Name _____

Computer Architecture
EE 4720
Midterm Examination

Friday, 20 March 2015, 9:30–10:20 CDT

Problem 1	_____	(25 pts)
Problem 2	_____	(25 pts)
Problem 3	_____	(15 pts)
Problem 4	_____	(10 pts)
Problem 5	_____	(10 pts)
Problem 6	_____	(15 pts)
Exam Total	_____	(100 pts)

Alias _____

Good Luck!

Problem 1: [25 pts] Appearing on the next page is the *HW 3 Implementation*, one of the solutions to Homework 3, Problem 2, in which the `bgt` instruction resolves in `EX`. In the HW 3 Implementation there is a 1-cycle branch penalty when `bgt` is taken. (The branch penalty is the number of squashed instructions.) Suppose that based on benchmark analyses we find that `bgt` is mostly taken. For that we would like a New Implementation [tm] in which there is no penalty when `bgt` is taken, and a 1-cycle penalty when it's not taken. The PEDs below show execution examples for the HW3 and New implementations.

```
 # Cycle          0  1  2  3  4  5  6  7   HW 3 Impl, bgt taken, 1-cyc penalty
 bgt r1, r2  TARG  IF ID EX ME WB
 xor r3, r4, r5       IF ID EX ME WB
 or r6, r7, r8           IF IDx
TARG:
 and r9, r10, r11           IF ID EX ME WB
```

```
 # Cycle          0  1  2  3  4  5  6  7   HW 3 Impl, bgt not taken, no penalty
 bgt r1, r2  TARG  IF ID EX ME WB
 xor r3, r4, r5       IF ID EX ME WB
 or r6, r7, r8           IF ID EX ME WB
TARG:
 and r9, r10, r11
```

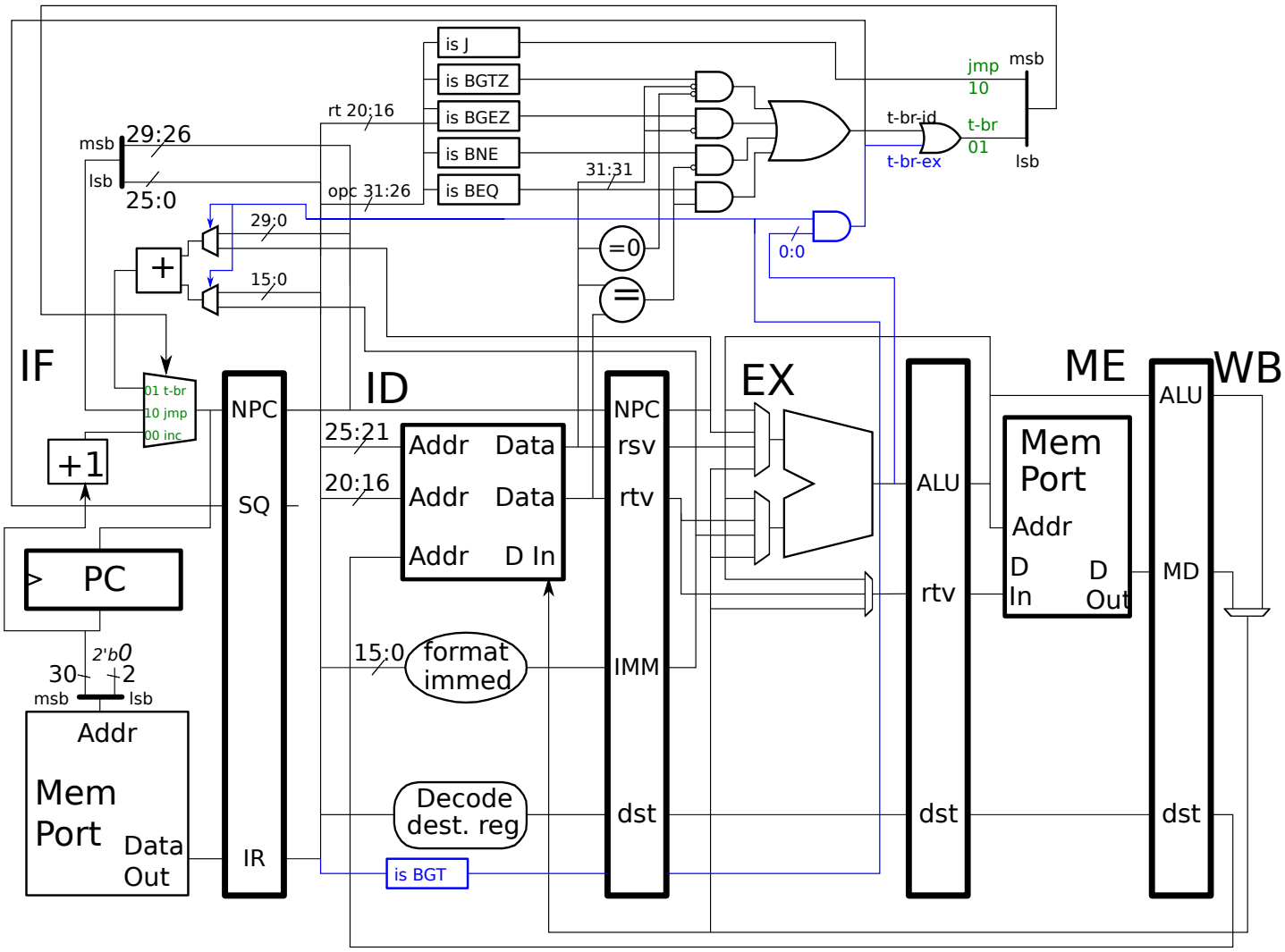
```
 # Cycle          0  1  2  3  4  5  6  7   New Impl, bgt taken, no penalty
 bgt r1, r2  TARG  IF ID EX ME WB
 xor r3, r4, r5       IF ID EX ME WB
 or r6, r7, r8
TARG:
 and r9, r10, r11        IF ID EX ME WB
```

```
 # Cycle          0  1  2  3  4  5  6  7   New Impl, bgt not taken, 1-cyc penalty.
 bgt r1, r2  TARG  IF ID EX ME WB
 xor r3, r4, r5       IF ID EX ME WB
 or r6, r7, r8           IF ID EX ME WB
TARG:
 and r9, r10, r11        IF IDx
 # Cycle          0  1  2  3  4  5  6  7
```

