LSU EE 4720 **Homework 1** Solution **Due: 20 February 2008**

Problem 1: Solve Fall 2007 Homework 2 without looking at the solution. Then look at the solution and give yourself a grade on a scale of $[0, 1]$. **Warning:** test questions are based on the assumption that homework problems were completed, so make a full effort to solve it without first consulting the solution.

Problem 2: The MIPS IV `movn` instruction is an example of a *predicated* instruction (predication will be covered later in the semester, but that material is not needed to solve this problem).

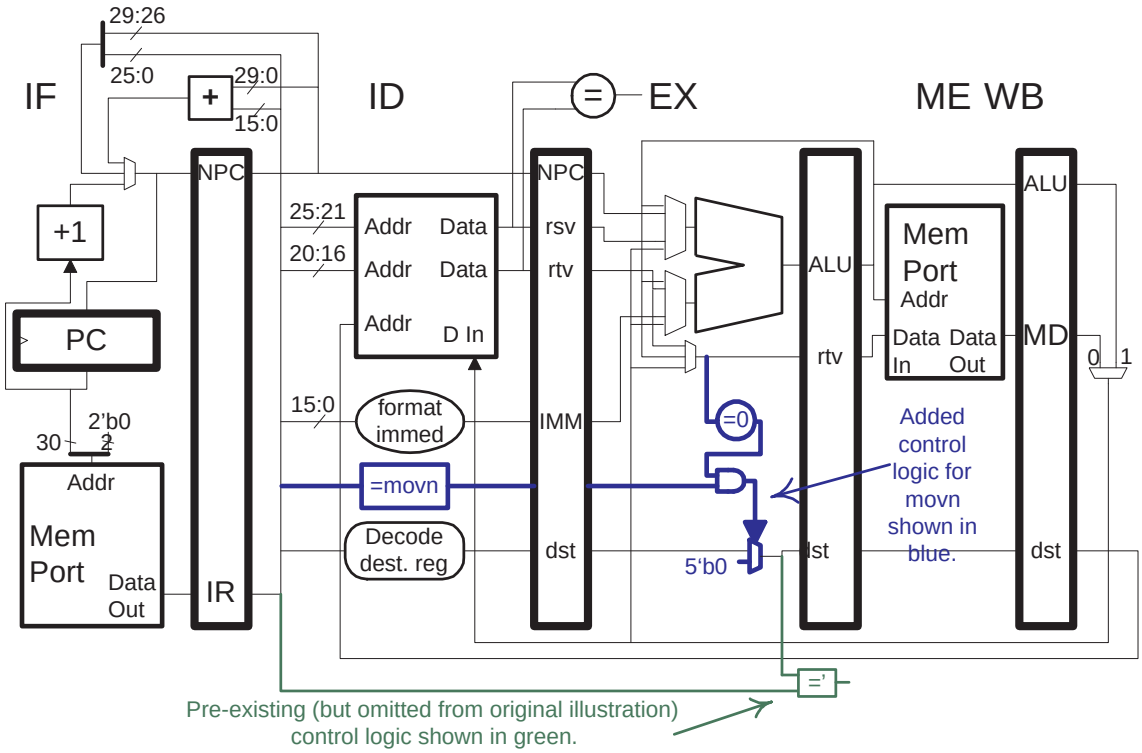
(*a*) Show how the `movn` instruction could be added to the implementation below inexpensively, but without impact on critical path. Take into account the new logic's impact on dependency testing (see the code sample below). Show all added control logic.

Changes for the `movn` instruction are shown in blue bold below. The logic shown in green was put in for the solution but would have appeared on the unsolved diagram if control logic were shown. That is, the green logic would be there with or without the `movn`.

The `movn` implementation below works as follows. The output of =movn box in `ID` is 1 if a `movn` instruction is present. In `EX` the ALU passes the `rs` value unchanged while the added =0 unit tests whether the `rt` value is zero. The and gate checks whether a `movn` instruction is in `EX` and whether the move should be cancelled, if so the mux substitutes a 0 for the destination register (suppressing the writeback), otherwise the `dst` register number is passed through unchanged. Note that the control logic for detecting bypasses examines the output of the mux.

This implementation will execute the code below without a stall.

In a lower-cost implementation (not illustrated) a comparison unit in the `ID` stage, already needed for branches, would be used. The code below would stall on such an implementation unless bypasses were added from `EX`.



(*b*) Show how the code below would execute on your implementation.

```
# Solution
# Cycle           0  1  2  3  4  5  6
 add  r1, r2, r3  IF ID EX ME WB
 movn r4, r5, r1     IF ID EX ME WB
 xor  r6, r4, r7        IF ID EX ME WB
```

(*c*) Suggest methods to eliminate any stalls encountered.

There are no stalls.