LSU EE 4720 **Homework 3** **Due: 3 November 2004**

Problem 1: Do Problems 1 and 2 From Spring 2004 Homework 3 `http://www.ece.lsu.edu/ee4720/2004/hw03.pdf`. After completing the problems look at the solution and assign yourself a grade. The maximum grade should be 10 points, divide the points between problems as you wish.

Problem 2: A new instruction, `copyTreg rt, rs`, will read the contents of register `rt` and `rs` and will write the contents of `rs` to the register number specified *by the contents* of register `rt` (not into register `rt`). For example,

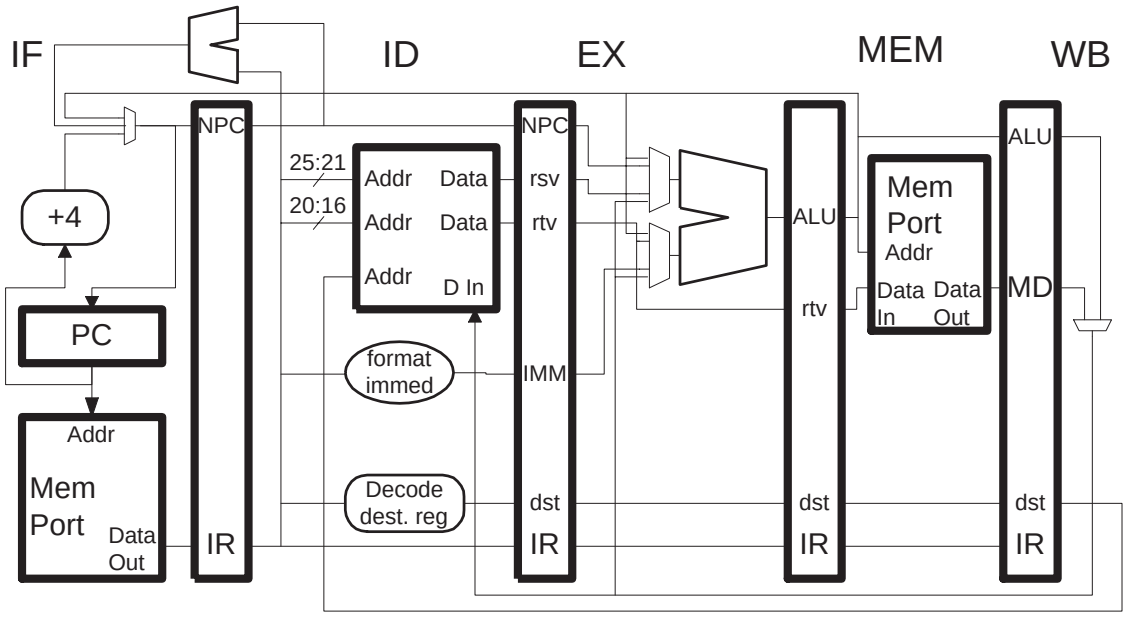
```
# Before:  $1 = 4,  $2 = 0x1234,  $4 = 0
copyTreg $1, $2
# After:   $1 = 4,  $2 = 0x1234,  $4 = 0x1234;
#          (Register $4 written with contents of register 2.)
```

Note that this is a variation on Midterm Exam 1 Problem 3, with the destination, rather than the source, being specified in a register.

(*a*) Modify the pipeline below to implement this instruction.

(*b*) Add the bypass connections needed so that the code below executes correctly.

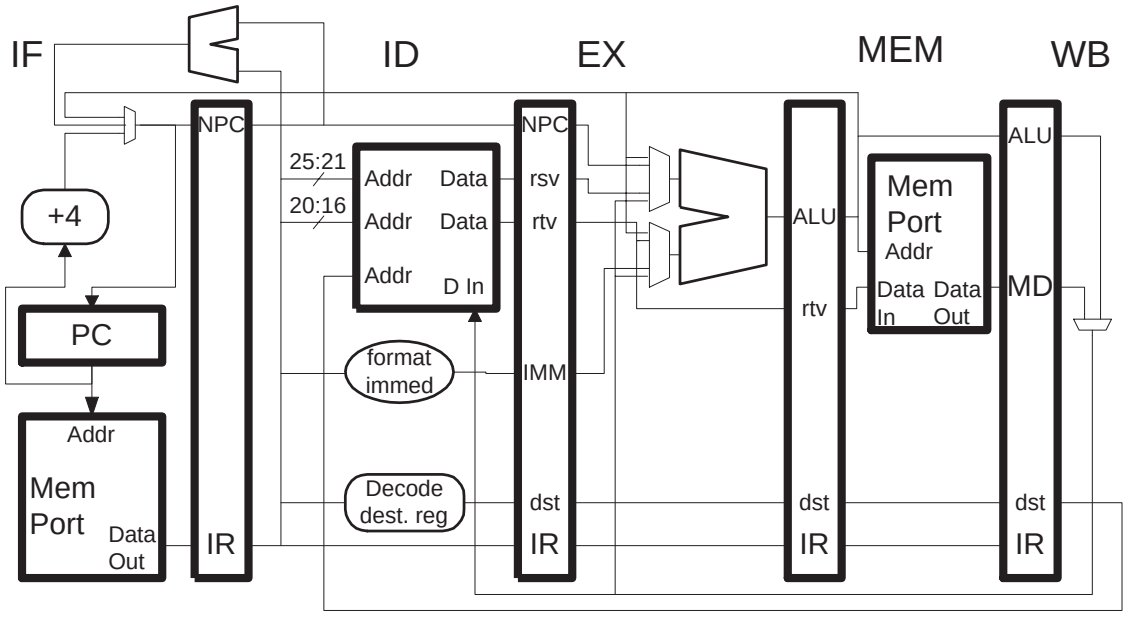
```
# Before:  $1 = 4,  $2 = 0x1234,  $4 = 0,  $5 = 0
addi $1, $0, 5
copyTreg $1, $2
# After:   $1 = 5,  $2 = 0x1234,  $4 = 0,  $5 = 0x1234;
```



Problem 3: In the problem above the register number to *write to* is in a register. Here consider `copyFreg rt, rs` in which, like the test question, the register to *copy from* is in a register. That is,

```
# Before:  $1 = 2,  $2 = 0x1234,  $4 = 0
copyFreg $4, $1
# After:   $1 = 2,  $2 = 0x1234,  $4 = 0x1234;
#          (Register $4 written with contents of register 2.)
```

Explain a difficulty in implementing this instruction on the pipeline below without vitiating its sublime elegance.



Problem 4: No, we are not vitiators. Instead consider `copyBreg rd` in which the source register to read is specified *in the* `rs` register of the preceding instruction, that value is written into the `rd` register of this instruction. (Okay, maybe we are vitiators.) For example,

```
# Before:  $1 = 2,  $2 = 0x1234,  $4 = 0
add      $0, $1, $0   # Instruction below uses rs ($1 here) of this insn.
copyBreg $4
# After:   $1 = 2,  $2 = 0x1234,  $4 = 0x1234;
#          (Register $4 written with contents of register 2.)
```

Implement this instruction on the pipeline above (from the previous problem).