2.29

$t1 = i , $s2 = result , $s0 = MemArray

slti $t2, $t1, 100
⇒ $t1이 100보다 작으면 1로 세팅
그렇지 않으면 0
bne $t2, $0, Loop
⇒ $t2와 $0이 다르면 Loop로 점프

⇒ for(int i=0 ; i < 100 ; i++)

코드 2번째 ~ 3번째 줄은
result($s2)에 $s1(로드 된 MemArray[i])
를 더해주고 $s0(현재 MemArray 위치)를
integer 사이즈 4만큼 증가시키고 있다.
∴ 문제의 코드는

```
for(int i=0; i<100; i++){
  result = result + MemArray[i];
}
```

2.39

immediate format으로 나타낼 수 있는 수는
16bits이고 문제에서 원하는 수는 32bits이다.

∴ lui, ori instructions 사용.

$$\underbrace{0010\ 0000\ 0000\ 0001}_{0\times2001}\ \underbrace{0100\ 1001\ 0010\ 0100}_{0\times4924}$$

∴ lui $t1, 0x2001
  ori $t1, $t1, 0x4924

3.13

Iteration	step	multiplier	multiplicand	product
0	Initial value	0001 0010	0000 0110 0010	0000 0000 0001 0010
1	1:0 ⇒ no operation	0001 0010	0000 0110 0010	0000 0000 0001 0010
	2,3 : shift right product	0001 0010	0000 0110 0010	0000 0000 0000 1001
2	1a : 1 ⇒ Prod = Prod + Mcand	0001 0010	0000 0110 0010	0110 0010 0000 1001
	2,3 : 〃	0001 0010	0000 0110 0010	0011 0001 0000 0100
3	1:0 ⇒ 〃	0001 0010	0000 0110 0010	0011 0001 0000 0100
	2,3 : 〃	0001 0010	0000 0110 0010	0001 1000 1000 0010
4	1:0 ⇒ 〃	0001 0010	0000 0110 0010	0001 1000 1000 0010
	2,3 : 〃	0001 0010	0000 0110 0010	0000 1100 0100 0001

5	1,2 : 〃	0001 0010	0000 0110 0010	0110 1110 0100 0001
	2,3 : 〃	0001 0010	0000 0110 0010	0011 0111 0010 0000
6	1,0 : 〃	〃	〃	0011 0111 0010 0000
	2,3 : 〃	〃	〃	0001 1011 1001 0000
7	1,0 : 〃	〃	〃	0001 1011 1001 0000
	2,3 : 〃	〃	〃	0000 1101 1100 1000
8	1,0 : 〃	〃	〃	0000 1101 1100 1000
	2,3 : 〃	〃	〃	0000 0110 1110 0100

$0000\ 0110\ 1110\ 0100_2 = 0664_{16}$

∴ 결과 : 0x0664

3.14

(hardware)

1 - product = product + multiplicand

2 - shift multiplicand left, multiplier right

3 - decide multiplication is done.

∴ 8 (bits) × 3 (operations per iteration) × 4 (time unit per operation)

= 96 time units

(software)

1 - decide to add

2 - Product = Product + multiplicand

3 - shift multiplicand left

4 - shift multiplier right

5 - Is repetition done?

∴ 8 (bits) × 5 (operation per iteration) × 4 (time unit per operation)

= 160 time units.

3.19 74÷21 8진수로 풀이(6bits)

Iteration	Step	Divisor	Remainder
0	initial values	010001	000000 111100
	shift remainder left	010001	000001 111000
1	1: remainder = remainder-Divisor	010001	110000 111000
	2b: remainder <0 ⇒ +Divisor, sll R, R0=0	010001	000011 110000
2	1: 〃	010001	110010 110000
	2b: 〃	010001	000111 100000
3	1: 〃	010001	110110 100000
	2b: 〃	010001	001111 000000
4	1: 〃	〃	111110 000000
	2b: 〃	〃	011110 000000
5	1: remainder =remainder-Divisor	010001	001101 000000
	2a: remainder>=0 ⇒ sll R, R0=1	010001	011010 000001
6	1: 〃	〃	001001 000001
	2a: 〃	〃	010010 000011
Done	shift left half of Rem right	〃	001001 000011

∴ Quotient = 3_8 = 3, remainder = 11_8 = 9

3.24

$63.25 = 111111.01_2$

$= 1.1111101_2 \times 2^5$

signed bit : 0 (+)

exponent : 5+1023 = 1028

$\therefore 0\ 100\ 0000\ 0100\ \underbrace{1111\ 1010\ 0000\ \text{--------}\ 0000}_{52bits}$

Floating point operation

a)

A를 8-bit IEEE 754 format으로 표현 ($A = 0.0010011_2$)

$A = 1.0011 \times 2^{-3}$

signed bit : 0 (+)

exponent : (Bias in this case : 7) $-3+7 = 4$

function : 001 (rounding)

$\therefore$ 0 0100 001

B를 IEEE754 format으로 표현 ($B = 18.0_{10}$)

$B = 10010.0 = 1.001 \times 2^4$, signed bit : 0

exponent : 4+7=11, function : 001 (rounding)

$\therefore$ 0 1011 001

b)

$A+B = 1.0011 \times 2^{-3} + 1.001 \times 2^{4}$

$= 0.00000001\,0011 \times 2^{4} + 1.001 \times 2^{4}$

$= 1.0010001\,0011 \times 2^{4}$

Slgned bit : 0

exponent : $4+7=11$, 1011

fraction : 001 (rounding)

$\therefore$ 0 1011 001