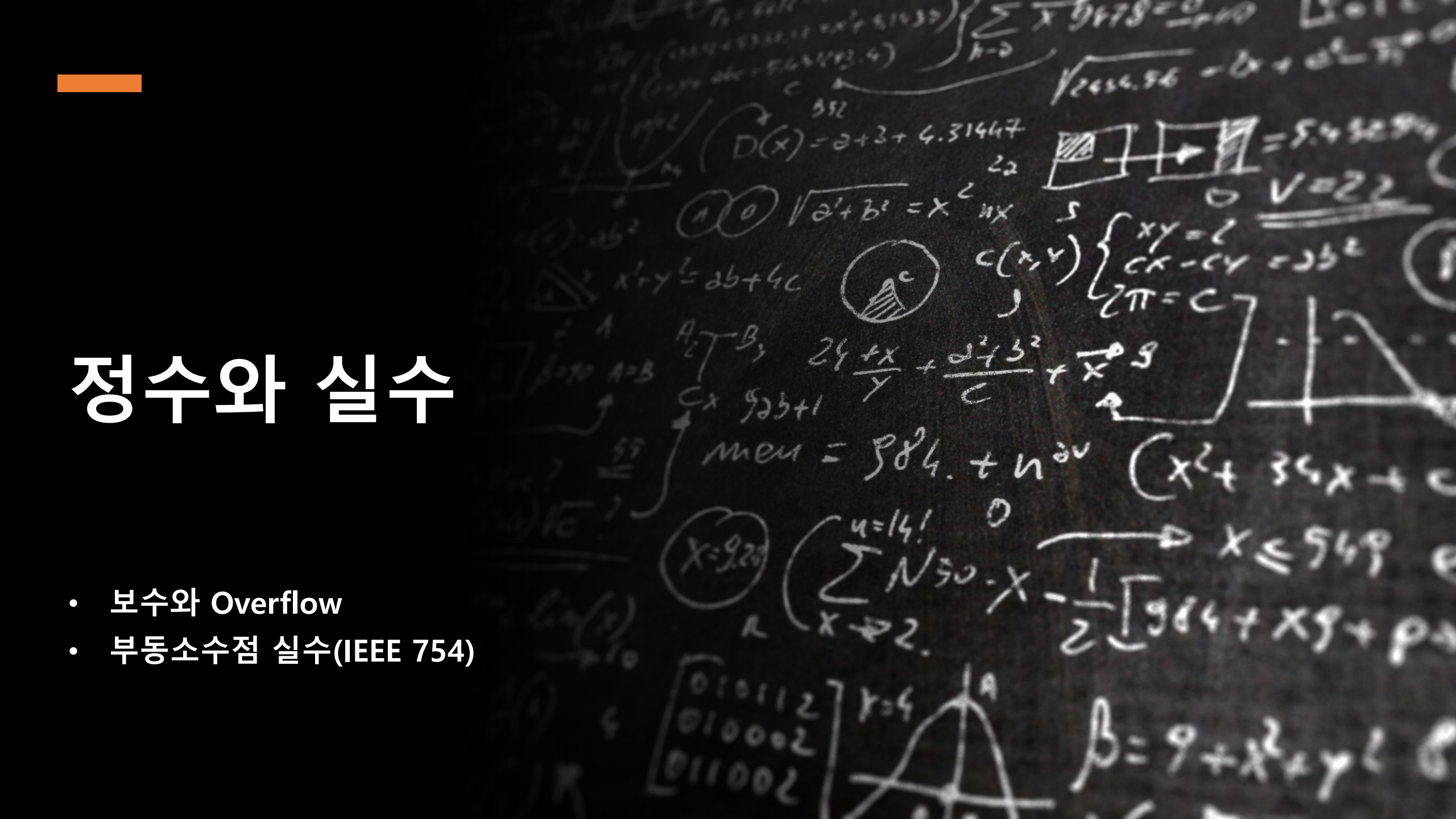
정수와 실수

- 보수와 Overflow
- 부동소수점 실수(IEEE 754)



정수

정수 리터럴

➔ 정수를 표현하는 방법

➔소수점 또는 10의 거듭제곱 없이 숫자문자로만 표현된 숫자

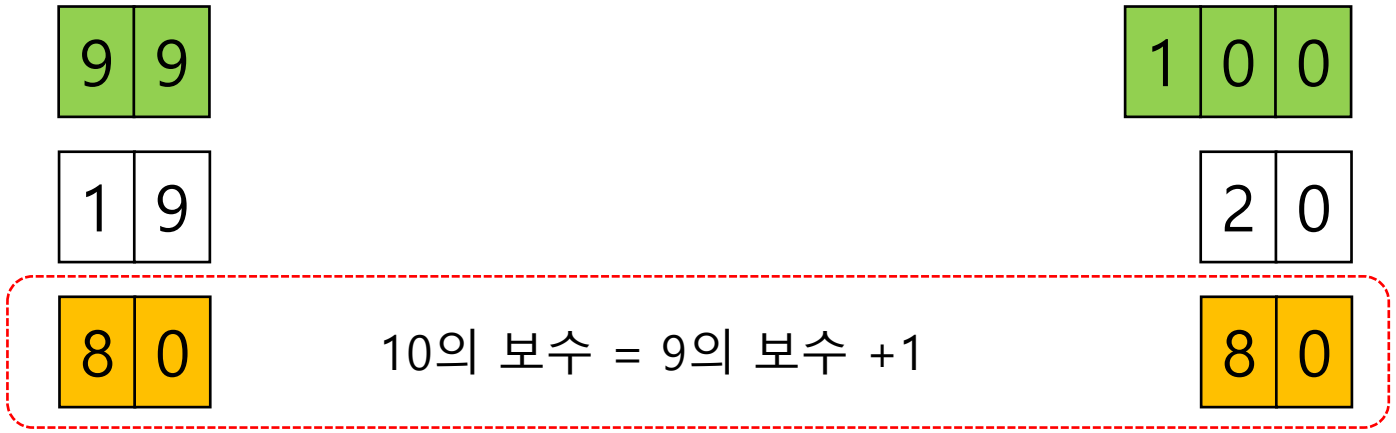
즉 소수부분이 없는 숫자 (예: 134)

보수: 특정한 수를 만들기 위한 보충해야하는 수

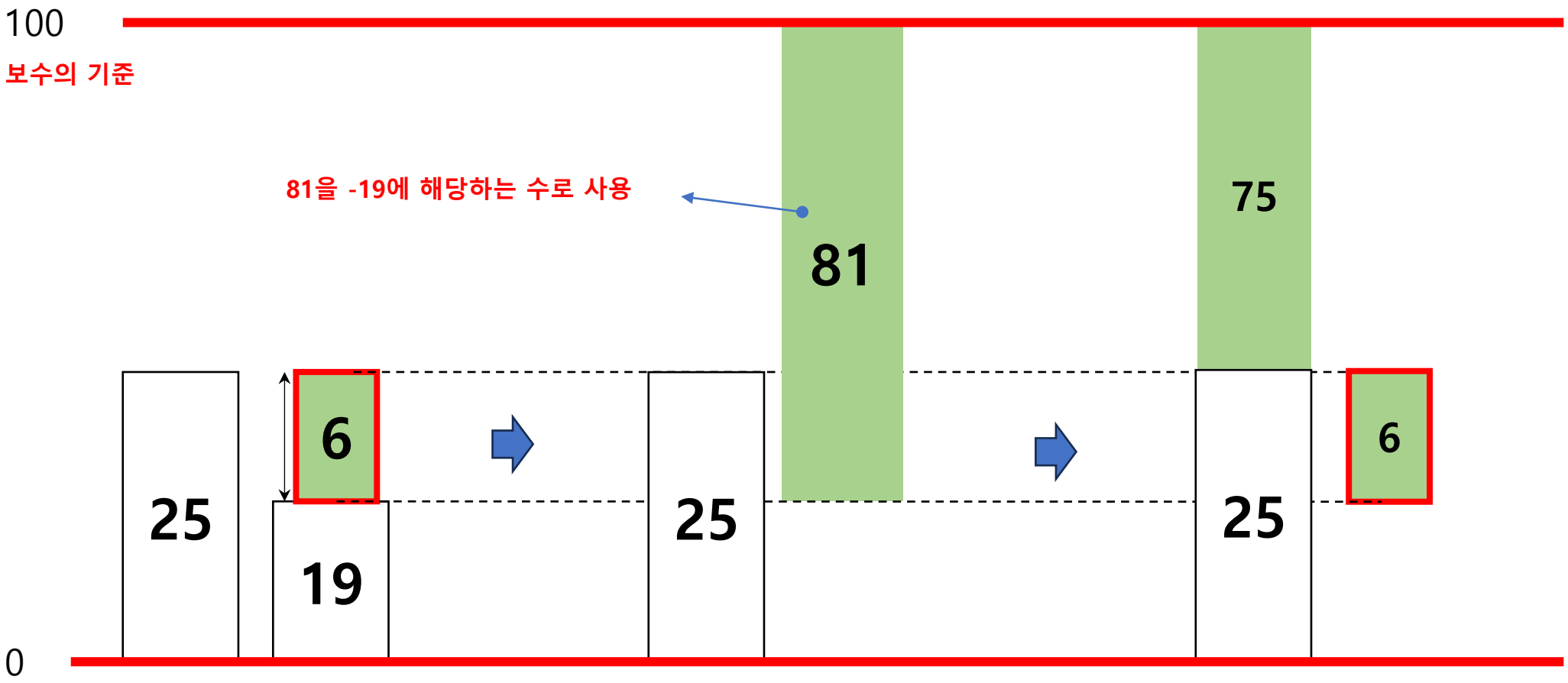
[10 진수]

