

Q1> [10 points]

Indicate whether the following explanation is true or not.

(For correct answer +2 points, for wrong answer -2 points)

- MIPS instruction set has 'blt' instruction which means 'branch if less than'. (O, X)
- 'lw' is the MIPS R-format instruction. (O, X)
- Intel's IA-32 CPUs adopt RISC philosophy. (O, X)
- MIPS processor needs adjunct processor to process floating-point format data. (O, X)
- The maximum number that can be represented by the IEEE 754 single-floating-point standard is $1.1111\ 1111\ 1111\ 1111\ 1111\ 111\ \times 2^{126}$. (O, X)

Q2> [5 points]

Fill in the blank space of the sentence with the appropriate terms.

- Tera byte is 10^{12} or 2^{40} bytes.
- Assembler is a program that translates symbolic instructions to binary instructions (machine code).
- In MIPS, \$ra register keeps a link to the calling site that allows a procedure to return to the proper address.
- Exponents 00000000 and 11111111 are reserved in IEEE 754 single floating-point standard.
- In IEEE 754 floating-point standard, denormalized number allow a number to degrade in significance until it becomes 0, called gradual underflow.

Q3> [12 points] (2 2 points)

Consider two different implementations of the same instruction set architecture. There are four classes of instructions, A, B, C, and D. The clock rate and CPI of each implementation are given in the following table.

Implementation	Clock Rate	CPI Class A	CPI Class B	CPI Class C	CPI Class D
P1	2.5GHz	2	1.5	2	1
P2	3GHz	1	2	1	1

Q3-1> Given a program with 10^6 instructions divided into classes as follows: 10% class A, 20% class B, 50% class C, and 20% class D, show the CPU time for executing the program. [4 points]

A>

Class A: 10^5 instr. Class B: 2×10^5 instr. Class C: 5×10^5 instr. Class D: 2×10^5 instr.

Time = No. instr. \times CPI/clock rate

Total time P1 = $(10^5 \times 2 + 2 \times 10^5 \times 1.5 + 5 \times 10^5 \times 2 + 2 \times 10^5)/(2.5 \times 10^9)$

= $6.8 \times 10^{-4} \text{ s} = 0.68 \times 10^{-3} \text{ s} = 680 \text{ us}$

Total time P2 = $(10^5 + 2 \times 10^5 \times 2 + 5 \times 10^5 + 2 \times 10^5)/(3 \times 10^9)$

= $4 \times 10^{-4} \text{ s} = 0.4 \times 10^{-3} \text{ s} = 400 \text{ us}$

Q3-2> What is the global CPI for each implementation? [4 points]

A>

CPI = time \times clock rate/No. instr.

CPI(P1) = $6.8 \times 10^{-4} \times 2.5 \times 10^9/10^6 = \underline{1.7}$

CPI(P2) = $4 \times 10^{-4} \times 3 \times 10^9/10^6 = \underline{1.2}$

Q3-3> Find the clock cycles required in both cases. [4 points]

A>

Clock cycles = CPI \times instruction count

Clock cycles (P1) = $1.7 \times 10^6 = 17 \times 10^5 = \underline{1,700,000}$

Clock cycles (P2) = $1.2 \times 10^6 = 12 \times 10^5 = \underline{1,200,000}$

Q4> [6 points]

This question is about the MIPS R-format instruction.

Describe the field structure and the role of each field. (Fill out all blanks)

op	rs	rt	rd	shamt	Funct
<u>6</u> bits	<u>5</u> bits	<u>5</u> bits	<u>5</u> bits	<u>5</u> bits	6 bits

Field name	Role
Op	<u>operation code (opcode)</u>
Rs	<u>source register number</u>
Rt	<u>source register number</u>
Rd	<u>destination register number</u>
Shamt	<u>shift amount</u>
funct	function code (extends opcode)

오류 1개 = -1 point

Q5> [16 points]

This question is about the MIPS PC-relative addressing mode.

Show the range of addresses that can be reached with two consecutive branch instructions. Assume that the PC is initially at address 0x00000000. The answer should be in hexadecimal format. (Note that, the immediate value of branch instruction should be interpreted as a signed integer number.)

Increment way

$$\rightarrow 1^{\text{st}} \text{ branch: } PC+4+4 \times (2^{15} - 1) = 0x0000_0000 + 4 + 4 \times (2^{15} - 1) = 0x0002_0000$$

$$\rightarrow 2^{\text{nd}} \text{ branch: } PC+4+4 \times (2^{15} - 1) = 0x0002_0000 + 4 + 4 \times (2^{15} - 1) = 0x0004_0000$$

Decrement way

$$\rightarrow 1^{\text{st}} \text{ branch: } PC+4+4 \times (-1) \times (2^{15}) = 0x0000_0000 + 4 + 4 \times (-1) \times (2^{15}) = 0xFFFE_0004$$

$$\rightarrow 2^{\text{nd}} \text{ branch: } PC+4+4 \times (-1) \times (2^{15}) = 0xFFFE_0004 + 4 + 4 \times (-1) \times (2^{15}) = 0xFFFC_0008$$

A>

0xFFFC_0008 [8 point] ~ 0x0004_0000 [8 point]

방향 반대시 각 -4 points

Q6> [12 points] (항목 당 1 point)

You will be asked to evaluate the following C code statements in MIPS assembly code.

```
for(i=0; i<a; i++)
    for(j=0; j<b; j++)
        D[4*j] = i + j;
```

The following table is incomplete version of the MIPS assembly code for the above C code. Fill out the blanks of table. Assume that the values of a, b, i, and j are in registers \$s0, \$s1, \$t0, and \$t1, respectively. Also, assume that register \$s2 holds the base address of the array D.

