CS 2734, Computer Organization II
Fall Semester, 2000
Second Examination

1. For this problem, you are to use a xerox of Figure 5.29. (Final datapath for the *single*-cycle implementation of MIPS.) You will be tracing through the path of the **`lw`** instruction on this single-cycle model. You should use a highlighter to trace the path the instruction and associated data takes through the diagram. (Do *not* show data traveling to "dead-end" components, which will eventually have no effect. In particular, do not show the computation of the branch address, which will not be used in this case.) Then write in the values for the *relevant* control signals. (Do *not* give control signals that serve to keep "dead-end" paths from having an effect.) **(30)**

   Use the following specific instruction:

   ```
   lw $t5, 92($t1)
   ```

   or in machine language form:

   ```
   0x8d2d005c                          (in hexadecimal)
   100011 01001 01101 00000 00001 011100 (fields in binary)
       35     9    13                92 (fields in decimal)
   ```

   Start at the left side, showing the PC coming in, and assume this instruction is read from the Instruction memory. Don't forget the handling of the PC by this instruction. Show what values are traveling along the different lines, assuming the following initial values:

   (a) **`$t1`** and **`$t5`** are registers numbers **`9`** and **`13`** (decimal), respectively.

   (b) The contents of register **`9`** is **`200`** (decimal).

   (c) The PC has value **`16`**.

2. For this problem, you are to use a xerox of Figure 5.33. (Final datapath for the *multi*-cycle implementation of MIPS.) You will be tracing through the path of the **`beq`** instruction on this multi-cycle model. You should use several colors of highlighters to trace the paths the instruction and associated data takes through the diagram. (Or you can trace these paths using more than one diagram.) Do *not* show data traveling to "dead-end" components, which will eventually have no effect. **(30)**

   For this diagram, you do *not* need to give the values of control signals.

   Below the diagram, or in some other way, carefully identify *which cycle* (or step) of handling the instruction belongs to each part of the highlighted datapath (just for data, not control). Thus you should identify *Cycle 1*, *Cycle 2*, *Cycle 3*, and perhaps *Cycle 4* and *Cycle 5* (if the instruction uses Cycles 4 and 5).

   Use the following specific instruction:

   ```
   beq   $t2, $t5, LabelA
   ```

   or in machine language form:

   ```
   0x114d0004                          (in hexadecimal)
   000100 01010 01101 00000 00000 000100 (fields in binary)
        4    10    13                 4 (fields in decimal)
   ```

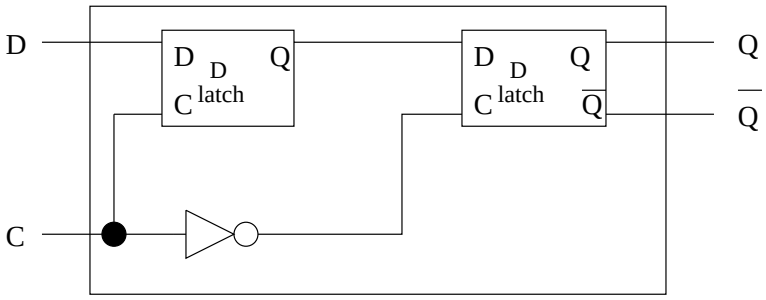
Start at the left side, showing the PC value coming in, and assume this instruction is read from the Instruction Memory. Show what values are traveling along the different lines, assuming the following initial values:

(a) **`$t2`** and **`$t5`** are register numbers **`10`** and **`13`** (decimal), respectively.

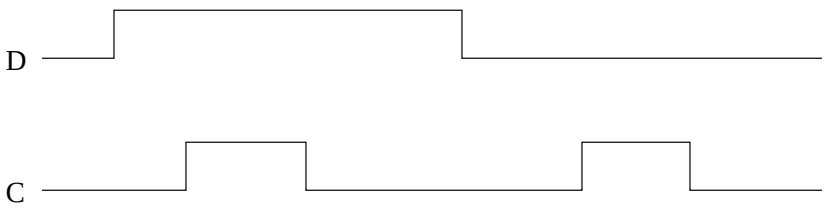
(b) Assume that the contents of each of these registers is **`5234`**, so you should assume that the branch is taken.

(c) Assume the PC has value **`20`** (decimal) initially. On the proper line, give the *final* PC value, assuming the branch is taken. Don't forget to highlight the parts related to the PC as well as the rest of the instruction. *Be sure to identify the different cycles.*

3. Consider the following diagram of a D flip-flop: **(15)**



D flip-flop



Q ——— . . .

This D flip-flop is constructed from two D latches.

(a) Under what circumstances does the signal D go through the first D latch? (Just give the answer, without referring the the inside of the D latch.)

(b) Give the output Q of the D flip-flop as it would appear in the figure above. (That is, fill in the output signal Q.)

4. Using the MIPS multi-cycle implementation as a model, give a description of what happens during a MIPS *exception* because of an *illegal instruction.* In more detail: **(25)**

(a) What part of the MIPS hardware (multi-cycle implementation) detects that an illegal instruction has occurred?

(b) What happens to the **Cause register**?

(c) What happens to the **EPC register**?

(d) How does the hardware manage to get the desired value into the EPC register?