COSE 222 Exam-I (Spring'15) April 28 (Tue), 2015.

(Note: Make it sure that the exam paper is of 6 pages, and this is a 90 min. exam)

Name:
ID no.:

1.**(10pt)** Assuming the MIPS instruction set, answer the questions below with a short explanation.

(1). (5pt.) suppose that the program counter (PC) contains 32-bit, which equals the value 24 in decimal, pointing to a "`beq`" instruction. Is it possible that the next instruction to be executed is at the memory address either (a)28 or (b)8,000,036 in decimal? If your answer is "yes", show the immediate field value for the instruction.

(2). (5pt) Now suppose that the program counter (PC) contains the 24 in decimal, pointing to a "j" instruction. Is it possible that the next instruction to be executed is at the memory address either (a)28 or (b)8,000,036 in decimal? If your answer is "yes", show the immediate field value for the instruction.

COSE 222 Exam-I (Spring'15) April 28 (Tue), 2015.

(Note: Make it sure that the exam paper is of 6 pages, and this is a 90 min. exam)

Name:
ID no.:

1.**(10pt)** Assuming the MIPS instruction set, answer the questions below with a short explanation.

(1). (5pt.) suppose that the program counter (PC) contains 32-bit, which equals the value 24 in decimal, pointing to a "beq" instruction. Is it possible that the next instruction to be executed is at the memory address either (a)28 or (b)8,000,036 in decimal? If your answer is "yes", show the immediate field value for the instruction.

(a) Yes, because the immediate field value can be **0** or branch can be **Not Taken**
(b) No, because the immediate field value must be **2,000,002** but is larger than 65,535 = 2^{16}

Opcode(6bit)	Rs(5bit)	Rt(5bit)	Immediate field(16bit)

Jump destination of "beq" instruction is following
PC=PC+4+(Immediate value<<2)

(2). (5pt) Now suppose that the program counter (PC) contains the 24 in decimal, pointing to a "j" instruction. Is it possible that the next instruction to be executed is at the memory address either (a)28 or (b)8,000,036 in decimal? If your answer is "yes", show the immediate field value for the instruction.

(a) Yes, because the immediate field value can be **7**
(b) Yes, because the immediate field value can be **2,000,009**

Opcode(6bit)	Immediate field(26bit)

Jump destination of "j" instruction is following
PC={PC[31:28],(Immediate value<<2)[27:0]}

2. .(10pts.)
Two decimal numbers X= 7 and Y= -4 will be multiplied using the Booth algorithm.
(1). (5pts) Assuming 4-bit 2's complement binary number representation, Show the booth encoding of Y and state how many times of Shift, Add, and Subtract will happen in the multiplication of X by Y, i.e., X*Y, using the Booth Algorithm.

(2). (5pt) Instead of X*Y, the number of operations for the multiplication may change if we exchange multiplicand and multiplier, i..e. Y*X. Explain which produces less number of operations by showing how many Add and Subtract operations will happen for Y*X using the Booth Algorithm.

2. .(10pts.)
Two decimal numbers X= 7 and Y= -4 will be multiplied using the Booth algorithm.
(1). (5pts) Assuming 4-bit 2's complement binary number representation, Show the booth encoding of Y and state how many times of Shift, Add, and Subtract will happen in the multiplication of X by Y, i.e., X*Y, using the Booth Algorithm.

P	Y		
0000	11000	00	shift
0000	01100	00	shift
0000	00110	10	sub
1001	00110		& shift
1100	10011	11	shift
1110	0100 = -28		

shift : 4
add : 0
sub : 1

(2). (5pt) Instead of X*Y, the number of operations for the multiplication may change if we exchange multiplicand and multiplier, i..e. Y*X. Explain which produces less number of operations by showing how many Add and Subtract operations will happen for Y*X using the Booth Algorithm.