Label	MIPS assembly Instruction			
LOOP1:	addi	\$t0	\$0	0
	<u>beq</u>	\$0	\$0	<u>TEST1</u>
	addi	\$t1	\$0	0
	<u>beq</u>	\$0	\$0	<u>TEST2</u>
LOOP2:	add	\$t3	\$t0	\$t1
	<u>sll</u>	\$t2	<u>\$t1</u>	<u>4</u>
	<u>add</u>	\$t2	\$t2	<u>\$s2</u>
	sw	<u>\$t3</u>	0	(\$t2)
TEST2:	addi	\$t1	\$t1	1
	slt	\$t2	\$t1	\$s1
	<u>bne</u>	\$t2	\$0	LOOP2
	addi	\$t0	\$t0	1
TEST1:	slt	\$t2	\$t0	\$s0
	<u>bne</u>	\$t2	\$0	LOOP1

Q7> [12 points] (항목 당 1 point)

This question is about the integer division computation with the optimized divider shown in Fig. Q7.

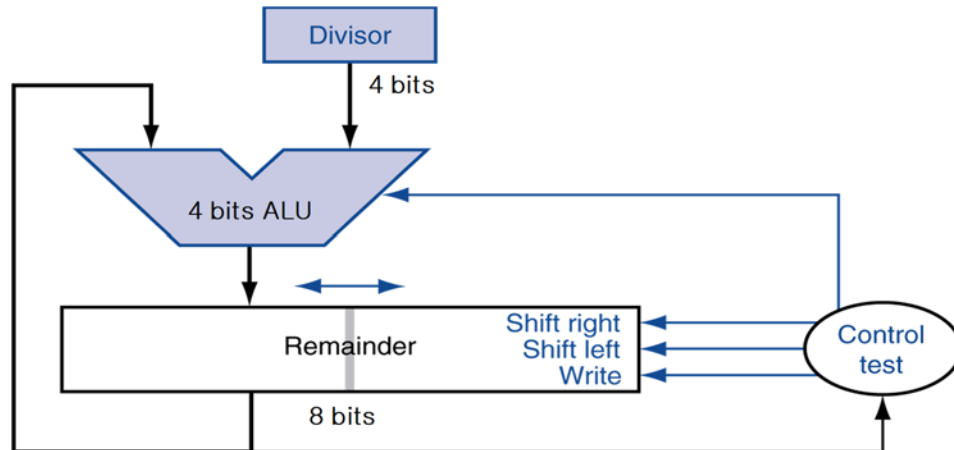


Fig. Q7

The following table is incomplete version of division computation for $0111 \div 0010$. Fill out the blanks of table and show remainder and quotient in unsigned binary integer format.

Iteration	Step	Divisor	Remainder
0	Initial values	0010	0000 0111
	Shift Rem left 1	0010	0000 1110
1	Rem = Rem - Div	0010	1110 1110
	Rem < 0 \rightarrow + Div, sll R, R0 = 0	0010	0001 1100
2	Rem = Rem - Div	0010	1111 1100
	Rem < 0 \rightarrow + Div, sll R, R0 = 0	0010	0011 1000
3	Rem = Rem - Div	0010	0001 1000
	Rem \geq 0 \rightarrow sll R, R0 = 1	0010	0011 0001
4	Rem = Rem - Div	0010	0001 0001
	Rem \geq 0 \rightarrow sll R, R0 = 1	0010	0010 0011
	Shift left half of Rem right 1	0010	0001 0011

Remainder: 0001

Quotient: 0011

Q8> [10 points] (과정 점수 없음)

Write down the hexadecimal representation of the decimal number 63.25, assuming the IEEE 754 single precision format.

$$63.25 \times 10^0 = 111111.01 \times 2^0$$

normalize, move binary point 5 to the left

$$1.1111101 \times 2^5$$

$$\text{sign} = \text{positive}, \text{exp} = 127 + 5 = 132$$

Final bit pattern: 0 1000 0100 1111 1010 0000 0000 0000 000

$$= 0100\ 0010\ 0111\ 1101\ 0000\ 0000\ 0000\ 0000 = \underline{0x427D0000}$$

No hexadecimal : -3 points

Q9> [10 points]

Calculate the following equation with guard, round, and sticky bits which are adopted in the IEEE 754. Consider 4-bits precision for fraction. The final result should be in normalized form. For rounding, 'round to nearest even' should be used. You should indicate guard, round, and sticky bits in your answer.

