

Bashundhara
Exercise Book
Write Your Future

Sadman
~~(A)~~ (G) MICRO (A)
011221592

C.W
23.1.24

CA

(Class → If) software CS ← 27 TM *

Introduction

→ for assignment

→ answer (your work)

C.W
27.1.24

CA

* MIPS

→ sum = a + b [regular]

→ { mov A, a ← block off H language
 mov B, b [Assembly] ←
 add A, B : add. ←
 mov sum, C : add sum ←
 ← test.

ROM → BIOS

→ 64 SRAM DRAM & disk

→ Instruction set (RISC)

→ Instructions → memory

→ [memory bus] → [bus] → [bus]
 ← monitor

* MIPS → 32 register (\$0 - \$31)

↓
WikiBooks.org

(Number, name, comments)

* (\$2 - \$25) → important for lectures.

[Registers] etc. → memory

* Hello World → print?

→ [Glossary] etc. → memory

.data

myStr: .ascii "Hello World\n"

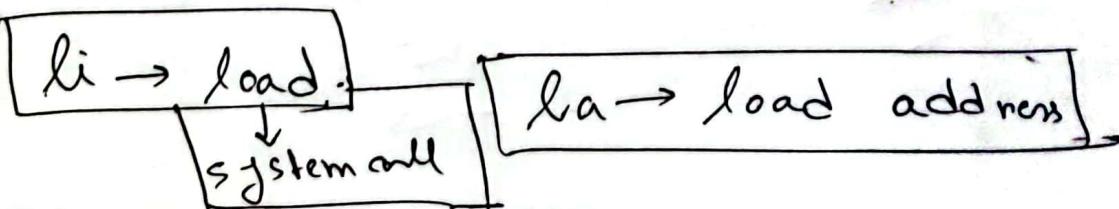
.text

main:

li \$v0, 4 → value

la \$a0, myStr → address

syscall → execute/print



`li $V0, 10` → exit
~~store value~~ → ~~exit~~
~~syscall~~

~~CW
3.2.2.4~~

~~Performance~~
~~Instruction~~

* Response time

\swarrow op~~d~~ code \swarrow operand

~~CA~~

* Throughput

\rightarrow performance = $\frac{1}{\text{Execution time}}$

\rightarrow X is w times faster than Y.

$$\frac{\text{Execution time } X}{\text{Execution time } Y} = \frac{\text{Performance } X}{\text{Performance } Y} \Rightarrow w$$

* Fetch → Decryption → Load → perform → save

↓
output

CPU Time → CPU clock cycles \times clock cycles time

$$CPU \text{ time} = \frac{CPU \text{ clock cycles}}{\text{clock rate}}$$

$$\text{clock cycles} = \text{Instruction count} \times$$

$$\boxed{\text{cycles per instruction}} \rightarrow CPI$$

$$CPU \text{ time} = \frac{\text{Instruction count} \times CPI}{\text{clock rate}}$$

$$\times \text{switch wait time} \times \text{clock cycle time}$$

$$\times \text{Instruction count} \times CPI$$

$$\frac{\text{Instruction count} \times CPI}{\text{clock rate}} \times \text{switch wait time}$$

\downarrow wait time \leftarrow waiting for bus \leftarrow waiting for memory \leftarrow waiting for disk

$$(\text{clocks} \times \text{cycles per clock}) \times \text{wait time}$$

CW
6.2.2

CA

* Performance depends on:

- Algorithm
- programming language
- compiler
- instruction set architecture

CW
10.2.2

* operation OP₁, OP₂, OP₃

OP₁ = OP₂ operation OP₃.

$$2^9 = 512$$

* h = 230;

$$A[10] = h + B[i]$$

hi

~~add~~, \$S₃, ~~\$S₁~~, 230

h = \$S₂

A = \$S₀

B = \$S₂

i = \$S₃

h(\$t0), \$S₂, \$S₁, \$S₀, \$S₃, \$t0

~~W~~ ~~15~~
meili \$12, \$53, ~~\$100~~ ~~16~~
dw \$10, ~~10~~ ~~\$10~~ (\$52)
add ~~10~~ ~~10~~ \$10, \$52, ~~\$10~~ ~~16~~
sw \$10, 40 (\$50)

~~W~~ ~~15~~
£70 - £70 = £70 walking ~~8~~
£70 walking - £70 = £70

$$\boxed{£50 = 8}$$

- 28 = 2
028 = A
- 28 = 0
£28 = 1
does not fit

$$[1]A + 28 = [0]A$$

088 - ~~28~~ = 28 ~~id~~

(28) ~~100~~ ~~100~~ ~~100~~ ~~100~~
014, 014, 014, 014 ~~660~~

11.2.24

online class - (2, 2, 3)

CA

C to Mips

$a = b + c + d$

$\rightarrow \text{add } a, b, c$

$\text{add } a, a, d$

~~add \$d + \$c = \$a~~

~~\$a = \$b + \$c~~

~~destination variable~~

~~variable~~

operation name

~~\$a = \$a + \$d~~

~~2 source variable~~

32 registers in mips:

\$0, \$1, ..., \$31

~~\$0~~

\$s0, \$s1 \rightarrow save register

\$t0, \$t1 \rightarrow temporary register.

\$0 \rightarrow zero register.

$f = (g+h) - (i+j)$

add \$t0, \$s1, \$s2

$g \rightarrow \$s_1$

$h \rightarrow \$s_2$

add, \$t1, \$s3, \$s4

$i \rightarrow \$s_3$

$j \rightarrow \$s_4$

sub, \$s0, \$t0, \$t1

P.T.O

(E_{i+1}) addition

* $a = b + 2$

addi \$50, \$52, 2

→ add immediate
instruction

* $a = b - 2$

addi \$50, \$52, -2

→ sign in register 52

* $a = 7$

lfp ... lfp op

addi \$50, \$zero, 7

* $a = 7 + 2x$ (constant) \leftarrow lfp lfp

addi \$50, \$zero, 7

addi \$50, \$50, 2

* $a = b + c$ [b and c] $(i+i) - (d+p) = 2$

addi \$50, \$51, 2

and \$50, \$51, \$52

$i \leftarrow i$

lfp lfp op d = 2

PT.

op

* $a = b + c$ [b or c]

ori \$s_0, \$s_1, \$s_2

and \$s_0, \$s_1, \$s_2

* $a = b + 2$

ori, \$s_0, \$1, 2

and \$s_0, \$s_1, \$s_2 : A H

* $a = (b+c)$ [nor]

nor \$s_0, \$s_1, \$s_2

* $a = b \gg 2$ [shift right 2 times]

A and \$s_0, \$s_1, \$s_2 : A H

srl \$s_0, \$s_1, 2 + s.c. mode

Program B Performance down fast with

Relative performance

$$\frac{P_x}{P_y} = \frac{E_j}{E_x} \text{ (Response time / Execution time)}$$

($E_j = O(N \cdot S + S^2)$)

$$\frac{P_x}{P_y} = \frac{C_p}{C_j} \text{ (Throughput / Clock Rate)}$$

$C_p = \frac{C_p}{C_j}$ = clock cycles

- * CPU time = CPU clock \times clock cycle time
 $\frac{\text{cycles}}{2 \times 10^9}$ \times 2×10^{-9} \approx
- * CPU time = $\frac{\text{CPU clock cycles}}{\text{clock rate}}$ \approx

A: 2 GHz clock, 10s CPU time

Designing B

- \rightarrow 6s CPU time
- \rightarrow Can do faster clock but comes 2×10^9 clock cycles

How fast must computer B clock be?

$$\rightarrow \text{CPU clock cycles} \rightarrow 10 \times (2 \times 10^9) \times 6 \times 10^9$$

$$\text{new clock cycle} \rightarrow 2.4 \times 10^{10} [1.2 \times \text{cc A}]$$

$$\text{so new rate}_{\text{out}} \rightarrow \frac{x^9}{x^9}$$

$$\text{clock rate} = \frac{2.4 \times 10^{10}}{6} = 4 \times 10^9 \text{ Hz}$$

* ~~CPU time = CPU clock cycles ×~~

* Clock cycles = Instruction count × Cycles per instruction

* CPU time = Instruction Count × CPI × Clock cycle time

Instruction count × CPI

→ ~~Instruction count × CPI × Clock Rate~~

A: Cycle time = 250 ps & CPI = 2.0
Clock rate = $\frac{1}{250 \text{ ps}}$
(ex. B) n = 500 ps , CPI = 2.2

Same for A. ➤ Which is faster? p.v.A
➤ How much

A

CPU time

$$= g_c \times 2.0 \times 250 \text{ ps}$$

$$= g_c \times 500 \text{ ps}$$

\downarrow faster

B

CPU time

$$= g_c \times 2.2 \times 500 \text{ ps}$$

$$= g_c \times 600 \text{ ps}$$

$$\frac{B}{A} = \frac{g_c \times 600}{g_c \times 500}$$
$$\Rightarrow B = 1.2 \times A$$

#

Instruction	add	sub	mul
CPI for Instruction	2	2	3
IC (in sequence)	1	2	2
IC = 2 + 4 + 2 = 8			

Avg CPI for seq 1 & 2
Avg 9/3 = 3

~~1.~~

$$\text{cycle: } (2 \times 2) + (18 \times 2) + (2 \times 3)$$

$$\rightarrow 10$$

$$\begin{aligned} \text{Avg CPI} &= \frac{10}{2+1+2} \\ &= \frac{10}{5} \\ &= 2 \end{aligned}$$

~~2.~~

$$\begin{aligned} \text{cycle: } & (4 \times 2) + (2 \times 2) + \\ & (2 \times 3) \end{aligned}$$

$$\begin{aligned} \text{Avg CPI} &= \frac{9}{6} \\ &= 1.5 \end{aligned}$$

$$\begin{aligned} \text{MIPS} &= \frac{\text{Instruction count}}{\text{CPU time} \times 10^6} \\ &= \frac{\text{JC}}{\text{Clock rate} \times \text{CPI} \times 10^6} \\ &= \frac{\text{JC} \times \text{Clock rate}}{\text{Clock rate} \times \text{CPI} \times 10^6} \end{aligned}$$

Lecture - (4, 5), 6.

* CMOS

$$\text{Power} = \text{Capacity load} \times \underline{\text{Voltage}} \times \text{Frequency}$$

enhance power decrease the voltage & increase the frequency in such a way that the power will not change.

Amalgamated time

$$T_{\text{improved}} = \frac{T_{\text{affected}}}{\text{improvement factor}} + T_{\text{unaffected}}$$

$$T_{\text{improved}} = \frac{100}{2} + 100 = 150$$

overall time
100 minus
[affected time]

if improvement factor negative, that will not possible.

But positive value improvement possible.

$$T_{\text{improved}} = \frac{100}{2} + 100$$

$$(150) \text{ or } 100$$

(v, 1) write

MIPS

~~lw \$t0, 8(\$r1)~~ + $A[8]$

$lw = \$s_2$

$A = \$s_3$

~~lw \$t0, 32(\$r3)~~

~~add \$t0, \$t0~~

~~sw \$t0, 32(\$r3)~~

~~add \$t0, \$s_2, \$t0~~

~~sw \$t0, 48(\$r3)~~

~~lw \$t0, 230~~

~~A[10] = h + B[i];~~

~~h = ~~\$s_2~~~~

~~A~~s_2~~ = \$s_3~~

~~addi \$t0, \$s_2, 230~~

~~mult \$t0, \$s_0, 9~~

~~add \$t0, \$s_0, 9~~

~~add \$t0, \$t1 (\$s_4)~~

~~addi \$t0, \$t0, 2~~

~~sw \$t0, 40(\$r3)~~

sel \$t2, \$t0, 2

shift left one time \rightarrow multiply by 2

$$\begin{array}{c|c} \text{w} & \text{w} \\ \text{w} & \text{w} \end{array} \xrightarrow{\text{shift left}} \begin{array}{c|c} \text{w} & \text{w} \\ \text{w} & \text{w} \end{array} \xrightarrow{\text{shift left}} \begin{array}{c|c} \text{w} & \text{w} \\ \text{w} & \text{w} \end{array}$$

$w^2 + i \quad n^2p \quad n^2q \quad n^2r + 2r$

if ($i = j$) $f = g + h$ $f = \$S_0$

else $f = g - h$ $g = \$S_2$

\Rightarrow begin $\$S_3, \S_4 , IF $i = \$S_3$

sub $\$S_0, \$S_2, \$S_2$ $j = \$S_4$

j EXIT $n^2p \quad n^2q \quad n^2r \quad b60$

IF:

add $\$S_0 + \$S_2, \$S_2$ $n^2p \quad n^2q \quad n^2r \quad b62$

EXIT:

P.T.O

if $i < j$ then $f = g + h$
~~if ($i < j$) $f = g + h;$~~ $f = s_0$
~~else $f = g - h$~~ $g = s_1$
 $h = s_2$
 $i = s_3$
~~set $\$t_0, \$s_3, \$s_4$~~ $j = s_4$ #
~~beq $\$t_0, \$zero, \#EISE (i == j) Zi$~~
~~sub $\$s_0, \$s_2, \$s_2$~~ $\boxed{\begin{array}{l} \text{If } f \neq 0 \\ \text{false} \end{array}}$
~~J EXIT~~ $\boxed{\begin{array}{l} \text{true} \\ \text{else} \end{array}}$
~~IF: add $\$s_0, \$s_2, \$s_2$~~
 $\rightarrow \# \text{ true else}$
 $\rightarrow \# \text{ true else}$
~~add $\$s_0, \$s_2, \$s_2$~~ ~~TIKE 6~~
~~J EXIT~~ ~~TIKE 7~~
~~ELSE: sub $\$s_0, \$s_2, \$s_2$~~
 ~~TIKE 6~~
~~EXIT:~~ ~~TIKE 7~~

0.79

$f = g - h$

else $f = g + h$

remove the equal sign

converted code
same
 $\text{if } (i < i) f = g - h$
 $\text{else } f = g + h$

$\Rightarrow \text{set } \$t_0, \$S_3, \$S_4 \text{ FIXE 5}$

~~neg \$t_0, \$zero, ELSE~~: FIXE

~~\$t_0, \$S_1, \$S_2~~ similar to output A

~~IF: \$t_0, \$S_1, \$S_2~~ FIXE 5

~~IF: \$t_0, \$S_1, \$S_2~~ FIXE 5

~~sub \$S_0, \$S_1, \$S_2~~ FIXE 5

J EXIT FIXE 5

ELSE: add ~~\$S_0, \$S_1, \$S_2~~ and : FIXE 5

EXIT: ~~add \$S_0, \$S_1, \$S_2~~ FIXE 5

: FIXE

if ($i < j$ and $i != k$) $f = g + h$;

else ~~$f = g - h$~~ $\rightarrow f = g$ and $h = 0$

\Rightarrow drops off error

slt \$t₀, \$\$₃, \$\$₄

lne \$\$₃, \$\$₅, JF

beq \$t₀, \$zero, ELSE

ELSE: sub \$\$₀, \$₁, \$₂

J EXIT.

