수 체계와 데이터 표현

❍ 진법

❖ 진법

- 사용할 수 있는 숫자의 개수와 위치 값을 정의해주는 수 체계
- 사용할 수 있는 숫자의 개수는 해당 진법과 같음
- 사용할 수 있는 숫자는 0에서 시작해서 해당 진법의 수보다 1 적은 수까지

❖ 사용할 수 있는 숫자

- 10진법 : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- 2진법 : 0과 1
- 8진법 : 0, 1, 2, 3, 4, 5, 6, 7
- 16진법 : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

❍ 자리 값

- ❑ 모든 수의 각 숫자는 자리 값을 가지고 있음

10진수 $365 = 3\times10^2 + 6\times10^1 + 5\times10^0$

2진수 $1011 = 1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0$

8진수 $567 = 5\times8^2 + 6\times8^1 + 7\times8^0$

16진수 $AB1 = A\times16^2 + B\times16^1 + 1\times16^0$

10진수 $0.258 = 2\times10^{-1} + 5\times10^{-2} + 8\times10^{-3}$

2진수 $0.101 = 1\times2^{-1} + 0\times2^{-2} + 1\times2^{-3}$

8진수 $0.34 = 3\times8^{-1} + 4\times8^{-2}$

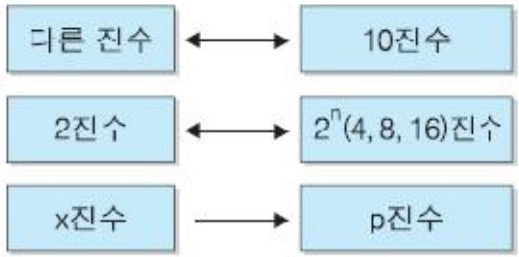
16진수 $0.1\mathrm{A} = 1\times16^{-1} + A\times16^{-2}$

❑ 진수 간의 값 비교

[표 2-1] 진수 간의 값 비교

10진수	2진수	8진수	16진수
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12

❍진수 변환 형태

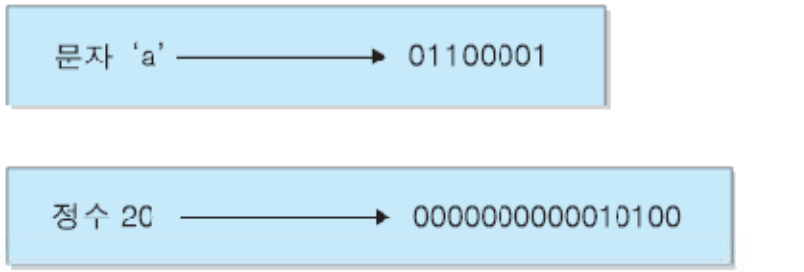


[그림 2-1] 진수 변환 형태

❍ 다른 진수의 10진수로의 변환

$$27.42_8 = 2\times8^1 + 7\times8^0 + 4\times8^{-1} + 2\times8^{-2}$$
$$= 16 + 7 + 4\times\frac{1}{8} + 2\times\frac{1}{8^2}$$
$$= 16 + 7 + 0.5 + 0.03125$$
$$= 23.53125_{10}$$
$$AF.8_{16} = A\times16^1 + F\times16^0 + 8\times16^{-1}$$
$$= 10\times16^1 + 15\times16^0 + 8\times\frac{1}{16^1}$$
$$= 160 + 15 + 0.5$$
$$= 175.5_{10}$$

❍ 컴퓨터 내부에서의 문자 'a'와 정수 20의 표현



[그림 2-9] 컴퓨터 내부에서의 문자 'a'와 정수 20의 표현

❖ 오늘날의 컴퓨터는 문자, 정수, 실수, 그림, 소리, 동영상 등의 모든 정보를 2진수 형식으로 표현한다. 초기의 컴퓨터에서는 10진수를 사용하여 정보를 표현했으나 고가의 장치, 연산 처리 능력의 저하 그리고 비안정성 등의 문제가 있었다. 이런 문제점을 해결하기 위해 안정성이 뛰어난 2진수 형식으로 정보를 표현하게 되었다. 0 또는 1의 2진수 개념은 모든 전기적인 장치의 on/off와 딱 맞는 개념이다.

❖ byte = 8 bit

❖ 하나의 0 또는 1을 비트(bit, binary digit)라 하는데 컴퓨터에서 정보를 나타내는 최소 단위이며, 비트 8개를 묶어 바이트(byte)라 한다.

