Computer Architecture

BLG 322E

Homework 4 Report

KAAN KARATAŞ
karatask20@itu.edu.tr

Faculty of Computer and Informatics Engineering
Department of Computer Engineering
Date of submission: 10.05.2024

1. Question 1

1.1. a

256 KB = 2^{18} => Main Memory
1 KB = 2^{10} => Cache Memory
16 Byte = 2^4 => Words
2^{10} / 2^4 = 2^6 => Frames
2^6 / 2 = 2^5 => Sets
$18 - 4 - 5$ = 9 => Tag Bits

Physical address => 18-bit
Tag => 9-bit
Cache Set Number => 5-bit
Word number => 4-bit

1.2. b

The tag memory has 64 rows since each frame has one tag ($2^{10}/2^4 => 2^6$).
Tag length => 9 bits.
Extra => Flag, Valid and Dirty bits.
No aging bit is required since FIFO implementation instead of Least Recently Used (LRU).

Therefore, Tag memory size => $2^6 * (9 + 1 + 1 + 1)$ => $64 * 12$

1.3. c

A[0][0 - 9] => $00210 => 00 0000 0010 0001 0000 - Set 1, Frame 1
Cache Miss (Empty Cache)
A[1][0 - 9] => $00410 => 00 0000 0100 0001 0000 - Set 1, Frame 2
Cache Miss (Empty Cache)
A[2][0 - 9] => $00820 => 00 0000 1000 0010 0000 - Set 2, Frame 1
Cache Miss (Empty Cache)
A[3][0 - 9] => $01020 => 00 0001 0000 0010 0000 - Set 2, Frame 2
Cache Miss (Empty Cache)
A[4][0 - 9] => $02010 => 00 0010 0000 0001 0000 - Set 1, Frame 1
Cache Miss (Full Frame) - Replace A[0][0 - 9] - FIFO Implementation
A[5][0 - 9] => $04010 => 00 0100 0000 0001 0000 - Set 1, Frame 2

Cache Miss (Full Frame) - Replace A[1][0 - 9] - FIFO Implementation
A[6][0 - 9] => $08020 => 00 1000 0000 0010 0000 - Set 2, Frame 1
Cache Miss (Full Frame) - Replace A[2][0 - 9] - FIFO Implementation
A[7][0 - 9] => $10020 => 01 0000 0000 0010 0000 - Set 2, Frame 2
Cache Miss (Full Frame) - Replace A[3][0 - 9] - FIFO Implementation

8 Cache misses (4 Empty Cache + 4 Full Frame)
4 Replacements

1.4. d

If we use column-wise iteration, in the first 4 iterations, reading from A[0-3][0] results in cache misses, leading to the caching of A[0-3] arrays into Sets 1-2. After the fourth iteration, following iterations also end up with cache misses, resulting in the replacement of frames within Sets 1-2. This occurs because A[0-1] and A[4-5] arrays both use Set 1, while A[2-3] and A[6-7] arrays both use Set 2. Therefore, there will be (8x10) - 4 = 76 replacements within arrays that share the same set.

On the other hand, row-wise iteration ensues only 4 replacements and 8 cache misses. However, column-wise iteration results in 76 replacements and 80 cache misses, signifying that all read operations are treated as misses. This fact depicts that column-wise iteration incurs much more load when reading data compared to row-wise iteration.

1.5. e

In order to improve the hit ratio, an easy strategy can be implemented by adjusting the start addresses of the arrays so that each array is sequentially located in main memory. To exemplify, if we assume A[0] starts at address $0000 then A[1] should follow with the address of $000A. This arrangement makes sure that all matrices shall fit within 5 frames in the cache, considering there are 80 words and a block size of 16 words. Thus, with only just 5 cache misses occurring, all array elements can be cached. Allowing the 5 blocks to be cached in different frames and sets, thus avoiding replacement of blocks in the following iterations.

2. Question 2

64 KB = 2^{16} => Main Memory
1 KB = 2^{10} => Cache Memory
8 Byte = 2^3 => Words
2^{10} / 2^3 = 2^7 => Frames
2^7 / 2 = 2^6 => Sets
$16 - 3 - 6$ = 7 => Tag Bits

Physical address => 16-bit
Tag => 9-bit
Cache Set Number => 6-bit
Word number => 3-bit

C = A + B:
- Read A
- Read B
- Write to C

2.1. a

For the fastest case, all variables should be in the same block of main memory to be transferred to the same cache frame on the first miss. Set numbers and tag values should be the same.

A => 0000 0000 0000 0000 or $0000 => Read miss t_b
B => 0000 0000 0000 0001 or $0001 => Read hit t_c
C => 0000 0000 0000 0010 or $0010 => Write hit t_m

Total access time = Block transfer time + Cache access time + Memory access time
t_a = t_b + t_c + t_m

2.2. b

For the slowest case, all variables should be in different blocks of main memory to be transferred to different cache frames. Set numbers should be the same but tag values should be different.

A => 0000 0000 0000 0000 or $0000 => Read miss t_b

B => 0000 0100 0000 0000 or $0001 => Read miss t_b

C => 0000 1000 0000 0000 or $0010 => Write miss t_b + t_m

Total access time = 3 * Block transfer time + Memory access time

t_a = t_b + t_b + t_b + t_m