IF: add \$\$₀, \$₂, \$₁

EXIT:

\Rightarrow Another solution:

slt \$t₀, \$\$₃, \$\$₄

lne \$t₀, \$zero, L₁

sub ~~\$\$₀~~, \$₂, \$₁ \rightarrow L₁ has ~~\$\$₀~~

J EXIT.

TI+3 6

L₁: lne \$\$₃, \$\$₅, JF

IF: add \$\$₀, \$₂, \$₁

EXIT:

TI+3 7

loop
whitec (sare[si] == i++)
if (sare[si] == i++)

for loop for what wanted
lw \$t0, \$20(\$s2)
loop: beq \$t0, \$s1, ~~inc~~ INC
J EXIT.

INC: addi \$s0, \$s0, 1

J LOOP

EXIT:

for (i=1; i<w) i++

addi \$s0, \$zero, 1

loop: set \$t0, \$s0, \$s2, INC

J EXIT

INC: addi \$s0, \$s0, 1

J LOOP

EXIT:

Lecture-7

CA

function

Argument reg \rightarrow \$a₀ - \$a₃ in w

Return Value reg \rightarrow \$v₀, \$v₁

Return Address \rightarrow \$r₁₅ in l

Stack pointer \rightarrow \$sp in r₁₄

Instructions:

jal func_address(\$r₁₅)

jr \$r₁₅

add (\$r_i + \$i), \$r_i in l

sub \$r_i, \$r_i in l

int, -23, 023, of R + 22:901

RTX 5

l, 023, 023, 9001:INC
9001:G
RTX

jal 4000

function. address

$$\$pa = PC + 4 = 2008 + 4 \rightarrow 2012$$

[next line]

* int func (int g, h, i, j)

[argument register]

* int func (int g, h, i, j) {

int f, a, b = 20, c = 30

$$a = b + c$$

$$f = (g + h) - (i + j)$$

$$f = a + f$$

return f;

g, h, i, j
a₀, a₁, a₂, a₃

f = \$50

a = \$51

b = \$52

c = \$53

for return f

add \$V0, \$50, \$zero

7

int main()

int a=20, b=5, c=2, d=3; //
 $a = 20$, $b = 5$, $c = 2$, $d = 3$

func (a,b,c,d); // func

$a = a + 3;$

$d = b + (4 * a) // \text{with this } *$

[stack for function]

before function call [part 1]

add i \$sp, -16 // empty place

sw \$s0, 0(\$sp)

sw \$s1, 8(\$sp)

sw \$s2, 16(\$sp) // $s+d = 0$

sw \$s3, 0(\$sp) // $(d+b) = t$

add \$sp, -16 // $t+d = t$

[i2 monitor]

P.T.O

L

after function call [part 1]

lw \$5, 0(\$sp)) ~~96T-9~~

lw \$3, 4(\$sp)

lw \$1, 8(\$sp)) ~~96T-10~~

lw \$0, 12(\$sp)

addi \$sp, \$sp, 16.) ~~96T-10~~

in \$ra.

- we will push the amount of
variables declared in the function
onto the stack

int func() {

int a, b, c, d

args - 90

local - 28

}

whichever variable is by

towards the end - local

variable = local

Lecture-8

SA

Machine code (Ans will not write)

R-Type (add, sub, and, or, nor,
sll, srl, slt, jro)

I-Type (lw, sw, andi, ori, addi,
(r2b,eqz,lne) r2p, wr)

J-Type (j, jal, r2p, r2b)

R-Type

OP	rs	rt	rd	shift	func
6 bit	5 bit	5 bit	5 bit	5 bit	6 bit

OP = opcode

rs = 1st source register

rt = 2nd ~ ~

rd = destination register

shift = shift amount

func = function

* add \$s₀, \$s₂, \$t₂ ~~276T + 15~~ 14

op	rs	rt	rd	shamt	func
0	17	10	16	X	32

000000 10001 01010 10000 00000 100000
 stic 3i stic 2 stic 2 - stic 3

* sll \$t₀, \$s₃, 20 ~~278~~ 205 028 002 08
 28 ← rt ← rd ← shamt ← rs ← op ← A1

op	rs	rt	rd	shamt	func
0	19	X	8	20	0

000600 10011 00000 01000 10100 000000

* jr \$ra ~~rs~~ ~~278~~ 028 ibus &

A15 tr ← 28 ← 90

op	rs	rt	rd	shamt	func
0	31	X	X	X	8

000600 11111 00000 00000 00000 0012000

H-J-Type

OP	rs	rt	for constant or address
000001	000000000001	010101000001	10001 00000000

6 bits 5 bits 5 bits 16 bits.

→ SW, \$50, 20 (\$51) → rs
 → rt

OP	rs	rt	C/A
0000000000101	0001000000000000	10001	00000000

→ andi \$50, \$51, 223 → rt C/A

OP	rs	rt	C/A
001100	10001	10000	0000000011011111

~~Pravak~~

bne \$s₁, \$s₂, 100



bits [for bne & beq] $\frac{N}{4}$

OP	rs	rt	C/A
5	18	17	$\frac{100}{4} = 25$
000101	1001 ⁰	10001	0000000000110001

J type

* J 2000

for J type
J, Jal = $\frac{N}{4}$

OP	address
2	$\frac{2000}{4} = 500$

6 bits ~~for 675 26785~~ 26 bits to program

op	Jal 10000
3	$\frac{10000}{4} = 2500$

000011	0000000000000100111000100
--------	---------------------------

1000 0000 0000 0000 0000 0000 0000 0000

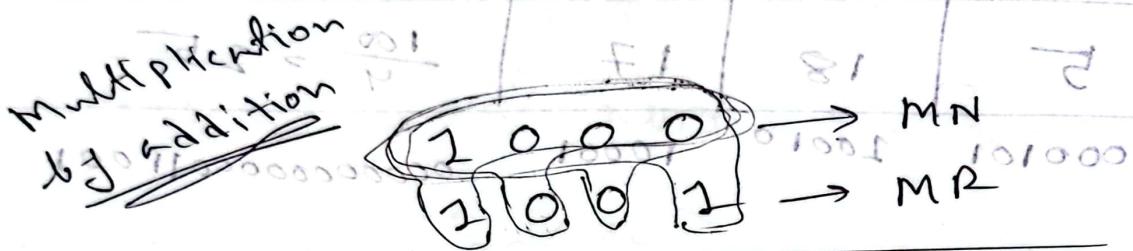
Lectures

drawn by

and arithmetic

Multiplication

~~Multiplication
by addition~~



$$\begin{array}{r}
 & 1 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & \times \\
 & 0 & 0 & 0 & 0 & \times & \times \\
 & 1 & 0 & 0 & 0 & \times & \times & \times \\
 \hline
 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
 & & 2 & 0 & 0 & - & 0 & 0 & 0 \\
 & & & & & & & & 90
 \end{array}$$

MR can be drawn by adding bits from right to left.

0 & 1 are called MN and MR respectively.

0 & 1 can be drawn by 1's complement.

0 & 1 can be drawn by 2's complement.

move 1's complement to 2's complement.

modular arithmetic

exp 2

Replace all the cross with 0.

→ 2 M+A shift left + ~~MN~~ 2st time
shift right MR 2 $\begin{array}{r} 0 \\ \times 2 \\ \hline 0 \end{array}$
 $\begin{array}{r} 0 \\ \times 2 \\ \hline 0 \end{array}$

011000 01011

0100 (MN)

0011 (MR)

(Multiplication)

Inputs $\begin{array}{r} 011000 \\ 0100 \\ \hline 010000 \end{array}$ step-1
→ 2 $\begin{array}{r} 101000 \\ 0000 \\ \hline 0000 \end{array}$ 00020
→ 2 $\begin{array}{r} 0000 \\ 0000 \\ \hline 0000 \end{array}$ 30
M+A $\begin{array}{r} 0000 \\ 0000 \\ \hline 0000 \end{array}$ 4
 $\begin{array}{r} 00011000 \\ 000000 \\ \hline 00011000 \end{array}$

→ 2 $\begin{array}{r} 00011000 \\ 000000 \\ \hline 00011000 \end{array}$

M+A 10000 11100

→ 2 00000 11000

→ 2 00000 10000

→ 2 00000 00000

→ 2 00000 00000

(M+A) 00000 00000

Optimised multiplication

steps

• 0 other steps will be

$$M \downarrow \quad Q \downarrow \\ 26 \times 6 \quad \xrightarrow{\text{with 5 bit architecture}} \quad 6 \times 26$$

if ($Q_0 = 1$)

$A + M, SR$

$Q_0 = 0, SR$

11010 00010

(written with 1's)

	$(n \cdot m)$	0010	th
C	$\cancel{(A \cdot n)}$	11000	
O	$\cancel{00000}$	0000010 - start	
$2 \{ 0$	00000×00011	$\cancel{00000} \times 00011$ SR	
$2 \{ 0$	$11010 \times \cancel{00011}$	$\cancel{00011} A + M$	
0	01101	$\cancel{00001} SR$	
$3 \{ 2$	00111	00001 A + M	
0	10011	10000 SR	
$4 \{ 0$	01001	11000 SR	
$5 \{ 0$	00100	11100 SR	

(Ans)

Division

steps

$$\begin{array}{r}
 \text{D} = 0 \\
 -92 \quad M+A \\
 \hline
 -92 \quad Q = 0
 \end{array}$$

$$\begin{array}{r}
 0 \quad M \\
 15 \div 7 \\
 \hline
 1111 \quad 0111
 \end{array}$$

$$\begin{array}{l}
 \text{SLL}(C, A, Q) \\
 A = A - M \\
 \times S1 \\
 (2, Q_0 = 0, A = A + M) \\
 L1Q3 \quad Q_0 = 1
 \end{array}$$

$$\begin{array}{r}
 D \\
 -4100 \\
 \hline
 -92 \quad M+A \\
 \hline
 -4100 \quad 0100 \\
 -92 \quad 1001 \\
 \hline
 -M+A \\
 \hline
 -92 \quad 0010 \\
 -92 \quad 0010 \\
 \hline
 \end{array}$$

$$\begin{array}{r}
 A \\
 0000 \\
 0000 \\
 0000 \\
 0001 \\
 0001 \\
 2020 \\
 0982 \\
 0021 \\
 0111 \\
 0000 \\
 00001 \\
 2020 \\
 0001 \\
 \hline
 \end{array}$$

$$\begin{array}{r}
 Q \\
 01111 \\
 01111 \\
 111 \square \\
 2220 \\
 2220 \\
 2220 \\
 220 \bullet \square \\
 2200 \\
 2200 \\
 100 \bullet \square \\
 2001 \\
 001 \bullet \square \\
 001 \bullet 0, Q_0 = 0 \\
 0010 \\
 \hline
 \end{array}$$

Reminder A

Multiplication

$$13 \times 3 \\ M \quad Q \\ 1101 \quad 0011$$

$1101 \times 0011 = 11100011$

\downarrow (Carry)

$$Q_0 = 1$$

$$A + M, S R$$

$$Q_0 = 0 \quad V = 0 \quad S \quad R$$

No carry until second column
(Col 2)

0000

1101

0110

0011

X

1001

0011

1101

0110

1002

X

1100

0100

1110

X

1110

0010

0111

X

0111

$$\rightarrow A + M$$

$$\rightarrow S R$$

$$\rightarrow A + M$$

$$\rightarrow S R$$

$$\rightarrow S R$$

$$\rightarrow S R$$

$$\rightarrow P$$

$$\rightarrow$$

1

2

3

4

x	x	x	x	x
x	x	x	x	x
x	x	x	x	x
x	x	x	x	x

$(X/0)$ same
No wire will connect

Single Cycle

	Reg Dst	Reg SPC	Mem for reg write	Mem write	Reg write	ALU SPC	ALU Dst	ALU OPT	ALU OPO	Branch	Mem	Mem write	Reg write	Reg Dst
Jmp	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Ret PC	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Reg														
Reg														

wait until fun

~~ALU Opt~~

1011

~~1010~~

M

1100

D

E

X E1

B

A

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

?