❖ 1비트로 나타낼 수 있는 정보는 0 또는 1의 두 가지가 되고, 2비트로 나타낼 수 있는 정보는 00, 01, 10, 11의 네 가지가 된다. 그리고 3비트로 나타낼 수 있는 정보는 000, 0001, 010, 011, 100, 101, 110, 111의 여덟 가지가 된다.

[표 2-5] 비트 표현

1비트	2비트	3비트
0	00	000
1	01	001
	10	010
	11	011
		100
		101
		110
		111

❖ 일반화하면 n비트로 가지의 정보를 표현할 수 있게 된다. 이것은 다음 절의 중요한 개념이 된다. 컴퓨터에서 문자를 어떻게 표현하는지에 대해 살펴보자.

❍상식 : 숫자의 단위 표현

❍소수점 이하 표현

- ❑ 1/10 = deci
- ❑ 1/100 = centi

1cm = 1m * 1/100

❍컴퓨터에서의 단위

- 1 bit = { 0, 1 }
- 1 byte = 8 bits (2^8 = 256 가지 표현)
  - ❖ 영문글자 ascii 코드를 표현하는데 필요한 크기 = 8bit =byte
- 1 Kbyte = 2^10 bytes = 1,024 bytes (= 1KB)
- 1 Mbyte = 1,024 Kbytes = 2^20 bytes = 1,024*1024 bytes
- 1 Gbyte = 1,024 Mbytes = 2^30 bytes (= 1GB)
- 1 Tbyte = 1,024 Gbytes = 2^40 bytes (= 1TB)

❍데이터 타입과 표현가능한 수의 범위

종류	크기	표현범위
byte	1byte	-128~127
short	2byte	-3만2천~3만2천
int	4byte	-21억~21억
long	8byte	-900경~900경

- 엑사(exa): 10^{18}(100경), 기호: E
  페타(peta): 10^{15}(1000조), 기호: P
  테라(tera): 10^{12}(1조), 기호: T
  기가(giga): 10^{9}(10억), 기호: G
  메가(mega): 10^{6}(100만), 기호: M
  킬로(kilo): 10^{3}(1000), 기호: k
  헥토(hecto): 10^{2}(100), 기호: h
  데카(deca): 10^{1}(10), 기호: da
  데시(deci): 10^{-1}(10분의 1), 기호: d
  센티(centi): 10^{-2}(100분의 1), 기호: c
  밀리(mili): 10^{-3}(1000분의 1), 기호: m
  마이크로(micro): 10^{-6}(100만분의 1), 기호: μ(그리스 문자 뮤)
  나노(nano): 10^{-9}(10억분의 1), 기호: n
  피코(pico): 10^{-12}(1조분의 1), 기호: p
  펨토(femto): 10^{-15}(1000조분의 1), 기호: f
  아토(ato): 10^{-18}(100경분의 1), 기호: a

❑ 컴퓨터에서의 숫자 표현

❍ 예 : 컴퓨터의 표현

- ❑ 32 bit 컴퓨터 ?
- ❑ 64 bit 컴퓨터 ?

제조사	인텔
제품명	코어 i5-2500
공정기술	32nm High-K
소켓방식	LGA1155
작동속도	3.30GHz / MAX 3.7GHz
코어형태	Quad-Core (4T / 4C)
캐쉬용량	L1 Cache 4 x 32KB / 2 x 32KB L2 Cache 4 x 256KB Last Level Cache 6MB
내장 메모리 컨트롤러	듀얼채널 DDR3-1333, 최대 16GB
그래픽 코어	인텔 그래픽스 HD 2000 (EU 6개)
코어 클럭	800MHz / Max 1,100MHz
그래픽카드 지원	단일 x16배속 or 듀얼 x8 / x8배속
운영모드	32 / 64bit
TDP	95W
특징	Intel HD Graphices 2000, Turbo Boost 2.0, SSE4.2 Quick Sync Video, AES-IN, Intel AVX, Last Level Cache Virtualization(VT-X), IMC

