate = 10/12 = 83.3%

MRU

3 0000 001 1 miss
180 1011 010 0 miss
43 0010 101 1 miss
2 0000 001 0 hit
191 1011 111 1 miss
88 0101 100 0 miss
190 1011 111 0 miss
14 0000 111 0 miss
181 1011 010 1 hit
44 0010 110 0 miss
186 1011 101 0 miss
253 1111 110 1 miss

***The last digit of the tag is three bits**

Tag*	Data	Tag*	Data	Tag*	Data	Tag*	Data
0x0000 00B2	M[180..181]	0x0000 00B5, 0x0000 0054, 0x0000 00B7, 0x0000 0007, 0x0000 0026, 0x0000 00B5, 0x0000 00F6	M[190..191], M[88..89], M[190..191], M[14..15], M[44..45], M[186..187], M[252..253]	0x0000 0001	M[2..3]	0x0000 0025	M[42..43]

Miss rate – 10/12 = 83.3%

Best miss rate possible = 9/12 = 75%, There are only three references that could hit.

Multilevel caching is an important technique to overcome the limited amount of space that a first level cache can provide while still maintaining its speed. Consider a processor with the following parameters.

Base CPI, no memory stalls	Processor speed	Main memory access time	First-level cache miss rate per instruction	Second-level cache, direct-mapped speed	Global miss rate with second-level cache, direct-mapped	Second-level cache, eight-way s et associative speed	Global miss rate with second-level cache, eight-way set associative
1.5	2 GHz	100 ns	7 %	12 cycles	3.5 %	28 cycles	1.5 %

5.7.5 It is possible to have an even greater cache hierarchy than two levels. Given the processor above with a second level direct-mapped cache, a designer wants to add a third level cache that takes 50 cycles to access and will reduce the global miss rate to 1.3%. Would this provide better performance? In general, what are the advantages and disadvantages of adding a third level cache?

$CPI_{L1L2} = CPI_{base} + L2_{search} * L2_{hit\_time} + (Main\_memory_{search} * Main\_memory_{hit\_time})$/cycle time
$CPI_{L1L2} = 1.5 + 0.07 * 12 + (0.035 * 100\ ns)/0.5\ ns = 1.5 + 0.84 + 7 = 9.34$

$CPI_{L1L2L3} = CPI_{base} + L2_{search} * L2_{hit\_time} + L3_{search} * L3_{hit\_time} + (Main\_memory_{search} * Main\_memory_{hit\_time})$/cycle time
$CPI_{L1L2L3} = 1.5 + 0.07 * 12 + 0.035 * 50 + (0.013 * 100\ ns)/0.5\ ns = 1.5 + 0.84 + 1.75 + 2.6 = 6.69$
Adding the L3 would provide better performance as indicated by the lower CPI. The advantage is better performance, the disadvantage is that it takes up space that could be used for other functionality.