CS224
Section No.: 1
Spring 2020
Lab No.: 6
Your Full Name/Bilkent ID: Erdem Ege Eroğlu / 21601636

1)

Case 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 bits[1]	Byte Offset Size in bits[2]	Block Replacement Policy Needed (Yes/No)
1	4	1	32	4	256	17	8	2	2	no
2	4	2	32	4	128	18	7	2	2	yes
3	4	4	32	8	32	19	5	3	2	yes
4	4	Full	32	8	1	24	0	3	2	yes
5	32	1	16	4	4096	14	12	2	1	no
6	32	2	16	4	2048	15	11	2	1	yes
7	32	4	8	16	512	16	9	4	0	yes
8	32	Full	8	16	1	25	0	4	0	yes

2)

a-

Instruction	Iteration No.				
	1	2	3	4	5
lw $t1, 0xA4($0)	Compulsory Miss	Conflict Miss	Conflict Miss	Hit	Hit
lw $t2, 0xAC($0)	Compulsory Miss	Hit	Hit	Hit	Hit
lw $t3, 0xA8($0)	Compulsory Miss	Hit	Hit	Hit	Hit

b-

main memory size: 2^32 bits

inst length: 32 bits

Cache is direct mapped (N=1) and has 16 words

Number of blocks in cache = 16 / 4 =4 blocks

Number of sets 4 / 1 = 4

1 word 4 byte

Byte offset = log2(2^2) = 2

Number of words in a single block = 4

Block offset = log2(4) = 2

Index size is log2(2^2) = 2

Tag size 32 – (2+2+2) = 26

32*4+26+1=155bits

c-

Two 4x1 mux

zero or gate

one and gate

one equality comparator

3)

a-

Instruction	**Iteration No.**				
	1	**2**	**3**	**4**	**5**
lb $t1, 0xA4($0)	Compulsory Miss	Capacity Miss	Capacity Miss	Capacity Miss	Capacity Miss
lb $t2, 0xAC($0)	Capacity Miss	Capacity Miss	Capacity Miss	Capacity Miss	Capacity Miss
lb $t3, 0xA8($0)	Capacity Miss	Capacity Miss	Capacity Miss	Capacity Miss	Capacity Miss

b-

C = 8. N = 2 So, 8/4=4.

Log2(4) = 2

block size is 1 word

Block offset is 0.

Tag = 32-2-2=28

28+1+30+1+30 =118 bits

118x4=472 bits

c-

One 2x1 mux

One or gate

Two and gate

Two equality comparators

4)

```
.data
	menu: .asciiz "Type a selection number.\n1 for entering the dimension and fill the matrix.\n2 for entering position to find\n3 for exitting!!\nSelection number: "
	warning: .asciiz "Type incorrect selection. Please try again.\n"
	newLine: .asciiz "\n"
	size: .asciiz "Type a number to be size(N): "
	takeValues: .asciiz "Enter values for matrix: \n"
	position: .asciiz "Enter the row anc col number(x,y): "
	rowSumText: .asciiz "Row major summation: "
	colSumText: .asciiz "Col major summation: "
	dene: .asciiz " AAAAAAAAA"
.text

top:
	# menu
	la $a0, newLine
	li $v0, 4
	syscall

	la $a0, menu
	li $v0, 4
	syscall

	li $v0, 5 #take selection from usee
	syscall
	move $a0,$v0

	beq $a0, 1, create
	beq $a0, 2, find
	beq $a0, 3, exit

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

```
        syscall

        j top

create:
        # take size
        la $a0, size
        li $v0, 4
        syscall
        li $v0, 5
        syscall
        move $s0,$v0 # s0 = N

        mul $s1, $s0, $s0 #s1 = NxN

        move $a0, $s1 # array is allocated with size
        li $v0, 9
        syscall

        move $s2, $v0 # s2 = array

        la $a0, takeValues
        li $v0, 4
        syscall

        addi $t2, $zero, 1 # i
        addi $t3, $zero, 1 # j

        move $t0, $s0
        move $t1, $s2
        move $t0, $s1

whileI: # i
        bgt $t2, $s0, whileIDone
        whileJ: # j
                bgt $t3, $s0, whileJDone
                move $t4, $t3
                subi $t4, $t4, 1
                mul $t4, $t4, $s0
                sll $t4, $t4, 2
                move $t5, $t2
                subi $t5, $t5, 1
                sll $t5, $t5, 2
                add $t6, $t4, $t5
```

```
                add $t7, $t6, $t1

                li $v0, 5
                syscall
                sw $v0, 0($t7)

                addi $t3, $t3, 1
                j whileJ
        whileJDone:
                addi $t2, $t2, 1
                addi $t3, $zero, 1
                j whileI
whileIDone:
        j top

find:
        la $a0, position
        li $v0, 4
        syscall

        li $v0, 5
        syscall
        move $t0, $v0 #  i
        li $v0, 5
        syscall
        move $t1, $v0 # j

        subi $t1, $t1, 1
        mul $t1, $t1, $s0
        sll $t1, $t1, 2
        subi $t0, $t0, 1
        sll $t0, $t0, 2
        add $a0, $t1, $t0

        add $a0, $a0, $s2

        lw $a0, 0($a0)
        li $v0, 1
        syscall
        j top

exit:
        li $v0, 10
        syscall
```