❍ 하드디스크


빠르다!
강력하다!
유일하다!
Barracuda® Green 2TB
• New Smart Align Technology 씨게이트가 개발한 스마트 정렬 신기술로 어드밴스 포맷 섹터 전환이 수월하고 빠르게 사용 가능합니다.
• SATA 6Gb/s 기존 3Gb/s에 비해 2배 더 빠른 입출력을 보장합니다.
• 2TB / 64MB Cache 현존 최고 용량의 저장공간과 캐쉬 메모리를 탑재했습니다.
• Green Technology 2.1dB의 저소음 동작으로 하드디스크의 소음에서 해방되세요.
• 글로벌 3년 무상 A/S 세계 어디에서도 3년 동안 언제든지 무상으로 A/S를 받을 수 있습니다.
Specification
• 용량(GB) 2TB / 1.5TB
• 인터페이스 SATA 6Gb/s NCQ
• 캐시(MB) 64 MB
• 지속 데이터 전송 속도 최대 144MB/s
• 회전 속도 5900 RPM
• 유휴 소음(bel) 2.1 bel
• 탐색 소음(bel) 2.3 bel

ASCII

- 각 문자를 7비트로 표현
- 총 128(2^7)개의 문자를 표현

1	0 0 0 0 0 0 0
2	0 0 0 0 0 0 1
3	0 0 0 0 0 1 0
4	0 0 0 0 0 1 1
⋮	⋮
127	1 1 1 1 1 1 0
128	1 1 1 1 1 1 1

[그림 2-10] 7비트로 128가지를 달리 표현할 수 있다.

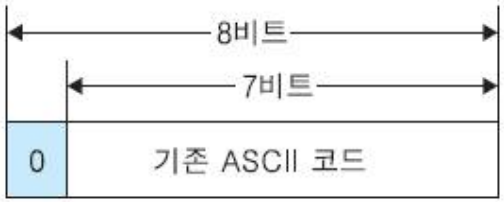
❖ ASCII 문자 코드

[표 2-6] ASCII 문자 코드

0000000	NUL	0100000	Space	1000000	@	1100000	`
0000001	SOH (Start of Heading)	0100001	!	1000001	A	1100001	a
0000010	STX (Start of Text)	0100010	"	1000010	B	1100010	b
0000011	ETX (End of Text)	0100011	#	1000011	C	1100011	c
0000100	EOT (End of Transmission)	0100100	$	1000100	D	1100100	d
0000101	ENQ (Enquiry)	0100101	%	1000101	E	1100101	e
0000110	ACK (Acknowledge)	0100110	&	1000110	F	1100110	f
0000111	BEL (Bell)	0100111	'	1000111	G	1100111	g
0001000	BS (Backspace)	0101000	(	1001000	H	1101000	h
0001001	HT (Horizontal Tabulation)	0101001	)	1001001	I	1101001	i
0001010	LF (Line Feed)	0101010	*	1001010	J	1101010	j
0001011	VT (Vertical Tabulation)	0101011	+	1001011	K	1101011	k
0001100	FF (Form Feed)	0101100	,	1001100	L	1101100	l
0001101	CR (Carriage Return)	0101101	-	1001101	M	1101101	m
0001110	SO (Shift Out)	0101110	.	1001110	N	1101110	n
0001111	SI (Shift In)	0101111	/	1001111	O	1101111	o
0010000	DLE (Data Link Escape)	0110000	0	1010000	P	1110000	p
0010001	DC1 (Device Control 1)	0110001	1	1010001	Q	1110001	q
0010010	DC2 (Device Control 2)	0110010	2	1010010	R	1110010	r
0010011	DC3 (Device Control 3)	0110011	3	1010011	S	1110011	s
0010100	DC4 (Device Control 4)	0110100	4	1010100	T	1110100	t
0010101	NAK (Negative Acknowledge)	0110101	5	1010101	U	1110101	u
0010110	SYN (Synchronous Idle)	0110110	6	1010110	V	1110110	v
0010111	ETB (End of Transmission Block)	0110111	7	1010111	W	1110111	w
0011000	CAN (Cancel)	0111000	8	1011000	X	1111000	x
0011001	EM (End of Medium)	0111001	9	1011001	Y	1111001	y
0011010	SUB (Substitute)	0111010	:	1011010	Z	1111010	z
0011011	ESC (Escape)	0111011	;	1011011	[	1111011	{
0011100	FS (File Separator)	0111100	<	1011100	\	1111100	|
0011101	GS (Group Separator)	0111101	=	1011101	]	1111101	}
0011110	RS (Record Separator)	0111110	>	1011110	^	1111110	~
0011111	US (Unit Separator)	0111111	?	1011111	_	1111111	DEL

