CS151B/EE M116C

Computer Systems Architecture

Winter 2005 Midterm Exam II

Instructor: Prof. Lei He

It is a closed-book exam, except that a single-sided letter-size summary sheet is allowed.

There are total SEVEN pages including this cover page. Check whether you have all pages. If not, let the proctor know right now.

Good luck!

Student name (print) ________________________ ID ______________

Problem I: short problems _____________ of 15 points

Problem II: single cycle implementation _____________ of 18 points

Problem III: data dependency _____________ of 12 points

Problem IV: control logic for forwarding _____________ of 15 points

Total _______________ of 60 points

Problem 1. (15 points) Explain terms or answer short problems. For example, Program counter (PC) is the register containing the address of the instruction in the program being executed. (Hint: if you do not know how to explain the term precisely, you may use examples to explain).

(1) Explain the concept of delayed load AND give an example piece of code

The loaded data is available one clock cycle after the load instruction. For example in the code:

```
lw    $1,    0($2)
add   $3,    $1,    $4
```

The loaded data $1 from the first instruction is not available right after the instruction. For the add instruction, we need to add nop before it or stall one clock cycle to wait for $1 to be available.

(2) Explain the concept of loop unrolling and why we perform loop unrolling

Unroll the loop n times and rename the registers to make a big loop. Loop unrolling leads to more instructional level parallelism and therefore improve performance.

(3) Name three techniques (in either software or hardware) to resolve branch hazard or reduce performance loss of branch hazard

Stall until branch is known, guess branch direction, reduce branch delay, delayed branch (always execute instruction after branch).

(4) State the conditions to share hardware between different stages of multi-cycle implementation

A piece of hardware can be shared if it is used by a same instruction in different clock cyles. For instance, PC increment and R-type execution both make use of the same ALU, but they do so on different cycles.

(5) A single-cycle implementation may be divided into five stages for pipelining. Compare the average CPI between single-cycle and ideal pipelining implementations and explain why pipelining may improve performance.

The average CPI of both single-cycle and ideal pipelining implementations is 1. But the critical path length of the single-cycle implementation is much larger (usually N times larger, where N is the number of pipeline stages, N=5 in the problem) than the ideal pipeline. Therefore, pipeling can improve performance.

Problem 2 (18 points)