19에 대한 9의 보수 (19에 대한 99의 보수)
➔ 99을 만들기 위해 19에 보충해주어야 하는 수
➔ 80

19에 대한 10의 보수 (19에 대한 100의 보수)
➔ 100을 만들기 위해 19에 보충해주어야 하는 수
➔ 81



빼기의 결과를 더하기를 이용하여 얻기 위해 보수사용



25-19

25+81

(25+75)+6

=100+6

보수의 기준(100) 삭제

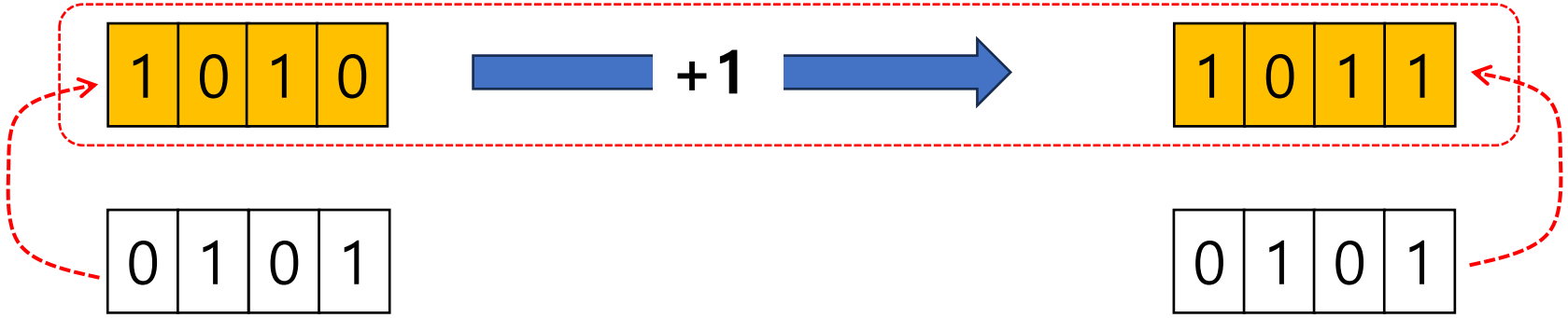
25-19=(25+81)= 106 → 6

[2 진수]

0101에 대한 1의 보수
(0101에 대한 1111의 보수)
➔ 1111을 만들기 위해 0101에 보충해주어야 하는 수
➔ 1010

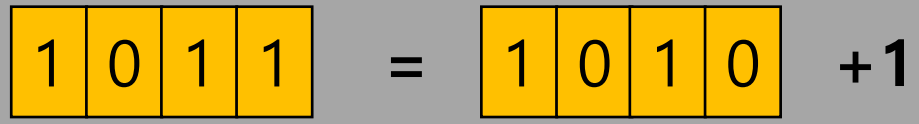
0101에 대한 2의 보수
(0101에 대한 10000의 보수)
➔ 10000을 만들기 위해 0101에 보충해주어야 하는 수
➔ 1011