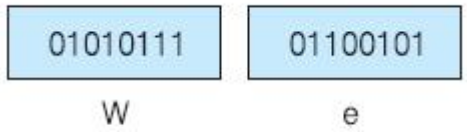
ASCII

- 확장(extended) ASCII
  - 256(2^5)개의 문자를 표현
  - 기존 7비트 ASCII 코드에는 가장 왼쪽에 0을 추가



[그림 2-11] 기존 ASCII의 확장 ASCII로의 표현

- ASCII로 표현한 "We"
  - ASCII는 각 나라별 언어는 표현이 불가능



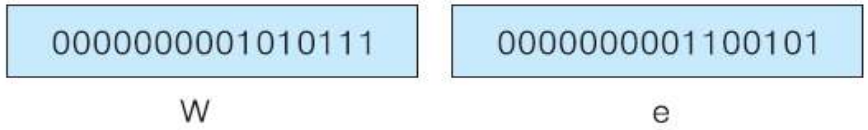
[그림 2-12] ASCII로 표현한 "We"

❍유니코드

- ❖ 각 나라별 언어를 모두 표현하기 위해 나온 코드 체계
- ❖ 문자를 16비트로 표현하므로 최대 65,536자를 표현
- ❖ 영문자, 숫자에 대한 유니코드
  (http://www.unicode.org/charts/PDF/U0000.pdf)

[그림 2-13] 영문자, 숫자에 대한 유니코드

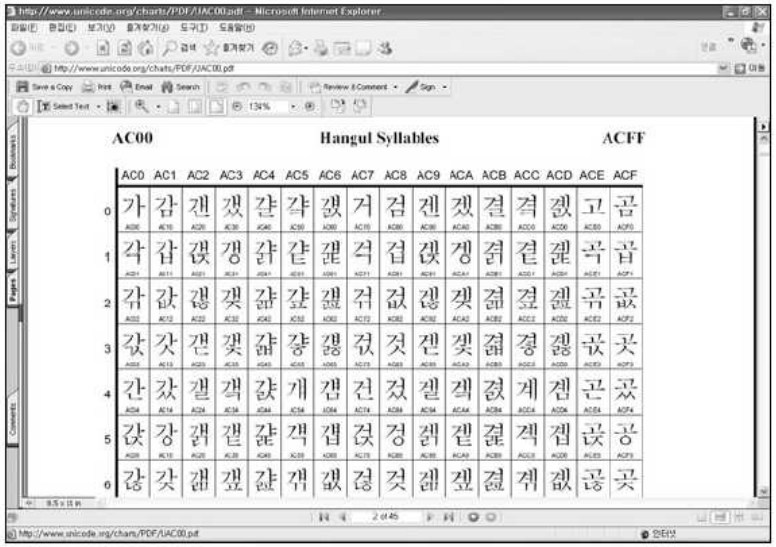
- ❖ 유니코드로 표현한 “We”



[그림 2-14] 유니코드로 표현한 “We”

❍유니코드

❖ 한글에 대한 유니코드
(http://www.unicode.org/charts/PDF/UAC00.pdf)



[그림 2-15] 한글에 대한 유니코드

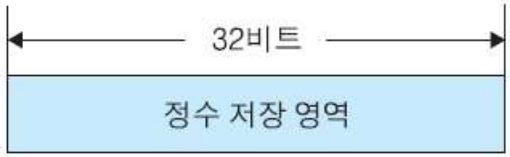
❖ 유니코드로 표현한 "한글"

1101010101011100	1010111000000000
한	글

[그림 2-16] 유니코드로 표현한 "한글"