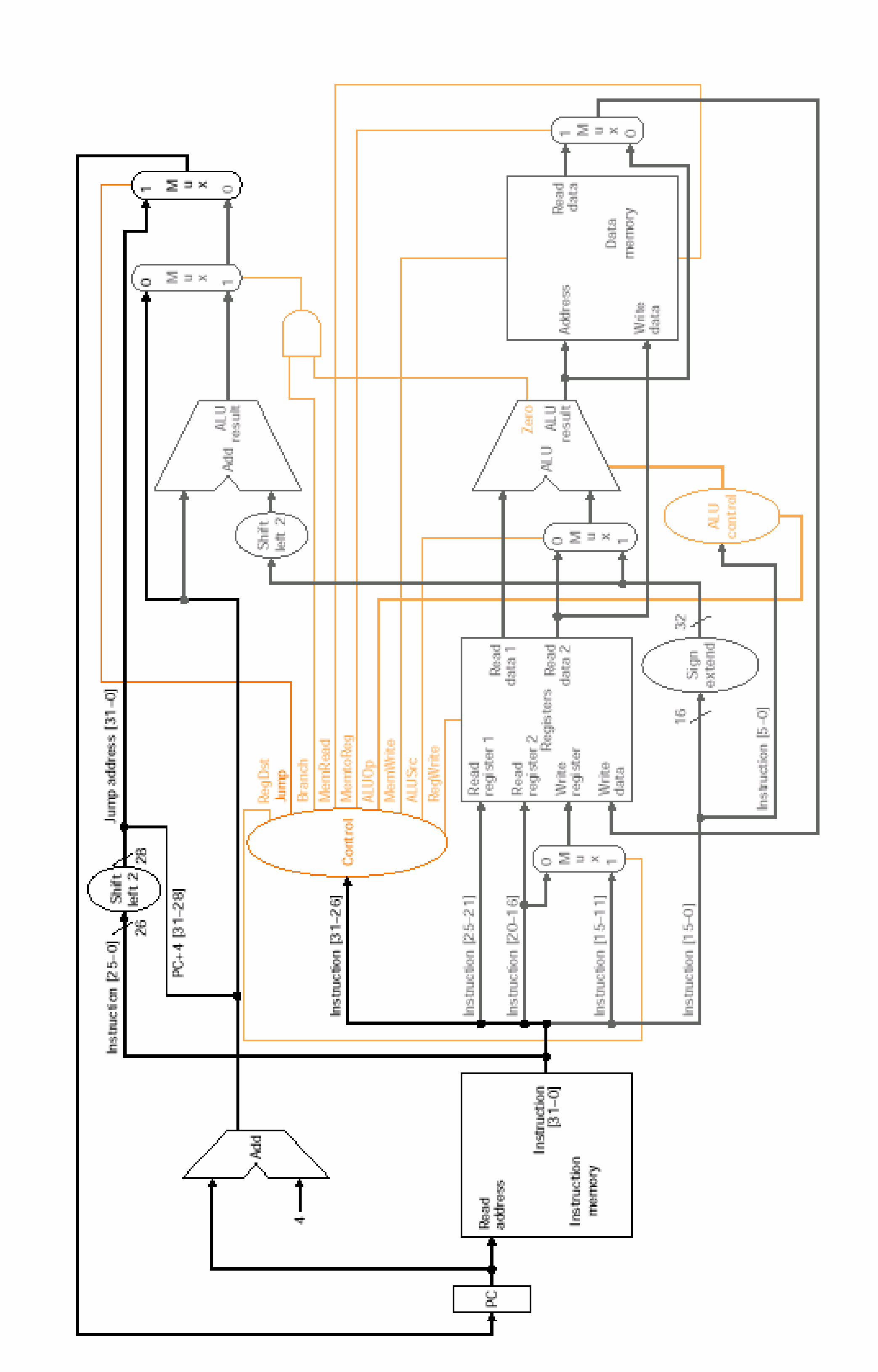
Perform the following:

(1) Find the values for all control signals in the single cycle implementation for the following instructions.

	R-type	**ori**	**lw**	**sw**	**beq**	**jump**
Opcode	00 0000	00 1101	10 0011	10 1011	00 0100	00 0010
RegDst	**1**	**0**	**0**	**x**	**x**	**x**
ALUSrc	**0**	**1**	**1**	**1**	**0**	**x**
MemtoReg	**0**	**0**	**1**	**x**	**x**	**x**
RegWrite	**1**	**1**	**1**	**0**	**0**	**0**
MemWrite	**0**	**0**	**0**	**1**	**0**	**0**
NPC_sel	**0**	**0**	**0**	**0**	**1**	**0**
Jump	**0**	**0**	**0**	**0**	**0**	**1**
ExtOp	**X**	**0**	**1**	**1**	**x**	**x**

(2) We wish to add the instructions subi (sub immediate) to the single-cycle machine. What changes do we need to make to the **control signals** and the **datapath**?

	Subi
Opcode	x
RegDst	0
ALUSrc	1
MemtoReg	0
RegWrite	1
MemWrite	0
NPC_sel	0
Jump	0
ExtOp ALU op	1 Sub


PC
4
Add
Read address
Instruction [31–0]
Instruction memory
Instruction [25–0]
Shift left 2
26
28
Jump address [31–0]
PC+4 [31–28]
Instruction [31–26]
Control
RegDst
Jump
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Instruction [25–21]
Instruction [20–16]
Instruction [15–11]
Instruction [15–0]
Instruction [5–0]
Read register 1
Read register 2
Write register
Write data
Registers
Read data 1
Read data 2
16
32
Sign extend
ALU control
Shift left 2
Add
ALU result
ALU
Zero
ALU result
Address
Write data
Read data
Data memory
M u x
0
1

Problem 3 (12 points):
Assume that we have a five-stage machine same as the one in textbook. For the following code,

(a) sub $2, $5, $4
(b) add $4, $2, $5 **$2 depends on (a)**
(c) lw $2, 100($4) **$4 depends on (b)**
(d) add $5, $2, $4 **$2 depends on (c), $4 depends on (b)**