?

?

?

?

?

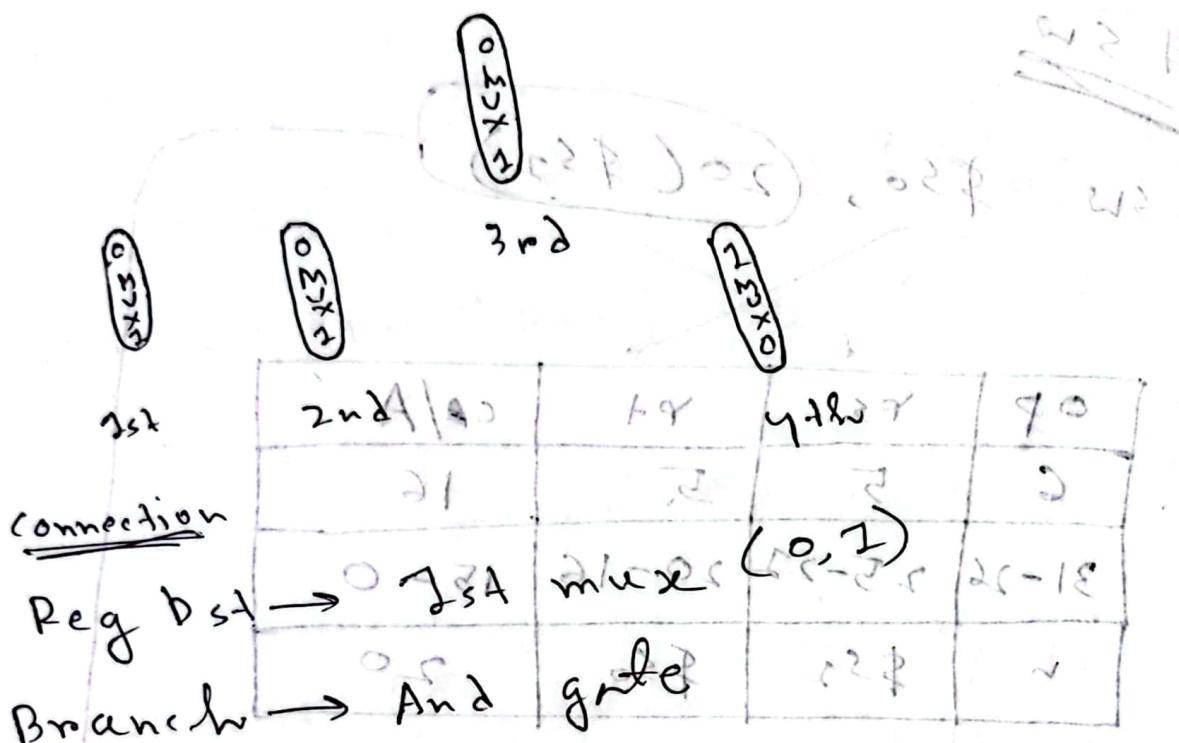
?

?

?

?

?



Mem Read → 3rd box (0,1)

Mem to Reg → 4-to-1 mux (1,0)

AIV OP → AIV-control → 3rd Adder

Mem Write → 3rd box

AIV src → 2nd mux (0,1)

Reg Write → 2nd box

(forward address between boxes)

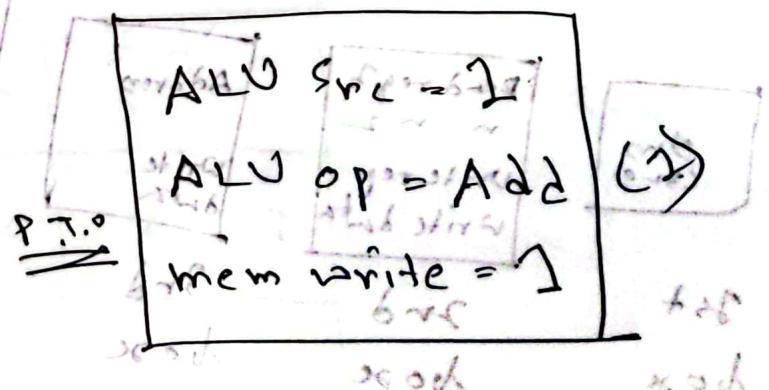
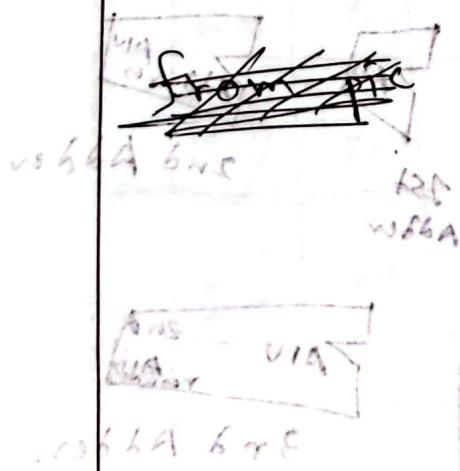


SW

SW \$50, 20(\$52)

OP	RSB ₁₆	RT	CA/Ans
C	5	5	16
31-26	25-22	20-16	15-0
w	\$52	\$50	20

1. Read (20-16) [Register]
2. Execute addition operation of mem (25-22) & (25-0) ← 90 VIA ALU [ALU]
each bit of result mem
3. store value of [20-16] in executed address [Data memory]



from pic VIA

$\{ws(31-26) \rightarrow$ control

$ws(25-21) \rightarrow$ Read reg 2 \rightarrow Read data 1

$ws(20-16) \rightarrow$ Read reg 2 \rightarrow Read data 2

$ws(15-0) \rightarrow$ mux(2) \rightarrow ALU

PC \rightarrow Address \rightarrow mux(0)

VIA \rightarrow bus \rightarrow PC

VIA \rightarrow bus \rightarrow PC

same for lwe / Add / beg

cor66A \rightarrow bus

what bus

(f) \rightarrow bus

read data 2 \rightarrow ALU result

control \rightarrow (0-15) wrb

Address \rightarrow Address

bus write \rightarrow (5-20) wrb

for data \rightarrow (5-20) wrb

(g) \rightarrow bus \rightarrow (0-15) wrb

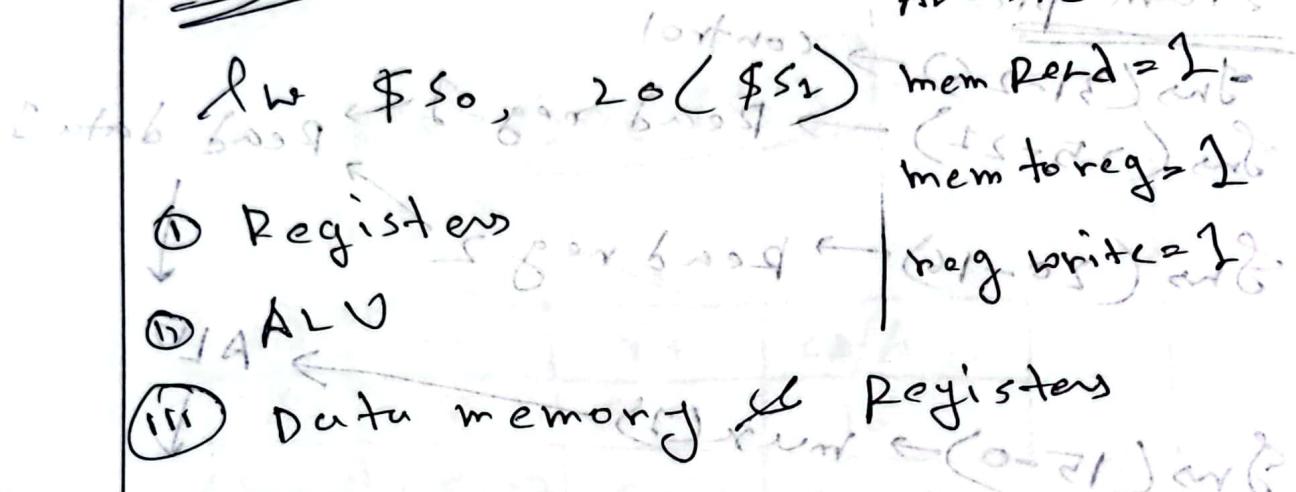
for

ofire

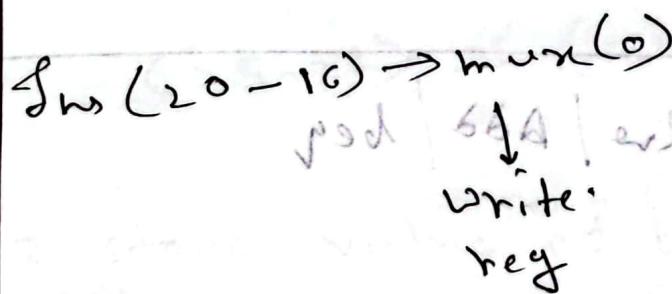
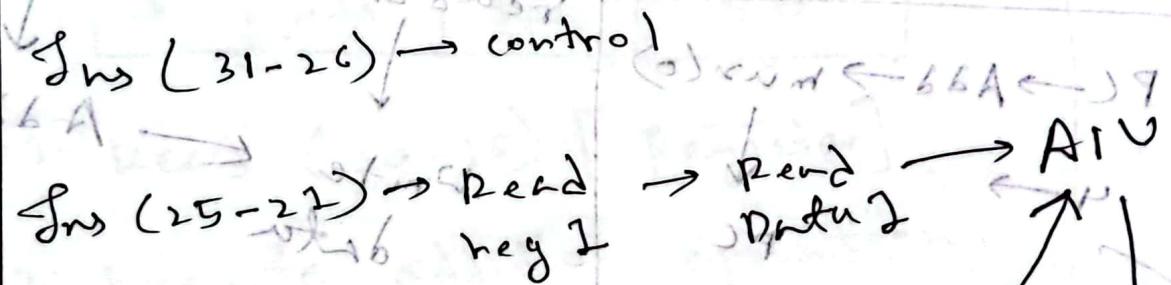
whid

(f) \rightarrow bus \rightarrow (0-21) wrb

~~LW~~



~~PIA~~



~~write Data~~

~~lw (15-0) → mux(1)~~

Address

↓
Read Data

↓
mux(1)

Add

$f_{ws}(31-26) \rightarrow \text{control}$

$f_{ws}(25-22) \rightarrow \text{Read reg. 1}$

$f_{ws}(20-26) \rightarrow \text{Read reg. 2}$

$f_{ws}(15-11) \rightarrow \text{mux}(2)$

$P + 9 = 39$ select A6 = 39

(cont'd) [01-05] & [55-59] \leftarrow write data

$f_{ws}(15-0) \rightarrow \text{A10}$

control

[01A] [01-05]

read data \rightarrow A10

1. $ds-18$

2. $ds-18$

3. $ds-18$

4. $ds-18$

5. $ds-18$

6. $ds-18$

[55-59] \leftarrow write data

Reg ds = 1

A10 op = 2C1A

Reg write op = 3A

L = whenever

Beav

66A

beg \$50, \$52, 2000 (25-16) arb

OP	rs	rt	(CC/Akt)
31-26	\$52	\$50	$\frac{2000}{4} = 500$
"	25-22	20-16	15-0

② $[25-22] = [20-16]$ true arb

③ equal, $PC = JA$, false $PC = PC + 4$

④ Read $[25-22]$ & $[20-16]$ (Registers)

⑤ execute subtraction operation

of $[25-22]$ & $[20-16]$ (A1U)

A1U3 rs = 110

A1U op is sub (2)

branch = 2

PC

→ mux(1)

pic

7 → 25

ED

Im3 (31-26) → control

0000 0 0

↑
zero

Im3 (25-21) → Read reg 1

Read data 1

↑
AIV

Im3 (20-16) → n n 2

Im3 (15-0) → shift left

bitwise & (0.18) 38 → (0.18) 38
left 1
(0.18) ↓

Add

AIV result

(0.18)