❍정수 표현하기

❖ 컴퓨터의 기억 공간은 제한적이므로 한 정수를 나타낼 기억 영역도 제한적. 시스템에 따라 약간의 차이가 있지만, 대부분 32비트로 정수를 표현



[그림 2-27] 32비트 정수

❖ 2의 보수(2's complement) 표기법 : 컴퓨터에서 정수를 표현하는 방법

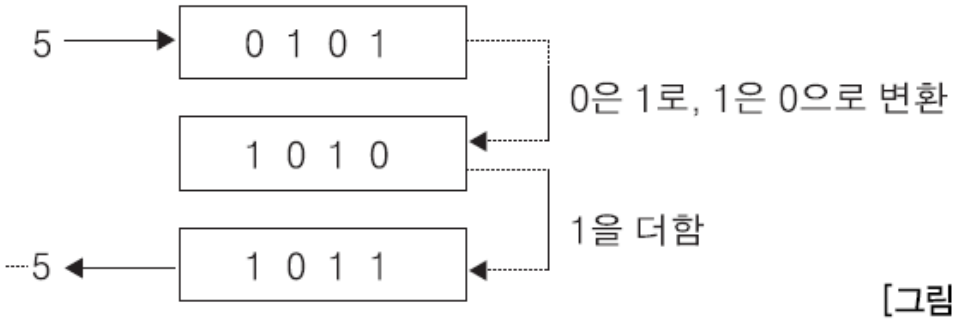
❖ 양수에 대한 2의 보수 표현

[표 2-8] 양수에 대한 2의 보수 표현

표현	값
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0

❍정수 표현하기

❑ -5의 2의 보수 표현 과정



[그림 2-28] -5의 2의 보수 표현 과정

❍정수 표현하기

❑ 음수에 대한 2의 보수 표현

[표 2-9] 음수에 대한 2의 보수 표현

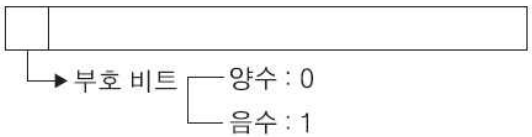
표현	값
1111	-1
1110	-2
1101	-3
1100	-4
1011	-5
1010	-6
1001	-7
1000	-8

❑ 8비트의 2의 보수 표현

[표 2-10] 8비트의 2의 보수 표현

표현	값
01111111	127
01111110	126
01111101	125
⋮	⋮
00000001	1
00000000	0
11111111	-1
11111110	-2
⋮	⋮
10000001	-127
10000000	-128

❑ 2의 보수 표현의 부호 비트



[그림 2-29] 2의 보수 표현의 부호 비트

❑ 표현할 수 있는 수의 범위

$$-2^{n-1} \sim 2^{n-1}-1$$

❍정수의 덧셈과 뺄셈

❑ 정수의 덧셈

❖ 16비트로 정수를 표현한다고 가정

❖ 6 + 7

```
6 => 0000000000000110
7 => 0000000000000111
```

```
  ···(1) (1) (0)
  0000000000000 0  1  1  0
+ 0000000000000 0  1  1  1
  0000000000000 1 (1)1 (1)0 (0)1
```

[그림 2-30] 6과 7의 덧셈

❖ 6 + (−7)

```
 6 => 0000000000000110
-7 => 1111111111111001
```

```
  0000000000000110
+ 1111111111111001
  1111111111111111
```

[그림 2-31] 6과 −7의 덧셈

❑ 정수의 덧셈

❖ (-6) + 7

```
-6 => 1111111111111010
 7 => 0000000000000111

  1111111111111010
+ 0000000000000111
------------------
 10000000000000001
 ↳ 무시
```

[그림 2-32] -6과 7의 덧셈

❖ (-6) + (-7)

```
-6 => 1111111111111010
-7 => 1111111111111001

  1111111111111010
+ 1111111111111001
------------------
 11111111111110011
 ↳ 무시
```

[그림 2-33] -6과 -7의 덧셈

❖ 30000 + 30000

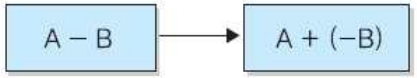
```
30000 => 0111010100110000

  0111010100110000
+ 0111010100110000
------------------
  1110101001100000
```

[그림 2-34] 30000과 30000의 덧셈

❑ 정수의 뺄셈

❖ 컴퓨터 내부에서의 뺄셈은 덧셈을 하는 하드웨어인 가산기를 이용



❖ 6 − 7

```
  0000000000000110
+ 1111111111111001
------------------
  1111111111111111
```

[그림 2-35] 6에서 7의 뺄셈

❖ −6 − 7

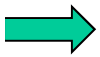
```
  1111111111111010
+ 1111111111111001
------------------
1 1111111111110011
  ↳ 무시
```

[그림 2-36] −6에서 7의 뺄셈

❖ −30000 − 30000

- 최상위 bit를 넘어가는 자리 올림수 1을 무시하면 5536이 되어 오버플로우 발생

```
  1000101011010000
+ 1000101011010000
------------------
1 0001010110100000
  ↳ 무시
```



```
0001010110100000
```

