TOPIC NAME _____ DAY _____ TIME _____ DATE / /

Lecture 3

R type: (Add, Sub) (rd, rs, rt)

A = B + C ($s0 = 16, $s1 = 17, $s2 = 18)

32 add $s0, $s1, $s2;

OP	rs	rt	rd	shamt	funct
0	17	18	16	0	32
000000	10001	10010	10000	00000	100000
6 bit	5 bit	5 bit	5 bit	5 bit	6 bit

I type: (lw, sw, addi, subi, beq, bne): (rt, offset(rs))

A[5] = A[10] + B ($s0 = 16, $s0 = 16, $s1 = 17)

35 lw $t0, 4x10($s0); (des(rt) = $t0, source(rs) = 4x10($s0))

32 add $t0, $t0, $s1;

45 sw $t0, 4x5($s0); (source(rt) = $t0, des(rs) = 4x5($s0))

OP	rs	rt	offset
35	16	8	40
100011	10000	01000	0000000000101000
6 bit	5 bit	5 bit	16 bit

OP	rs	rt	offset
45	16	8	20
101101	10000	01000	0000000000010100

J type:

J Lab1 (100)

OP.	offset/address
2	100
000010	00000000000000000001100100
6 bit	26 bit

GOOD LUCK

TOPIC NAME : DAY
TIME DATE / /

$A[3] = A[2] + B + 5$ (A: $s_0 = 16$, B: $s_1 = 17$)

Assembly code:

```
35  lw   $t0, 2x4($s0)
32  add  $t0, $t0, $s1
8   addi $t0, $t0, 5
45  sw   $t0, 3x4($s0)
```

machine code:

op	rs	rt	offset
35	16	8	8
100011	10000	01000	0000000000001000
6 bit	5 bit	5 bit	16 bit

op	rs	rt	rd	shamt	funct
0	8	17	8	0	32
000000	01000	10001	01000	00000	100000

op	rs	rt	immediate
8	8	8	5
001000	01000	01000	0000000000000101

op	rs	rt	offset
45	16	8	12
101101	10000	01000	0000000000001100

GOOD LUCK

TOPIC NAME _____ DAY _____

TIME _____ DATE : / /

Booth Algo:

$2 \times (-3)$

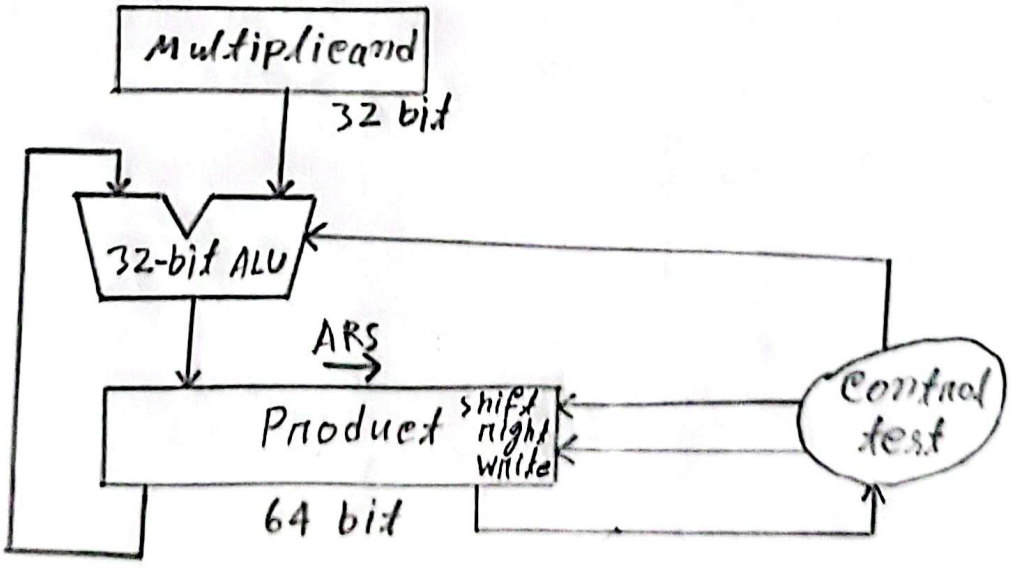
↓ Mn ↓ Mx

$2 \Rightarrow 0010$

$-3 \Rightarrow 1101$

Mx_0	B
0	0 → NOP
0	1 → Add
1	0 → Sub
1	1 → NOP

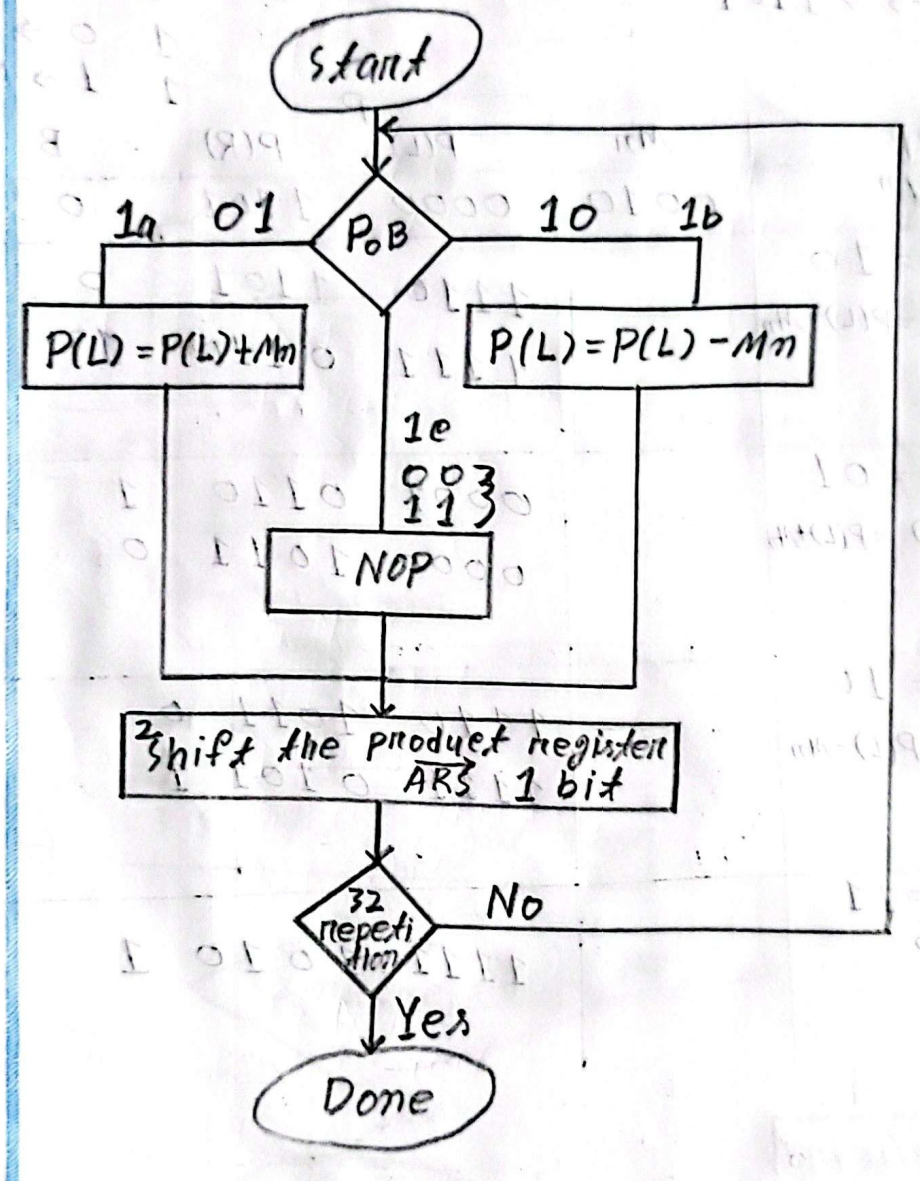
iteration	step	Mn	P		
			P(L)	P(R)	B
0	init^n	0010	0000	1101	0
1	1. P₀B = 10 1b. P(L) = P(L) − Mn 2. ARS → PB		1110 1111	1101 0110	0 1
2	1. P₀B = 01 1a. P(L) = P(L) + Mn 2. ARS → PB		0001 0000	0110 1011	1 0
3.	1. P₀B = 10 1b. P(L) = P(L) − Mn 2. ARS → PB		1110 1111	1011 0101	0 1
4.	1. P₀B = 11 1c. NOP 2. ARS → PB		1111	1010	1