P	X		
0000	01110	10	sub
0100	01110		& shift
0010	00111	11	shift
0001	00011	11	shift
0000	10001	01	add
1100	10001		& shift
1110	0100 = -28		

shift : 4
add : 1
sub :1

3.(20pt) We will follow the IEEE standard floating point number representation but for the sake of simplicity with less number of bits. Let's assume that we have 16bit floating point number representation : **sign-bit|5-bit exponent with bias|10-bit fraction**
Note that the bias would be 15 and that the significand will be 1+(10-bit fraction). Also, all 0's for the 16bit representation is reserved for representing the zero and all 1's for the exponent (with bias) is reserved for special cases, e.g., result from division by 0.

(1).(8pt)
(a) What would be the smallest positive number that can be represented?

(b). What would be the largest positive number that can be represented?

(2). (12pt)
Now you are adding the above two numbers, the smallest and the largest positive numbers. What is the addition result, if
(a). one sticky bit and two bits of the guard and rounding are available as in the IEEE standard.

(b). none of the guard, rounding, and sticky bits are available

3.(20pt) We will follow the IEEE standard floating point number representation but for the sake of simplicity with less number of bits. Let's assume that we have 16bit floating point number representation : **sign-bit|5-bit exponent with bias|10-bit fraction**
Note that the bias would be 15 and that the significand will be 1+(10-bit fraction). Also, all 0's for the 16bit representation is reserved for representing the zero and all 1's for the exponent (with bias) is reserved for special cases, e.g., result from division by 0.

(1).(8pt)
(a) What would be the smallest positive number that can be represented?

0 | 00001 | 00000 00000

1.0×2^{-14}

(b). What would be the largest positive number that can be represented?

0 | 11110 | 11111 11111

$1.11111\ 11111 \times 2^{15}$

(2). (12pt)
Now you are adding the above two numbers, the smallest and the largest positive numbers. What is the addition result, if
(a). one sticky bit and two bits of the guard and rounding are available as in the IEEE standard.

0 | 11110 | 11111 11111

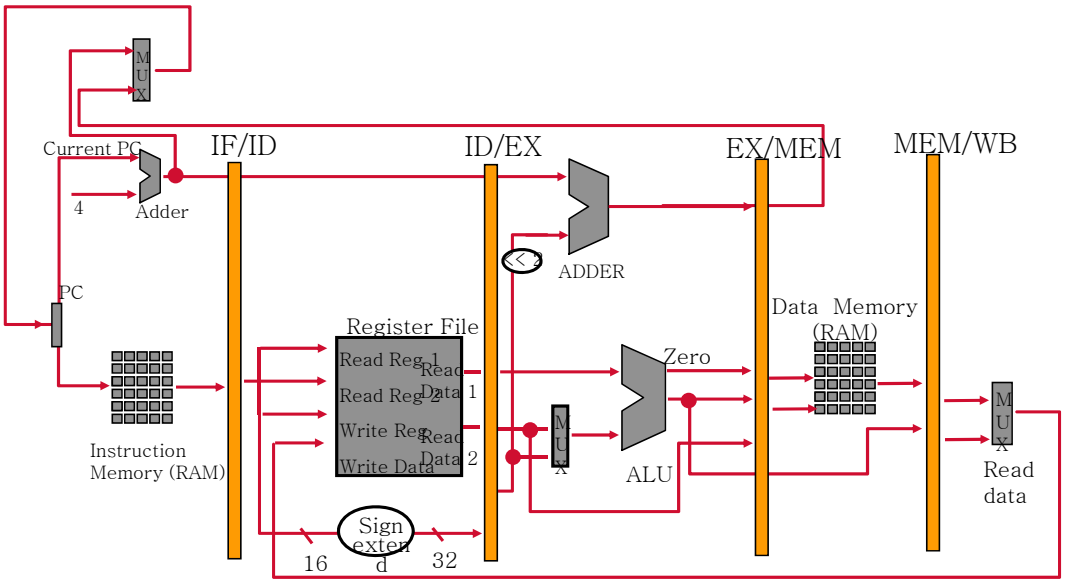
1.11111 11111 0(g) 0(r) 1(s) $\times 2^{15}$

