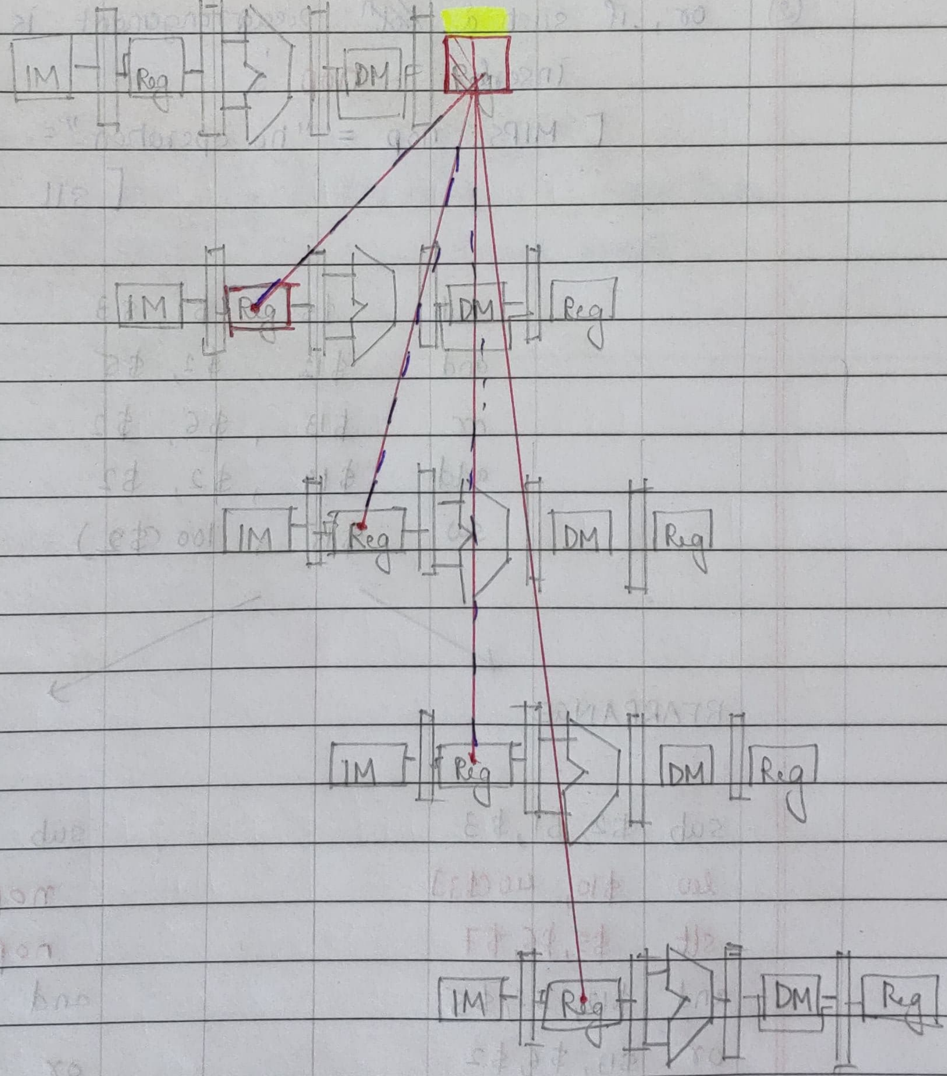
U19CS012

Q1) Draw the pipelining execution for the following code and detect and resolve the hazard, if any.

Time(in clock cycles) ⟶

	CC1	cc2	cc3	cc4	cc5	cc6	cc7	cc8	cc9
Value of register $2	10	10	10	10	10/-20	-20	-20	-20	-20

[Program execution order]



sub $S2, $1, $3
↓
EX stage

data hazard

and $12, $2, $5
↓
ID stage

or $13, $6, $2

and $14, $2, $2

sw $15, 100($2)

Data Hazard

[--- ↳ Data hazard]

U19CS012

Software Sol^{n}:

① - By rearranging to insert independent instructions that would otherwise have a data hazard between them

OR

② or, if such a ~~soln~~ rearrangement is not possible, insert nop's

[MIPS: nop = "no operation" = 0000...0 (32 bits)]

[sll $0, $0, 0]

```
sub     $2,   $1, $3
and     $12 , $2, $5
or      $13 , $6, $2
add     $14 , $2, $2
sw      $15 , 100 ($2)
```

REARRANGE

```
sub $2, $1, $3
lw  $10, 40($3)
slt $5, $6, $7
and $12, $2, $5
or  $13, $6, $2
add $14, $2, $2
sw  $15, 100($2)
```

NOP

```
sub  $2, $1, $3
nop
nop
and  $12, $2, $5
or   $13, $6, $2
add  $14, $2, $2
sw   $15, 100($2)
```

- Such compiler solutions may not always be possible, and 'nop''s slows the machine down

Q2> Reorder code to avoid Pipeline stall

```
lw   $t0, 0($t1)
lw   $t2, 4($t1)
sw   $t2, 0($t1)
sw   $t0, 4($t1)
```

[RAW]
data hazard

Re-ordered code:

```
lw   $t0, 0($t1)
lw   $t2, 4($t1)
sw   $t0, 4($t1)
sw   $t2, 0($t1)
```

No Hazard

[Interchange ∴ No Data Hazard Now]

———— × ————