You may work in pairs for this assignment. If you choose to work with a partner, make sure only one of you makes a submission a solution and that the file lists names and PIDs for both of you as described in the assignment below.

Prepare your answers to the following questions in a plain text file. Submit your file to the Curator System by the posted deadline for this assignment. No late submissions will be accepted.

You will submit your answers to the Curator System (www.cs.vt.edu/curator) under the heading MIPS04.

For questions 1 through 3, refer to the pipeline design with forwarding, shown below, which supports execution of the following MIPS instructions: `add`, `sub`, `and`, `or`, `slt`, `sw`, and `lw` so long as no stalls are needed to resolve load-use hazards.

Remember: this pipeline design does not include (load-use) hazard detection hardware, so it can forward operands but it cannot introduce stalls to deal with situations that forwarding alone will not handle.
1. Assume we have the 5-stage pipelined datapath described above, with forwarding hardware but without the (load-use) hazard-detection hardware. Consider executing the following sequence of instructions on this datapath:

   add $t5, $t3, $t2 # 1.1
   sw $t5, 0($t2)     # 1.2
   lw $t3, 8($t5)     # 1.3
   add $t4, $t3, $t5 # 1.4
   lw $t2, 4($t4)     # 1.5
   add $t1, $t4, $t3 # 1.6
   add $t3, $t2, $t1 # 1.7

   a) [18 points] Identify all of the data hazards that can be resolved with forwarding alone; that is, data hazards that do not require inserting a stall (nop).

      For each such data hazard, state the number of the writing instruction, the number of the reading instruction, the register involved, and indicate whether the forwarding will be from the EX/MEM interstage buffer or from the MEM/WB interstage buffer.

   b) [12 points] Even with forwarding, the given sequence of instructions cannot execute correctly, in the order given without inserting at least one nop instruction. Show how to achieve correct execution, on this datapath, by inserting the minimal number of nop instructions, but not reordering the instructions.

      Would any forwarding still be required? If yes, state the number of the writing instruction, the number of the reading instruction, the register involved, and indicate whether the forwarding will be from the EX/MEM interstage buffer or from the MEM/WB interstage buffer.

2. Assume we have the 5-stage pipelined datapath described above, with forwarding hardware but without the (load-use) hazard-detection hardware. Consider executing the following sequence of instructions on this datapath:

   lw  $t1, 8($t0)   # 2.1
   lw  $t2, 0($t1)   # 2.2
   add $t3, $t1, $t2 # 2.3
   add $t4, $t0, $t1 # 2.4
   add $t2, $t3, $t7 # 2.5
   sw  $t4, 0($t1)   # 2.6
   add $t5, $t7, $t0 # 2.7

   a) [12 points] Identify all of the data hazards that cannot be resolved with forwarding alone; that is, data hazards that do require inserting stalls.

      For each such data hazard, state the number of the writing instruction, the number of the reading instruction, and the register involved.

   b) [8 points] Can the given sequence of instructions be reordered so that they will execute correctly, using forwarding but without inserting any nop instructions to deal with that hazard? Justify your answer.

3. [20 points] Suppose that it was decided to add a new stage, BRNCH, to the pipeline between the current EX and MEM stages. The BRNCH stage would contain the hardware for calculating the branch target address and the AND gate that determines whether the branch is taken. All the other existing hardware would remain in the same stages as in the current design. There would now be an interstage buffer called EX/BRNCH between EX and BRNCH, and an interstage buffer called BRNCH/MEM between BRNCH and MEM. Each pipeline stage would still require one clock cycle (which might be shorter, but that's not relevant to this question). See the diagram after question 5.

   What effect, if any, would this change have on the design of the Forwarding logic for the pipeline?
For questions 4 and 5, refer to the pipeline design with forwarding and (load-use) hazard detection, shown below, which supports execution of the following MIPS instructions: add, sub, and, or, slt, lw, and sw.

4. [20 points] Assume we have the 5-stage pipelined datapath described above, with both forwarding hardware and (load-use) hazard-detection hardware. Consider executing the following sequence of instructions (same as question 2) on this datapath:

   lw    $t1, 8($t0)      # 4.1
   lw    $t2, 0($t1)      # 4.2
   add   $t3, $t1, $t2    # 4.3
   add   $t4, $t0, $t1    # 4.4
   add   $t2, $t3, $t7    # 4.5
   sw    $t4, 0($t1)      # 4.6
   add   $t5, $t7, $t0    # 4.7

Where would the Hazard Detection unit insert stalls (bubbles)?
5. [10 points] Recall the scenario from question 3:

Suppose that it was decided to add a new stage, BRNCH, to the pipeline between the current EX and MEM stages. The BRNCH stage would contain the hardware for calculating the branch target address and the AND gate that determines whether the branch is taken. All the other existing hardware would remain in the same stages as in the current design. There would now be an interstage buffer called EX/BRNCH between EX and BRNCH, and an interstage buffer called BRNCH/MEM between BRNCH and MEM. Each pipeline stage would still require one clock cycle (which might be shorter, but that's not relevant to this question).

Assume that any changes to the Forwarding logic that would be needed (from your answer to question 3) have been made.

What effect would the change above have on the design of the (load-use) Hazard Detection logic for the pipeline?

Here's a high-level diagram of the organization of the modified 6-stage pipeline: