CS 224

Fall 2023–2024

Lab 6 Preliminary Report

Hüseyin Uzun

21702559

Section 3

Part 1

No:	Cache Size KB	N way cache	Word Size in bits	Block size (no. of words)	No. of Sets	Tag Size in bits	Index Size (Set No.) in bits	Word Block Offset Size in bits1	Byte Offset Size in bits2	Block Replacement Policy Needed (Yes/No)
1	128	1	32	4	2^13	15	13	2	2	No
2	128	4	32	16	2^9	17	9	4	2	Yes
3	128	Full	32	16	1	26	0	4	2	Yes
4	256	2	64	8	2^11	15	11	3	3	No
5	256	4	64	32	2^8	16	8	5	3	Yes
6	256	Full	16	16	1	27	0	4	1	Yes

Part 2

Memory Address Accessed (hex)	Set No.	Hit (yes/no)
00 00 20 24	00	No
00 00 20 42	00	No
00 00 20 68	01	No
00 00 20 04	00	No
00 00 20 0C	01	No
00 00 20 4C	01	No

Part 3

Memory Address Accessed (hex)	Set No.	Hit (yes/no)
00 00 20 2C	01	No
00 00 20 48	01	No
00 00 20 44	00	No
00 00 20 0C	01	No
00 00 20 04	00	No
00 00 20 0C	01	Yes

Part 4

a)

Physical memory = 2^32

Block offset = log(Block size) = log(2^6) = 6

Index size = 1024/64 = 16

16 / 2 = 8

log8= 3

Tag = 32 – 9 = 23

b) 64 byte

tag 23 bits

V 1 bit

512 + 23 + 1 = 536 bits

c)

set size = 8 x 536

SRAM size = 8 x 8 x 536

d)

There is no change to SRAM size because it only changes cache's performance.

Part 5

```
# CS224
# Lab 6 Preliminary Part 5.
# Section 3
# Hüseyin Uzun
# 21702559

.text
main:
        li $v0, 4
        la $a0, prompt
        syscall



        li $v0, 5
```

```
        syscall
        move $s1, $v0
        mul $s0, $s1, $s1
        li  $a0, 4
        mul $a0, $a0, $s0

        li $v0, 9
        syscall
        move $s2, $v0



        move $s7, $s0
        move $s5, $s2
        li $s6, 1

    loop:
        sw $s6, 0($s5)
        addi $s7, $s7, -1
        addi $s6, $s6, 1
        addi $s5, $s5, 4
        bne $s7, $zero, loop



menu:
    li $v0, 4
    la $a0, menuPrompt
    syscall
```

```
    li $v0, 5
    syscall

    beq $v0, $zero, exit
    beq $v0, 1, row
    beq $v0, 2, column
    beq $v0, 3, element

exit:
        li $v0, 4
        la $a0, exitPrompt
        syscall
        li $v0, 10
        syscall
row:

        move $s3, $s2
        li $t1, 4
        mul $s4, $t1, $s1
        li $t2, 0
        li $a0, 0
        move $t3, $s1
rowAddLoop:

        lw $a1,  0($s3)
        add $a0, $a0, $a1
        add $s3, $s3, $s4
```

```
        subi $t3, $t3, 1
        bne $t3, $zero, rowAddLoop

        li $v0, 1
        syscall

        la $a0, comma
        li $v0, 4
        syscall

        li $a0, 0
        move $t3, $s1
        addi $t2, $t2, 1
        move $s3, $s2
        mul $t6, $t1, $t2
        add $s3, $s3, $t6
        bne $t2, $s1, rowAddLoop
        j menu

column:
        move $s3, $s2
        li $t1, 4
        mul $s4, $t1, $s1
        move $t3, $s1
        move $t4, $s1
        li $t2, 0
        li $a0, 0
```

```
columnAddLoop:
        lw $a2,  0($s3)
        add $a0, $a0, $a2
        add $s3, $s3, $t1
        addi $t2, $t2, 1
        bne $t2, $s1, columnAddLoop

        li $v0, 1
        syscall

        la $a0, comma
        li $v0, 4
        syscall

        li $a0, 0
        li $t2, 0
        subi $t3, $t3, 1
        bne $t3, $zero, columnAddLoop
        j menu

element:
        move $s3, $s2
        li $t2, 4

        li $v0, 4
        la $a0, rowPrompt
```

```
syscall
li $v0, 5
syscall
move $t0, $v0

li $v0, 4
la $a0, columnPrompt
syscall
li $v0, 5
syscall
move $t1, $v0

addi $t1, $t1, -1
addi $t0, $t0, -1
mul $t1, $t1, $s4
mul $t0, $t0, $t2

add $t1, $t0, $t1
add $s3, $s3, $t1



li $v0, 4
la $a0, elementPrompt
syscall
lw $a0, 0($s3)
li $v0, 1
syscall
```

j menu

```
 .data
matrix: .space 1024
prompt: .asciiz "Enter matrix size: "
exitPrompt: .asciiz "Program is done."
result: .asciiz "Summation: "
menuPrompt: .asciiz "\n1. Obtain summation of matrix elements row-major (row by row) summation\n2. Obtain summation of matrix elements column-major (column by column) summation\n3. Display desired elements of the matrix by specifying its row and column member\n0. Exit\n "
rowPrompt: .asciiz "Row: "
columnPrompt: .asciiz "Column: "
elementPrompt: .asciiz "The element is: "
comma: .asciiz " , "
```