(0.18) 11 + 39

66A < 39

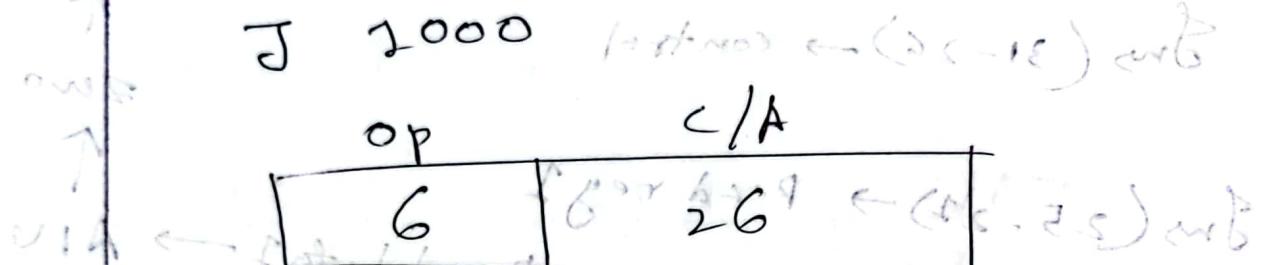
Max word

↓

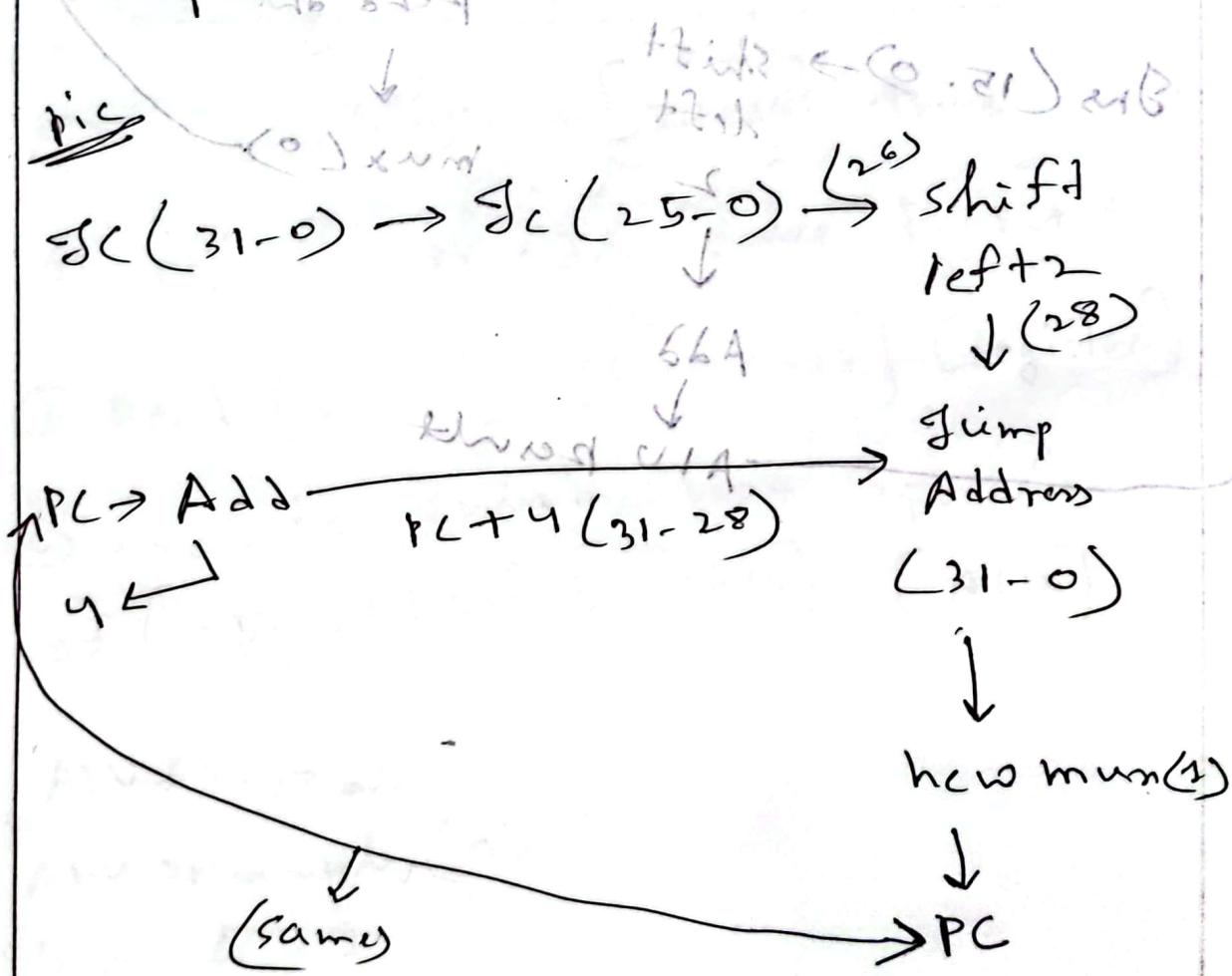
39

(max)

* Jump



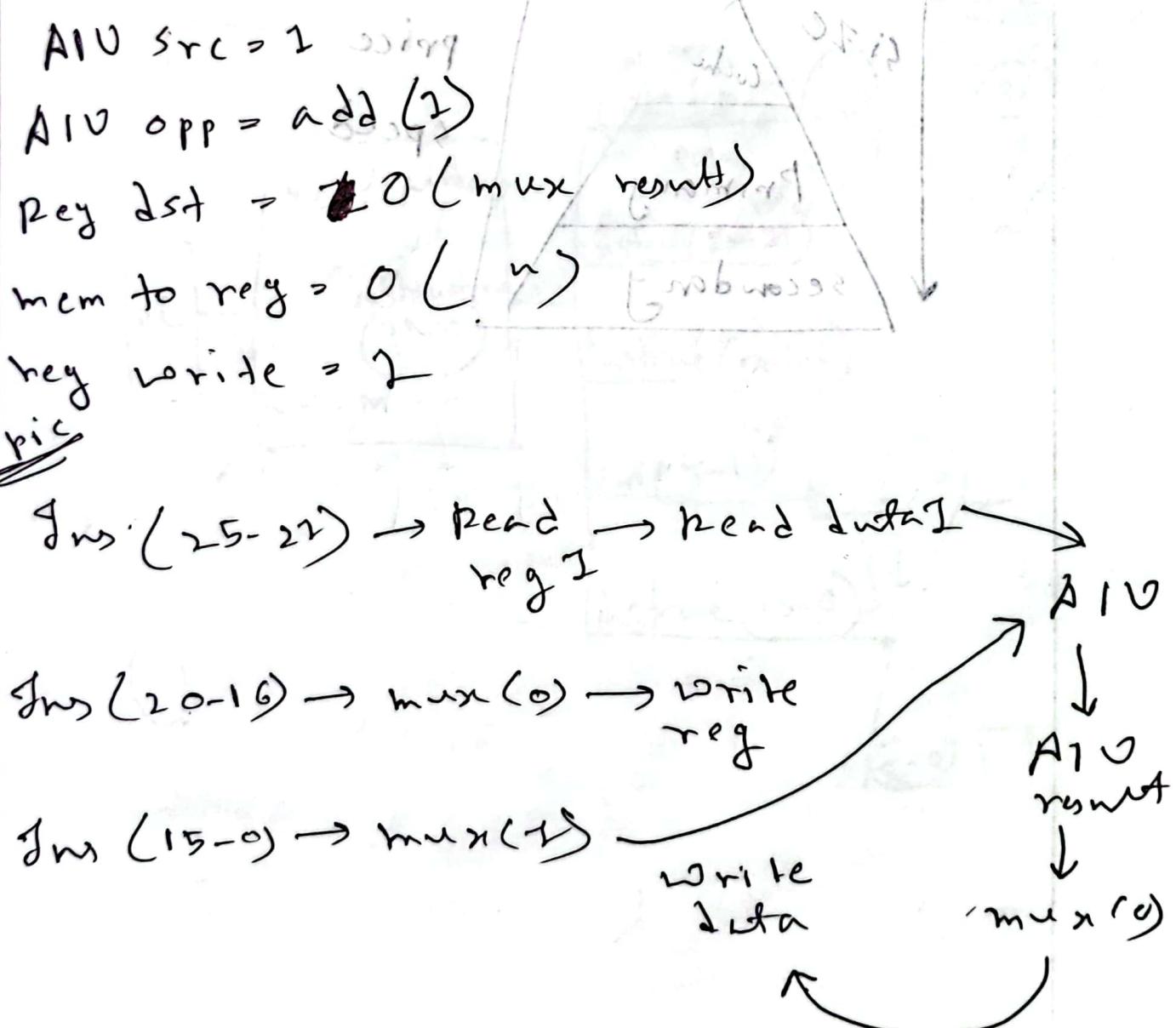
Jump = $\frac{1}{2}$ word



dst

addi \$50 \$52, 1

op	rs	rt	c/a
31-26	\$52	\$50	1
n	25-22	20-216	25-0



~~6.19
2.97 4.24~~

~~6.26~~

~~(A)~~

~~6.02P~~

~~0.24~~

~~1660~~

~~H~~

Primary → Ram, Rom

~~A~~

~~for~~

~~28~~

~~90~~

Secondary → HDD, SSD

~~424~~

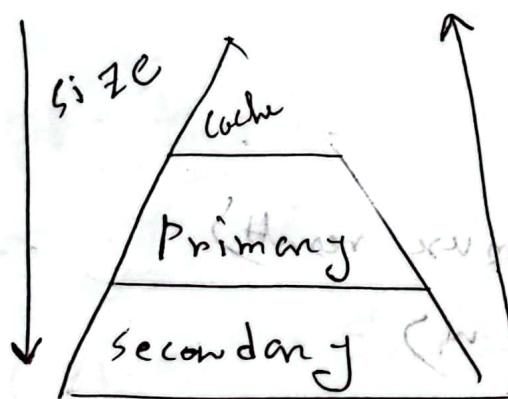
~~25-18~~

Tertiary → CD, DVD, Pendrive

~~346-05~~

~~45-25~~

~~w~~



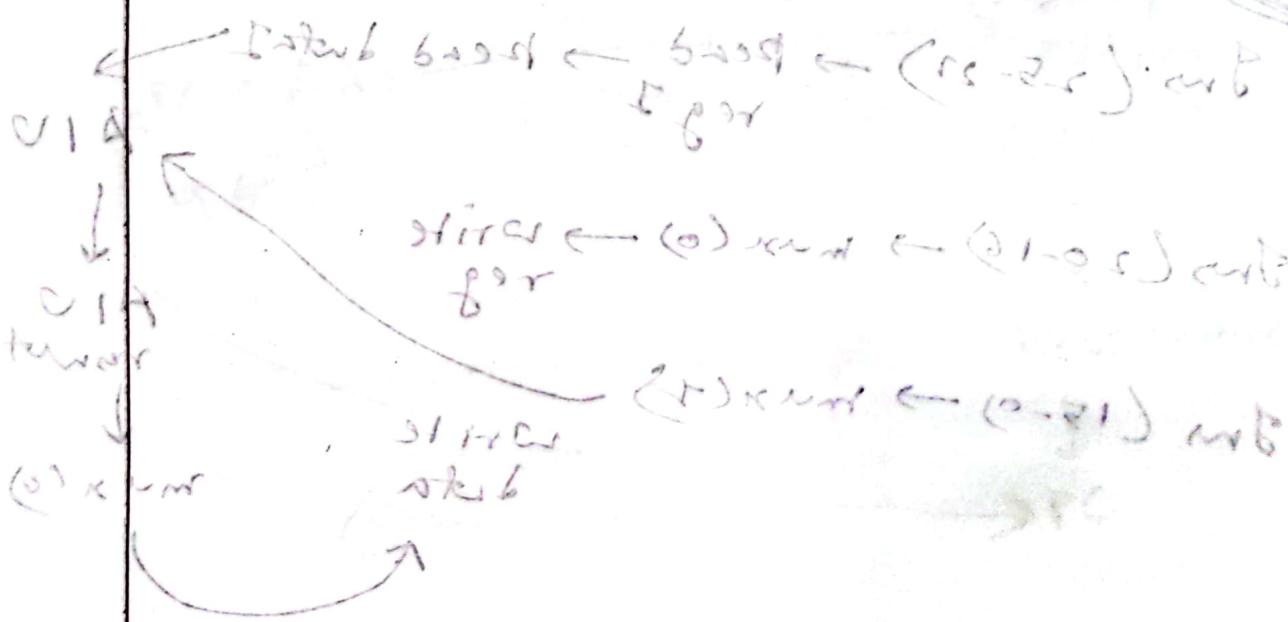
price low > high

speed slow = fast

low = high of mind

slow = strings for

high



~~CW~~

~~4.5.2.3~~

~~a)~~

$$\text{Per lock} = \frac{16}{w}$$

$$522 \times 1024$$

$$= 2^{16}$$

$$(0-21) \text{ entries}$$

$$2^w = 32768$$

$$\log_2(32768)$$

$$2^w =$$

$$\log_2(32768)$$

$$(0-15) \text{ entries}$$

$$2^w = 32768$$

$$\cdot 15$$

$$w = 15$$

$$w = 15$$

~~(Q)~~

Ex. F1, P, C : (8 blocks of size 4)

