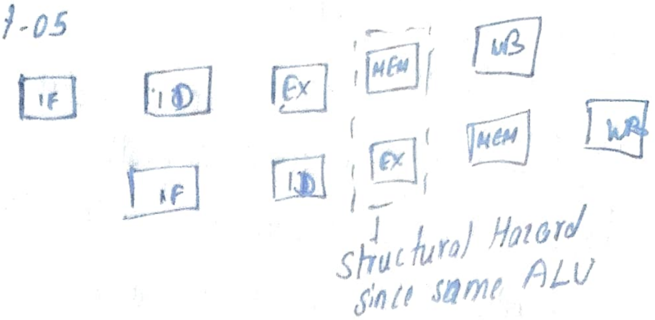
1) add3 r0, r1, r2

add3 r3, r4, r5

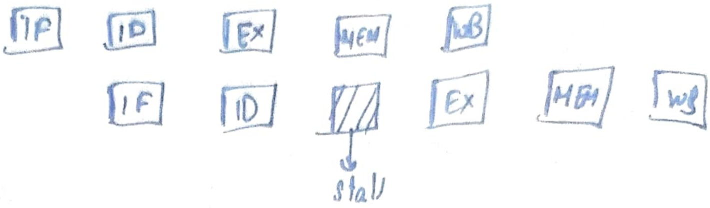
Jul-05



This solution is correct
-Bhaskar

∴ add3 r0, r1, r2

add3 r3, r4, r5



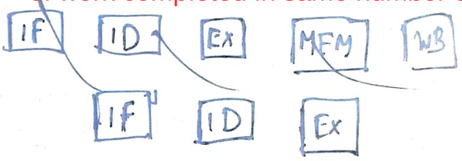
2) Yes, due to add3, considering only the structural stalls caused by it, we get atmost 1 structural stall due to add3, which matches with 2 instructions of (add, add) but the structural stall needn't occur always, add3 improves performance

This answer is correct
-Bhaskar

If we make the simplifying assumption that add3 is followed by an instruction using ALU, then perf with/without add3 will be the same - same amount of work completed in same number of cycles

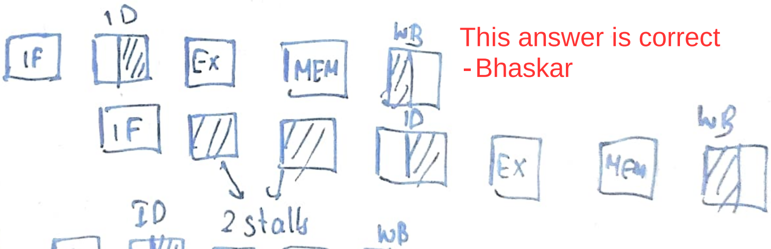
~~3. lw $t0 4($sp)~~

~~sw $t1 8($t0)~~



3. lw $t0 4($sp)

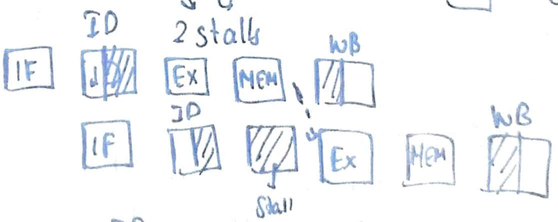
sw $t1 8($t0)



This answer is correct
-Bhaskar

5. a),b) ~~addi~~ lw $sp ~~$sp 4~~ 0($s0)

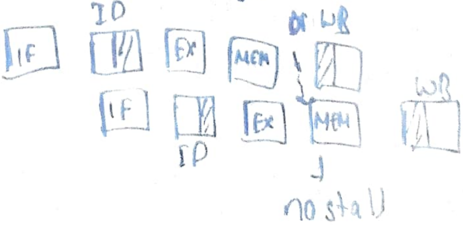
lw $t0 0($sp)



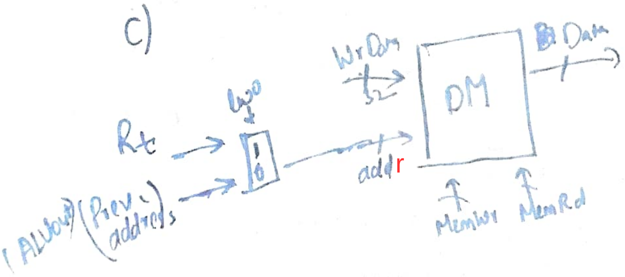
This ans is correct, except that usually we won't lw from memory to $sp
-Bhaskar

lw $sp 0($s0)

lw $t0 $sp



c)



This answer is correct
-Bhaskar

Forwarding to Exstage

4) 1) EX/MEM to EX

```
add r1 r2 r3
add r5 r1 r4
```

2) MEM/WB to EX

```
add r1 r2 r3
nop
add r5 r1 r4
```

Forwarding to Mem Stage

3) MEM/WB to MEM

```
lw $t2 0($t1)
sw $t2 0($t3)   (with a stall)
```

4) Post WB to MeM

```
add R1, R2, R3
nop
sw 0(R6), R1
```