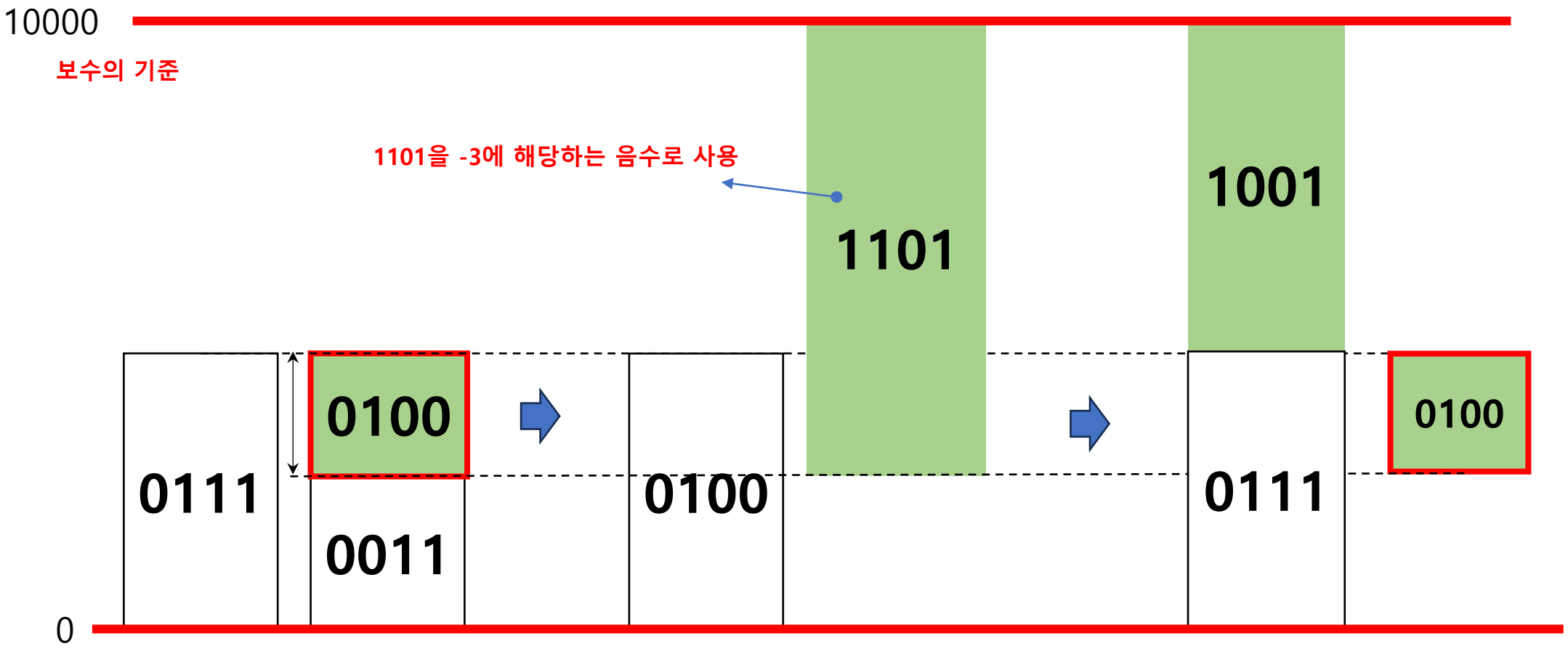
0101에 대한 1의 보수



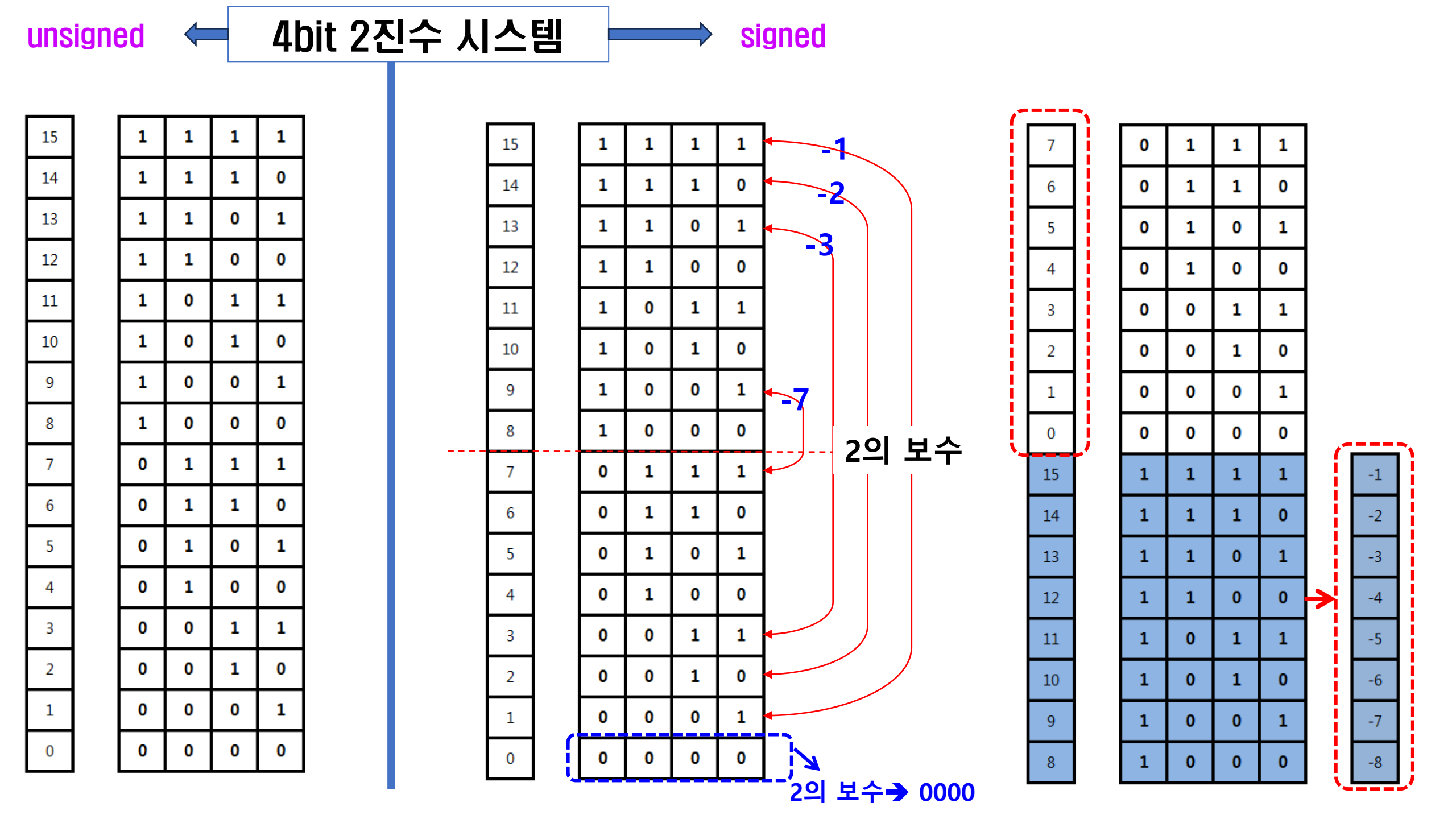
0101에 대한 2의 보수

2의 보수 = 1의 보수 +1





0111- 0011=(0111+1101)= 10100 → 0100


unsigned
4bit 2진수 시스템
signed
15 1 1 1 1
14 1 1 1 0
13 1 1 0 1
12 1 1 0 0
11 1 0 1 1
10 1 0 1 0
9 1 0 0 1
8 1 0 0 0
7 0 1 1 1
6 0 1 1 0
5 0 1 0 1
4 0 1 0 0
3 0 0 1 1
2 0 0 1 0
1 0 0 0 1
0 0 0 0 0
-1
-2
-3
-7
2의 보수
2의 보수➔ 0000
7 0 1 1 1
6 0 1 1 0
5 0 1 0 1
4 0 1 0 0
3 0 0 1 1
2 0 0 1 0
1 0 0 0 1
0 0 0 0 0
15 1 1 1 1
14 1 1 1 0
13 1 1 0 1
12 1 1 0 0
11 1 0 1 1
10 1 0 1 0
9 1 0 0 1
8 1 0 0 0
-1
-2
-3
-4
-5
-6
-7
-8

2의 보수 표현 (4bit)

7	0	1	1	1
6	0	1	1	0
5	0	1	0	1
4	0	1	0	0
3	0	0	1	1
2	0	0	1	0
1	0	0	0	1
0	0	0	0	0
-1	1	1	1	1
-2	1	1	1	0
-3	1	1	0	1
-4	1	1	0	0
-5	1	0	1	1
-6	1	0	1	0
-7	1	0	0	1
-8	1	0	0	0

2의 보수

2 – 4

=2 + (-4)

=0010+1100

=1110

= -2

4bit 2진수 시스템

unsigned에는 2의 보수를 반환하는 단항연산자 – 를 사용할 수 없다!!!