GOOD LUCK

TOPIC NAME : _____ DAY : _____

TIME : DATE : / /



GOOD LUCK

TOPIC NAME : DAY
TIME : DATE : / /

Divide Version 2

$Dn = 7 \Rightarrow 0111$

$D = 2 \Rightarrow 0010$

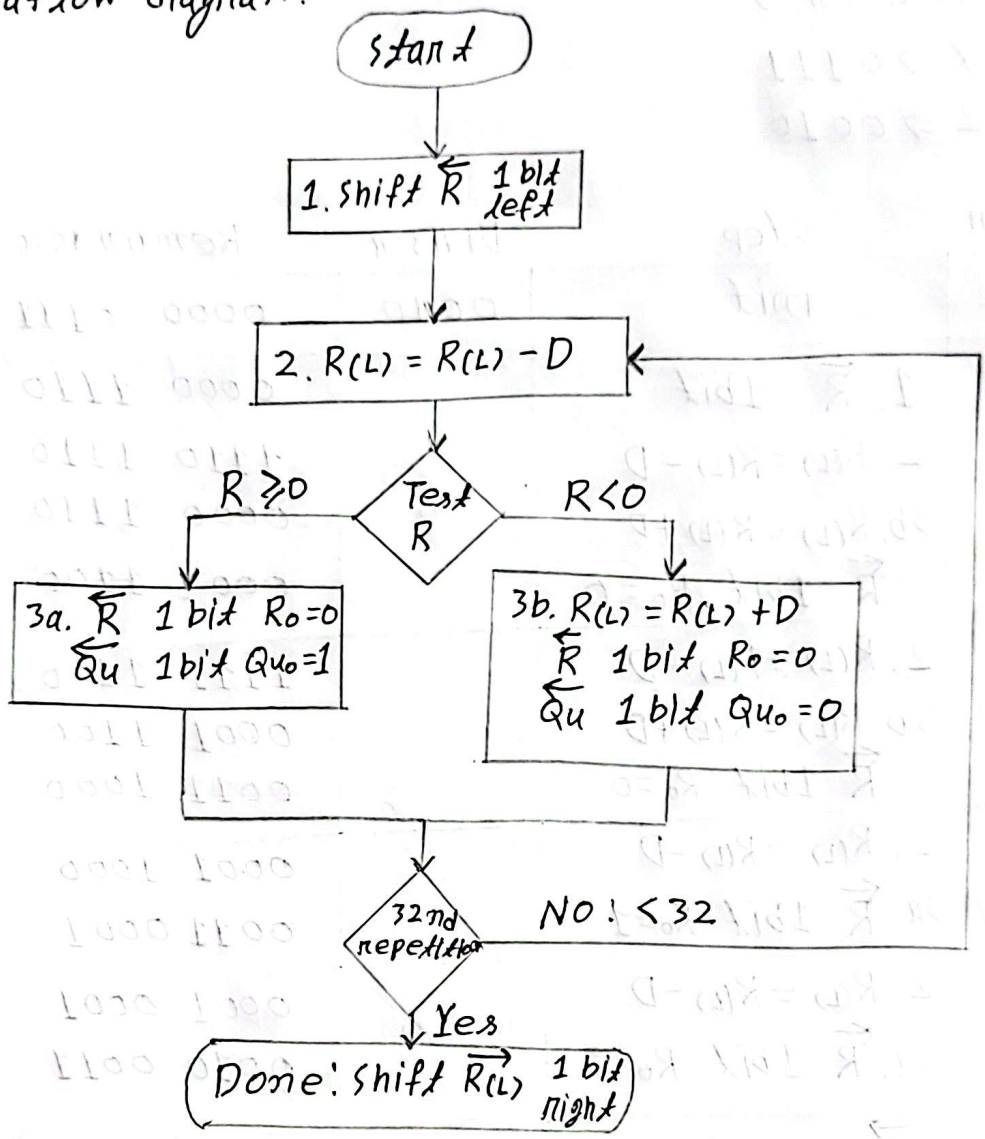
iteration	step	Quotient	Divisor	Remainder
0	init	0000	0010	0000 0111
1.	1. $\overleftarrow{R}$ 1 bit 2. $R_{(L)} = R_{(L)} - D$ 3b. $R_{(L)} = R_{(L)} + D$ $\overleftarrow{R}$ 1 bit $R_0 = 0$ $\overleftarrow{Qu}$ 1 bit $Qu_0 = 0$	0000		0000 1110 1110 1110 0000 1110 0001 1100
2.	2. $R_{(L)} = R_{(L)} - D$ 3b. $R_{(L)} = R_{(L)} + D$ $\overleftarrow{R}$ 1 bit $R_0 = 0$ $\overleftarrow{Qu}$ 1 bit $Qu_0 = 0$	0000		1111 1100 0001 1100 0011 1000
3.	2. $R_{(L)} = R_{(L)} - D$ 3a. $\overleftarrow{R}$ 1 bit $R_0 = 0$ $\overleftarrow{Qu}$ 1 bit $Qu_0 = 1$	0001		0001 1000 0011 0000
4.	2. $R_{(L)} = R_{(L)} - D$ 3a. $\overleftarrow{R}$ 1 bit $R_0 = 0$ $\overleftarrow{Qu}$ 1 bit $Qu_0 = 1$	0011		0001 0000 0010 0000
	$\overrightarrow{R_{(L)}}$ 1 bit			0001 0000