[그림 2-37] −30000에서 30000의 뺄셈

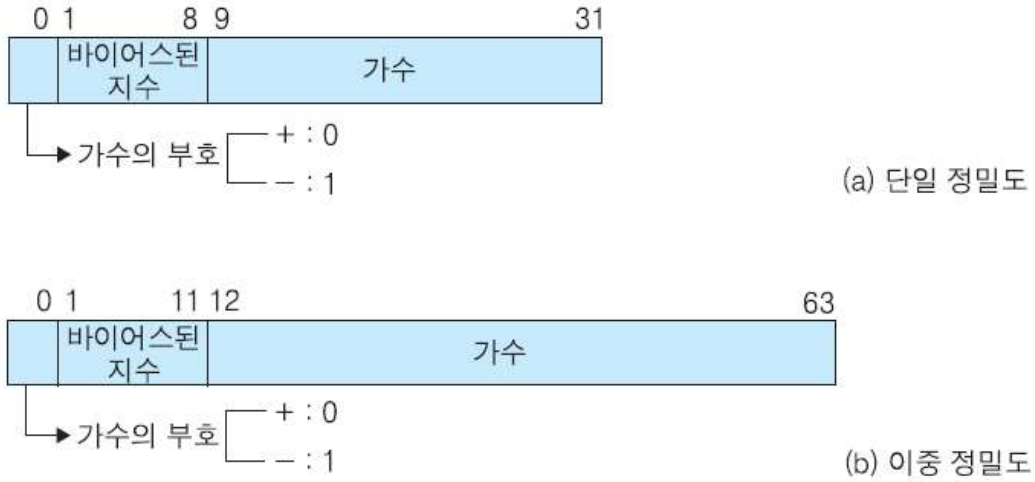
❍실수 표현하기

- ❑ 컴퓨터 내부에서는 제한된 공간에 2진수 형태로 표현

$$m \times r^e$$

(m : 가수 r : 밑수 e : 지수)

- ❑ IEEE 754 실수 표현 형식

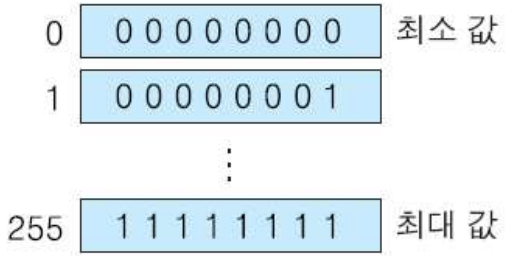


[그림 2-38] IEEE 754 실수 표현 형식

❍실수 표현하기

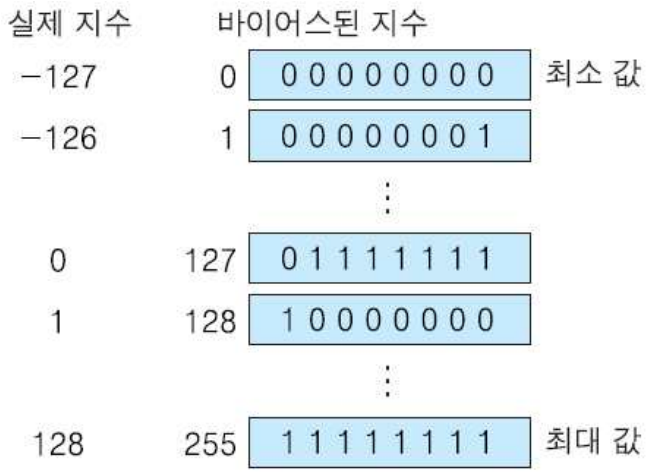
- 예 : $-0.001101_2 \times 2^2$
- 정규화 : $\pm 1.???_2 \times 2^? \Rightarrow -1.101_2 \times 2^{-1}$
- 지수 처리 : 지수 + 바이어스 => 바이어스 된 지수

❖ 바이어스 개념이 없을 경우 지수 표현 범위



[그림 2-39] 바이어스 개념이 없을 경우의 지수 표현 범위

❖ 바이어스가 127인 경우의 지수 표현 범위



[그림 2-40] 바이어스가 127인 경우의 지수 표현 범위

❍실수 표현하기

❖ IEEE 754 표준에서 바이어스는 127

$-1+127 \Rightarrow 126$

[그림 2-41] 간략하게 표현한 실수 변환 예제

- 가수의 부호를 나타내야 하는데, 가수가 음수므로 첫 번째 bit가 1이 됨
- 8비트 영역에 바이어스 된 지수 126을 나타냄
- 가수를 우측 23bit에 표현하면 되는데, 가수 1.101에서 소수점 아래 부분인 101을 나타내면 된다. 나머지 가수 부분은 0으로 채운다.

[그림 2-42] 실수 변환 결과

Thank you