2^m words \rightarrow $m = \log_2(4)$ $\rightarrow m = 2$ (number of blocks)

total \Rightarrow block \Rightarrow address of memory

Cache \Rightarrow block \Rightarrow address of memory

4.5.24

CA

Cache

\rightarrow upper level means cache memory

* lower \rightarrow Ram. \rightarrow tag bits
 \rightarrow index

Cache 8 block

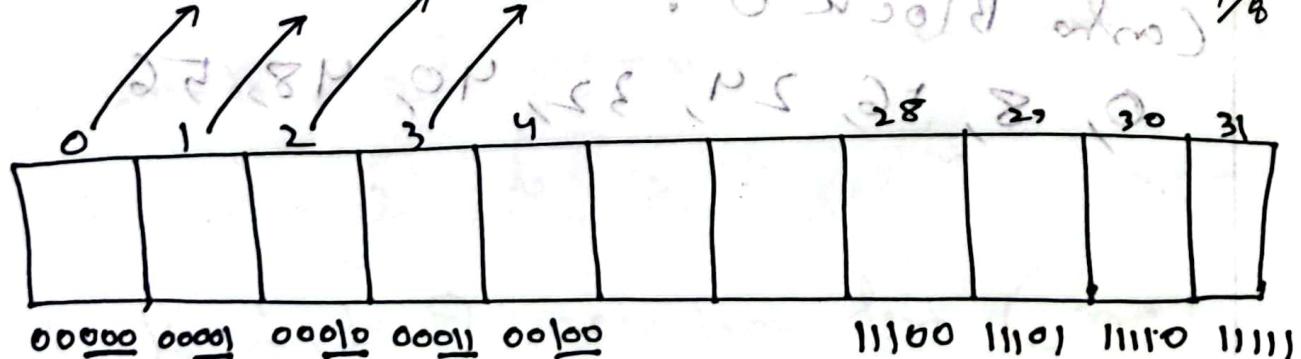
Ram 32 block

Cache block

000	001	010	011	100	101	110	111
00	00	00	00	-	-	8	-
01	-	-	-	-	-	-	8
10	-	-	-	-	-	-	-
11	-	-	-	-	-	-	-

Value of Ram index

$32/8 = 4$



Cache Block 1: 1, 9, 17, 25

Cache Block 2: 2, 10, 18, 26

[always plus 8]

[5 bits of Last
3 bits match with
the block number]

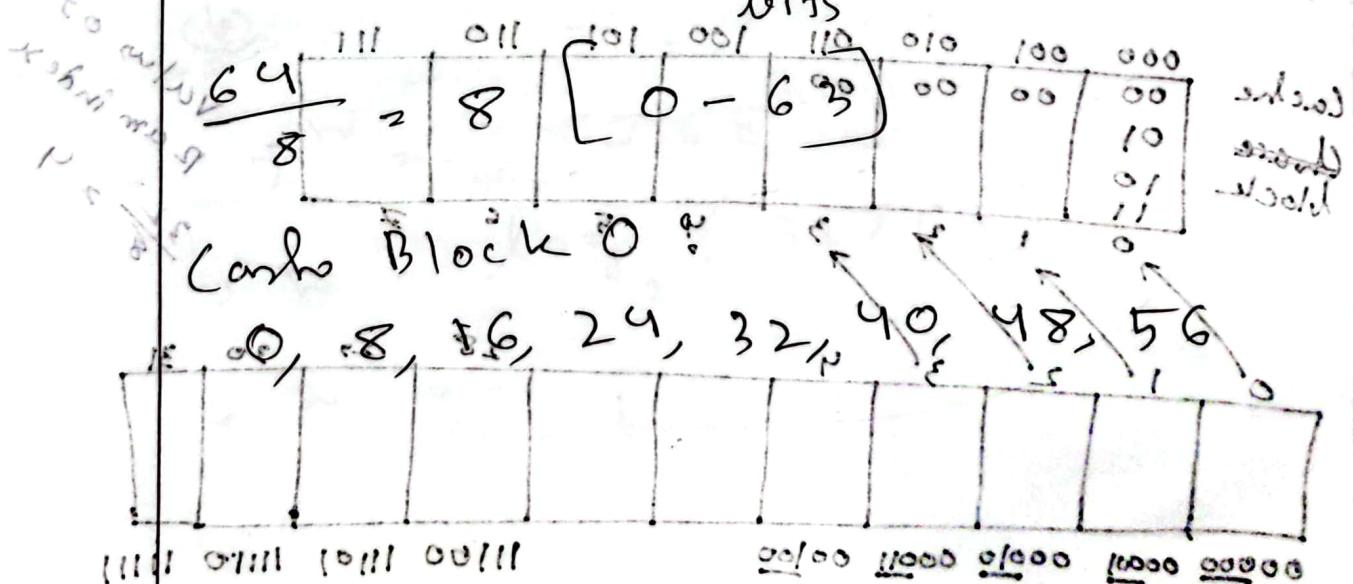
* Cache block \rightarrow 8

main block \rightarrow 64

(0-7) 00000
(8-15) 00010
(16-23) 00001
(24-31) 00011

$2^6 = 64$ ~ word

000 000
tag \rightarrow 3 bits
index \rightarrow 5 bits



Cache works like RAM blocks \rightarrow word address

Cache Blocks

for 8 blocks

000	001	010	011	100	101	110	111
000				(St + w)			
001							
010							
011							
100							
101							
110							

→ tag bits

Cache Blocks

4, 12, 20, 28, 36, 44, 52, 60

* From slide.

Size of block

Cache size = 2^n blocks of $(m+2)$ words

Block size = 2^{m+2} words

1 word = 4 bytes = 4 words

8 blocks $\div 2^3$ blocks = selection bit 3

2 words = 2^2 words = ~ ~ 1

4 bytes = 2^2 bytes = ~ ~ 2

which byte? selection bit ($m+2$)

Index = selection bit

* 32 bit byte addressable

(tag \rightarrow total bits - selection bit)

$$\text{tag} \Rightarrow [32 - (w + m + 2)]$$

* total size of cache

$$\Rightarrow 2^w \times [1 + \{32 - (w + m + 2)\}] + (2^m \times 32)$$

number of blocks

valid

tag

number of words

total size

* slide (2^w, 2^m)

Selection bit = index + offset

$$(n) \quad (m+2)$$

(start + n) bit will be Post offset

~~word bytes~~

~~EE5~~

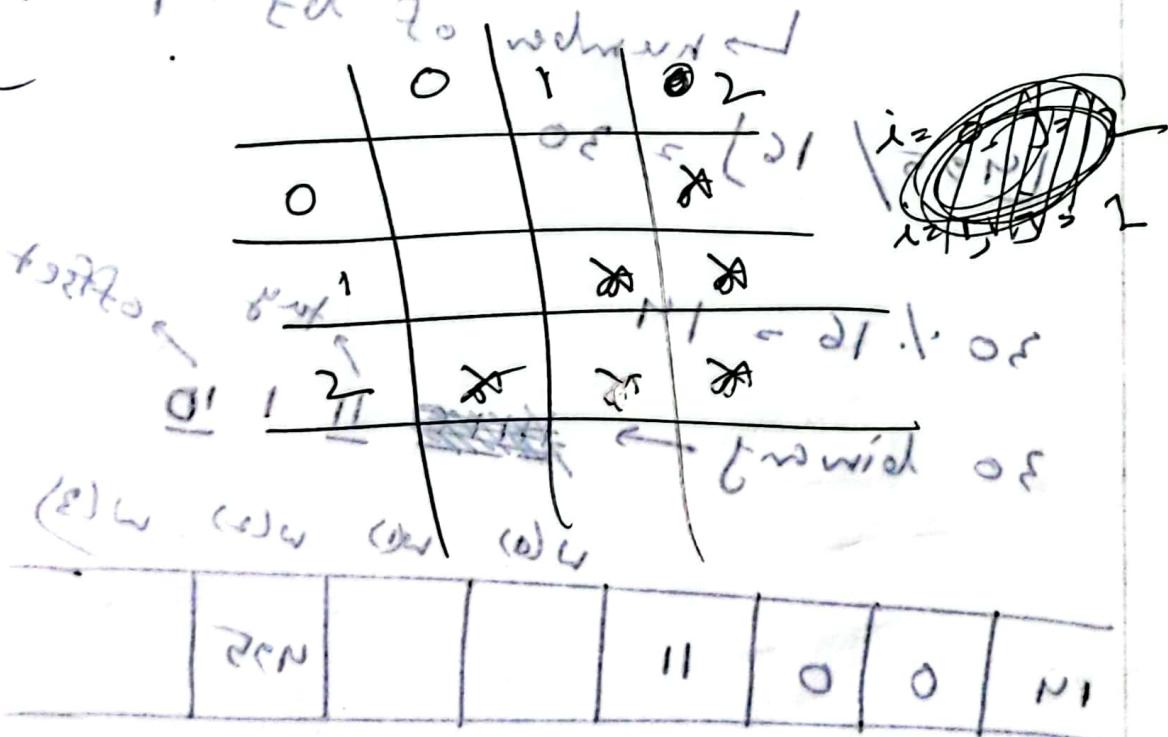
~~1000~~

Block copied from old \ new \ s1

(Block Address \times num of bytes) -
(Block Address + $s \times s1$) + $s1 \times s1$

{ Block n = $n \times n \times s1$ } + (num of bytes - 1)

Block Address = $\frac{\text{address}}{\text{bytes}}$
Block number = $\frac{\text{block address}}{\text{total blocks}}$



~~W
252~~

~~233~~

~~80CA~~
~~frame~~

16 bytes / block ~~bytes~~ ~~short~~

→ ~~Added to next + next (6x2 bytes)~~
~~16x16 + (16x2) + (6x2 bytes)~~
~~bytes bits~~

~~for next~~ ~~block~~ ~~+ (next + next + next)~~
→ 256 + 4 bytes + 4 bytes }
= 264 bytes.

~~blocks~~
~~blocks~~
~~blocks~~
1024 bytes → 64 bytes per block
16 bytes → number of bytes per block

