CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

BILKENT UNIVERSITY

COMPUTER SCIENCE

CS 224 : COMPUTER ORGANIZATION

PRELIMINARY DESIGN REPORT

LAB 6

SECTION 4

ALPEREN CAN

21601740

19.12.2019

CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

QUESTION 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

CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

QUESTION 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)

V = 1 Bit

2 Data Sections, each of 32 bits.

Tag = 27 bits

Therefore, 1 block is 92 bits.

Since there are 4 blocks, total cache memory size is 92 x 4 = 368 bits.

c)

1 AND Gate

1 Equality Comparator

1 2x1 Multiplexer

CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

QUESTION 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)

To implement LRU,

Cache = 1 bit

V = 1 bit

Tag = 30 bits

Data = 32 bits

There are 2 V sections, 2 tag sections and 2 data sections in the cache. Therefore, total cache memory is:

(1 + 30 + 32) x 2 + 1 = 127 bits

c)

2 AND Gates

1 OR Gate

2 Equality Comparators

1 2x1 Multiplexer

CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

QUESTION 4

Access time for L1 = 1 clock cycle

Access time for L2 = 4 clock cycles

Access time for Main Memory = 40 clock cycles

Miss rate for L1 = 20%

Miss rate for L2 = 5%

AMAT = $1 + 0.20 [4 + 0.05 \times (40)]$ = 2.2 clock cycles

With 4 GHz clock rate, the time needed for 10^{12} instructions is:

$10^{12} \times 2.2 \times (0.25 \times 10^{-9}) = 550$ seconds

CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

QUESTION 5

```
# Author : Alperen CAN

.text

start:      jal menu
            li $v0,10
            syscall

menu:       la $a0,prompt3
            li $v0,4
            syscall

            la $a0,prompt4
            li $v0,4
            syscall

            la $a0,prompt5
            li $v0,4
            syscall

            la $a0,prompt6
            li $v0,4
            syscall

            la $a0,prompt7
            li $v0,4
            syscall
```

```
                la $a0,prompt8
                li $v0,4
                syscall

                la $a0,prompt9
                li $v0,4
                syscall

                la $a0,prompt10
                li $v0,4
                syscall

                li $v0,5
                syscall

                beq $v0,1,option1
                beq $v0,2,option2
                beq $v0,3,option3
                beq $v0,4,option4
                beq $v0,5,option5
                beq $v0,6,option6
                beq $v0,7,option7

option1:        la $v0,4
                la $a0,prompt1
                syscall
                li $v0,5
                syscall
```

```
                blez $v0,retake         # N should be greater than zero.
                move $t0,$v0            # t0 is the matrix size N in terms of dimension.

                la $a0,prompt19
                li $v0,4
                syscall

                move $a0,$t0
                li $v0,1
                syscall

                la $a0,prompt21
                li $v0,4
                syscall

                j menu

retake:         la $a0,prompt23         # N should be greater than zero
                li $v0,4
                syscall

                j option1

option2:        blez $t0,notValid       # The current N should be greater than zero.

                mul $t1,$t0,$t0
                mul $a0,$t1,4
```

```
                li $v0,9            # Allocates the array by syscall 9
                syscall

                move $t1,$v0        # t1 is starting address of the array

                jal initArray
                j menu

notValid:       la $a0,prompt22     # N should be greater than zero
                li $v0,4
                syscall

                j menu

option3:        jal accessElement
                j menu

option4:        jal rowMajorSum
                j menu

option5:        jal columnMajorSum
                j menu

option6:        jal displayRowCol
                j menu

option7:        li $v0,10
                syscall
```

CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

```
##################################### SUBPROGRAM initArray ###################################


initArray:      mul $t2,$t0,$t0     # t2 is number of elements
                move $t3,$t1        # t3 is the address

                la $a0,prompt20
                li $v0,4
                syscall

initLoop:       beqz $t2,initLoopDone
                la $a0,prompt2
                li $v0,4
                syscall

                li $v0,5
                syscall

                sw $v0,0($t3)
                addi $t3,$t3,4
                subi $t2,$t2,1
                j initLoop

initLoopDone:   jr $ra
```

CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

```
################################## SUBPROGRAM columnMajorSum ##################################

columnMajorSum:     mul $t2,$t0,$t0         # t2 is number of elements
                    move $t3,$t1            # t3 is the address
                    li $t4,0                # t4 is the sum

columnSumLoop:      beqz $t2,columnSumLoopDone
                    lw $t5,0($t3)
                    add $t4,$t4,$t5
                    addi $t3,$t3,4
                    subi $t2,$t2,1
                    j columnSumLoop

columnSumLoopDone:  la $a0,prompt11
                    li $v0,4
                    syscall

                    li $v0,1
                    move $a0,$t4
                    syscall

                    jr $ra

################################## SUBPROGRAM rowMajorSum  ##################################

rowMajorSum:        mul $t2,$t0,$t0         # t2 is number of elements
                    li $t4,0                # t4 is the sum
                    mul $t6,$t0,4           # t6 is Nx4
```

```
                li $t8,0
                li $s0,0

rowMajLoop1:    beq $t8,$t0,rowMajLoop1Done
                move $t7,$t0
                mul $s0,$t8,4
                move $t3,$t1            # t3 is the address
                add $t3,$t3,$s0

    rowMajLoop2:        beqz $t7,rowMajLoop2Done
                        lw $t5,0($t3)
                        add $t4,$t4,$t5
                        add $t3,$t3,$t6
                        subi $t7,$t7,1
                        j rowMajLoop2

    rowMajLoop2Done:    addi $t8,$t8,1
                        j rowMajLoop1

rowMajLoop1Done:    la $a0,prompt12
                    li $v0,4
                    syscall
                    move $a0,$t4
                    li $v0,1
                    syscall
                    jr $ra
```

CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

```
################################### Subprogram accessElement ###################################

accessElement:      move $t2,$t0          # t2 is N
                    move $t3,$t1          # t3 is the address

                    la $a0,prompt13
                    li $v0,4
                    syscall

                    la $a0,prompt14
                    li $v0,4
                    syscall

                    li $v0,5
                    syscall
                    move $t4,$v0          # t4 is the row no

                    la $a0,prompt15
                    li $v0,4
                    syscall

                    li $v0,5
                    syscall
                    move $t5,$v0          # t5 is the column no

                    subi $t6,$t5,1        # t6 is column no-1
                    mul $t7,$t0,4
```

```
                mul $t7,$t6,$t7         # t7 is (Nx4)x(colNo-1)
                subi $t6,$t4,1
                mul $t6,$t6,4
                add $t3,$t3,$t7
                add $t3,$t3,$t6

                lw $t8,0($t3)

                la $a0,prompt16
                li $v0,4
                syscall

                li $v0,1
                move $a0,$t8
                syscall

                jr $ra

################################## Subprogram displayRowCol ######################################
##This subprogram displays either the row or column. ( Due to the comment of Mr Özcan Öztürk at Unilica)##
 ##################################################################################################

displayRowCol:  move $t2,$t0            # t2 is N
                move $t3,$t1            # t3 is the address

                la $a0,prompt13
                li $v0,4
                syscall
```

```
                la $a0,prompt17
                li $v0,4
                syscall

askAgain:       la $a0,prompt14
                li $v0,4
                syscall

                li $v0,5
                syscall
                move $t4,$v0            # t4 is row no

                la $a0,prompt15
                li $v0,4
                syscall

                li $v0,5
                syscall
                move $t5,$v0            # t5 is column no

                beq $t4,$t5,invalid
                beq $t4,0,displayCol
                beq $t5,0,displayRow

invalid:        la $a0,prompt18
                li $v0,4
                syscall
                b askAgain
```

```
displayCol:         subi $t6,$t5,1          # t6 is colno-1
                    mul $t7,$t0,4
                    mul $t6,$t6,$t7
                    add $t3,$t3,$t6         # array address now starts from top of column

displayColLoop:     beqz $t2,displayColLoopDone

                    lw $a0,0($t3)
                    li $v0,1
                    syscall

                    la $a0,space
                    li $v0,4
                    syscall

                    addi $t3,$t3,4
                    subi $t2,$t2,1
                    j displayColLoop
displayColLoopDone: jr $ra

displayRow:         move $t2,$t0            # t2 is N
                    move $t3,$t1            # t3 is the address
                    subi $t6,$t4,1
                    mul $t6,$t6,4
                    add $t3,$t3,$t6         # array address now starts from the beginning of the row
                    mul $t7,$t0,4           # t7 is Nx4
```

CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

```
displayRowLoop:         beqz $t2,displayRowLoopDone

                        lw $a0,0($t3)

                        li $v0,1

                        syscall

                        la $a0,space

                        li $v0,4

                        syscall

                        add $t3,$t3,$t7

                        subi $t2,$t2,1

                        j displayRowLoop

displayRowLoopDone:     jr $ra
```

CS 224
Section 4
Fall 2019
Lab 6
Alperen CAN / 21601740

```
.data
space:          .asciiz " "
newLine:        .asciiz "\n"
prompt1:        .asciiz "Enter the matrix size N in terms of dimension : "
prompt2:        .asciiz "Enter number: "
prompt3:        .asciiz "\nWhat do you want to do?"
prompt4:        .asciiz "\n(1) Enter the size N for an NxN matrix "
prompt5:        .asciiz "\n(2) Allocate the array with the current N value "
prompt6:        .asciiz "\n(3) Access the matrix element at [x,y]"
prompt7:        .asciiz "\n(4) Obtain Row-Major summation"
prompt8:        .asciiz "\n(5) Obtain Column-Major summation"
prompt9:        .asciiz "\n(6) Display the whole row or column elements"
prompt10:       .asciiz "\n(7) Exit\n"
prompt11:       .asciiz "The column-major sum is : "
prompt12:       .asciiz "The row-major sum is : "
prompt13:       .asciiz "Note that row and column numbers start from 1"
prompt14:       .asciiz "\nEnter row no   : "
prompt15:       .asciiz "Enter column no: "
prompt16:       .asciiz "The searched element is : "
prompt17:       .asciiz "\nTo display row, enter 0 to col no. To display col,enter 0 to row no."
prompt18:       .asciiz "This subprogram displays either the row or column. Enter 0 to one of them."
prompt19:       .asciiz "\n*** N is set to "
prompt20:       .asciiz "Initialize the array\n"
prompt21:       .asciiz " ***"
prompt22:       .asciiz "\n*** First, please enter an N value greater than zero. ***"
prompt23:       .asciiz "*** The N value should be greater than zero ***\n."
```