Name _____

Computer Architecture
EE 4720
Midterm Examination

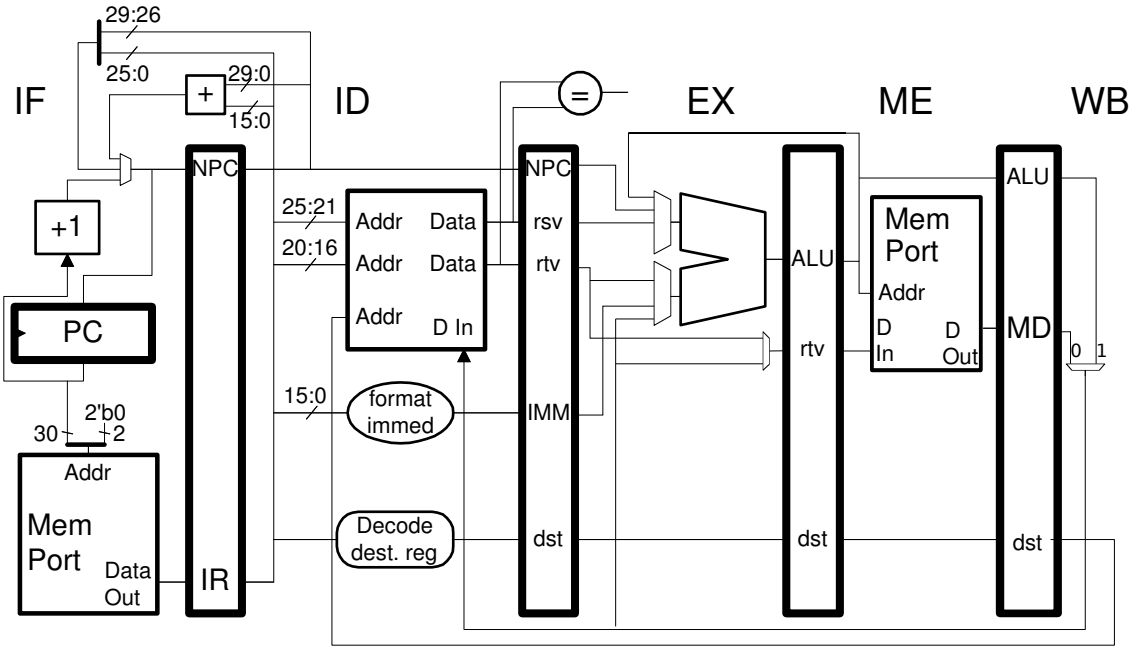
Friday, 23 March 2012, 9:40–10:30 CDT

Problem 1	_____	(20 pts)
Problem 2	_____	(15 pts)
Problem 3	_____	(15 pts)
Problem 4	_____	(20 pts)
Problem 5	_____	(15 pts)
Problem 6	_____	(15 pts)
Exam Total	_____	(100 pts)

Alias _____

Good Luck!

Problem 1: [20 pts] The MIPS code below executes on the illustrated **seemingly** familiar implementation. If you look closely you'll notice that certain bypass paths that are present in the five-stage implementation usually used in class are missing from the diagram below.



(*a*) Show the execution of the code below on the illustrated implementation for enough iterations to determine CPI. Determine the CPI.

☐ Execution diagram.

☐ Double-check for bypass paths, stall when necessary.

☐ Compute CPI.

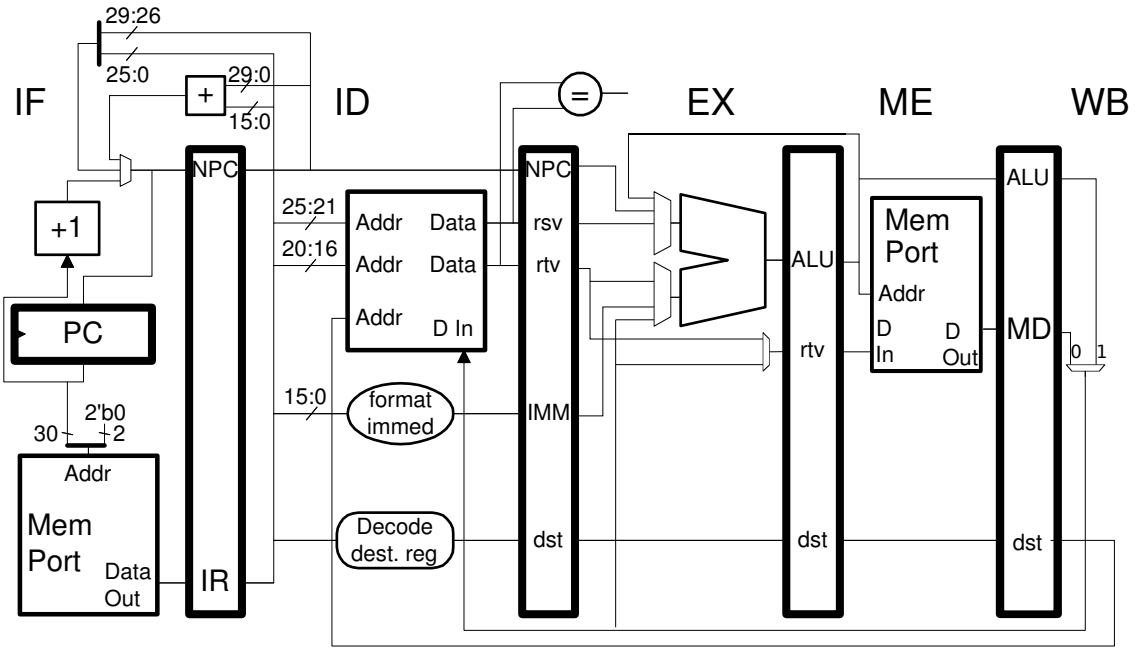
```
LOOP:
 lw  r2, 0(r4)

 add r1, r2, r3

 sw r1, 4(r4)

 bne r4, r5  LOOP

 addi r4, r4, 8
```

Problem 1, continued:

(*b*) The code above should have encountered at least one of the missing bypass paths. Choose one of those missing bypass paths and design control logic for it. The `ID`-stage control logic should generate a 1-bit signal `STALL` that will be logic `1` when the instruction in `ID` will have to stall because of the missing bypass path.

☐ Show which missing bypass path the control logic is for.

☐ Control logic for missing bypass.



Problem 2: [15 pts] Consider a MIPS implementation in which the memory port is split into two stages, `Ma` and `Mb`. With this change the clock frequency can increase from 1 GHz to 1.2 GHz; with this added stage our implementation is now six stages. A sample execution appears below.

```
add r2, r3, r4 IF ID EX Ma Mb WB
lw r1, 4(r2)      IF ID EX Ma Mb WB
```

(*a*) Ignoring the cost of `Ma` and `Mb`, how is the cost of the pipeline changed in this design? Consider bypass paths.

☐ Cost change, ignore `Ma` and `Mb`, consider bypass, etc.

(*b*) Provide an example of a code fragment which will execute more slowly with the six-stage pipeline. Assume that all reasonable bypass paths are provided.

☐ Code fragment that will run more slowly.

(*c*) Provide an argument that this change is good.

☐ This change is good because:

Problem 3: [15 pts] Answer the following questions about a `blt` instruction.

(*a*) MIPS lacks an instruction such as `blt r1, r2 TARG` (branch if `r1` less than `r2`). For the questions below, which ask about the suitability of such an instruction, consider implementations similar to the five-stage pipeline covered in class.

☐ Explain why adding such an instruction would slow such implementations even though `beq r1,r2 TARG` is okay.

☐ Using a PED, explain why there would be no problem adding a `blt` instruction if the ISA had two delay slots.

(*b*) The code fragment below includes the hypothetical MIPS instruction, `blt`.

```
# Hypothetical MIPS Instruction
blt r1, r2, targ
xor r4, r5, r6
```

☐ Show an equivalent MIPS code fragment without using `blt`.

☐ Show an equivalent SPARC V8 code fragment.

Problem 4: [20 pts] For the ISA design tradeoff questions below consider the following data: A typical program executes 10^{10} dynamic instructions, of these 1.38×10^9 are branches (half of which are taken), 9.12×10^7 are indirect jumps (`jr` and `jalr`), 2.22×10^8 are direct jumps (`j` and `jal`).

Consider a debate by those developing the MIPS ISA: don't include format J, which means no `j` and no `jal` instructions.

(*a*) Which MIPS instructions can be used to replace `j` and `jal` in the code below, paying attention to the hints in the target names?

☐ Replacement for `j` and `jal` in code below, paying attention to target names.

```
j NOT_TOO_FAR_AWAY
xor r1, r2, r3


jal A_FAR_AWAY_PROCEDURE
xor r1, r2, r3
```

(*b*) Determine how much longer (as a percentage or absolute time) it would take to run the typical program described above without the format J instructions. Assume execution is always at 1 CPI. (A formula is okay.)

☐ Percent increase in execution time without format J:

(*c*) What parts of out implementation would be unnecessary without format J? Approximately how many bypass paths could you add with the cost saved by not having the format J instructions?

☐ What parts would be eliminated?

☐ How many bypass paths could be added with cost savings?

Problem 5: [15 pts] Answer the following ISA design questions.

(*a*) Suppose that *one of the first* modern computer designers, working in the 1940s, made the following statement: "Let's define an 'Instruction Set Architecture' [the last three words spoken slowly, for emphasis] and then later design some hardware 'im-ple-men-ta-tions' of that architecture." Would that be a good idea? Explain. *Hint: Pay attention to the slanted words.*

☐ ISA before implementation in 1940s, good or bad? Explain.

(*b*) CISC ISAs have variable-length instructions.

☐ Benefit of variable-length instructions for CISC.

☐ Benefit of fixed-length instructions for RISC.

Problem 6: [15 pts] Answer each question below.

(*a*) In our π program demo we found that turning optimization on reduced execution time by half (yay!) but reduced the instruction count even further, from 325 million to 100 million dynamic instructions. This means that optimization *increased* (made worse) CPI from 2.77 to 4.93. *Note: The data is from an earlier semester, so the numbers won't exactly match.*

☐ Optimization is supposed to eliminate stalls, why did the CPI go up?

☐ What category of instruction was completely eliminated from the loop body of the π program by optimization.

(*b*) SPECcpu currently has two sets of results, *base* and *peak*. Consider a third set, *shrink-wrap*, for people who buy mass-marketed software (say, buying a word processor at an office supply store).

☐ Who should use the peak numbers and who should use the base numbers?

☐ Explain how the run and reporting rules might be modified for the new shrink-wrap results.