CornellX: ENGRI1210x The Computing Technology Inside Your Smartphone

KarenWest ▼

Courseware Course Info Discussion Wiki Progress Discussion Guidelines Resources Exploring Engineering Syllabus How to Use Jade

Help

LIMITATIONS DUE TO DATA DEPENDENCES

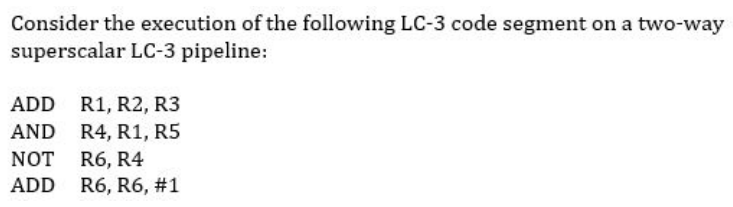
3:11 / 3:11 1.0x

Download transcript .txt

Show Discussion

New Post

1. CHECK YOUR UNDERSTANDING



Consider the execution of the following LC-3 code segment on a two-way superscalar LC-3 pipeline:

```
ADD   R1, R2, R3
AND   R4, R1, R5
NOT   R6, R4
ADD   R6, R6, #1
```

Help

1 A. CHECK YOUR UNDERSTANDING (1/1 point)

Assume that the first ADD and the AND are in the ID stage of the pipeline, and the NOT and the second ADD are in IF. Which of the following accurately describes the actions that the pipeline will take in the next clock cycle?

- The first ADD and the AND will move together into EX, and the NOT and second ADD will move together into ID.
- The two ADDs will move into EX, the AND will remain in ID, and the NOT will move into ID.
- The first ADD and the AND will move together into EX, the NOT will move into ID, and the second ADD will remain in IF.
- [✓ The first ADD will move into EX, the AND will remain in ID, and the NOT and second ADD will remain in IF.] ✓

EXPLANATION

The immediate problem is that the data dependence involving R1 between the first ADD and the AND prevents them from moving together into EX, thereby eliminating options 1 and 3.

Option 2 would require the second ADD to skip ID to move into EX in the next cycle. Also, the program would operate incorrectly due to the data dependence between the NOT and this second ADD involving R6.

Option 4 describes the correct handling of this case. Moving only the first ADD into EX creates the necessary separation between the two instructions to allow bypassing of R1 from MEM to EX. That requires the other three instructions to remain in their pipeline stages.

Final Check Save Hide Answer *You have used 1 of 2 submissions*

Show Discussion New Post

1 B. CHECK YOUR UNDERSTANDING (1 point possible)

Which of the following describes the action that a smart compiler can take to speed up the execution of this code while maintaining correct operation?

- Switch the order of the first ADD and the AND.
- Switch the order of the AND and the NOT.
- [✓ Place the second ADD right after the first ADD.] ✗
- A smart compiler cannot reorder the instructions in this code and maintain correct operation. ✓

Help

EXPLANATION

Because each of the first three instructions produces a result that is used by the very next instruction, these instructions must execute in their original order. Any reordering by the compiler would cause the program to execute incorrectly.

Hide Answer *You have used 2 of 2 submissions*

Show Discussion New Post