$$1.0000 \times 2^0 + 1.0001 \times 2^{-5}$$

The diagram shows the binary addition of 1.0000×2^0 and 0.000010×2^0 . The result is 1.000010×2^0 . The bits are labeled as follows: 'g' (guard) for the 5th bit after the decimal point, 'r' (round) for the 4th bit, and 's' (sticky) for the 5th bit. The final result is 1.0001×2^0 .

$$\begin{array}{r} 1.0000 \times 2^0 \\ + 0.000010 \times 2^0 \\ \hline 1.000010 \times 2^0 \\ \hline 1.0001 \times 2^0 \end{array}$$

g r s

A>

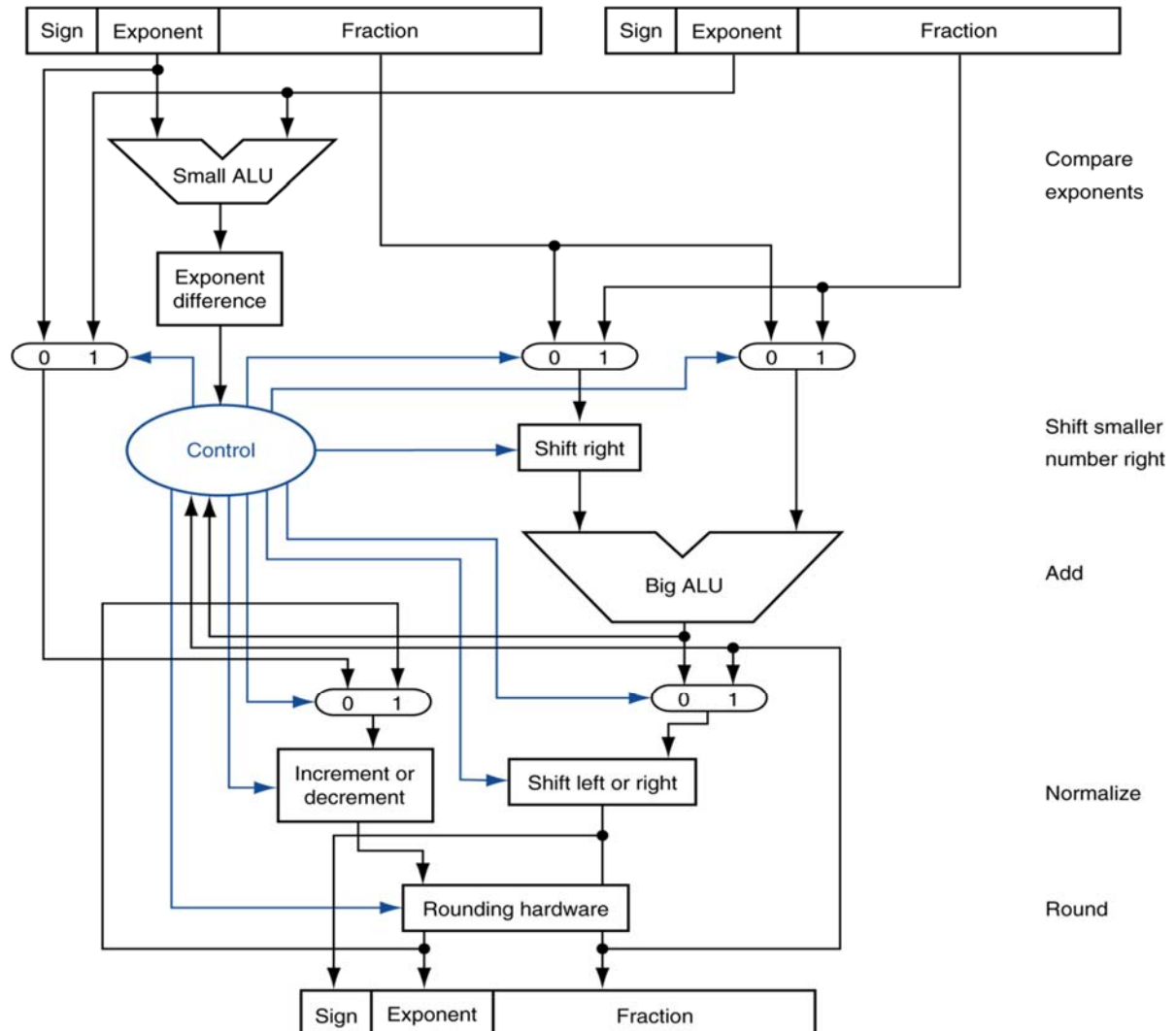
Guard, round, sticky bit를 명시 각 2점

최종 정답 4점

Q10> [7 points]

The following figure shows a floating point adder hardware.

Explain why the floating point adder hardware needs two multiplexers in the normalization step.



A>

Re-normalization을 위해서, Rounding 결과 다시 normalization이 필요한 경우가 발생 할 수 있으므로.

의미는 통하나 애매하거나 틀린 내용이 나타난 경우 3점