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 `beq` 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.) Indicate the values of relevant control signals, without highlighting the control line. (Do *not* give control signals that serve to keep “dead-end” paths from having an effect.)

Use the following specific instruction:

```
beq  $t2, $t5, Loop
```

or in machine language form:

```
0x114d0008 (in hexadecimal)
000100 01010 01101 00000 00000 00100 (fields in binary)
 4   10   13      8 (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. (The branch will be taken because the two register values are equal.)

(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.
2. For this problem, you are to use one or more xeroxes of Figure 5.33. (Final datapath for the multi-cycle implementation of MIPS.) You will be tracing through the path of the sw instruction on this multi-cycle model. You should use a highlighter in one color 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, in cycle 2 do not show the computation of the branch address, which will not be used in this case.)

For this diagram, do not 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:

\[
\text{sw } \$t2, 40(\$t1)
\]

or in machine language form:

- Binary: 0xad2a0028
- Hexadecimal: 0xad2a0028
- Decimal: 43 9 10 40

Start at the left side, showing the PC coming in, and assume this instruction is read from the Instruction memory. Be sure to identify the different cycles. Don’t forget the PC. Show what values are traveling along the different lines, assuming the following initial values:

(a) \$t1 and \$t2 are registers numbers 9 and 10 (decimal), respectively.
(b) The contents of register 9 is 104 (decimal), and of register 10 is 57 (decimal).
(c) The PC has value 32.
3. This question is concerned with exceptions and interrupts, as described in the text for the multi-cycle implementation.

(a) Explain very briefly how Vectored Interrupts work. Does MIPS use vectored interrupts? (Just “Yes” or “No.”)
(b) In case of an MIPS because of an undefined instructions, what value ends up in the EPC register?

4. Consider the following diagram of a D flip-flop:

This D flip-flop is constructed from two D latches. Recall that a D latch lets the signal D go through if the clock C is asserted (the D latch is open), and the output does not change if the clock C is deasserted (the D latch is closed). Such a D latch is said to be transparent.

(a) 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.)
(b) Explain very carefully why the output is what it is. (Your explanation should refer to the specifics of the diagram above, including the NOT gate and the two D latches.)