(1) Name all data dependencies

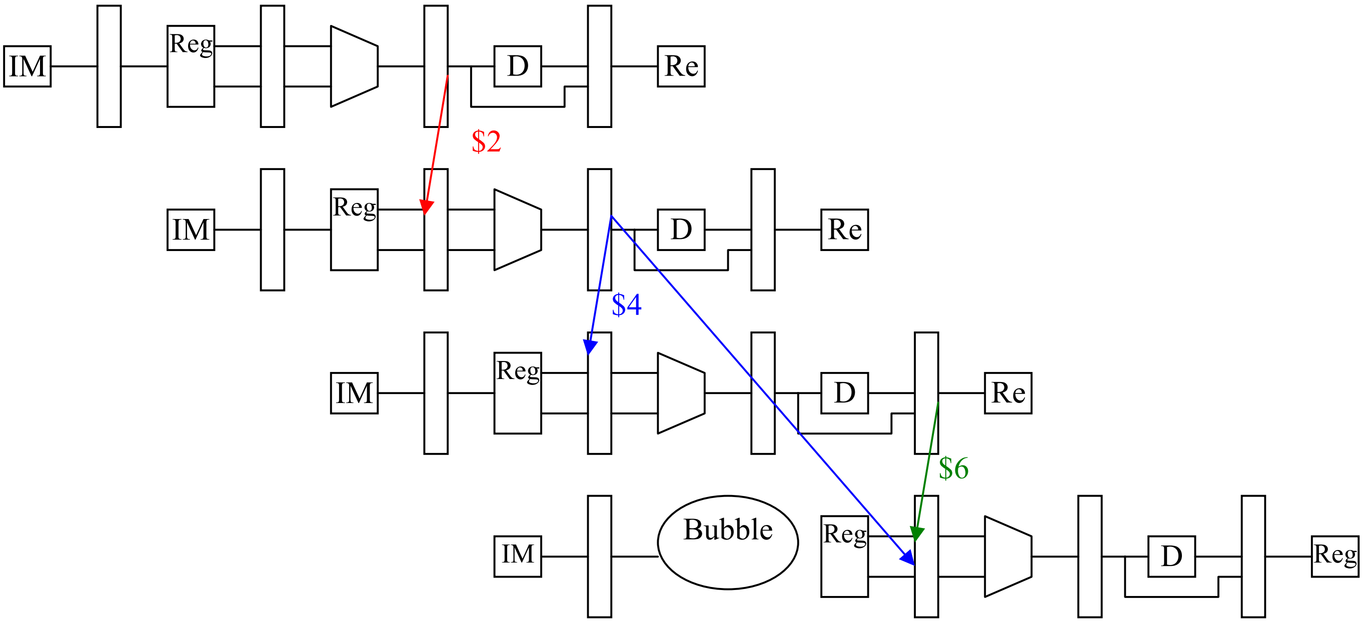
See the read bold words.

(2) Which data hazards can be resolved by renaming? Write down the code after renaming and with minimal data hazards

Write after write can be resolved by renaming. After renaming, the code will look like the following, $2 in (c) and (d) is renamed $6:

(a) sub $2, $5, $4
(b) add $4, $2, $5
(c) lw $6, 100($4)
(d) add $5, $6, $4

(3) After renaming, which data hazard can be resolved via forwarding? Illustrate all the forwarding using 5-stage pipelining figures similar to those in the textbook.



Problem 4 (15 points) Data forwarding

Considering data forwarding for the pipeline below, state how to generate control signals for MUX A. I.e., use plain English **AND** logic function such as *EX/MEM.RegisterRd != 0* to explain when the control signal for MUX A should be 00, 01 and 10 respectively.