(b). none of the guard, rounding, and sticky bits are available

0 | 11110 | 11111 11111

$1.11111\ 11111 \times 2^{15}$

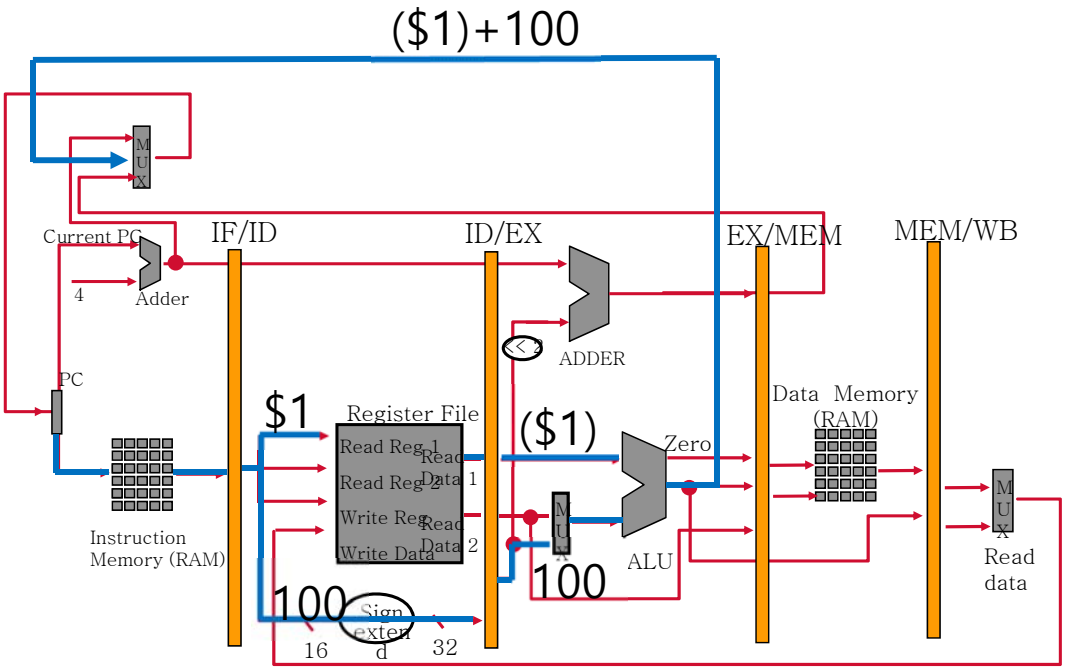
4. (20pt)
The below is a simplified MIPS processor diagram with intermediate latches between the pipeline stages.



(1). (10pt) Now you are asked to modify the data path to include **jr** instruction with 16-bit offset. **jr** causes the next instruction to be the one at specified in a source register, i.e. PC = (Reg $1)+100 for **jr 100($1)**. Draw necessary modification clearly on the data path figure above (no need to show control path change)

(2). (10pt) With your modified data path, state that how many bubbles (pipeline stall cycles) you need between a **jr** instruction and the next instruction to be executed, i.e. the target instruction.

4. (20pt)
The below is a simplified MIPS processor diagram with intermediate latches between the pipeline stages.



(1). (10pt) Now you are asked to modify the data path to include **jr** instruction with 16-bit offset. **jr** causes the next instruction to be the one at specified in a source register, i.e. PC = (Reg $1)+100 for **jr 100($1)**. Draw necessary modification clearly on the data path figure above (no need to show control path change)

Above blue lines are answers.

(2). (10pt) With your modified data path, state that how many bubbles (pipeline stall cycles) you need between a **jr** instruction and the next instruction to be executed, i.e. the target instruction.

IF	ID	EX	Mem	WB			
	stall	stall	IF	ID	EX	Mem	WB

2 bubbles

5. **(20pt)** Consider the following code that will be executed on the MIPS implemented in 5-stage pipeline, IF, ID, EX, MEM, WB, as in the text.

```
add    $3,$3,$4
lw     $2,100($3)
add    $2,$3,$6
```

(1). (10pt) Assume the MIPS pipeline implementation is without data forwarding and no delayed slot, show how many cycles need to be spent to finish the code. To show the pipeline stalls, when and where, insert "nop" instructions into the code and also state the CPI for the code.

