University of California, Berkeley
College of Engineering
Computer Science Division — EECS

Fall 1999 John Kubiatowicz

Homework Quiz (HW #5)
SOLUTIONS

October 27, 1999
CS152 Computer Architecture and Engineering

This quiz covers one of the problems from homework #5.
Good Luck!

Your Name:	
SID Number:	
Discussion Section:	

Total:	

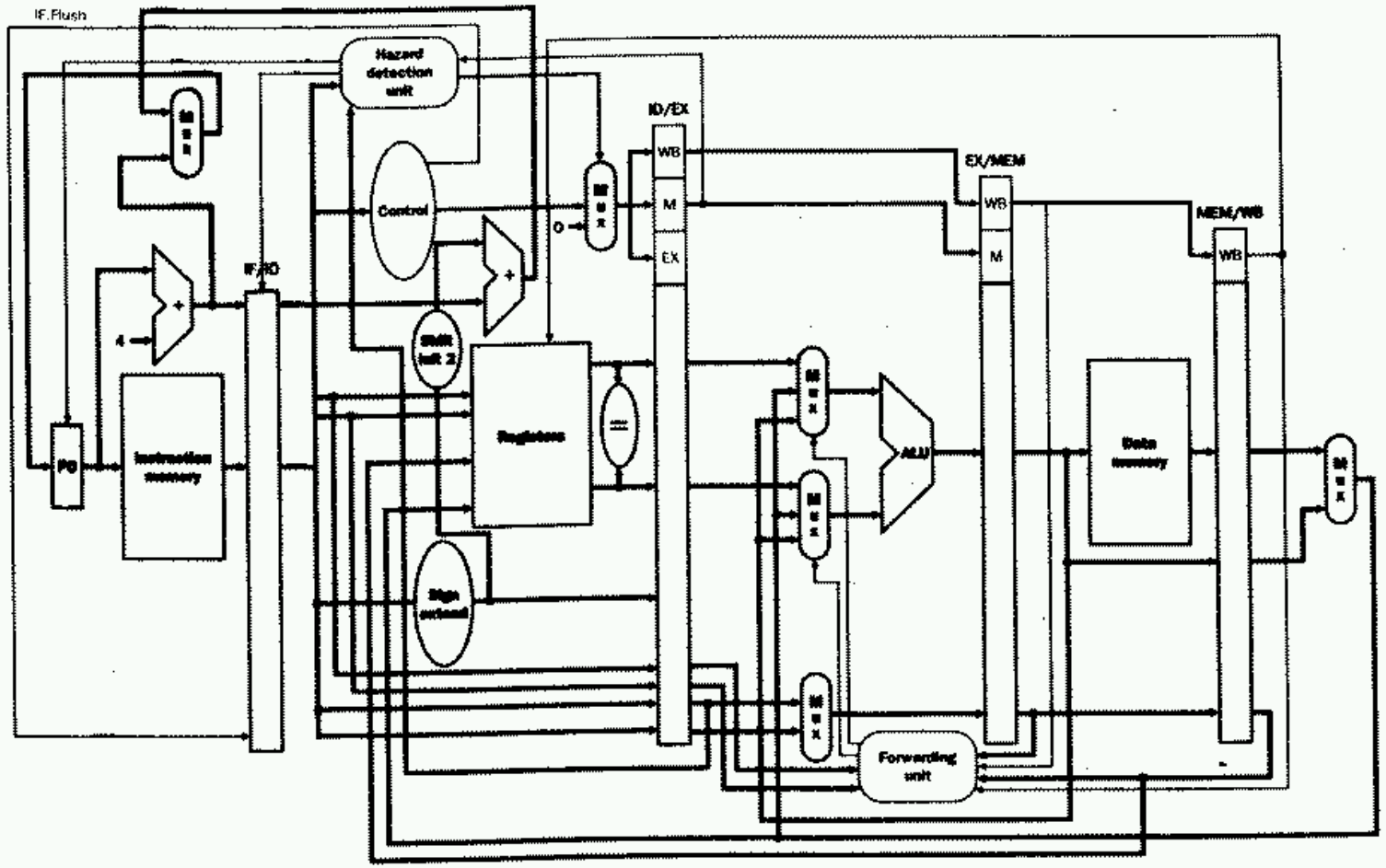

FIGURE 6.51 Datapath for branch, including hardware to flush the instruction that follows the branch. This optimization moves the branch decision from the fourth pipeline stage to the second; only one instruction that follows the branch will be in the pipe at that time. The control lines IF.Flush turns the fetched instruction into a nop by zeroing the IF/ID pipeline register. Although the flush line is shown coming from the control unit in this figure, in reality it comes from hardware that determines if a branch is taken, labeled with an equal sign to the right of the registers in the ID stage.

In problem 6.27, you were asked which stage branch decisions must be made to reduce the branch delay to a single cycle.

1. In which stage must branch decisions be made in order to support only one branch delay slot? Justify your answer.

   Branch decisions must occur in the ID stage. This is most easily seen by assuming a forward dependency between the ID stage and a fetch two instructions later:

2. Figure 6.51 from one of the printings of the book is reproduced on the previous page. Consider the following instruction sequence:

   ```
   sub $2, $4, $5
   beq $2, $3, somewhere
   ```

   Why doesn't this code sequence work properly on this hardware (this is a bug in the book!)?

   This code sequence doesn't work because there is no forwarding into the comparison operator (marked as an oval with "=") in it.

3. Enhance the data path so that the code sequence of (2) works. Don't worry about control:

   There are two solutions to this.

   Simplest: put two forwarding muxes in the ID stage which feed into the comparison block. The inputs to each side of the muxes includes (1) the normal register input, (2) a value from the END of the execute stage (i.e. after the ALU), and (3) an input from the END of the memory stage.

   Better: Change the timing of the forwarding logic. Move the forwarding muxes from after the Id/EX register to before the ID/EX register. Similarly, all of the inputs to these muxes must be moved from after registers to before them (so, for instance, you will have a mux feedback path coming directly from the output of the ALU). Now, place the equality comparison hardware directly after the muxes.

4. Suppose that the execute stage of the original hardware (figure 6.51) was the critical path (longest stage). What is the critical path after the modification for question #3 (be careful!)?

   Given the "simplest" solution, the critical path is now from the beginning of the EX stage, through the computation forwarding muxes, through the ALU, back through the forwarding for the equal comparison (in the ID stage), through the equal comparison, through the PC mux. For the "better" solution, the delay through the ALU and into the forwarding muxes will be the same as the original critical path (through forwarding muxes and into the ALU). However, the critical path will now include the equal comparison and the PC mux.