$495 / 16 = 30$

$30 \div 16 = 14$

30 binary → ~~1110010~~

→ ~~1110010~~ \uparrow \uparrow offset
byte byte

$w(0) w(1) w(2) w(3)$

14	0	0	11			495	
----	---	---	----	--	--	-----	--

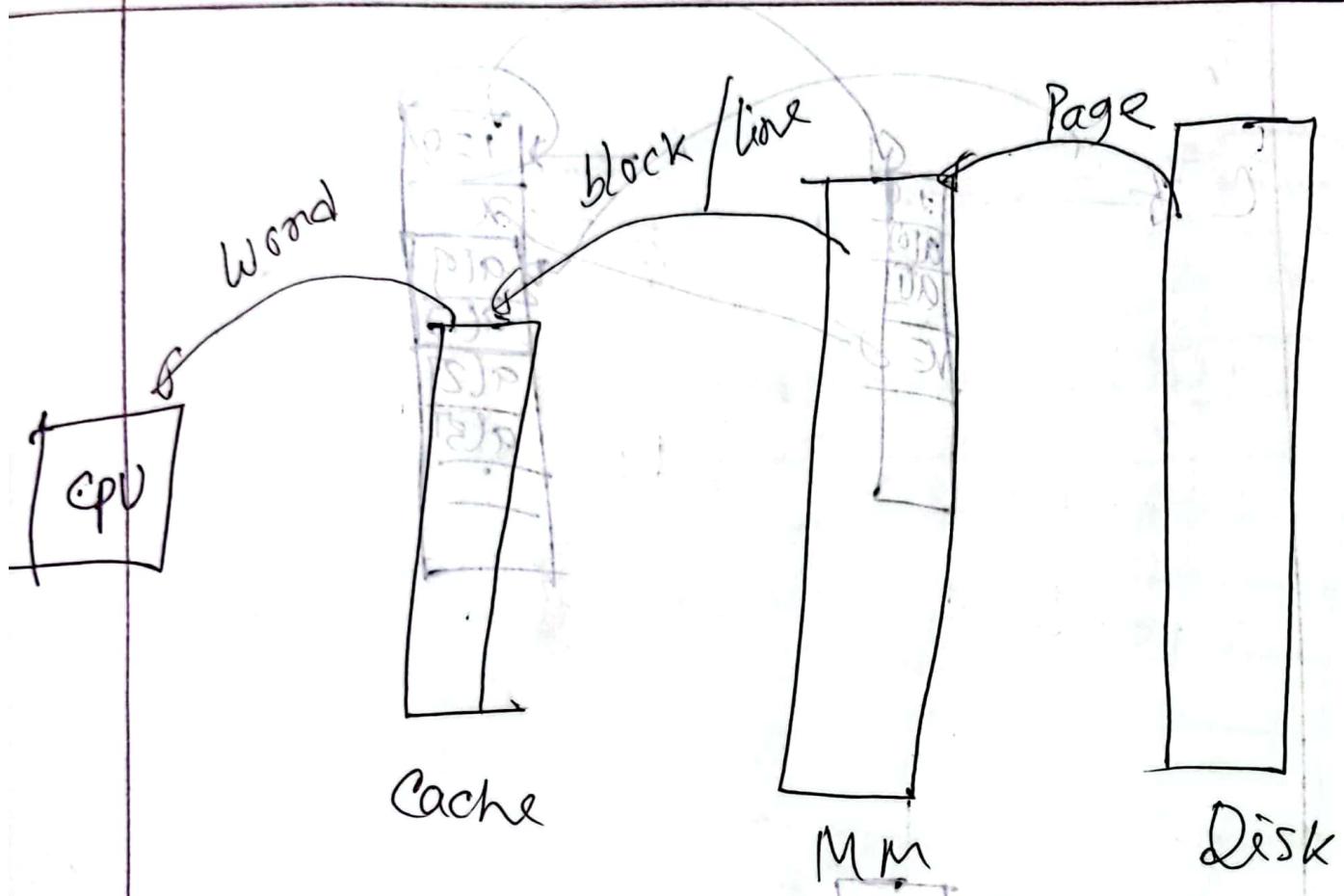
8 word per block,

$$\frac{56}{64}$$

4 word per block,

$$\frac{48}{64}$$

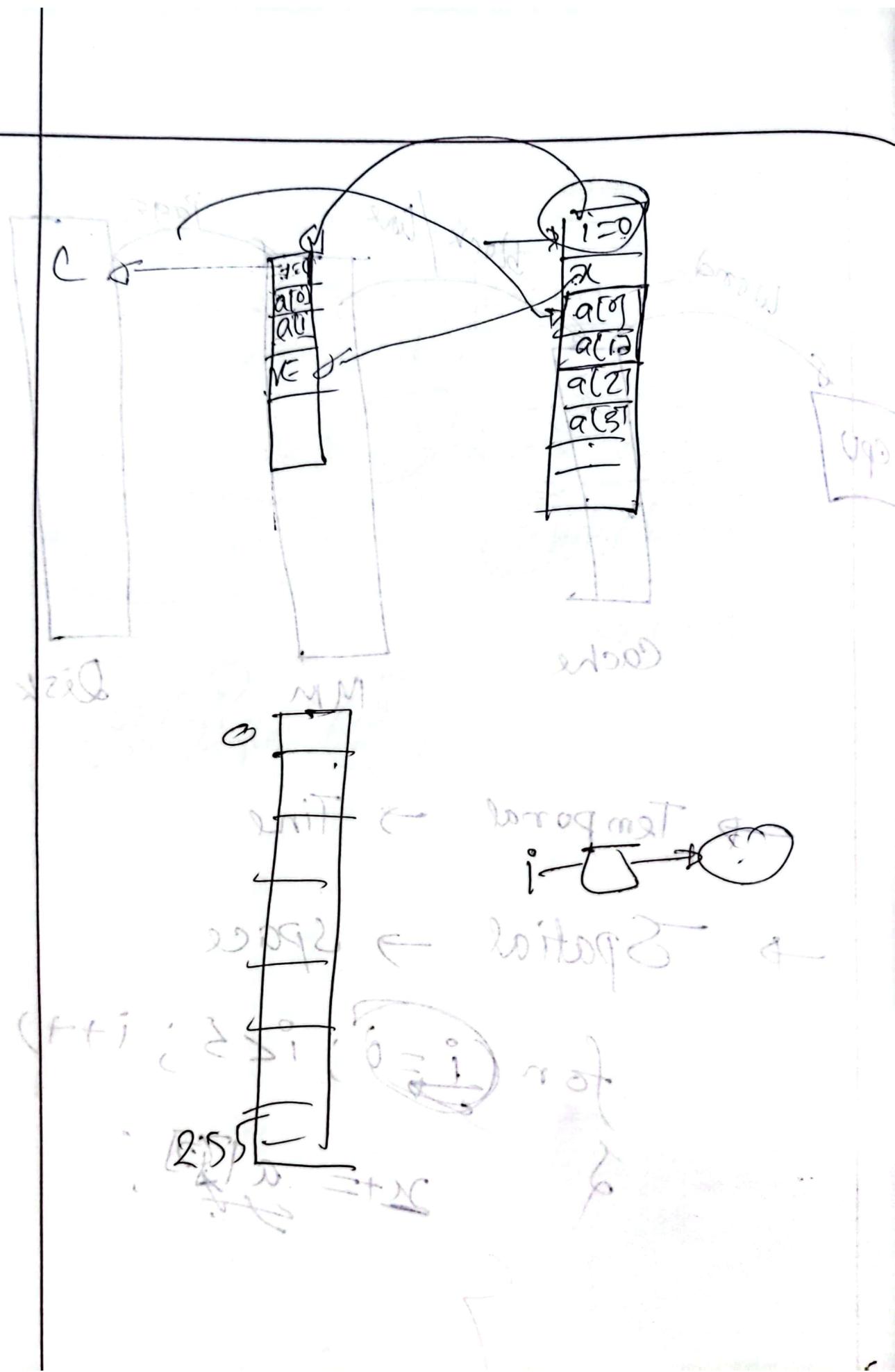




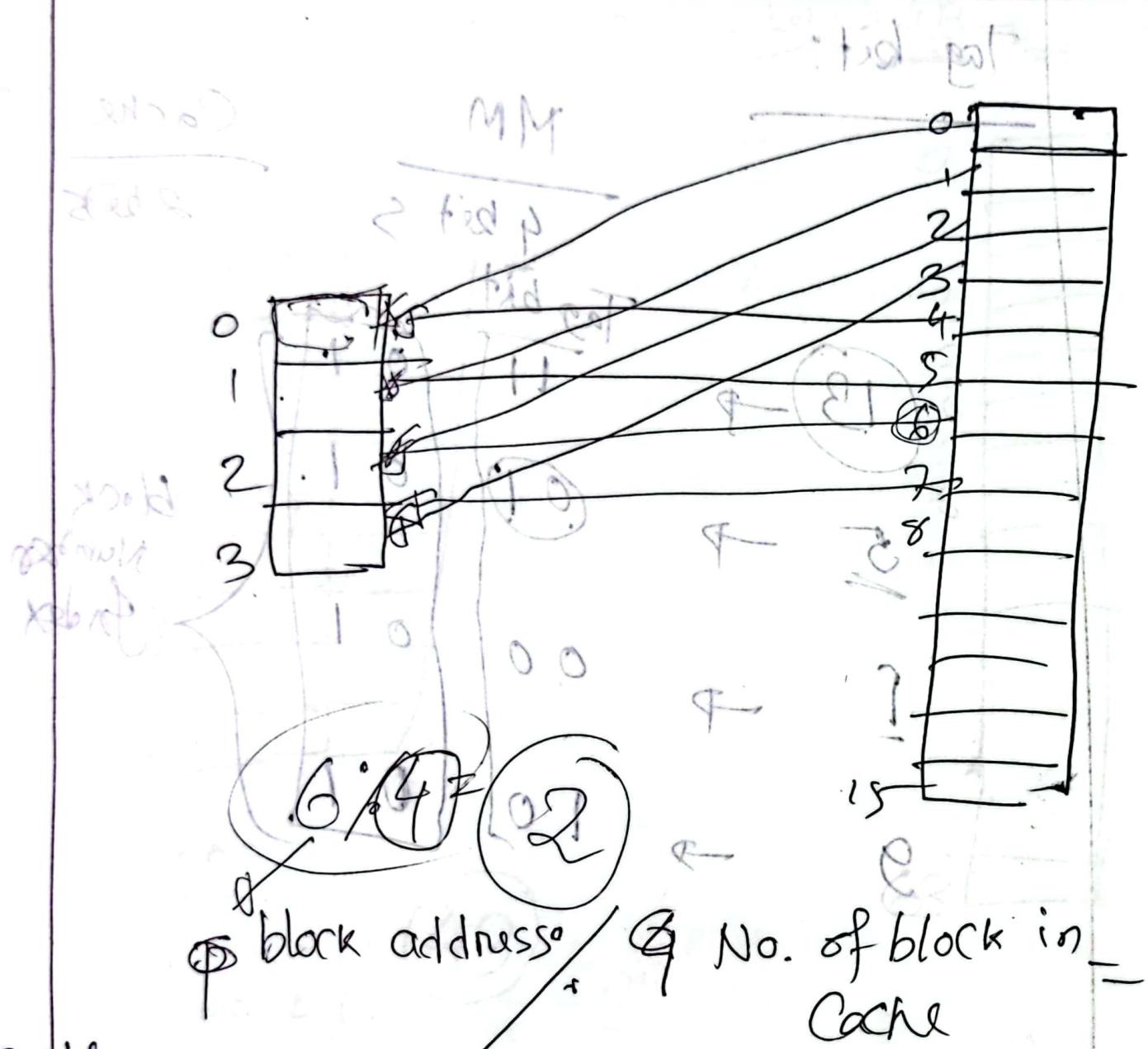
\rightarrow Temporal \rightarrow Time
 \rightarrow Spatial \rightarrow Space

for $i = 0$; $i \leq 5$; $i++$
 $x = a[i]$

g



(13)



Cache

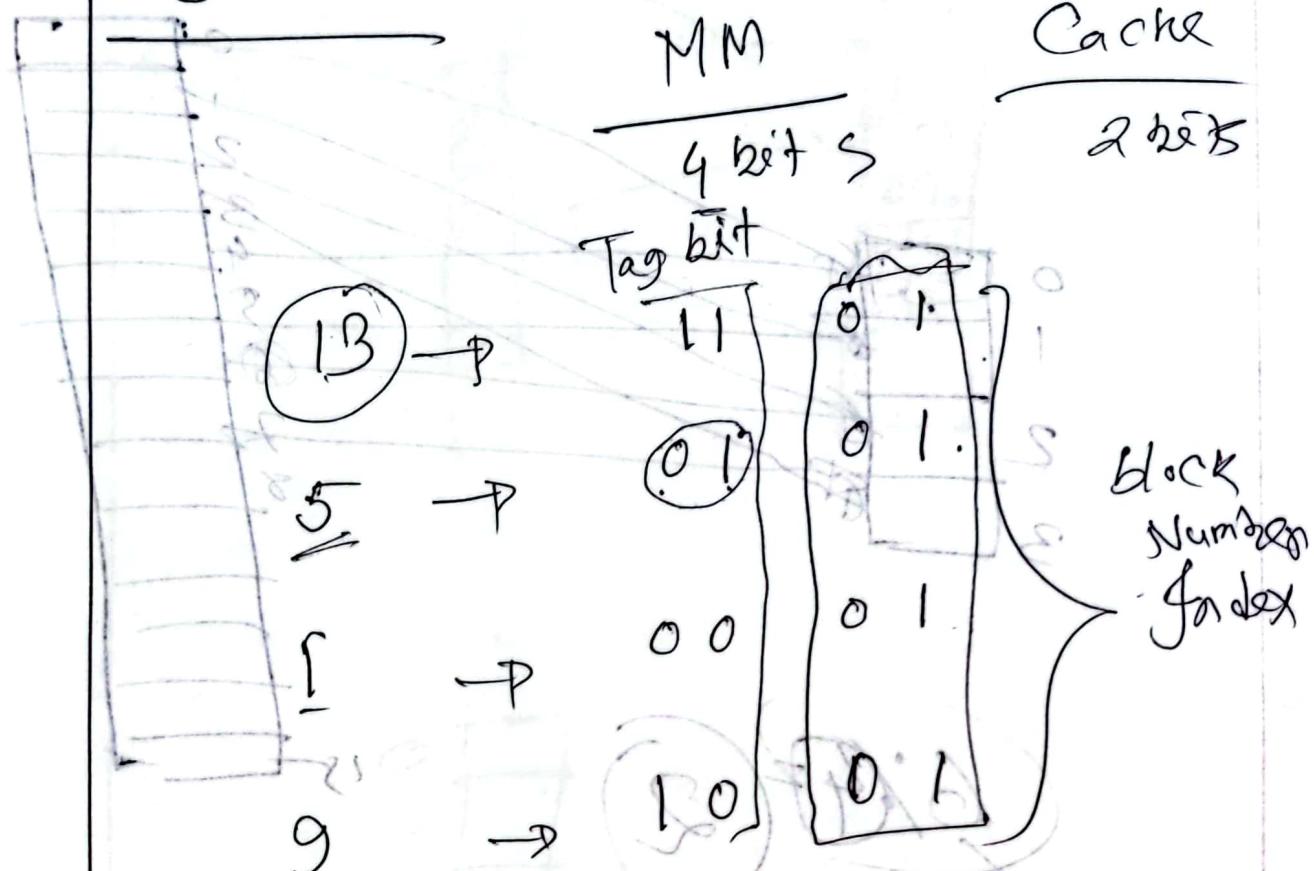
0: 0, 4, 8, 12

1: 1, 5, 9, 13

2: 2, 6, 10, 14

3: 3, 7, 11, 15

Tag bit:



Block Address
Index
Block Number

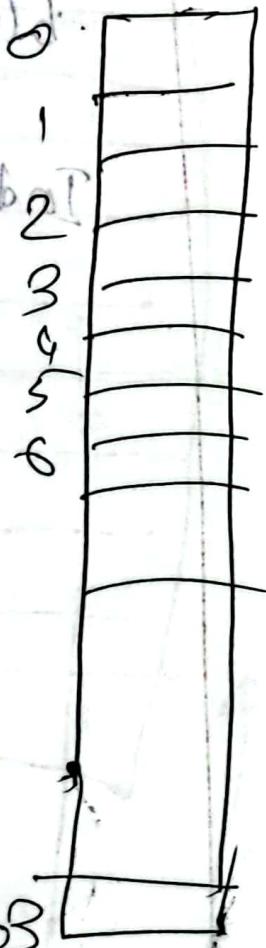
Cache:
S1, 8, p, 0 : 0

S1, e, 2, 1 : L

P, 01, 2, 8 : L

21, 11, f, 8 : E

Ex: $S = 8 \times 8 = 64$ bits \rightarrow MIPS \rightarrow 32 bit \rightarrow 4B \rightarrow 1 word