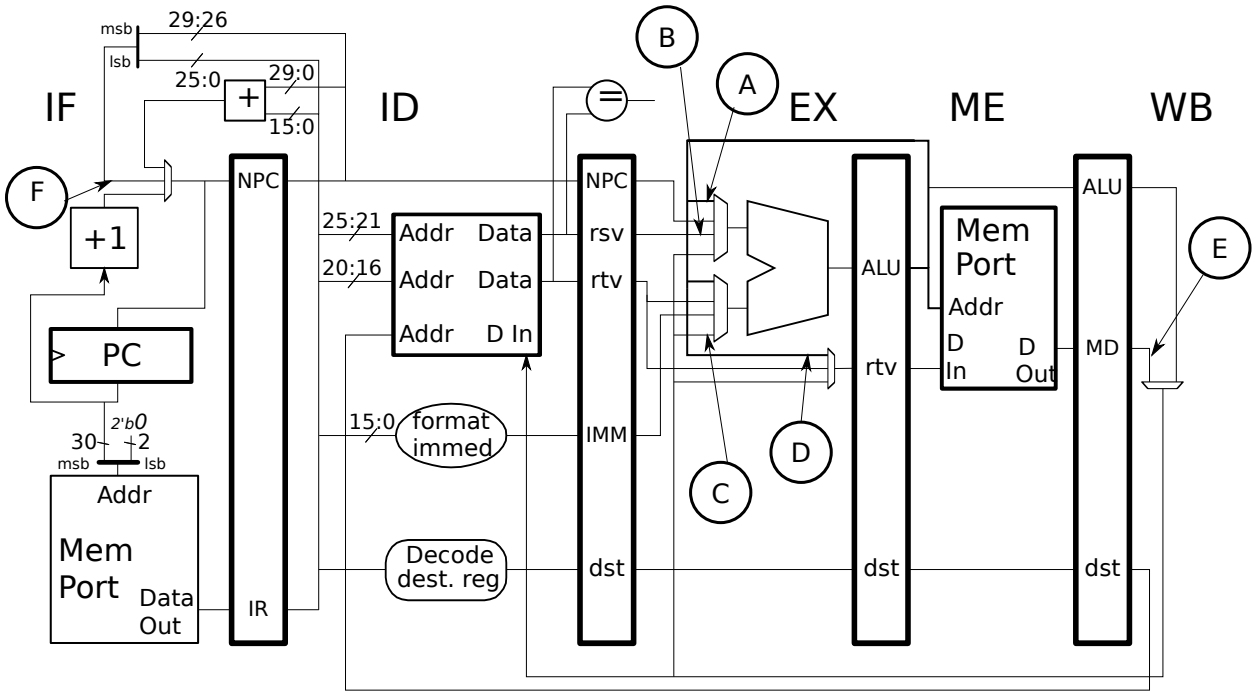
Problem 1, continued: Convert the HW 3 Implementation below into the New Implementation. *Hint: Calm down! A correct solution only requires three minor changes, one of those changes is substituting a mux input with a constant.*

☐ `ID` changes: `bgt` acts like it's always taken.

☐ `EX` changes: if `bgt` resolves not taken, fetch insn after delay slot insn.



Problem 2: [25 pts] In the implementation below several multiplexor inputs are labeled. For each labeled input write a program that uses it. A sample solution is provided for **A**.



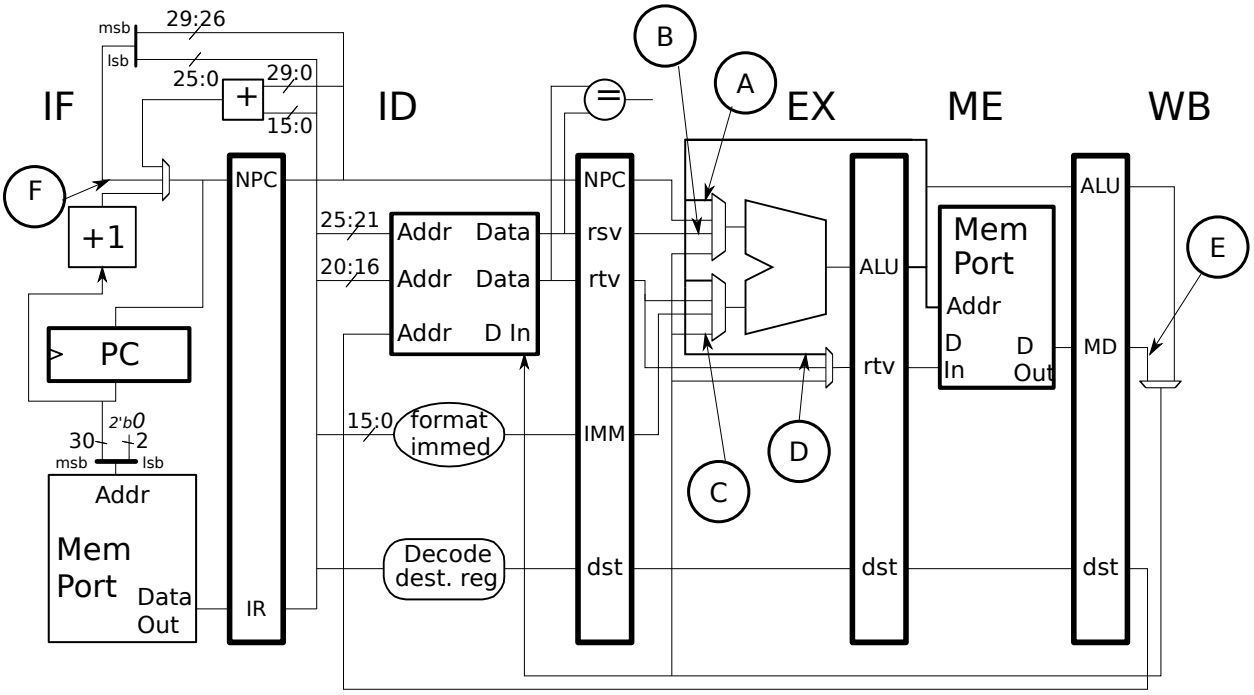
☑ Write a code fragment for mux input **A**. ☑ Mux input used in cycle 3, ☑ for register r1.

```
# SAMPLE SOLUTION -- Mux input A.
# Cycle          0  1  2  3  4  5
add r1, r2, r3   IF ID EX ME WB
sub r4, r1, r5      IF ID EX ME WB
```