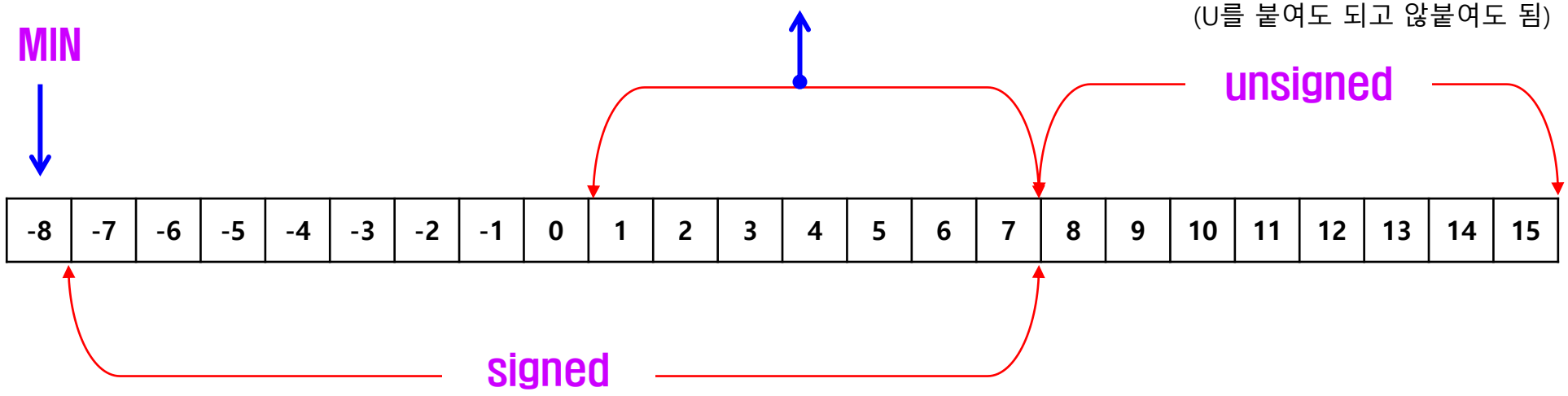
1U, 7U

숫자뒤에 U를 붙여 unsigned라는 것을 표시

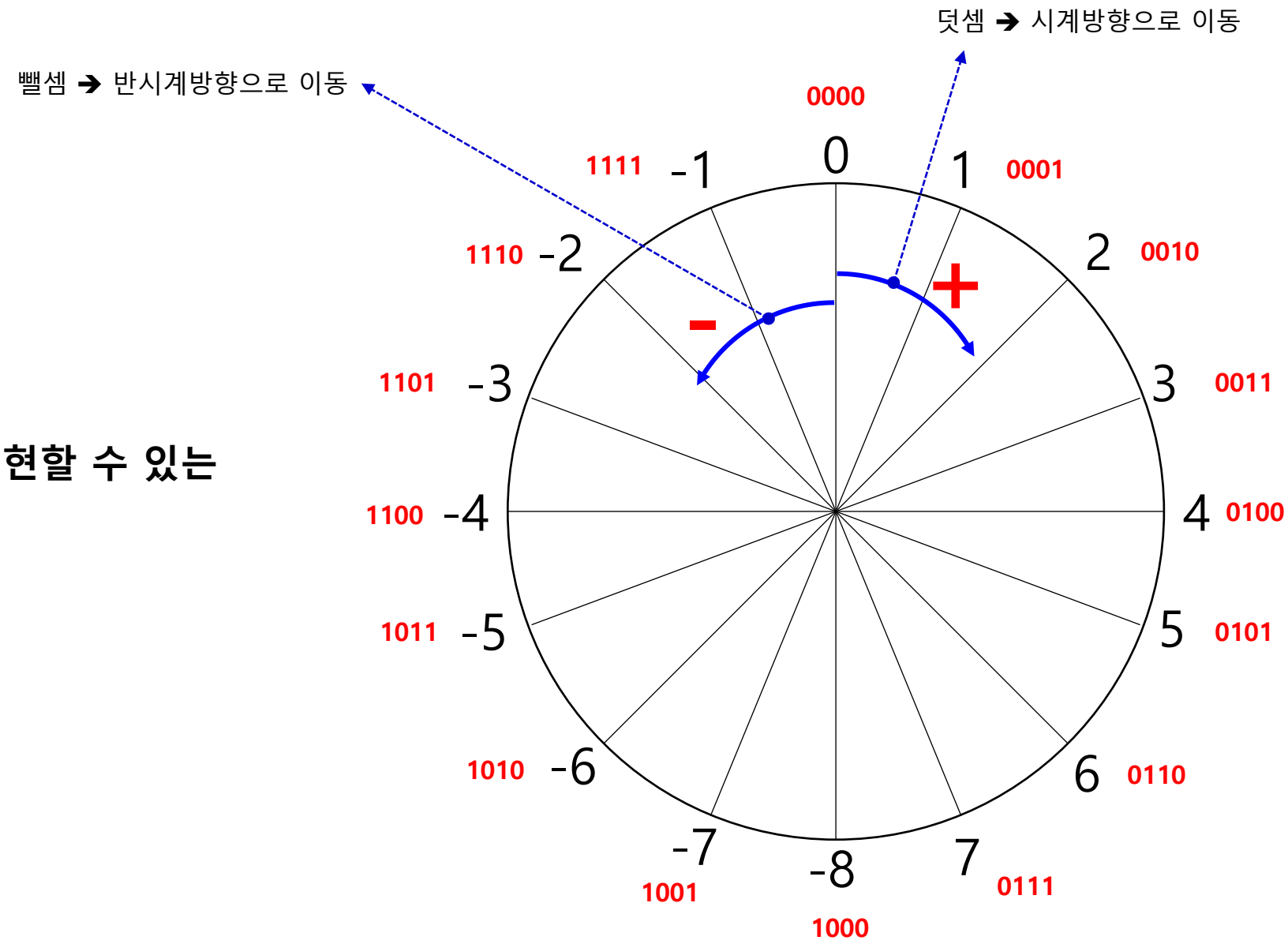
8==8U, 13==13U

다른 표시를하지 않아도 unsigned로 인식

(U를 붙여도 되고 않붙여도 됨)



4bit 2진수 시스템에서 덧셈과 뺄셈



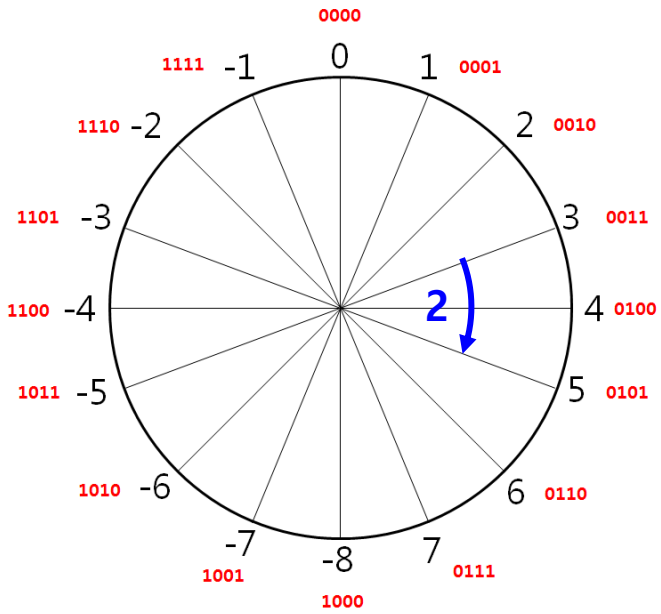
Overflow

➔ 계산 결과가 4비트로 표현할 수 있는

숫자 범위를 벗어나는 것

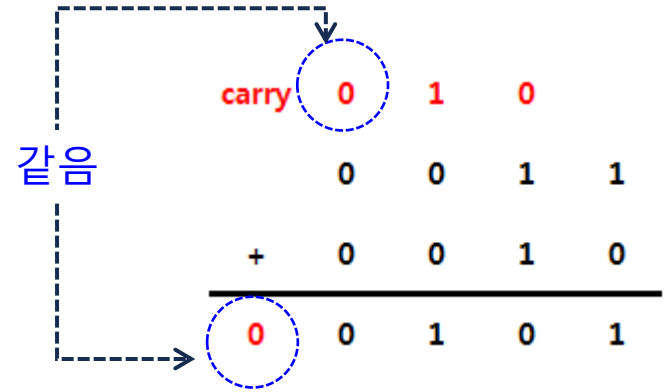
3+2 = 5

더한 결과 5가 4비트로 표현할 수 있는 숫자 범위(-8~+7)안에 있음
➔ No Overflow



Overflow 확인 방법
➔ 마지막 자리로 올라오는 올림수(carry)와 마지막 자리에서 다음 자리로 올려주는 올림수가 같으면 Overflow가 아니다.!!!

No Overflow !!

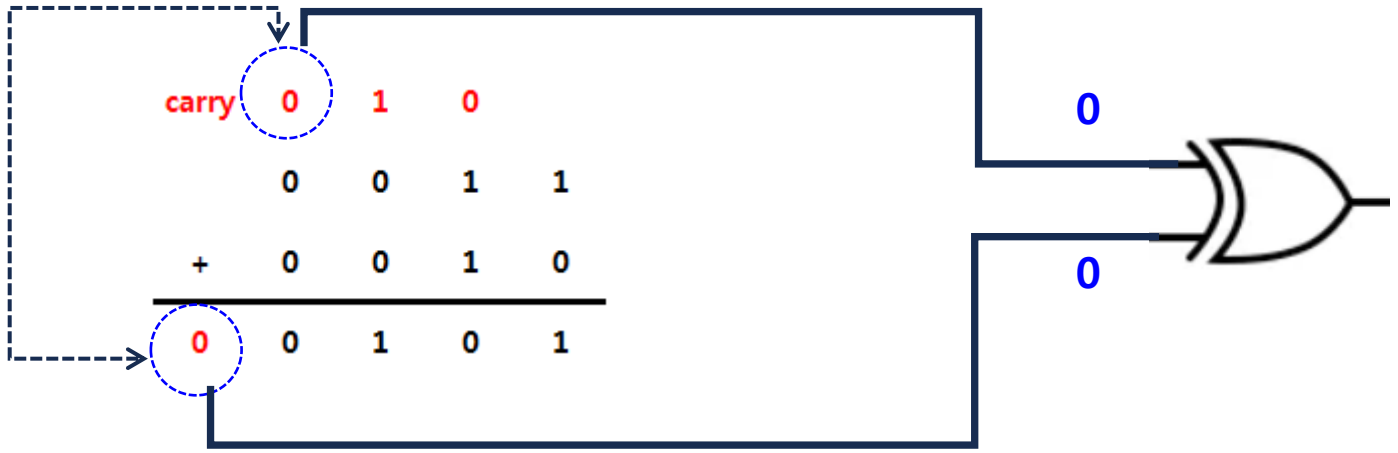


Overflow Check with XOR Gate

Overflow 확인은 논리회로의 XOR 게이트를 이용해서 할 수 있다.

