CS224
Section No: 1
Fall 2019
Lab No: 6
Oğuz Kaan İmamoğlu / 21702233

Preliminary Work

1.

No.	Cache Size KB	N way cache	Word Size	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	64	1	32 bits	4	2^{12}	16	12	2	2	No
2	64	2	32 bits	4	2^{11}	17	11	2	2	Yes
3	64	4	32 bits	8	2^{9}	18	9	3	2	Yes
4	64	Full	32 bits	8	1	27	0	3	2	Yes
9	128	1	16 bits	4	2^{14}	15	14	2	1	No
10	128	2	16 bits	4	2^{13}	16	13	2	1	Yes
11	128	4	16 bits	16	2^{10}	17	10	4	1	Yes
12	128	Full	16 bits	16	1	27	0	4	1	Yes

[1] **Word Block Offset Size in bits:** Log_2(No. of words in a block)
[2] **Byte Offset Size in bits:** Log_2(No. of bytes in a word)

2.

a)

Instruction	Iteration No.				
	1	2	3	4	5
lw $t1, 0x4($0)	Compulsory	Hit	Hit	Hit	Hit
lw $t2, 0xC($0)	Compulsory	Hit	Hit	Hit	Hit
lw $t3, 0x8($0)	Hit	Hit	Hit	Hit	Hit

b)

Valid -> 1 bit

Data -> 32 bits

Tag -> 27 bits

There are 4 sets, so total cache memory size: [(32*2) + 1 + 27] * 4 = 368

c)

1 2:1 mux

1 AND Gates

1 Equality comparators

3.

a)

Instruction	**Iteration No.**				
	1	**2**	**3**	**4**	**5**
lw $t1, 0x4($0)	Compulsory	Capacity	Capacity	Capacity	Capacity
lw $t2, 0xC($0)	Compulsory	Capacity	Capacity	Capacity	Capacity
lw $t3, 0x8($0)	Capacity	Capacity	Capacity	Capacity	Capacity

b)

There are 1 word, it means byte offset is 2, there are 1 set so set is 0 and tag is 30 bits. Thus, result is (1+30+32)*2 = 126

c)

1 2:1 mux

2 AND Gates

1 OR Gate

2 Equality comparators

4)

AMAT = T_{L1}+ $Miss_{L1}$(T_{L2}+$Miss_{L2}$*T_{MM}) = 2.2
4 GHz = 0.25ns

Time needed = 10^{12}*0.25*2.2ns = 550s

5)

```
# Oguz Kaan Imamoglu
# Course: CS224
# Section: 1
# Lab: 6

menu:
        la $a0,info1      # print string before result
    li $v0,4
    syscall

    la $a0,info2          # print string before result
    li $v0,4
    syscall

        la $a0,info3      # print string before result
    li $v0,4
    syscall

        la $a0,info4      # print string before result
    li $v0,4
    syscall

    la $a0,info5          # print string before result
    li $v0,4
    syscall

    la $a0,info6          # print string before result
    li $v0,4
    syscall
```

```
la $a0,choice          # print string before result
    li $v0,4
syscall

li $v0 5
    syscall
    move $t7 $v0  #t7 is the option

beq  $t7,1,first
    beq  $t7,2,second
    beq  $t7,3,third
    beq  $t7,4,fourth
    beq  $t7,5,fifth
beq  $t7,6,sixth


first:
la $a0,infofirst        # print string before result
li $v0,4
syscall


li $v0 5
    syscall
    move $s0 $v0  #s0 is size


second:
sll $a0 $v0 2    # sll performs $a0 = $v0 x 2^2
li $v0 9    #9 is the system code for service(sbrk) whoes work is
    syscall     #to allocate dynamic memory
```

```
    mul $s1 $s0 3
    li  $t0 0 #counter
    li  $t1 0 #base adress of array
    addi $sp, $sp, -4 # make space on stack to
    sw  $t1, 0($sp)

    loop2:

    la $a0,prompt2 # print string before result
    li $v0,4
    syscall

    li $v0 5
    syscall
sw $v0 0($t1)

addi $t1 $t1 1
addi $t2 $t2 4

    blt $t0 $s1 loop2
    lw  $t1, 0($sp)
    addi $sp, $sp, 4
    j menu

third:
j menu

fourth:
la $a0,prompt4      # print string before result
    li $v0,4
```

```
    syscall


li $v0 5
    syscall
    move $t7 $v0  #t7 is the row number


    li  $t0 0 #counter
    li  $s4 0 #result
    addi $sp, $sp, -4 # make space on stack to
    sw  $t1, 0($sp)
    subi $t7 $t7 1
    mul $t7 $t7 4
    add $t1 $t1 $t7


    loop4:
    lw $t4 0($t1)
    add $s4 $s4 $t4
    addi $t0 $t0 1
    mul $t5 $s0 4
add $t1 $t1 $t5
bgt $t0 $s0 menu
lw  $t1, 0($sp)
    addi $sp, $sp, 4
j loop4



fifth:
la $a0,prompt5      # print string before result
    li $v0,4
    syscall
```

```
li $v0 5
    syscall
    move $t7 $v0  #t7 is the col number

    li  $t0 0 #counter
    li  $s5 0 #result
    addi $sp, $sp, -4 # make space on stack to
    sw  $t1, 0($sp)
    mul $t6 $s0 4
    subi $t7 $t7 1
    mul $t7 $t7 $t6
    add $t1 $t1 $t7

    loop5:
    lw $t8 0($t1)
    add $s5 $s5 $t8
    addi $t1 $t1 4
    addi $t0 $t0 1
    bgt $t0 $s0 menu
    lw  $t1, 0($sp)
    addi $sp, $sp, 4
    j loop5



sixth:
la $a0,prompt4      # print string before result
    li $v0,4
    syscall

li $v0 5
```

```
    syscall
    move $t7 $v0  #t7 is the row number

    la $a0,prompt5 # print string before result
    li $v0,4
    syscall

li $v0 5
    syscall
    move $t9 $v0  #t9 is the col number

    subi $t9 $t9 1
    subi $t7 $t7 1

addi $sp, $sp, -4 # make space on stack to
    sw  $t1, 0($sp)

    mul $s5 $s0 4
add $t9 $t9 $s5
mul $t7 $t7 4
add $t1 $t1 $t9
add $t1 $t1 $t7
lw $s7 0($t1)

li  $v0, 1          # service 1 is print integer
    move $a0, $s7 # load desired value into argument register $a0, using pseudo-op
    syscall

j menu

                .data
```

```
prompt2:.asciiz "\n Enter matix element \n"
prompt4:.asciiz "\n Enter the row number \n"
prompt5:.asciiz "\n Enter the col number \n"
info1:.asciiz "\n 1)Enter the size of matrix (N) \n"
info2:.asciiz "2)Enter Matrix Elements \n"
info3:.asciiz "3)Display the content\n"
info4:.asciiz "4)Row major sum \n"
info5:.asciiz "5)Col major sum\n"
info6:.asciiz "6)Display desired elements \n"
choice:.asciiz "7) Enter your choice \n"
infofirst:.asciiz "\n Enter the size of matrix (N) \n"
```