

(15 pts) Exercise 4-81

- A.) Draw a pipeline stage diagram for the following sequence of instructions. You don't need fancy pictures – just text for each stage: ID, MEM, etc.
- SHOW the cycle number for each column, starting at cycle #1.

```
lw    $v0, 0($a0)
lw    $v1, 0($v0)
add   $a0, $a0, $v1
sub   $t0, $t0, $a0
```

B.) Now assume that the above sequence of instructions is repeated 1000 times (so the processor does a load, another load, an add, a sub, then back to a load, another load, an add, ...). There are no branches, just these 4 instructions repeated 1000 times. Do NOT reorder the code to improve efficiency.

i.) What is the total number of cycles needed to execute those 4000 instructions?

ii.) What is the average CPI for executing that sequence?

(15 pts) Exercise 4-82

- A.) Do the same thing as with Exercise 4-81, except assume that there is NO forwarding.

```
lw    $v0, 0($a0)
lw    $v1, 0($v0)
add   $a0, $a0, $v1
sub   $t0, $t0, $a0
```

- B.) Again assume that the above sequence of instructions is repeated 1000 times (so the processor does a load, another load, an add, a sub, then back to a load, another load, an add, ...). There are no branches, just these 4 instructions repeated 1000 times. Do NOT reorder the code to improve efficiency. As with immediately above, assume there is NO forwarding.

- i.) What is the total number of cycles needed to execute those 4000 instructions?

- ii.) What is the average CPI for executing that sequence?

(5 pts) Exercise #4-86

- Can you rewrite this code to eliminate stalls? Circle YES or NO (as usual, assume we do have forwarding)
 1. `lw $a0, 0($t1)`
 2. `lw $v0, 0($a0)`
 3. `sub $t1, $s2, $t3`
 4. `sw $v0, 4($s1)`
- If any improvement is possible (to reduce stalls), show the new version here:

(10 pts) Exercise 4-87: True or False? (circle for each)

1. A pipelined implementation will have a faster clock rate than a comparable single cycle implementation (TRUE or FALSE)
2. Pipelining increases performance by splitting up each instruction into stages, thereby decreasing the time needed to execute each instruction (TRUE or FALSE)
3. “Backwards” branches are likely to not be taken (TRUE or FALSE)
4. Dynamic branch prediction is done by the compiler (TRUE or FALSE)
5. Most modern processors use pipelining (TRUE or FALSE)