2 input XOR gate

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0



같음

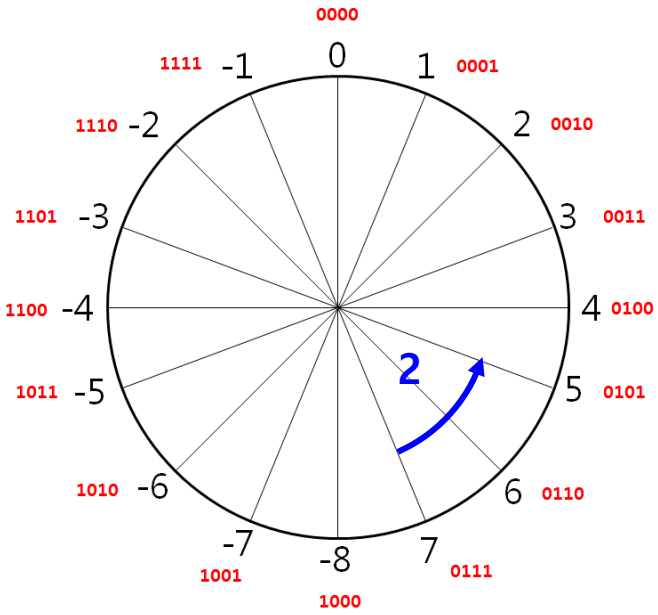
No Overflow !!

0 No Overflow !!

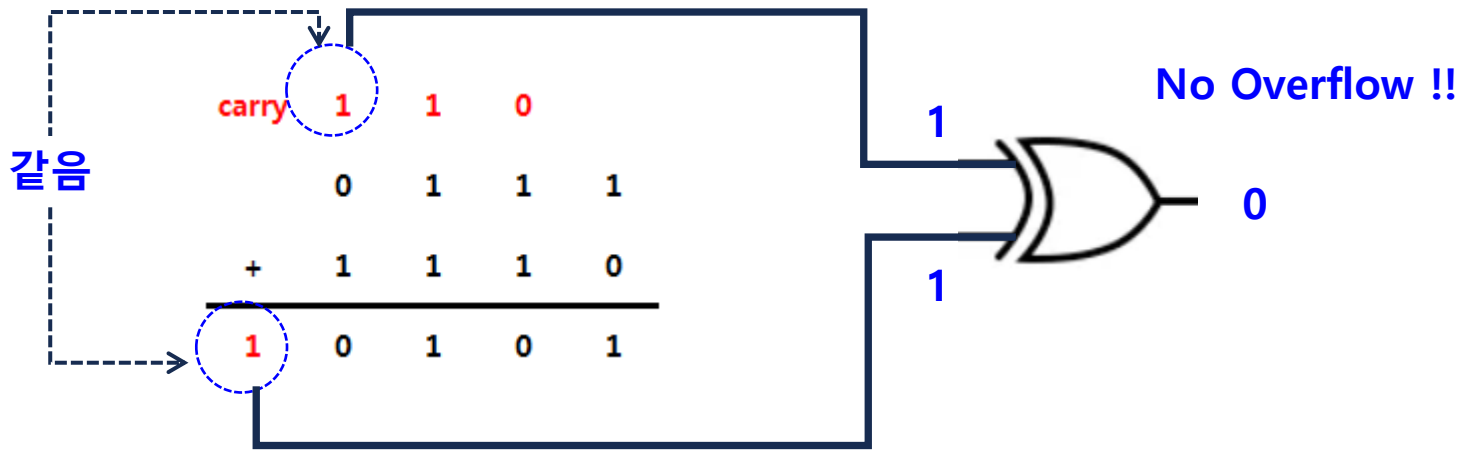
XOR 게이트의 출력 → 0이면 No overflow

XOR 게이트의 출력 → 1이면 overflow

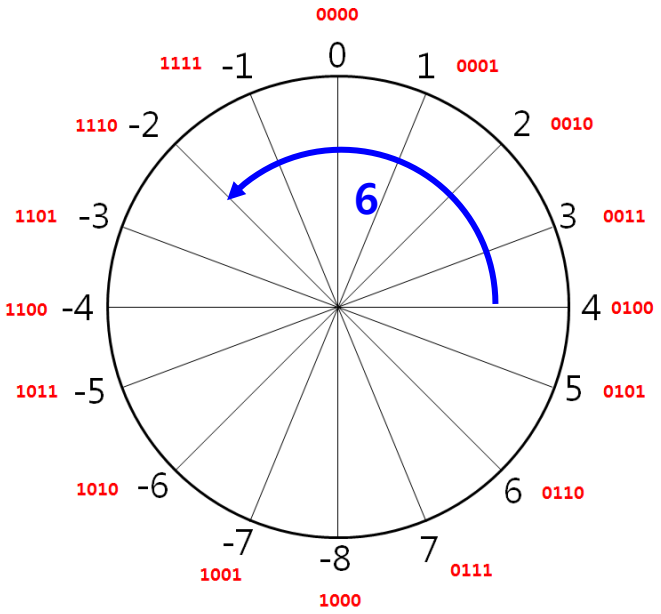
7-2=5



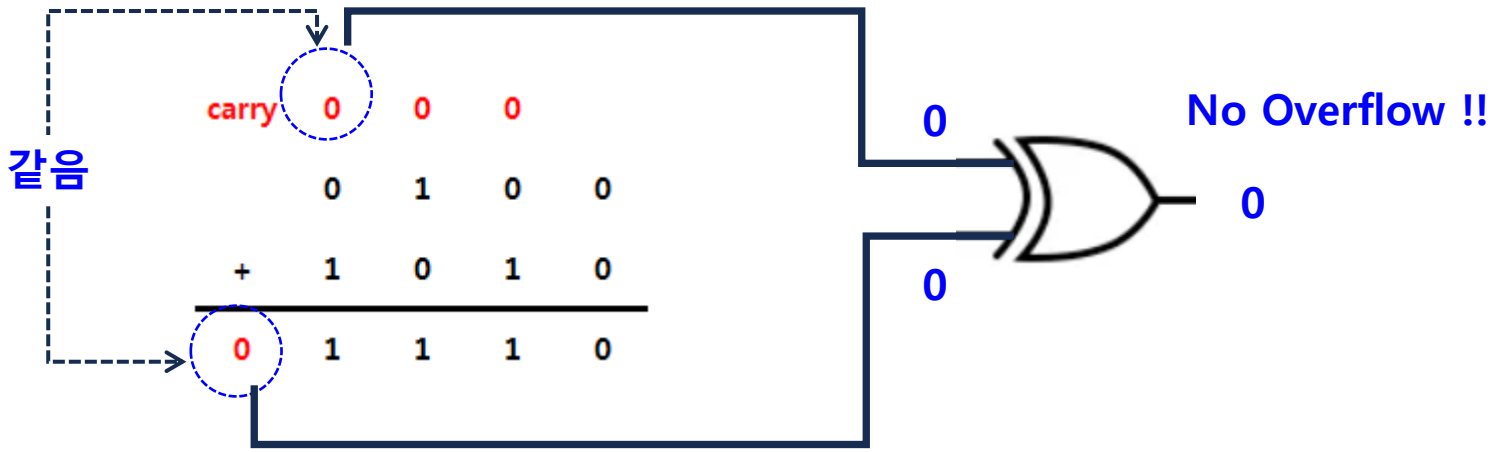
No Overflow !!



