



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

Help

PLEASE READ BEFORE DOING TLP HOMEWORK

As is typical for our homework problems, we are asking you to apply what you have learned to conditions you may not have encountered.

After completing the problem please be sure to pay attention to the explanations in "Show Answer", where Professor Albonesi has added some further content to extend your learning.

1. HOMEWORK

Instead of having two separate register files, one for each thread, a two-way multithreaded pipeline may implement a single register file and use register renaming (refer to homework problem 1 in Instruction Level Parallelism) to distinguish the registers in the two threads. For our two thread example, the logical registers might be renamed as follows:

Thread 1

```

LOOP ADD p1, p2, p3
      NOT p4, p1
      AND p11, p4, p3
      ADD p6, p11, 2
      BRz LOOP

```

Thread 2

```

ADD p7, p8, 0
ADD p10, p7, p9
AND p5, p10, 1
STR p5, p8, 0
ADD p10, p8, 0

```

Notice that the logical registers for the two threads are renamed to distinct physical registers even as they share a single physical register file.

1 A. HOMEWORK (1/1 point)

If the first two ADD instructions in each thread are in the EX stage, which of the following describes the action that will be taken by the pipeline when those two instructions move into MEM?

- ☐ The second instruction in each of the two threads (NOT and ADD) will remain in ID for one additional cycle.
- ☐ The second instruction in each of the two threads (NOT and ADD) will move into EX, but the third instruction in each thread (AND and AND), will remain in IF for one additional cycle.
- ☒ The second instruction in each of the two threads (NOT and ADD) will move together into EX, and the values of p1 and p7 will be bypassed from the MEM stage. ✓
- ☐ The second instruction in each of the two threads (NOT and ADD) will move together into EX, but the values of p1 and p7 will not be bypassed from the MEM stage.

EXPLANATION

The action that is taken is exactly the same as we had before. As the two ADD instructions move into MEM, their results (p1 and p7) can be bypassed to the NOT and ADD as they move into EX.

Help

Final Check

Save

Hide Answer

You have used 1 of 2 submissions

Show Discussion

 New Post**1 B. HOMEWORK** (1/1 point)

Register renaming in a multithreaded pipeline also permits the two threads to share a common issue queue, and the two ALUs, in a pipeline with dual-issue out-of-order execution.

The assignment of distinct physical register numbers to the two threads ensures that an instruction in the issue queue will not accidentally set its ready bit as a result of an issuing instruction from a different thread. Moreover, the sharing of both ALUs allows one thread to use both of them if the second thread cannot issue any instructions in a particular cycle. (This could be due to waiting for a value to return from memory, as we will see later in the course.)

Consider a slightly modified version of the above two threads:

Thread 1

```

LOOP ADD  p1, p2, p3
      NOT  p4, p14
      AND  p11, p4, p3
      ADD  p6, p11, 2
      BRz  LOOP

```

Thread 2

```


ADD  p7, p8, 0
ADD  p10, p7, p9
AND  p5, p10, 1
STR  p5, p8, 0
ADD  p10, p8, 0

```

Here, in the first thread, there is no data dependence between the first two instructions (ADD and NOT).

Assume that the instructions for both threads are stored in a common issue queue with dual-issue capability, and that one thread can use both ALUs if necessary. Further assume that the values of p2, p3, and p14 for the first two instructions of thread 1 are stored in the issue queue, but that the value of p8 for the first instruction in thread 2 is not available.

How might the issue queue handle this case?

- ☒ Issue the first two instructions of thread 1. 
- ☐ Issue the first two instructions of thread 2.
- ☐ Issue the first instruction from each of the threads.
- ☐ Issue no instructions.

EXPLANATION

Because the value of p8 is unavailable, no instructions from thread 2 are ready to issue. The first four instructions have data dependences that prevent them from issuing. The last instruction requires the value of p8, which is unavailable.

Because p2, p3, and p14 are available, and the first two instructions of thread 1 have no data dependences, they can issue together.

Help

Final Check

Save

Hide Answer

You have used 1 of 2 submissions

Show Discussion

 New Post

EdX offers interactive online classes and MOOCs from the world's best universities. Online courses from MITx, HarvardX, BerkeleyX, UTx and many other universities. Topics include biology, business, chemistry, computer science, economics, finance, electronics, engineering, food and nutrition, history, humanities, law, literature, math, medicine, music, philosophy, physics, science, statistics and more. EdX is a non-profit online initiative created by founding partners Harvard and MIT.

© 2015 edX Inc.

EdX, Open edX, and the edX and Open edX logos are registered trademarks or trademarks of edX Inc.

[Terms of Service and Honor Code](#)





[Privacy Policy \(Revised 10/22/2014\)](#)

POWERED BY
OPENedX

About edX

[About](#)[News](#)[Contact](#)[FAQ](#)[edX Blog](#)[Donate to edX](#)[Jobs at edX](#)

Follow Us

 [Facebook](#) [Twitter](#) [LinkedIn](#) [Google+](#) [Tumblr](#) [Meetup](#) [Reddit](#) [Youtube](#)