GOOD LUCK

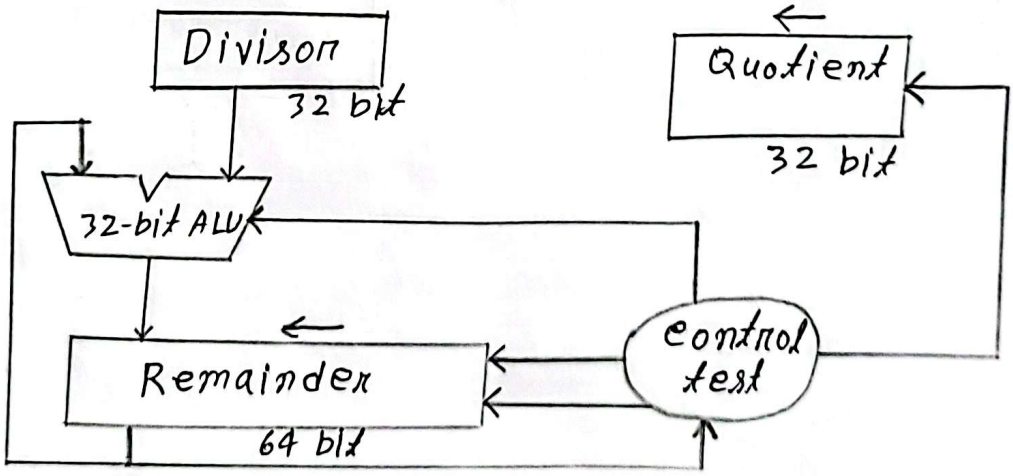
TOPIC NAME _____ DAY _____

TIME _____ DATE : / /

Dataflow diagram:



Hardware:



GOOD LUCK

TOPIC NAME : DAY :
TIME : DATE : / /

Divide Version 3 !

