Consider the following code sequence using hypothetical LC-3 SIMD instructions:

```
ADD.V   R1, R2, R3
NOT     R4, R5
ADD.V   R4, R4, #1
ADD.V   R5, R1, R4
```

As in the example in the video, the ADD.V instruction simultaneously performs the same operation on four 16-bit operands.

1 A. HOMEWORK (1/1 point)

Why did we not use a NOT.V instruction?

- A NOT cannot be performed on any more than 16 bits at a time.
- A NOT.V would create a data dependence with the previous instruction.
- A NOT.V would create a data dependence with the next instruction.
- (•) The operation performed on each bit position for a NOT is independent of any of the other bit positions. ✔

EXPLANATION

For a NOT instruction, the operation on each bit is independent of the other bits. Therefore, there is no need for a NOT.V instruction that operates on four 16-bit operands, as it would produce the exact same result as a NOT that operates on a single 64-bit operand.

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

Show Discussion | New Post

1 B. HOMEWORK (1/1 point)

Now consider the execution of this code on a 64-bit, two-way superscalar pipeline with an issue queue that supports dual-issue, out-of-order execution. Assume that the values of R2, R3, and R5 are stored in the issue queue with the first two instructions. Moreover, each of the two ALUs is 64 bits wide, and can perform four 16-bit additions in order to support the ADD.V instructions.

Which of the following describes how the issue queue would handle this instruction sequence?

- Each instruction would issue alone on four clock cycles.
- The first two instructions would execute on the first cycle, and the second two on the second cycle.
- The first two instructions would execute on the first cycle, the third instruction on the second cycle, and the fourth instruction on the third cycle. ✔
- The first two ADD.V instructions would execute on the first cycle, followed by the second and fourth instructions on the second cycle.

Help

EXPLANATION

The first two instructions can issue together in the first cycle since they have no data dependence. Because the third and fourth instructions are data dependent, they cannot issue together. The third instruction will issue in cycle two, and the fourth in cycle three.

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

Show Discussion

New Post