I. control signal = 00

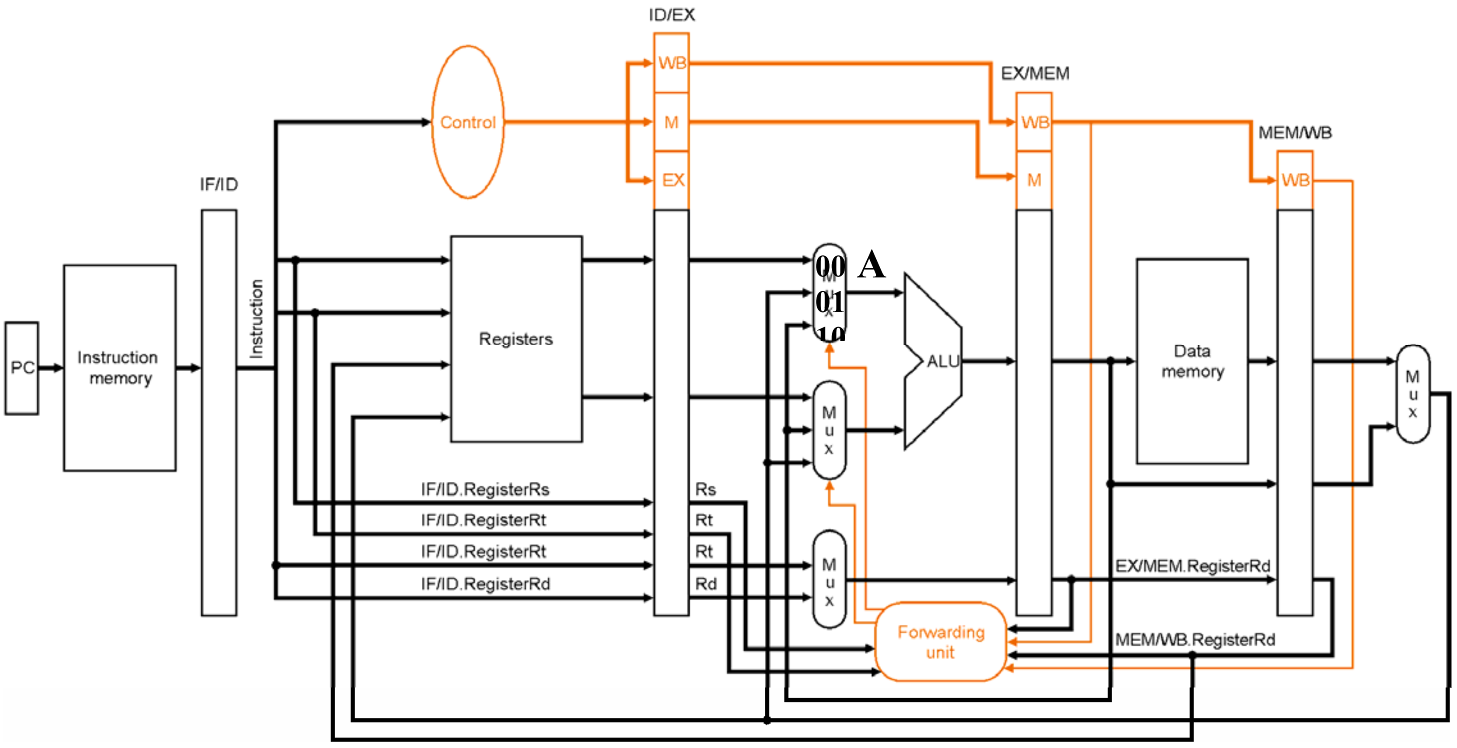
No data forwarding.
Neither condition below holds.

II. control signal = 01

Forward result from MEM/WB register.
If ((MEM/WB.RegWrite) &&
(IF/ID.RegisterRs != EX/MEM.RegisterRd) &&
(EX/MEM.RegWrite) // this condition is not in the book
(IF/ID.RegisterRs == MEM/WB.RegisterRd) &&
(IF/ID.RegisterRs != 0))

III. control signal = 10

Forward result from EX/MEM register.
If ((EX/MEM.RegWrite) &&
(IF/ID.RegisterRs == EX/MEM.RegisterRd) &&
(IF/ID.RegisterRs != 0))



ID/EX
WB
M
EX
EX/MEM
WB
M
MEM/WB
WB
Control
IF/ID
Instruction
PC
Instruction memory
Registers
00
01
10
A
M u x
ALU
M u x
M u x
Data memory
M u x
IF/ID.RegisterRs
IF/ID.RegisterRt
IF/ID.RegisterRt
IF/ID.RegisterRd
Rs
Rt
Rt
Rd
EX/MEM.RegisterRd
MEM/WB.RegisterRd
Forwarding unit