$D_n = 7 \Rightarrow 0111$

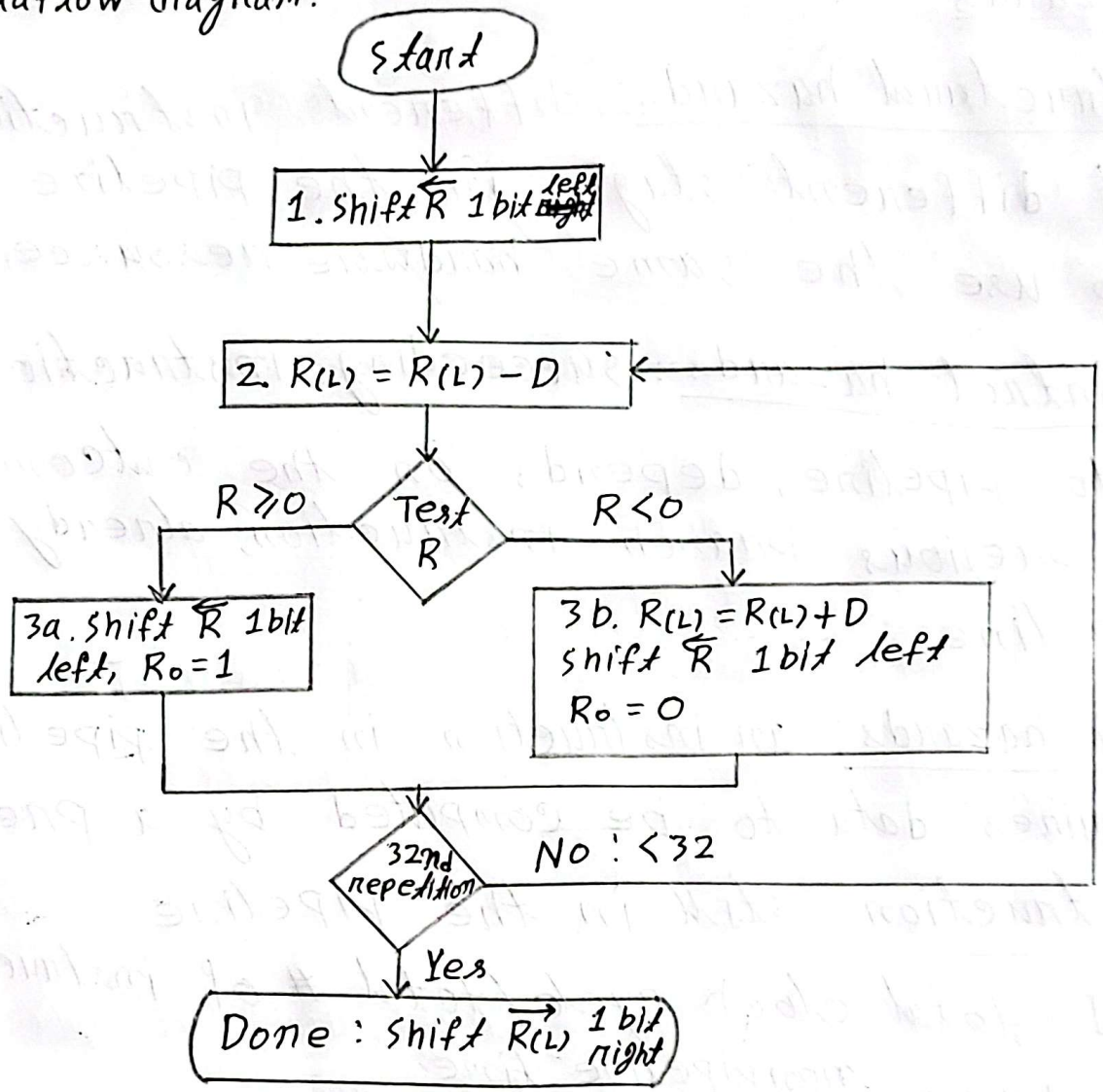
$D = 2 \Rightarrow 0010$

iteration	step	Divisor	Remainder
0	init	0010	0000 0111
1.	1. $\overleftarrow{R}$ 1bit		0000 1110
	2. $R_{(L)} = R_{(L)} - D$		1110 1110
	3b. $R_{(L)} = R_{(L)} + D$		0000 1110
	$\overleftarrow{R}$ 1bit $R_0 = 0$		0001 1100
2.	2. $R_{(L)} = R_{(L)} - D$		1111 1100
	3b. $R_{(L)} = R_{(L)} + D$		0001 1100
	$\overleftarrow{R}$ 1bit $R_0 = 0$		0011 1000
3	2. $R_{(L)} = R_{(L)} - D$		0001 1000
	3a $\overleftarrow{R}$ 1bit $R_0 = 1$		0011 0001
4.	2. $R_{(L)} = R_{(L)} - D$		0001 0001
	3a. $\overleftarrow{R}$ 1bit $R_0 = 1$		0010 0011
	$\overrightarrow{R_{(L)}}$		0001 0011

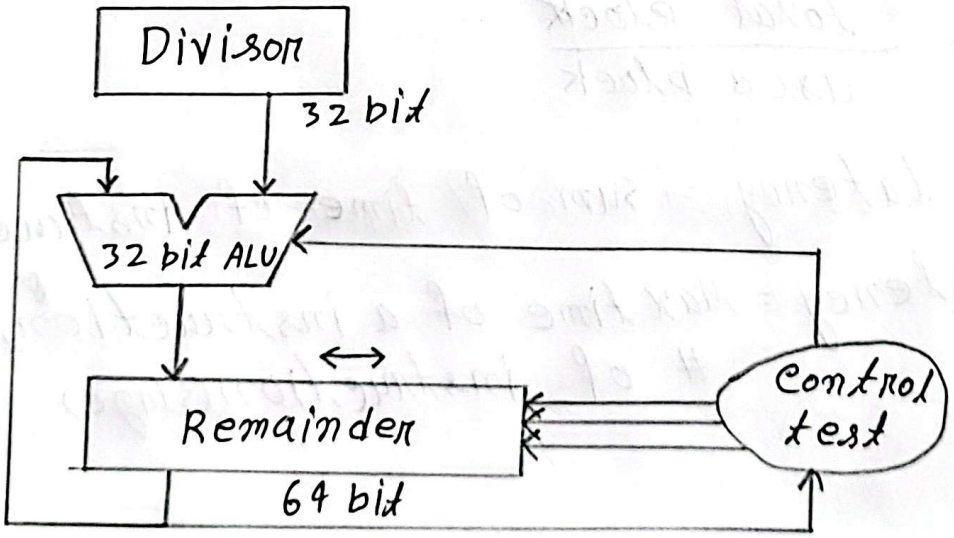
GOOD LUCK

TOPIC NAME : DAY
TIME : DATE : / /

Dataflow diagram:



Hardware:



GOOD LUCK

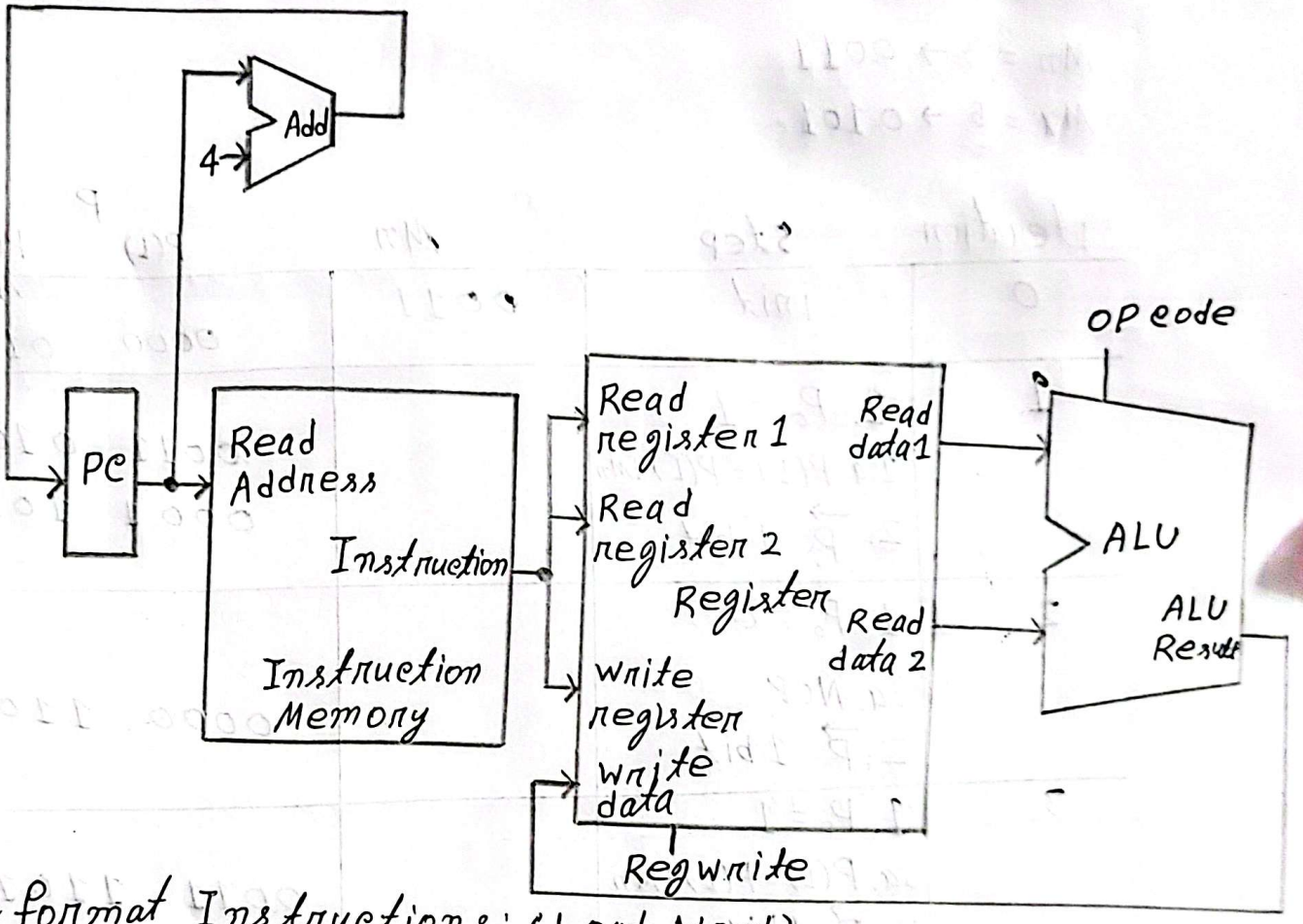
Read → lw
Write → sw

TOPIC NAME Data path

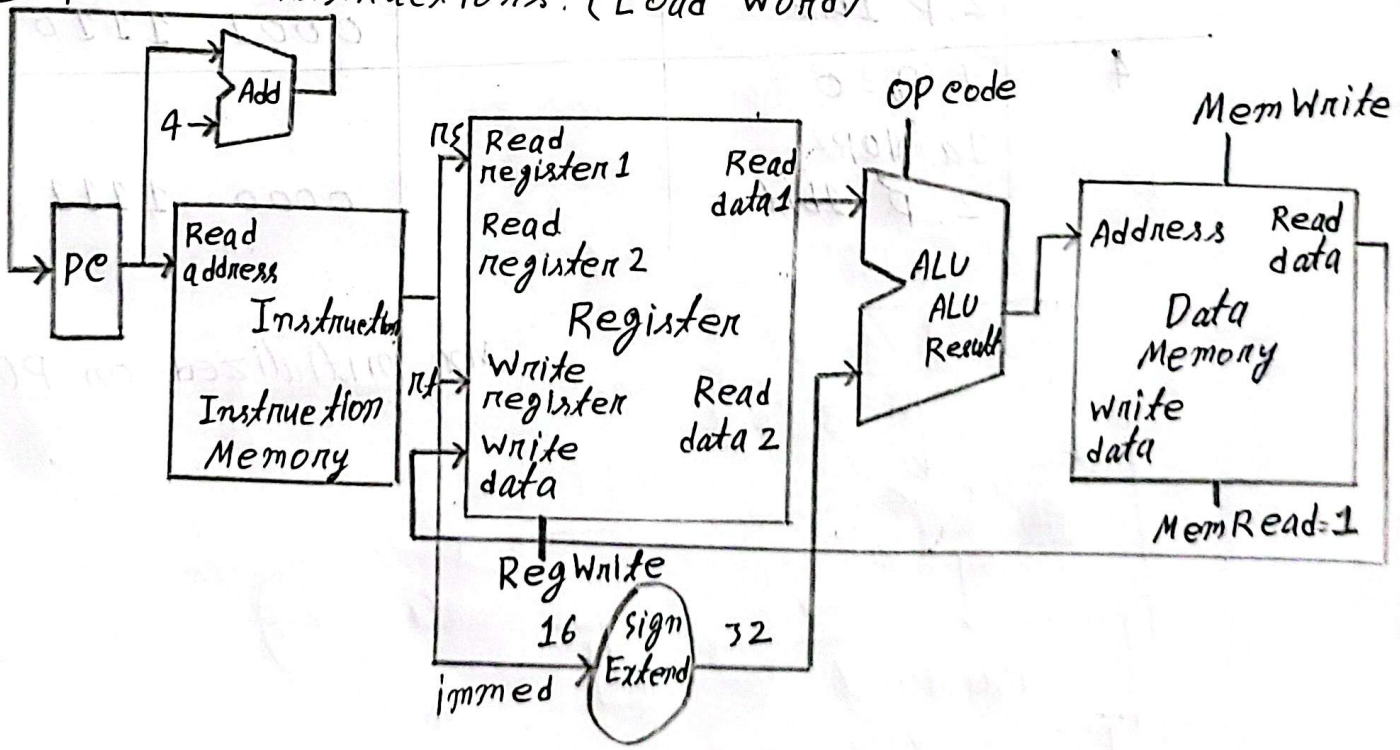
DAY

TIME DATE / /

R-format Instructions:



I-format Instructions: (Load word)

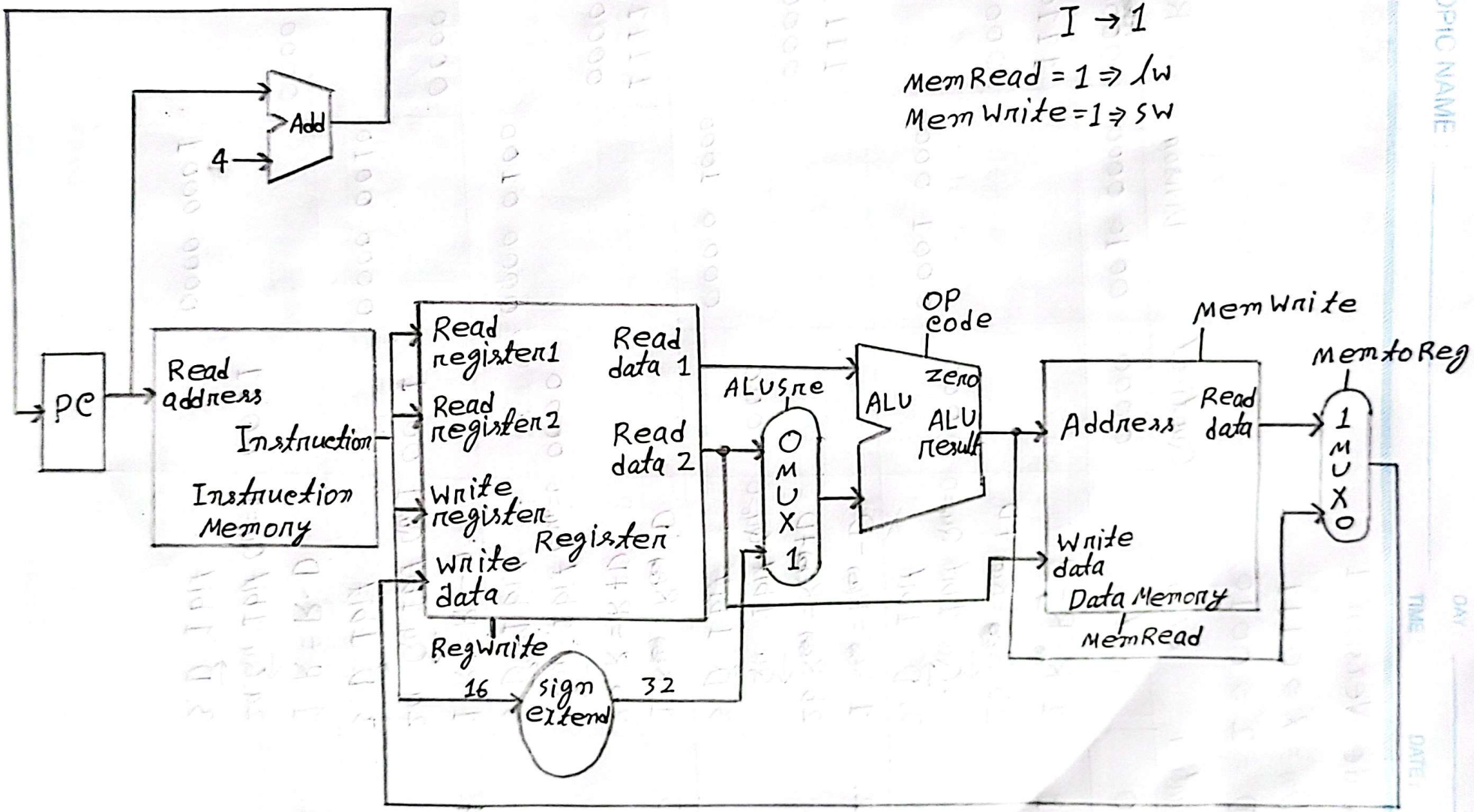


GOOD LUCK

Both R, I type

In MUX: R → 0
I → 1

MemRead = 1 ⇒ lw
MemWrite = 1 ⇒ sw



TOPIC NAME : Pipeline

DAY :

TIME : DATE : / /

Hazards :

Structural hazards: different instructions, at different stages, in the pipeline want to use the same hardware resource.

