University of Maryland Baltimore County
Department of Computer Science and Electrical Engineering

CMSC 411, Computer Architecture

Assignment #5 Solution — Due: Tuesday 11/28/17 in class

Question 1: *(30 Points)*

The table below lists parameters for different direct-mapped cache designs.

	Cache data size	Cache block size
i)	64 KB	1 word
ii)	64 KB	2 words

A) Calculate the total number of bits required for the cache listed in the table, assuming a 32-bit address.

i) Direct-mapped (block size = 1 word)

blocks = 64 KB / 4 bytes = 16 K blocks ➔ index = $\log_2$ (16K) = 14 bits

Tag size = 32 - Index - Byte offset - Word offset = 32 - 14 - 2 - 0 = 16 bits

Cache Size = 16 K [16 (Tag) + 1 (valid) + 1×32 (block size)] = 784K bits

ii) Direct-mapped (block size = 2 word)

blocks = 64 KB / (2×4) bytes = 8 K blocks ➔ index = $\log_2$ (8K) = 13 bits

Tag size = 32 - Index - Byte offset - Word offset = 32 - 13 - 2 - 1 = 16 bits

Cache Size = 8 K [16 (Tag) + 1 (valid) + 2×32 (block size)] = 648K bits

B) What is the total number of bits if the cache is organized as a 4-way associative with one word blocks?

Direct-mapped (block size = 16 word)

blocks = 64 KB / (16×4) bytes = 1 K blocks ➔ index = log2 (1K) = 1 bits

Tag size = 32 - Index - Byte offset - Word offset = 32 - 1 - 2 - 4 = 25 bits

Cache Size = 1 K [16 (Tag) + 1 (valid) + 16×32 (block size)] = 529K bits

Commented [AA1]: 3 points
Commented [AA2]: 3 points
Commented [AA3]: 3 points
Commented [AA4]: 1 point
Commented [AA5]: 3 points
Commented [AA6]: 3 points
Commented [AA7]: 3
Commented [AA8]: 1 point
Commented [AA9]: 3 points
Commented [AA10]: 3 points
Commented [AA11]: 3 points
Commented [AA12]: 1 point

Question 2: *(30Points)*

For a pipeline with a perfect CPI=1 if no memory-access related stall, consider the following program and cache behaviors.

Data reads per 1000 instructions	Data writes per 1000 instructions	Instruction cache miss rate	Data cache miss rate	Block size (byte)
200	160	0.20%	2%	8

A) For a write-through, write-allocate cache with sufficiently large write buffer (i.e., no buffer caused stalls), what's the minimum read and write bandwidths (measured by byte-per-cycle) needed to achieve a CPI of 2?

For a write-allocate cache, when a write instruction encounters a miss, the cache is first updated with missing block. Therefore, it will experience the same penalty as a read miss. With the write-through policy, there is no need to check dirty bit since any cache-write operation (1 cycle) will be executed concurrently with memory write (into the write buffer). If we denote to the read and write bandwidth (bit rate) by BW then the number of cycles required for populating a cache block form the memory

is 8/BW. If we denote to the number of instruction executed by the program by *I*, then we can find the miss rates as follows:

1. Data Cache read miss penalty = $I \times 200/1000 \times 2/100 \times (8/BW+1) = 0.004 \times I \times (8/BW+1)$ cycles
2. Data Cache write miss penalty = $I \times 160/1000 \times 2/100 \times (8/BW+1) = 0.0032 \times I \times (8/BW+1)$ cycles
3. Instruction Cache read miss penalty = $I \times 0.20/100 \times (8/BW+1) = 0.002 \times I \times (8/BW+1)$ cycles

To Achieve a CPI of 2, we can write I × CPI = Hit time + miss penalty. Therefore, we can write:

$I \times 2 = I + I \times (8/BW+1) \times (0.004+0.0032+0.002)$

Solving the above for BW yields $BW = 8 / 107.69 = 0.7428$ Byte/Cycle

B) For a write-back, write-allocate cache, assuming 30% of replaced data cache blocks are dirty, what's the minimal read and write bandwidths needed for a CPI of 2?

The write-back policy means that the cache and memory may have different contents if the block was dirty. Therefore, upon a cache miss we must check the dirty block and if the dirty bit is set, then we will have to write the block back to the memory before replacing it with the required block. With write-allocate, when a write miss occurs, we must populate the cache with the missing block, however, we have to check the dirty bit of the block being replaced in cache before fetching the required block. If the block is dirty, then we must write it to memory before fetching the new block. Therefore, the miss rates are updated as follows:

1. Data Cache read miss penalty = $0.004 \times I \times (8/BW+1) \times (1+30/100) = 0.0052 \times I \times (8/BW+1)$ cycles
2. Data Cache write miss penalty = $0.0032 \times I \times (8/BW+1) \times (1+30/100) = 0.00416 \times I \times (8/BW+1)$ cycles
3. Instruction Cache read miss penalty = $0.002 \times I \times (8/BW+1)$ cycles (no change because there is no write in the instruction cache)

To Achieve a CPI of 2, we can write I × CPI = Hit time + miss penalty. Therefore, we can write:

$I \times 2 = I + I \times (8/BW+1) \times (0.0052 +0.00416+0.002)$

Solving the above for BW yields $BW = 8 / 87.028 = 0.0919$ Byte/Cycle

Commented [A13]: 3 Points
Commented [A14]: 3 Points
Commented [A15]: 3 Points
Commented [A16]: 3 Points
Commented [A17]: 3 Points
Commented [A18]: 3 Points
Commented [A19]: 3 Points
Commented [A20]: 3 Points
Commented [A21]: 3 Points
Commented [A22]: 3 Points

Question 3: *(40 Points)*

Using the sequences of 32-bit memory read references, given as word addresses in the following table:

6	214	175	214	6	84	65	174	64	105	85	215

For each of these read accesses, identify the binary address, the tag, the index, and whether it experiences a hit or a miss, for each of the following cache configurations. Assume the cache is initially empty.