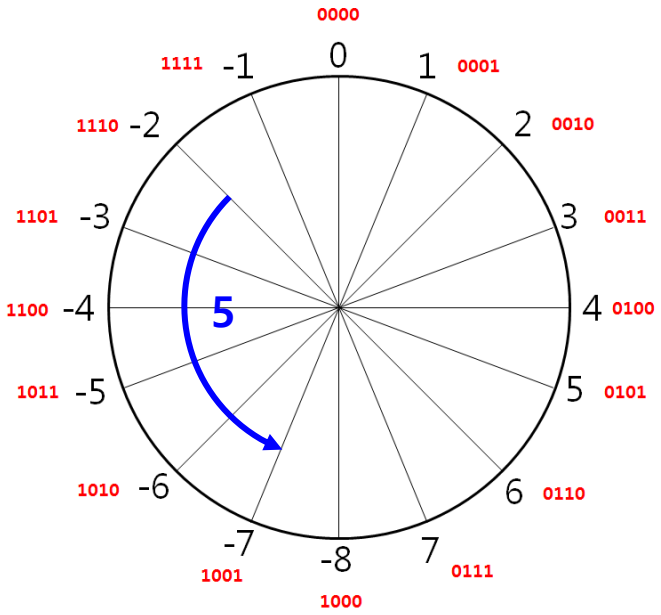
4-6=-2



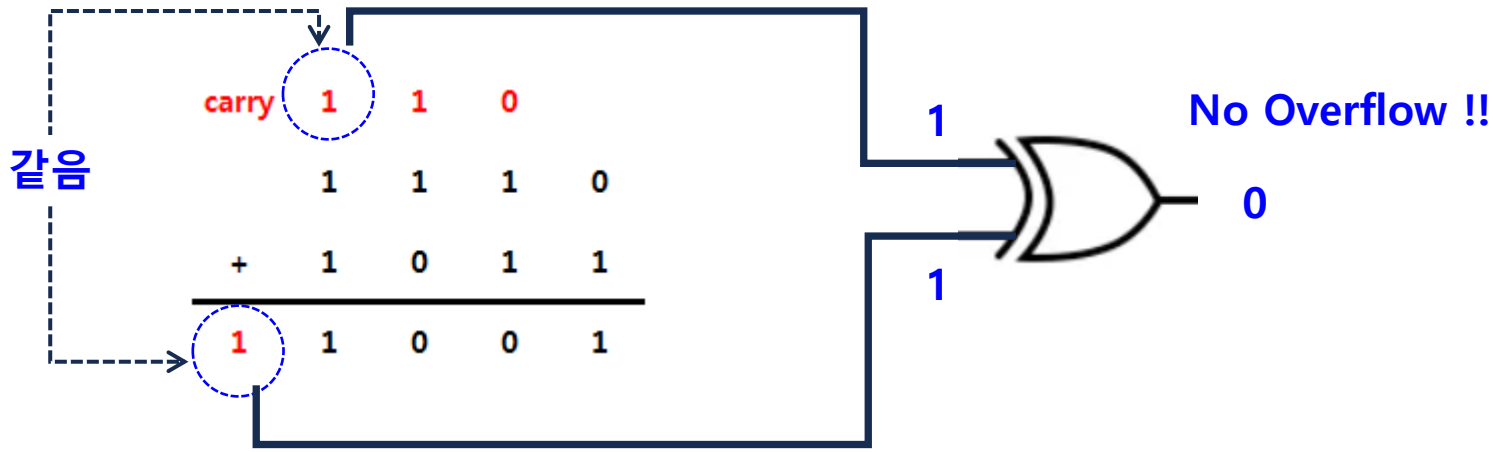
No Overflow !!



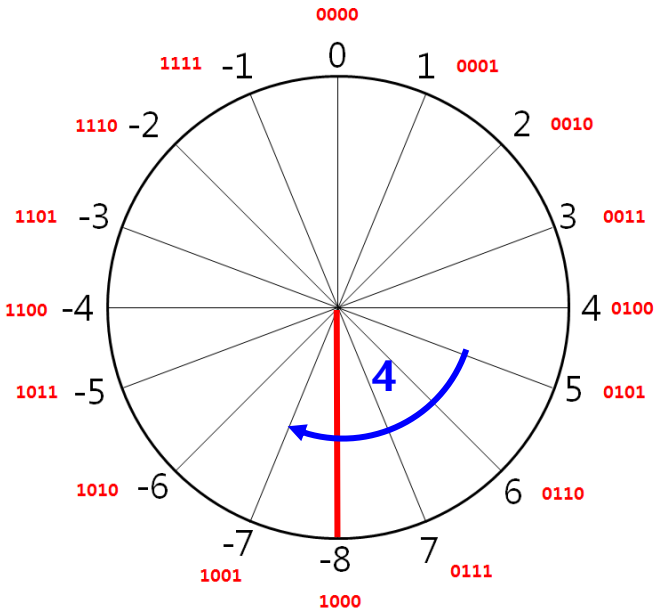
-2-5=-7



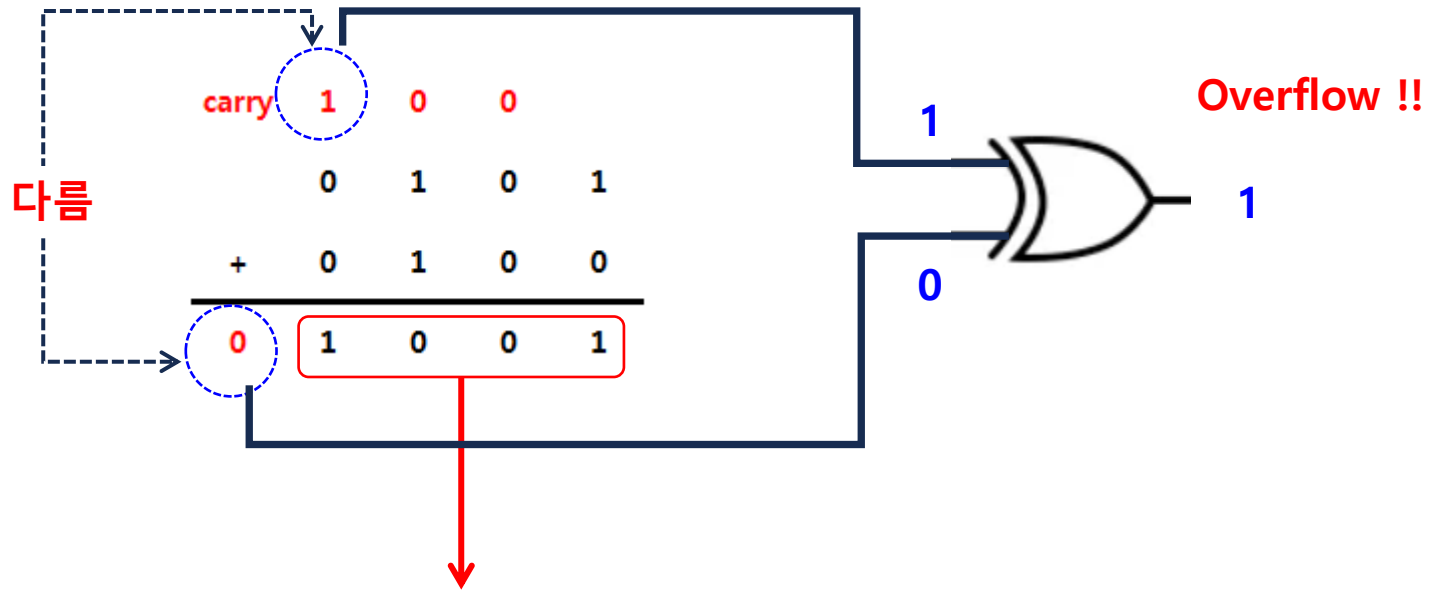
No Overflow !!