(2). (10pt) Now the pipeline is with data forwarding but without delay slots, show how many cycles are needed for the code. To show the pipeline stalls, when and where, insert "nop" instructions into the code and also state the CPI for the code.

5. **(20pt)** Consider the following code that will be executed on the MIPS implemented in 5-stage pipeline, IF, ID, EX, MEM, WB, as in the text.

```
add    $3,$3,$4
lw     $2,100($3)
add    $2,$3,$6
```

(1). (10pt) Assume the MIPS pipeline implementation is without data forwarding and no delayed slot, show how many cycles need to be spent to finish the code. To show the pipeline stalls, when and where, insert "nop" instructions into the code and also state the CPI for the code.

```
add    $3,$3,$4
nop
nop
lw     $2,100($3)
add    $2,$3,$6
```

9/5

add	**$3,$3,$4**	IF	ID	EX	Mem	WB				
nop			IF	ID	EX	Mem	WB			
nop				IF	ID	EX	Mem	WB		
lw	**$2,100($3)**				IF	ID	EX	Mem	WB	
add	**$2,$3,$6**					IF	ID	EX	Mem	WB

(2). (10pt) Now the pipeline is with data forwarding but without delay slots, show how many cycles are needed for the code. To show the pipeline stalls, when and where, insert "nop" instructions into the code and also state the CPI for the code.

```
add    $3,$3,$4
lw     $2,100($3)
add    $2,$3,$6
```

7/3

add	**$3,$3,$4**	IF	ID	EX	Mem	WB		
lw	**$2,100($3)**		IF	ID	EX	Mem	WB	
add	**$2,$3,$6**			IF	ID	EX	Mem	WB

6. (20pt) Consider the following code that will be executed on the MIPS 5-stage pipeline at 100Mhz cycle. It now has hazard detection and stall unit, which needs no explicit delay slot. Assume that branch can make its decision in the second, i.e., ID, stage. We will consider executing the code shown below with and without data forwarding.

```
loop  sub   $1,$4,$6
      add   $3,$3,$4
      add   $5,$3,$6
      lw    $2,100($3)
      beq   $2,$4,loop
```

(1). Assume the MIPS pipeline implementation is without data forwarding, show how many cycles need to be spent to finish a single iteration of the loop including the loop ending conditional, **beq $2,$4,loop**. You need to show the pipeline stalls; when and where, and also state the CPI for the single iteration.

(2). Now assume that the pipeline is with data forwarding. Show what is the resulting CPI for the single iteration with data forwarding?

6. (20pt) Consider the following code that will be executed on the MIPS 5-stage pipeline at 100Mhz cycle. It now has hazard detection and stall unit, which needs no explicit delay slot. Assume that branch can make its decision in the second, i.e., ID, stage. We will consider executing the code shown below with and without data forwarding.

```
loop   sub    $1,$4,$6
       add    $3,$3,$4
       add    $5,$3,$6
       lw     $2,100($3)
       beq    $2,$4,loop
```

(1). Assume the MIPS pipeline implementation is without data forwarding, show how many cycles need to be spent to finish a single iteration of the loop including the loop ending conditional, **beq $2,$4,loop**. You need to show the pipeline stalls; when and where, and also state the CPI for the single iteration.

sub	**$1,$4,$6**	IF	ID	EX	Mem	WB								
add	**$3,$3,$4**		IF	ID	EX	Mem	WB							
add	**$5,$3,$6**			IF	ID	stall	stall	EX	Mem	WB				
lw	**$2,100($3)**				IF	stall	stall	ID	EX	Mem	WB			
beq	**$2,$4,loop**					stall	stall	IF	ID	stall	stall	EX	Mem	WB

13/5

(2). Now assume that the pipeline is with data forwarding. Show what is the resulting CPI for the single iteration with data forwarding?

sub	**$1,$4,$6**	IF	ID	EX	Mem	WB						
add	**$3,$3,$4**		IF	ID	EX	Mem	WB					
add	**$5,$3,$6**			IF	ID	EX	Mem	WB				
lw	**$2,100($3)**				IF	ID	EX	Mem	WB			
beq	**$2,$4,loop**					IF	ID	stall	stall	EX	Mem	WB

11/5