000
010
011 { 011
100 { 011
101 { 011
110 { 011
111 { 011

43

3, 5

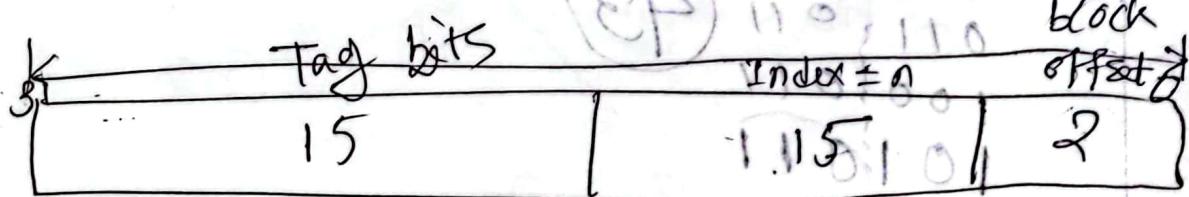
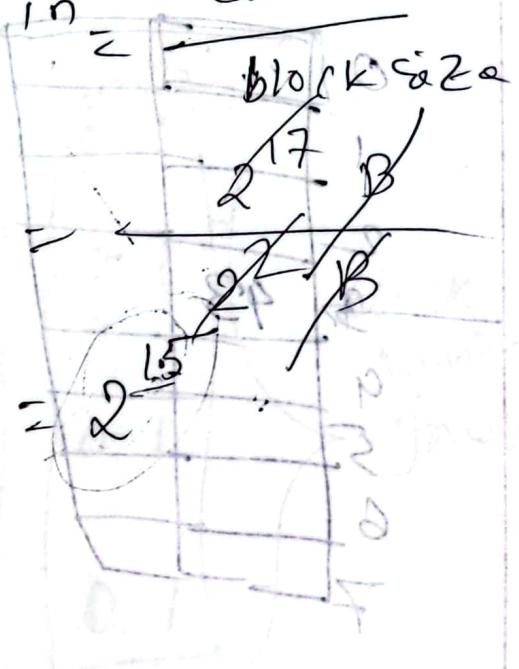
3, 11, 19, 27, 35, 43, 51, 59

Q. ~~CRU = 32 bit~~

$$\text{Cache size} = 128 \text{ kB} = 2^7 \times 2^{10} = 2^{17} \text{ B}$$

$$\text{block size} = 1 \text{ word} = 4 \text{ B} = 2^2 \text{ B}$$

Index/n / No. of blocks in Cache



Ex. 12, 8N, 28, FS, C1, H, E

CPA

$$16 = 2^4$$

Index

0 0 0 0 0 0

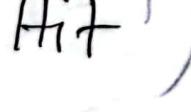
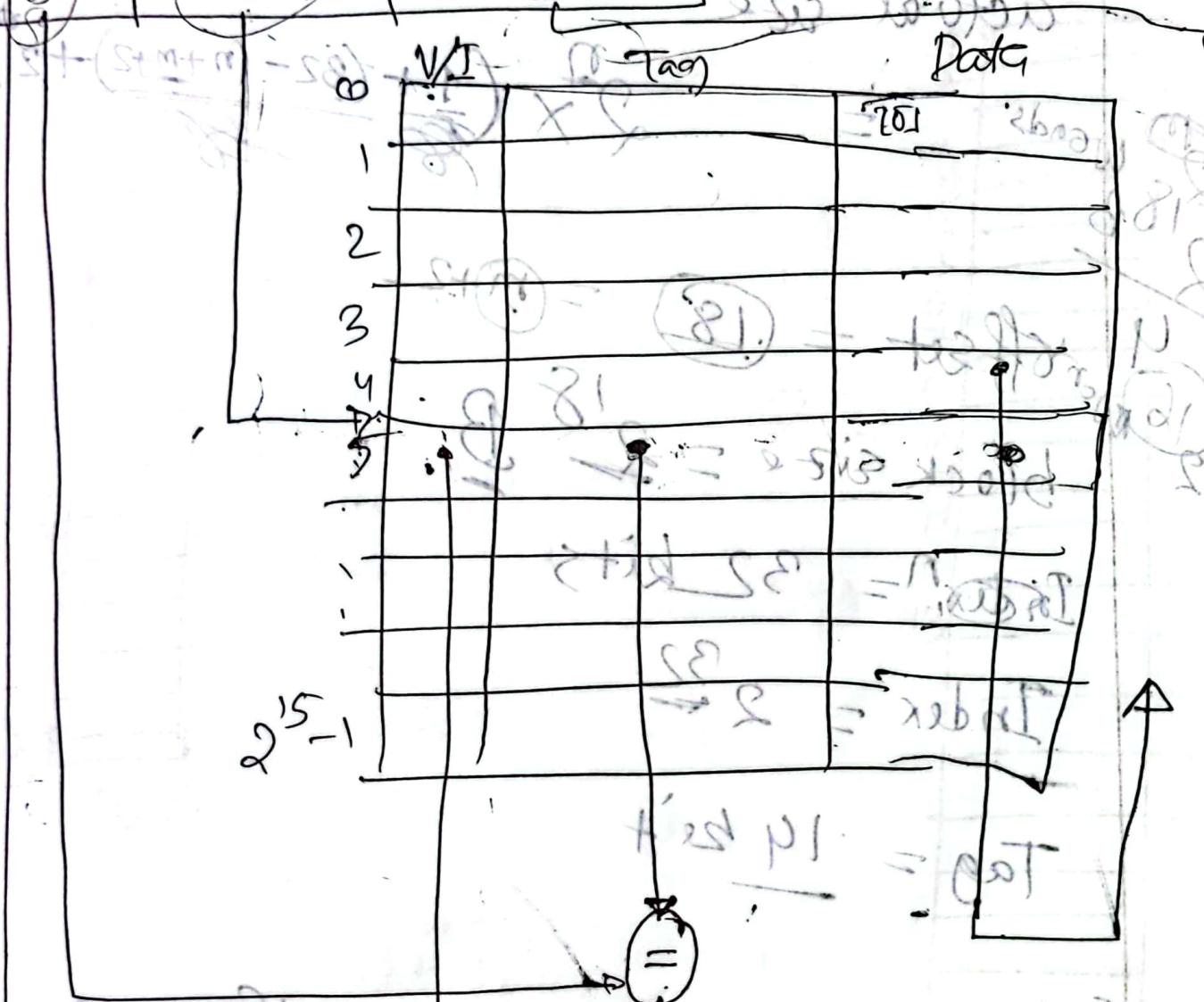
offset 0 0 1 0

reg.

15

12

start address



$$\text{ex: } \frac{((\text{start} + \text{size}) - \text{val})}{\text{size}} + 1 \times \text{size} = \text{Hit}$$

18

m+2

(2)

2^2

$$2^2 = 2k = m$$

actual size

10 words
~~2¹⁸B~~
2
4
16
~~= 2~~

~~Index~~, $n = 32$ bits

~~Index~~ = 2^{32}

Tag = 14 bits

(b)

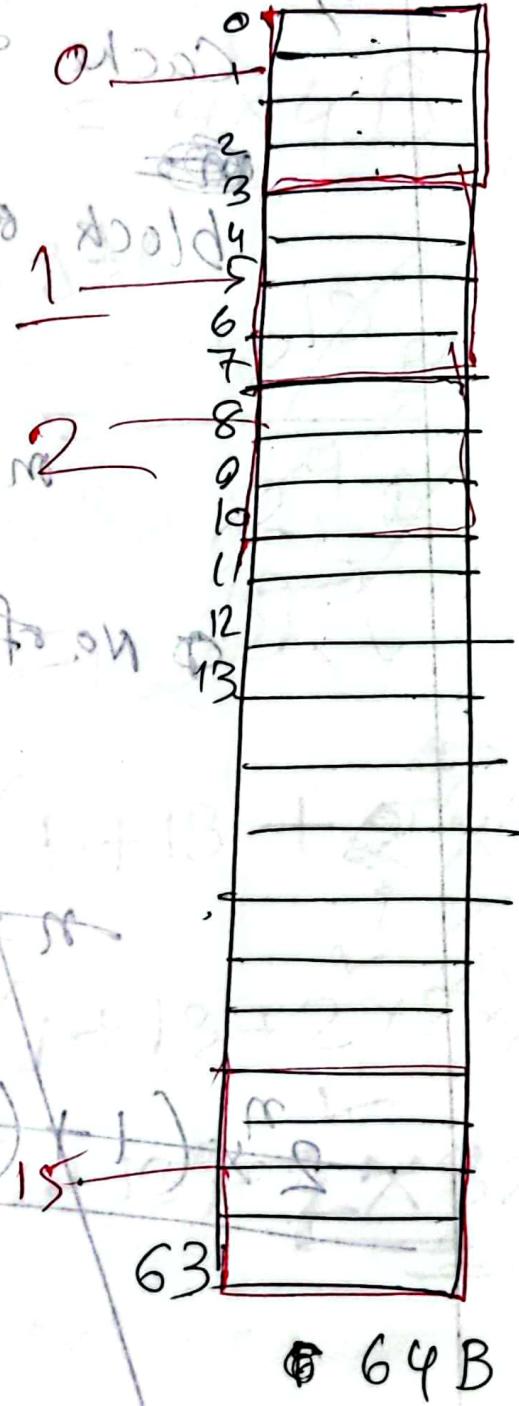
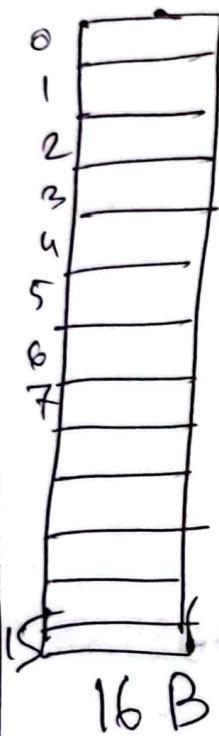
$$= 2^{32} \times (1 + 14 + 2^{18} \times 8) \text{ bits}$$

or

$$= 2^{32} \times (1 + \frac{1}{16} (64 - (32 + 16 + 2)) + 2^{16}$$

2

block size = 4B



$$= 4$$

$$\begin{aligned} \text{block size} &= 2^4 \text{ B} \\ &= \frac{2^4}{16} = 2 = 4 \text{ wtf} \end{aligned}$$

(91) 519 ← address (b)

Cache size $\leftarrow 2^{19} B$

$$\text{block size} = \frac{4}{4} B = 4 \times 4 B$$

$$= 12 B \text{ byte}$$

$$\therefore \text{No. of blocks} = \frac{2^{19} B}{2^4 B}$$

$$= 2^{15}$$

$$\text{cache actual size} = 2^{15} (1 + 13 + 2 \times 8)$$

$$= 2^{15} (1 + 13 + 2 \times 2 \times 8)$$

$$= 2^{15} (1 + 13 + \underline{\underline{2 \times 32}})$$

Cache block address = 111000000000000000000000

Cache block number = $2^{15}/2^4 = 2^{11}$

Cache block number = 100000000000000000000000

Cache block number = 100000000000000000000000

* block address = $\lfloor \frac{\text{given address}}{\text{block size}} \rfloor$

$$\lfloor \frac{1200}{16} \rfloor = 75$$

block number = $75 \% 64$
= 11

$$(8 \times 8 + 81 + 1) \text{ byte} = 256 \text{ bytes}$$

$$(8 \times 8 + 81 + 1) \text{ byte}$$

$$(8 \times 8 + 81 + 1) \text{ byte}$$

$$40 \text{ Cache size} = 4KB = 2^{12} B$$

$$\text{block size} = 4 \text{ word} = 4 \times 4 B = 2^4 B$$

$$\text{No. of blocks} = \frac{2^{12}}{2^4} = 2^8$$

1: \rightarrow

$$\text{block address} = \lfloor 1 / 2^4 \rfloor = 0$$

$$\text{block number} = 0 \% 2^8 = 0 \text{ Miss}$$

2: \rightarrow (miss)

$$\text{block address} = \lfloor 2 / 2^4 \rfloor = 0$$

$$\text{block number} = 0 \% 2^8 = 0 \text{ hit}$$

10: \rightarrow Hit

~~block address~~

4097: \rightarrow

$$\text{block address} = \lfloor 4097 / 2^4 \rfloor = 256$$

$$\text{block number} = 256 \% 2^8 = 0 \text{ Miss}$$

4098: \rightarrow Hit

15: \rightarrow Miss

~~(Q1)~~

Cache size = $S \times 2^8$ = 18×2^8 = 18 Kbytes

Block words = $2^8 \times 2^3$ = 32×8 = 256

No. of blocks = $\frac{18}{256} = 32768$

~~(Q2)~~

Block address = $42 / 2^3$ = 1

Block number = 1

~~(Q3)~~

$[0 - 0] = [2^3 - (2^3 + 2^8)]$

$0 = [N_S / S] (2^{15})$ (miss)

~~(Q4)~~

42 → miss

Block address = $\frac{42}{2^3} = 5$

Block number = $5 \cdot 32768 = 163840$

~~(Q5)~~ [S_{FFOP}] = 2^{13} bytes ← : FFOP

~~(Q6)~~ $0 \times 2^2 \times 2^3 = [40 \text{ units}]$

~~(Q7)~~ $+ 12 \leftarrow : 800\beta$

~~(Q8)~~ ← : ?