5+4=-7

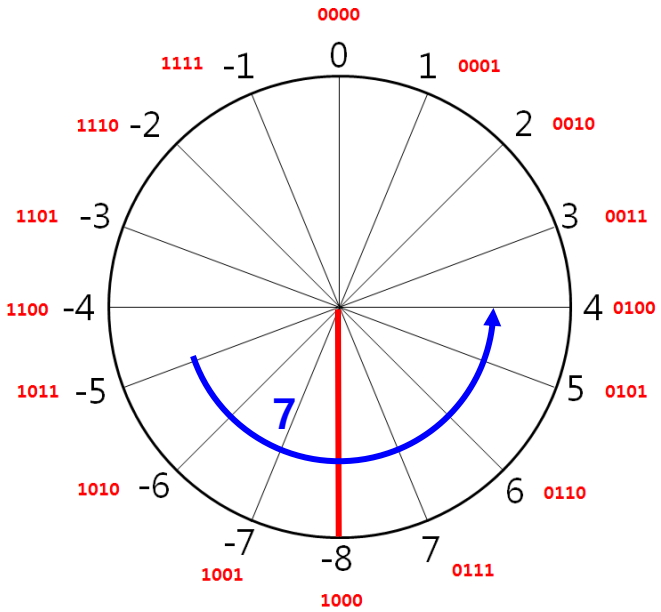


Overflow !!

