Name _____

Computer Architecture
EE 4720
Midterm Examination

Wednesday, 28 March 2007, 11:40–12:30 CDT

Problem 1	_____	(40 pts)
Problem 2	_____	(30 pts)
Problem 3	_____	(30 pts)
Exam Total	_____	(100 pts)

Alias _____

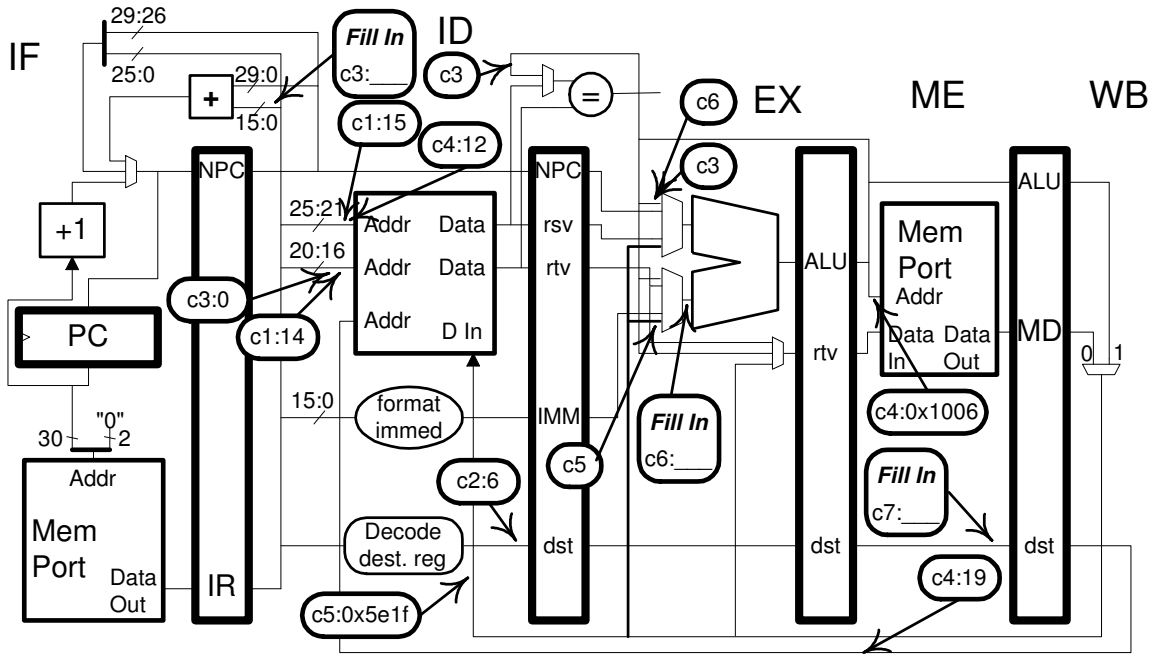
Good Luck!

Problem 1: In the MIPS implementation below some wires are labeled with cycle numbers and values that will then be present. For example, c1:15 indicates that at cycle 1 the wire will hold a 15. Other wires are labeled just with cycle numbers, indicating that the wire is used at that cycle. If a value on any labeled wire is changed the code would execute incorrectly. Note that the last instruction and the address of two instructions have been provided. [40 pts]

☐ Finish a program consistent with these labels.

☐ All register numbers and immediate values can be determined.

☐ Be sure to fill the three blocks marked *Fill In*.



	Cycle:	0	1	2	3	4	5	6	7	8
0x1000		IF	ID	EX	ME	WB				
			IF	ID	EX	ME	WB			
				IF	ID	EX	ME	WB		
					IF	ID	EX	ME	WB	
0x1030 sb r17, 3(r4)						IF	ID	EX	ME	WB
	Cycle:	0	1	2	3	4	5	6	7	8

Problem 2: Answer each question below.

(*a*) Show the encoding of the MIPS `xor` instruction below. In particular, show the name of each field in the encoded instruction, the fields' bit positions, and the fields' values. If you don't know the value of a field (there's one or two you're not expected to know) make up a value and label it "made up." *Hint: It is not cheating to look at the diagram for Problem 1.*

```
xor r5, r7, r10
```

☐ [10 pts] Encoding showing field names, bit positions, and values.

(*b*) Unlike some other benchmark suites, in SPECcpu the tester compiles the benchmarks (rather than having SPEC provide the benchmarks already compiled). [10 pts]

☐ How does this difference make SPECcpu valuable to those in the area of computer architecture?

☐ Why might this difference make SPECcpu less valuable to those looking for the fastest computer to run their favorite game?

(*c*) A company is considering adding a BCD data type to their new ISA. An analysis of a suite of Cobol programs, widely used by their customers, shows that the ISA's BCD data type would be extensively used. [10 pts]

☐ What more does the company need to know to make the decision?

Problem 3: Answer each question below.

(*a*) The MIPS code below loads a character from memory and places it in bit positions 15:8 of register `r3`. [10 pts]

```
lbu r1, 0(r2)   # Note: r2 can be any address.
sll r1, r1, 8
and r3, r3, r4  # r4 = 0xffff00ff
or  r3, r3, r1
```

☐ Explain how this code is similar to, and differs from, the operation performed by `lwl` and `lwr` (from Homework 1).

☐ **Given that MIPS already has `lwl` and `lwr`** is it worthwhile adding an instruction that performs the same operation as the code above? Explain, stating any necessary assumptions or made-up data.

(*b*) The first code fragment below, standard MIPS, uses a `slt` to perform a comparison for a branch. The second uses a proposed MIPS branch instruction that does the comparison itself and as a result the target is fetched one cycle sooner. The first execution is on *sImp*, an implementation of standard MIPS, the second execution is on *bImp*, an implementation of the proposed MIPS; bImp includes `blt` but is otherwise identical to sImp. [10 pts]

```
 # Standard MIPS  Cyc: 0  1  2  3  4  5  6  7
 slt r1, r2, r3        IF ID EX ME WB
 bne r1, r0,  TARG        IF ID EX ME WB
 nop                         IF ID EX ME WB
TARG: xor r4, r5, r6            IF ID EX ME WB

 # Proposed MIPS  Cyc: 0  1  2  3  4  5  6  7
 blt r2, r3, TARG      IF ID EX ME WB
 nop                      IF ID EX ME WB
TARG: xor r4, r5, r6         IF ID EX ME WB
```

☐ Why might the clock frequency of bImp be lower than sImp?

☐ How does knowing the percentage of branches in a program help determine if bImp can run the program faster than sImp (when compiled for the respective implementation)?

(*c*) All a compiler needs to know about the target (the implementation being compiled for) is its ISA. However, if the compiler also knows the implementation it can produce faster code. [10 pts]

☐ What's wrong with the following statement: *If the compiler also knows the target implementation it can make better register assignment decisions since it knows the exact number of registers available.*

☐ How can the compiler produce better code knowing operation latencies, such as the time needed for a `mul` instruction?