Homework Quiz (HW #5)
April 5, 1999
CS152 Computer Architecture and Engineering

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

<table>
<thead>
<tr>
<th>Your Name:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SID Number:</td>
<td></td>
</tr>
<tr>
<td>Discussion Section:</td>
<td></td>
</tr>
</tbody>
</table>

Total:               |          |
In problem 6.26, you were asked to consider a pipeline which does not support a delayed branch. Figure 6.51 from one of the printings of the book is reproduced on the previous page.

1. In this datapath, how many instructions must be “flushed” out of the pipeline when a branch is taken? Explain.

   One. The comparison for the branch is performed in the Decode stage, so there is only one additional instruction in the pipeline at that time.

2. What must be in the “Control” oval in order to support flushing (i.e. when does it decide to assert IF.Flush)? What exactly does the “IF.Flush” signal do?

   The “Control” oval must flush if
   A. There is a branch in the decode register, IF/ID.
   B. The branch condition is true.

   IF.Flush must cause the value being latched into the IF/ID register to be turned into a NOP. Note that, as you discovered in problem 6.26, this must only occur if you are not stalling the pipeline (i.e. if the IF/ID register is enabled for latching).

3. Consider the following instruction sequence:

   \[
   \text{sub } \$2, \$4, \$5 \\
   \text{beq } \$2, \$3, \text{somewhere}
   \]

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

   This datapath doesn’t forward to the branch equality hardware. Thus, the BEQ will look only at an old value of $2$, instead of the value from the sub instruction.

4. Can you fix this problem without increasing the number of instructions flushed on a taken branch? If so, how? If not, why not? You can use space on the back of this sheet if necessary.

   Yes. Simply change the timing of the forwarding. Move the 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 (but before the ID/EX register).