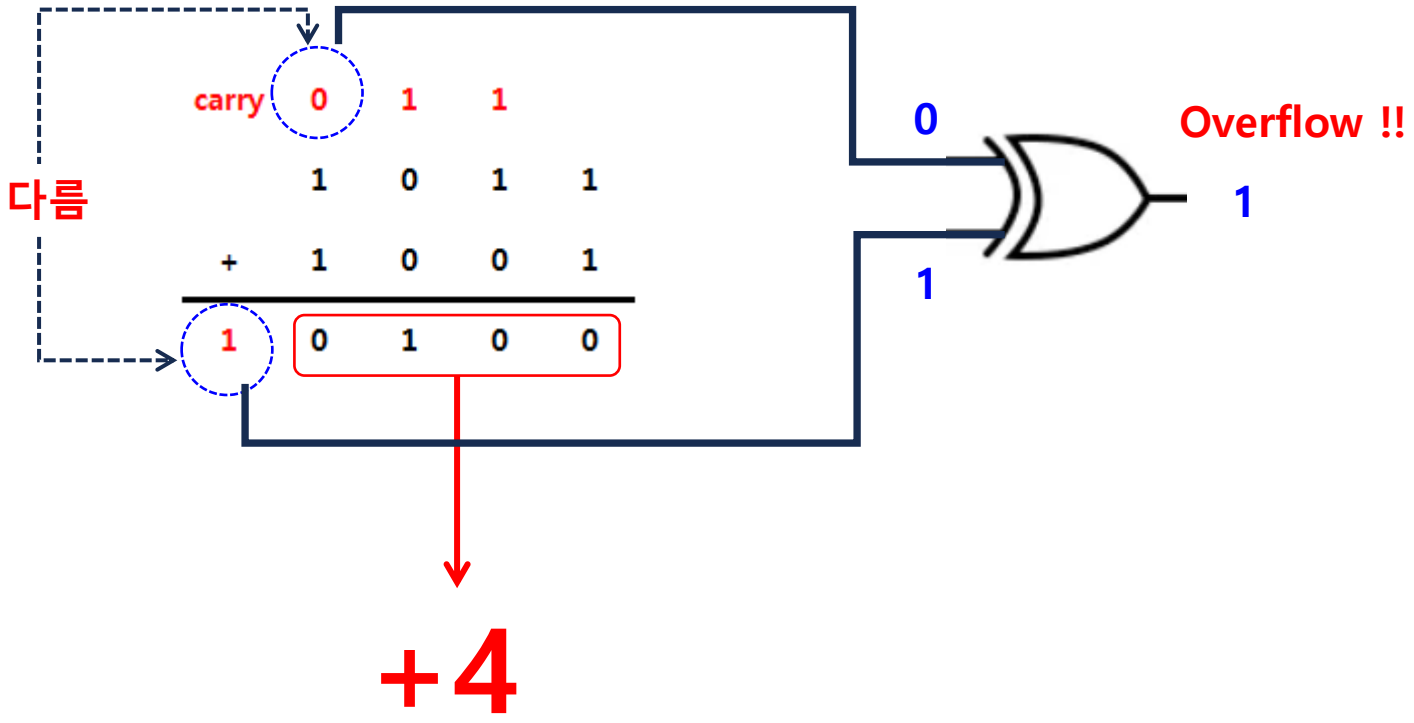


-7

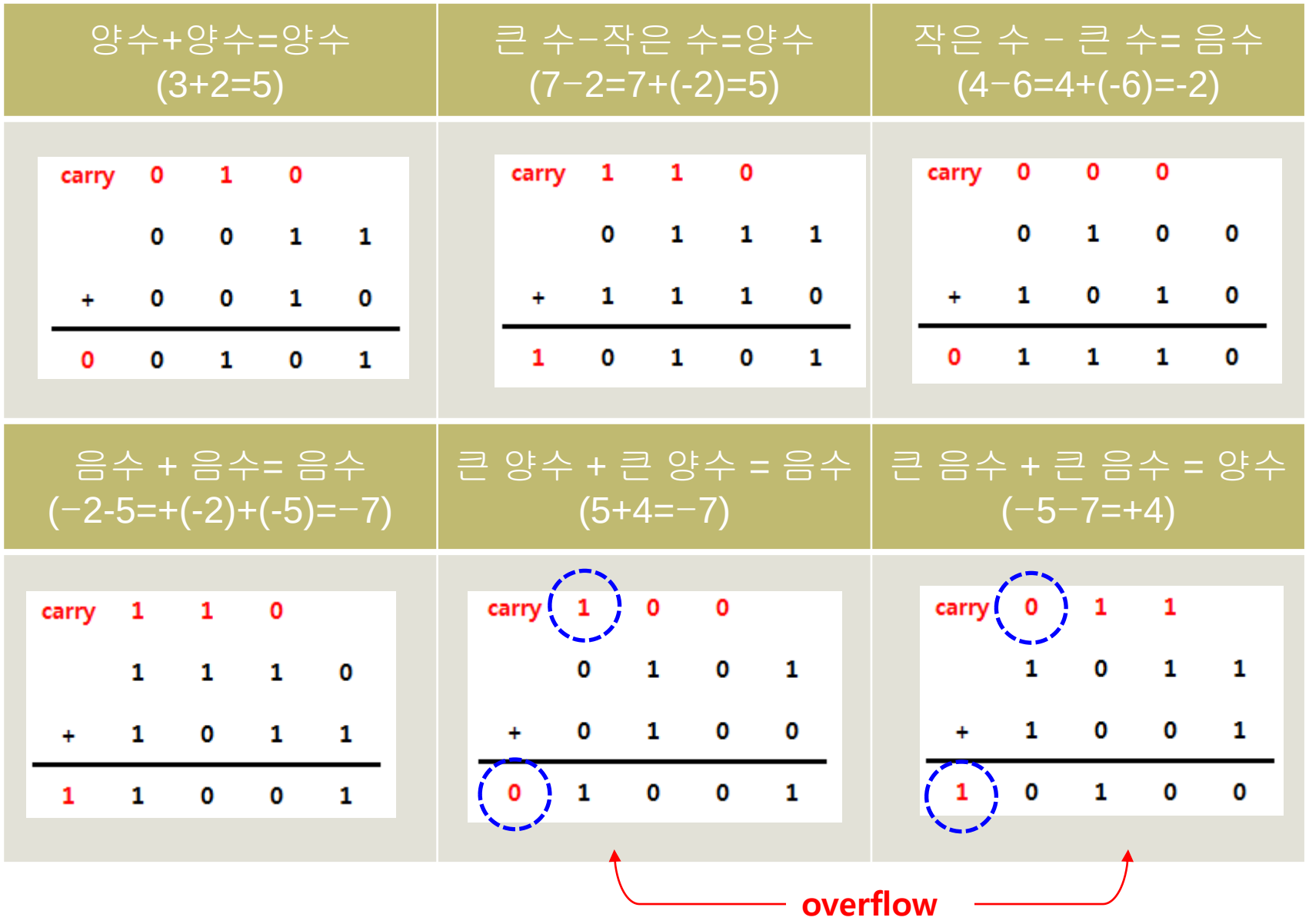
-5-7=+4



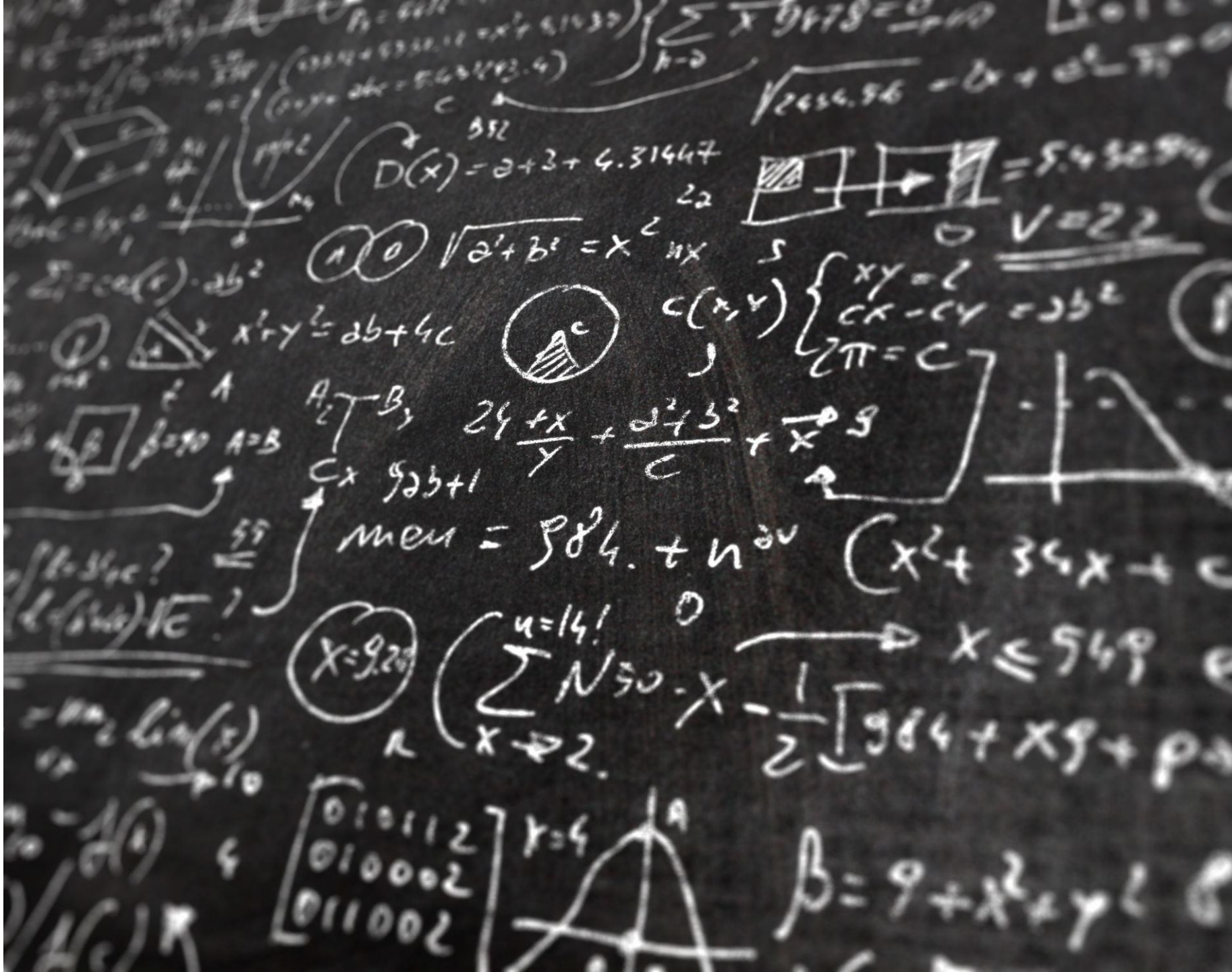
Overflow !!



+4


양수+양수=양수
(3+2=5)
carry 0 1 0
0 0 1 1
+ 0 0 1 0
0 0 0 1 0 1
큰 수-작은 수=양수
(7-2=7+(-2)=5)
carry 1 1 0
0 1 1 1
+ 1 1 1 0
1 0 1 0 1
작은 수 - 큰 수= 음수
(4-6=4+(-6)=-2)
carry 0 0 0
0 1 0 0
+ 1 0 1 0
0 1 1 1 0
음수 + 음수= 음수
(-2-5=+(-2)+(-5)=-7)
carry 1 1 0
1 1 1 0
+ 1 0 1 1
1 1 0 0 1
큰 양수 + 큰 양수 = 음수
(5+4=-7)
carry 1 0 0
0 1 0 1
+ 0 1 0 0
0 1 0 0 1
큰 음수 + 큰 음수 = 양수
(-5-7=+4)
carry 0 1 1
1 0 1 1
+ 1 0 0 1
1 0 1 0 0
overflow

부동소수점
실수



실수 리터럴

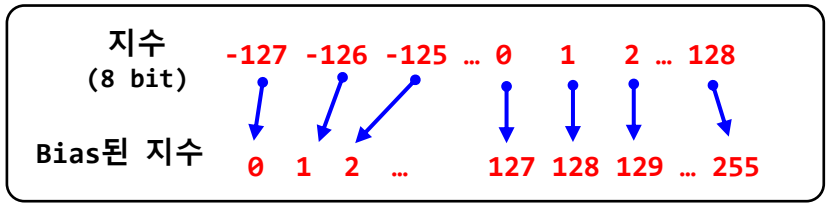
➔ 실수를 표현하는 방법

➔소수점으로 표현된 숫자 (예: 13.4 / 1. / .5)

➔10의 거듭제곱으로 표기하는 과학적 표기법이 적용된 숫자

(예: 123E-1 / 1e+3) 알파벳 e 또는 E ,대소문자 구분없음

IEEE 754 부동소수점 표현



정수부분이 1이 되도록 이진수 지수형식으로 표현

4바이트 부동소수점 수로 컴파일하라는 의미

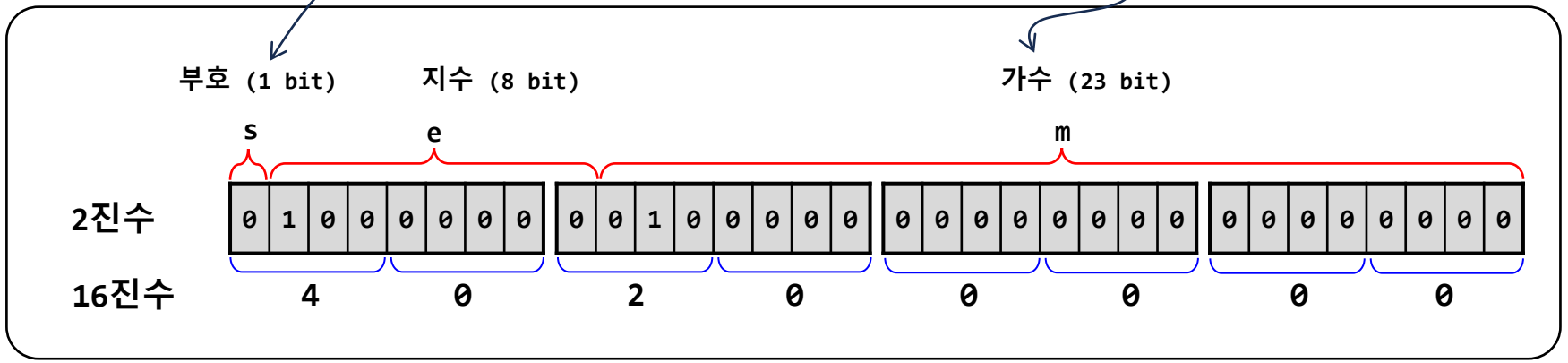
1+127 (지수에 bias 127을 더함)

2.5f (십진수) ➡ 10.1 (이진수) ➡ (1.01) * 2^{1} (정규화 (이진수 지수형식)) ➡ (-1)^{0} * (1.01) * 2^{128} (Bias된 지수)

부호 s (1 bit) : 0

가수 m (23 bit) : 010 0000 0000 0000 0000 0000

지수 e (8 bit) : 1000 0000



메모리에는 Little Endian 으로 저장됨.

메모리에 저장된 값 : 00 00 20 40