☐ Write a code fragment for mux input **B**. ☐ Mux input used in cycle _____, ☐ for register _____.

☐ Write a code fragment for mux input **C**. ☐ Mux input used in cycle _____, ☐ for register _____.

Problem 2, continued:



☐ Write a code fragment for mux input **D**. ☐ Mux input used in cycle _____, ☐ for register _____.

☐ Write a code fragment for mux input **E**. ☐ Mux input used in cycle _____, ☐ for register _____.

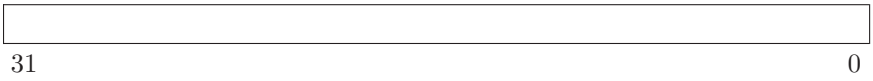
☐ Write a code fragment for mux input **F**. ☐ Mux input used in cycle _____.

Problem 3: [15 pts] Answer each question below.

(*a*) Show the encoding of the MIPS instructions below. The opcode for `beq` is `0x4` and the opcode for `lw` is `0x23`. *Hint: For the fields' bit positions see the implementation diagrams in previous problems.*

```
TARG:
 lw r1, 2(r3)
 beq r4, r5, TARG
```

☐ Encoding of `lw r1, 2(r3)`:



☐ Encoding of the `beq` above ☐ with the correct immediate field value.

(*b*) Many MIPS Format-R instructions, such as `add` and `sub`, have an opcode value of 0. Explain why they don't have their own opcode field values, such as `0x11` for `add` and `0x12` for `sub`.

☐ R-format instructions have opcode 0 because ...

☐ If R-format instructions each had their own opcodes that would be a problem because ...

Problem 4: [10 pts] Answer each question below.

(*a*) Describe what the dead-code elimination optimization is and provide an example.

☐ Description of dead-code elimination.

☐ Example.

(*b*) For instruction scheduling optimizations is it necessary or just helpful for the the compiler to know the implementation?

☐ Implementation: Necessary, important, not needed. ☐ Circle one ☐ and explain.

Problem 5: [10 pts] Answer each question below.

(*a*) A CISC ISA might have an instruction like `add (r1), r2, 8(r3)`, in which the source and destination come from memory.

☐ Explain why such an instruction is not suitable for a RISC ISA, ☐ refer to RISC goals in your answer.

(*b*) Describe a feature and goal of VLIW ISAs.

☐ VLIW feature:

☐ VLIW goal:

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

(*a*) SPECcpu benchmark results have two levels of tuning, *peak* and *base*. In one of these tuning levels the same optimization flags must be used for all benchmarks of the same language.

☐ Which tuning level is this for?

☐ What is the purpose for requiring the same flags?

(*b*) Explain how the following corruptions of SPECcpu would be (we hope) prevented.

☐ *The suite excludes benchmarks that perform poorly on Evil Company's products.* This won't happen because . . .

☐ *The SPECcpu results for Evil Company's System X are just made-up numbers.* This won't happen because . . .