Control hazards: succeeding instruction, to F into pipeline, depends on the outcome of a previous branch instruction, already in pipeline

Data hazards: an instruction in the pipeline requires data to be computed by a previous instruction still in the pipeline.

CPI = total clock cycle / total # of instructions

$$\text{speed up} = \frac{\text{nonpipeline time}}{\text{pipeline time}}$$

$$\text{Efficiency} = \frac{\text{total block}}{\text{used block}}$$

Nonpipeline latency = sum of time of instruction

pipeline latency = Max time of a instruction, @ (stage) * # of instruction (stage)

GOOD LUCK

TOPIC NAME _____ PAGE NO. _____ TIME _____ DATE : / /

Non pipeline time :

latency * # of instruction

pipeline time :

latency + (Max time of instruction * (# of instruction - 1))

TOPIC NAME : Memory.

DAY :

TIME : DATE : / /

Mapping functions:

1. Direct mapped: one unique cache location for each memory block.

   cache block address = (requested address) % (cache size)

2. Set Associative: each memory block can place in a unique set of cache location

   $$\#\text{ of set} = \frac{\text{cache size}}{\#\text{ of way}} = \frac{8}{2} = 4$$

   cache block address = (requested address) % (# of set)

3. Fully Associative: each memory block can place any where in cache

Miss ⇒ Data not in the cache, add the data.

Hit ⇒ Data present in the cache

For overwrite use LRU

12, 8, 25, 3, 8, 12, 11

For 11 overwrite it over 25

GOOD LUCK

TOPIC NAME : _____ DAY _____

TIME _____ DATE : / /

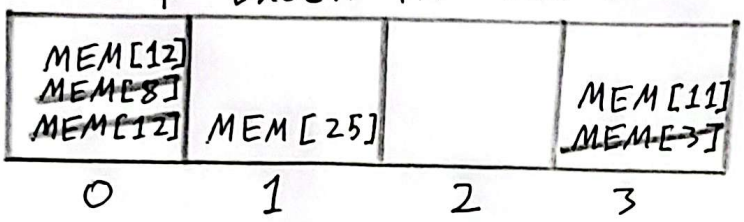
Mapping function:

12, 8, 25, 3, 8, 12, 11

For a cache with four 1-word blocks

1. Direct mapped

4 block in cache

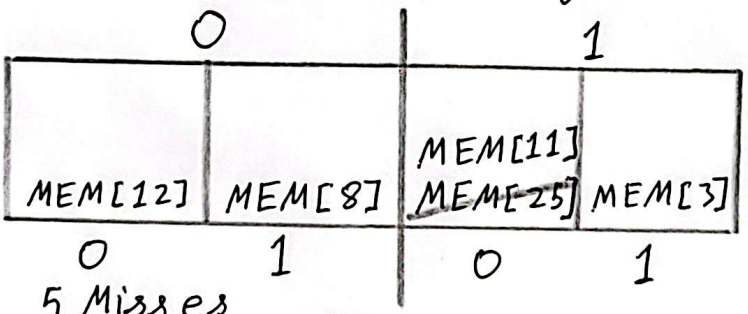


6 Misses

12 % 4 = 0 Miss
8 % 4 = 0 Miss
25 % 4 = 1 Miss
3 % 4 = 3 Miss
8 % 4 = 0 Hit
12 % 4 = 0 Miss
11 % 4 = 3 Miss

2. Set Associative (2-way):

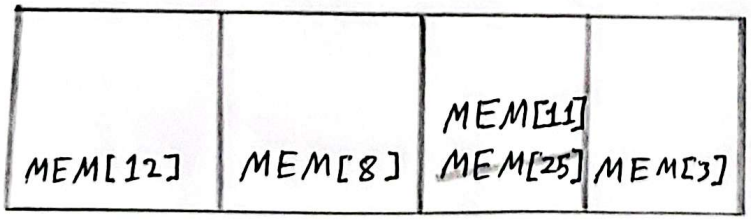
$$\#\text{ of set} = \frac{\text{cache size}}{\#\text{ of way}} = \frac{4}{2} = 2$$



5 Misses

12 % 2 = 0 Miss
8 % 2 = 0 Miss
25 % 2 = 1 Miss
3 % 2 = 1 Miss
8 % 2 = 0 Hit
12 % 2 = 0 Hit
11 % 2 = 1 Miss

3. Fully Associative:



12 → Miss
8 → Miss
25 → Miss
3 → Miss
8 → Hit
12 → Hit
11 → Miss

GOOD LUCK