3 → miss

$$\text{block add } \left(\frac{2^8 \times 7}{2^3} \right) = 0 \quad (\text{miss ratio})$$

660 words

$$\text{block num } 2826.1.32768 = 0 \quad (\text{miss ratio})$$

$$(\rightarrow \text{miss ratio}) \leftarrow \{0 - 17\}$$

$$\underline{15} \rightarrow \text{Hit} \quad (\text{miss ratio})$$

miss ratio

$$\underline{40} \rightarrow \text{Hit}$$

$$\underline{39} \rightarrow \text{miss} \quad (8821.1.5 = \text{miss ratio})$$

$$\text{probability} = \frac{4}{6.12} - (\text{miss ratio})$$

$$\text{probability} = \frac{4}{6.12} - 0.66$$

still on

$$\underline{60} \rightarrow \text{fill} \leftarrow \underline{21} \rightarrow \text{fill} \leftarrow \underline{18} \rightarrow \text{fill} \leftarrow \underline{15} \rightarrow \text{fill} \leftarrow \underline{12} \rightarrow \text{fill} \leftarrow \underline{9} \rightarrow \text{fill} \leftarrow \underline{6} \rightarrow \text{fill} \leftarrow \underline{3} \rightarrow \text{fill} \leftarrow \underline{0}$$

$$\text{cache size} = 2^8 \times 2^10 = 2^{18}$$

$$\text{block size} = 2^8 \times 2^4 = 2^{12}$$

$$\text{No of blocks} = \frac{2^{18}}{2^4} = 2^{14} = 16384$$

Element
addr.



12 (miss)

$$\text{block add} = \left(\frac{12}{2^4} \right) = 6 \text{ (miss)}$$

$$\text{block num} = 2^1 \cdot 16384 = 16384 \text{ (miss)}$$

$$(0 \text{ } 15) \rightarrow (0 \text{ } 0 \text{ } 0 \text{ } 2^{-1}) \text{ (miss)}$$

42 ~~miss~~

$$\text{block add} = \left(\frac{42}{2^4} \right) = 2 \text{ (hit)}$$

$$\text{block num} = 2^1 \cdot 16382 = 2 \text{ (hit)}$$

$$\text{Cache } 2 \times 2 - \{ 2 \times 2^4 + (2^4 - 1) \}$$

$$(32 - 47) \text{ = Cache miss}$$

$\frac{7}{2} \rightarrow \text{Hit}, \frac{15}{2} \rightarrow \text{Hit}, \frac{40}{2} \rightarrow \text{Hit}, \frac{39}{2} \rightarrow \text{Miss}$

$$\text{miss rate} = \frac{2}{6} = \frac{1}{3} \text{ or } 33.33\% \text{ misses}$$

$$18821 = \frac{4}{6} = \frac{815}{15} \text{ or } 54.33\% \text{ misses}$$

$$\text{Cache size} = 2^{15} \text{ B} = 2^7 \text{ B}$$

$$\text{block size} = 128 \text{ B} = 2^8$$

$$\text{No. of block} = \frac{2^15}{2^8} = 2^7$$

Data [128 bytes]

Address [07]

$$\text{Tag} = 17 \text{ bits}$$

Tag directory

$$2^8 \times 17 \text{ bit}$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
3	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
5	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
6	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
7	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
8	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
9	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
11	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
12	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
13	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
14	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
15	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
16	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
17	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
18	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
19	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
20	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
21	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
22	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
23	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
24	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
25	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
26	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
27	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
28	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
29	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
30	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
31	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

miss = (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31)

No. of miss in each row = 1/9

$$\text{Total Miss} = 4 \times 64 = 256$$

$$\text{Miss ratio} = \frac{256}{64 \times 63} = 0.0628$$

Spring 22

(a)

$$\text{cache size} = 2 \times 2^{10} = 2^{11}$$

$$\text{block size} = 2^5 = 32$$

$$\text{No. of block} = \frac{2^{11}}{2^5} = 2^6 = 64$$

15 → miss

$$\text{block address} = \lfloor 15 / 2^5 \rfloor = 0 \text{ word (0)}$$

$$\text{block number} = 0 \% 64 = 0$$

$$[0 - 31]$$

10 - hit $[0 - 31]$

4097 - miss

$$\text{block address} = \lfloor 4097 / 2^5 \rfloor = 128$$

$$\text{block number} = 128 \% 64 = 0$$

$$128 \times 2^5 - \{128 \times 2^5 + (2^5 - 1)\}$$
$$= 4096 - 4127 = 69 \text{ words}$$

~~4098~~ → hit [4096 or ~~4 + 27~~] N

~~7~~ → hit [0 - 32] N word

~~30~~ → miss [0 - 31] N word

$$\text{miss rate} = \frac{(1-p)^4 + p(1-p)^3}{6} = 0.33$$

(1111 111001)

cache size = 2

block size = 16 = 2^4 ← 8 con

No. of block = $\frac{2^4}{2^4} = 1$ ← 1

~~15~~ → miss
block address = $[15/2^4] = 0$

block number = $0 \cdot 1 \cdot 128 = 0$

with $[0 - 15]$ star and with

-1 + 16, star and with 16, star and with 16

~~19~~ → miss
block address = $[19/2^4] = 1$ ← 1

block number = $1 \cdot 128 = 1$ or

$$1 \times 2^4 - \{1 \times 2^4 + (2^4 - 1)\}$$

$$16 - 27$$

$$\begin{aligned} \underline{4097} \rightarrow \text{miss} & \quad \left. \begin{array}{l} \text{find } \leftarrow \text{soon} \\ \text{block address} = \left[\frac{4097}{2^4} \right] = 256 \end{array} \right\} \text{find } \leftarrow \text{soon} \\ \text{block number} &= 256 \times 128 = 0 \\ 256 \times 2^4 &= \{256 \times 2^4 + (64-1)\} \\ &= 4096 - 4111 \end{aligned}$$

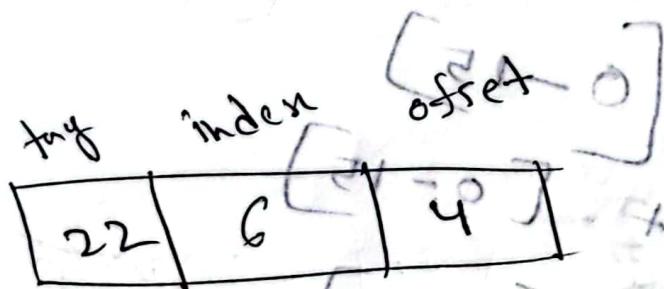
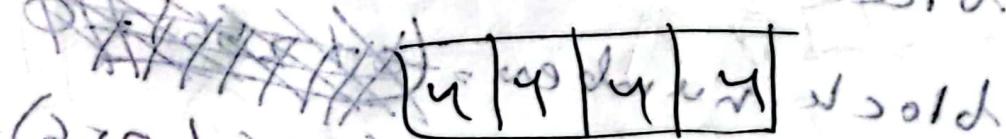
~~4098~~ → Hit $\frac{4}{4} = 21 = 3512$ words
~~51~~ → $\frac{4}{4} = 21 = 3512$ words
~~7~~ → Hit $\frac{4}{4} = 21 = 3512$ words
~~30~~ → Hit
miss rate $= \frac{3}{46} = 0.5$ words \leftarrow words
this miss rate is greater than the previous miss rate because in previous it can store more bytes so it maintain ~~coherency~~ spatial locality of 4×1 \rightarrow words words
 $T_{FS} = 217$

(a)

Tag size = 19 bits

Index $n = 5 = 317$ bits $\times N = 3512 \text{ words}$ Offset $n = 6$ bits $\times N = 3512 \text{ words}$ Number of block = 2 \rightarrow 1 word to 2 blocks

Number of block = 2

Bytes per block $= 2^{6 \times 2} = 2^6 \text{ bytes per block}$ Number of block $= 2^6 = 64$ blocksBytes per block $= 2^6 = 64 \text{ bytes per block}$ (21-0) \rightarrow 1 word(225.1225) \rightarrow 1 word

0 =

Spring - 23

Q

(a)

$$\text{Cache size} = \frac{\text{st. add}}{2^4} \times 2^4 = 2^8 \text{ words}$$

10 bits / 4 bits per word

$$\text{block size} = \frac{\text{st. add}}{2^4} \times 2^4 = 2^8 \text{ bytes}$$

10 bits / 4 bits per word

$$\text{No. of block} = \frac{2^{12}}{2^4} = 2^8 = 256$$

words

$$\text{word} = \frac{\text{miss}}{\text{st. add}} = \frac{2^8}{2^4} = 2^4 = 16$$

words

$$\text{block add} = \left\lceil \frac{1}{2^4} \right\rceil = 0$$

$$\text{block number} = 0 \cdot 256 = 0 \text{ block}$$

$$[0-15]$$

$$\Rightarrow \text{hit} [0-15] \text{ block}$$

$$10 \rightarrow \text{hit} [0-15]$$

$$1097 \rightarrow \text{miss}$$

$$\text{block add} = \left\lceil \frac{1097}{2^4} \right\rceil = 256$$

$$\text{block number} = \frac{1097}{256} = 4$$

$$\Rightarrow (256, 256)$$

$$= 0$$

~~COPYP~~

$$\text{step } 2^{56} \times 2^4 + \{2^{56} \times 2^4 + (2^4 - 1)\}$$

~~8 words~~

$$= (4096 - 4113)$$

~~(8 words) miss 4113~~

4096 → hit

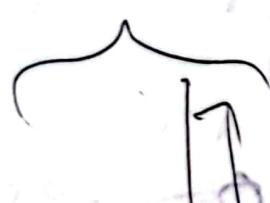
15 → hit

=

miss rate = $\frac{2}{(8+1)} = 0.125$ words

~~at cd re~~

$N_2 = \frac{2}{(8+1)} = 0.125$ words



$$0 = (581.25) = 560 \text{ words}$$

$$0 = (12.5) = 12 \text{ words}$$

$$0 = (1.5625) = 1.5625 \text{ words}$$

$$0 = (0.1953125) = 0.1953125 \text{ words}$$

$$0 = (0.0244140625) = 0.0244140625 \text{ words}$$



~~Cache size~~ = $2^{12} \times 8$ byte
~~= 2048~~ byte

~~Block size = (32 × 8)~~

~~[32 × 8]~~ ← ~~800~~
~~[32 × 8]~~ ← ~~81~~

~~No. of blocks = 8~~

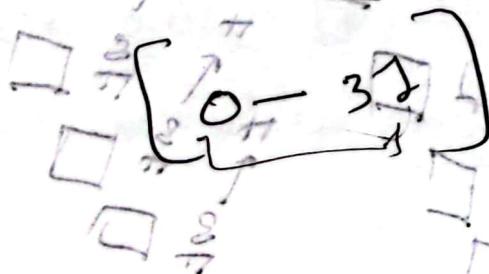
~~block size = (4 × 8)~~ → ~~short error~~
~~= 32 bytes~~

~~No. of blocks = 64~~

~~15 → miss~~

~~block add = (15 / 32) = 0~~

~~block num = (0 / 64) = 0~~



0.00125

10 → Hit

4097 → miss \rightarrow 885152 \rightarrow 2512 mod 32

block add $= \lfloor 4097 / 32 \rfloor$

$\rightarrow 83$ F \rightarrow 128 mod 32 to on

block num = $(128 \cdot 1.6^4)$

$\leftarrow r = m \cdot \frac{N}{r} = 50$

$[4097 - 4127] \rightarrow 10$

4098 → Hit

21 → miss \rightarrow 3512 mod 32

$[0 - 31]$

30 → Hit \rightarrow 3512 mod 32 post

$r + s + e + t + n$

?

$y_{max} +$

~~(a)~~
6 bytes
50 words

cache size = 524288 bytes
~~1 word~~ ~~4 bytes~~ ~~4 words~~ ~~4 bytes~~ ~~4 words~~

block size = 16 bytes
~~1 word~~ ~~4 bytes~~ ~~4 words~~

no of block = 32768 words

~~(12.1.8+1)~~ = min words

block size = $2^4 = m = 2$

2

block size = 2 m bits $\leftarrow 800\text{?}$

No of block = 2^{15} $n \leftarrow 15$

\therefore tag size = $32 - (15 + 2 + 2)$
 $= 13$ bits

$2^n \times (1 + 13 + 2^4 \times 32)$