A) A direct-mapped cache with 16 one-word blocks.

Address (Decimal)	**Binary Representation** (ignore last 24 bits, all 0)	**Tag**	**Index**	**Cache Data**	**H/M**
6	00000110	0000	0110	M(6)	M
214	11010110	1101	0110	replace M(6) with M(214)	M
175	10101111	1010	1111	M(214),M(175)	M
214	11010110	1101	0110	M(214),M(175)	H
6	00000110	0000	0110	replace M(124) with M(6)	M
84	01010100	0101	0100	M(6),M(175),M(84)	M
65	01000001	0100	0001	M(6),M(175),M(84),M(65)	M
174	10101110	1010	1110	M(6),M(175),M(84),M(65),M(174)	M

Address (Decimal)	Binary Representation (ignore last 24 bits, all 0)	Tag	Index	Cache Data	H/M
64	01000000	0100	0000	M(6),M(175),M(84),M(65),M(174), M(64)	M
105	01101001	0110	1001	M(6),M(175),M(84),M(65),M(174), M(64),M(105)	M
85	01010101	0101	0101	M(6),M(175),M(84),M(65),M(174), M(64),M(105),M(85)	M
215	11010111	1101	0111	M(6),M(175),M(84),M(65),M(174), M(64),M(105),M(85),M(215)	M

Commented [A23]: Each mistake -1 max deductible -8

B) A direct-mapped cache with two-word blocks and a total size of eight blocks.

Address (Decimal)	Binary Representation (ignore last 24 bits, all 0)	Tag	Index	Offset	Cache Data	H/M
6	00000110	0000	011	0	M(6) M(7)	M
214	11010110	1101	011	0	M(214) M(215)	M
175	10101111	1010	111	1	M(214) M(215), M(174) M(175)	M
214	11010110	1101	011	0	M(214) M(215), M(174) M(175)	H
6	00000110	0000	011	0	M(6) M(7), M(174) M(175)	M
84	01010100	0101	010	0	M(6) M(7), M(174) (175), M(84) M(85)	M
65	01000001	0100	000	1	M(6) M(7), M(174) M(175), M(84) M(85), M(64) M(65)	M
174	10101110	1010	111	0	M(6) M(7), M(174) M(175), M(84) M(85), M(64) M(65)	H
64	01000000	0100	000	0	M(6) M(7), M(174) M(175), M(84) M(85), M(64) M(65)	H
105	01101001	0110	100	1	M(6) M(7), M(174) M(175), M(84) M(85), M(64) M(65), M(104)M(105)	M
85	01010101	0101	010	1	M(6) M(7), M(174) M(175), M(84) M(85), M(64) M(65), M(104) M(105)	H
215	11010111	1101	011	1	M(214) M(215), M(174) M(175), M(84) M(85), M(64) M(65), M(104) M(105)	M

Commented [A24]: Each mistake -1 max deductible -8

C) A fully associative cache with two-word blocks and a total size of eight words. Use LRU replacement.

Address (Decimal)	Binary Representation (ignore last 24 bits, all 0)	Index	Offset	Cache Data	H/M
6	00000110	0000011	0	M(6) M(7)	M
214	11010110	1101011	0	M(6) M(7), M(214) M(215)	M
175	10101111	1010111	1	M(6) M(7), M(214) M(215), M(174,175)	M
214	11010110	1101011	0	M(6) M(7), M(214) M(215), M(174) M(175)	H
6	00000110	0000011	0	M(6) M(7), M(214) M(215), M(174) M(175)	H
84	01010100	0101010	0	M(6) M(7), M(214) M(215), M(174) M(175), M(84) M(85) - cache full	M
65	01000001	0100000	1	M(6) M(7), M(214) M(215), M(64) M(65), M(84) M(85) - cache full	M
174	10101110	1010111	0	M(6) M(7), M(174) M(175), M(64) M(65), M(84) M(85) - cache full	M
64	01000000	0100000	0	M(6) M(7), M(174) M(175), M(64) M(65), M(84) M(85) - cache full	H
105	01101001	0110100	1	M(104) M(105), M(174) M(175), M(64) M(65), M(84) M(85) - cache full	M
85	01010101	0101010	1	M(104) M(105), M(174) M(175), M(64) M(65), M(84) M(85) - cache full	H
215	11010111	1101011	1	M(214) M(215), M(214) M(215), M(84) M(85), M(64) M(65), M(104) M(105)	M

Commented [A25]: Each mistake -1 max deductible -8

D) A 2-way set associative cache with one-word block size and total size of 8 words, while applying LRU replacement policy.

Address (Decimal)	Binary Representation (ignore last 24 bits, all 0)	Tag	Set #	Cache Data and Timing Table	H/ M
6	00000110	000001	10		M
214	11010110	110101	10		M
175	10101111	101011	11		M
214	11010110	110101	10		H
6	00000110	000001	10		H
84	01010100	010101	00		M
65	01000001	010000	01		M
174	10101110	101011	10		M
64	01000000	010000	00		M
105	01101001	011010	01		M
85	01010101	010101	01		M
215	11010111	110101	11		M

Set #	Content	Accessed
00	M(84)	6
	M(64)	9
01	~~M(65)~~ M(85)	~~7~~ 11
	M(105)	10
10	M(6)	~~1~~ 5
	~~M(214)~~ M(174)	~~2~~ ~~4~~ 8
11	M(175)	3
	M(215)	12

Commented [A26]: Each mistake -1 max deductible -8