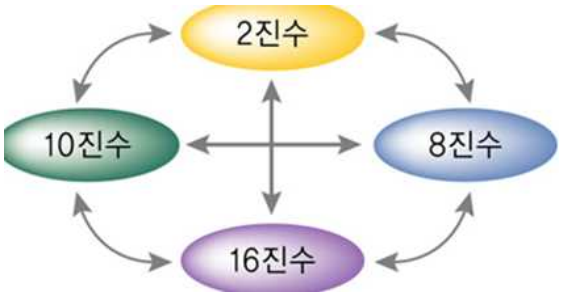
1. 수의 표현과 연산

1) 진법과 수의 구성

- 10진법: 0~9까지 사용하며 10을 한 자리의 기본 단위로 하는 진법
- 2진법: 0과 1의 조합으로 숫자를 표시하는 방법
- 8진법: 0~7까지 수로 표시하는 것이 8진법
- 16진법: 0~9까지 그리고 A~F까지를 사용하여 표시하는 진법

2) 수의 변환



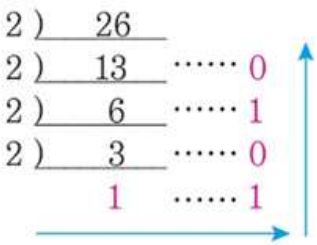
- 8진수를 10진수로 변환

$$(156)_8 = (110)_{10}$$
$$1 \times 8^2 + 5 \times 8^1 + 6 \times 8^0 = 110$$

- 2진수를 10진수로 변환

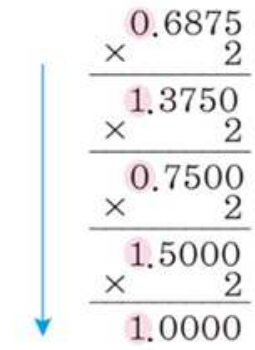
$$(11010)_2 = (26)_{10}$$
$$1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 26$$

- 10진수를 2진수로 변환



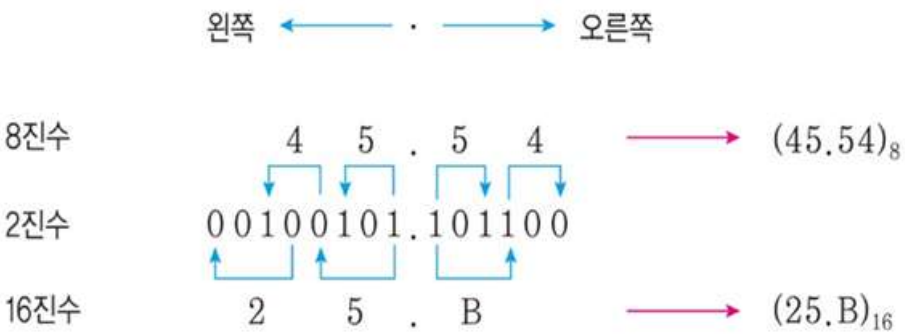
$$(26)_{10} = (11010)_2$$

- 소수를 2진수로 변환



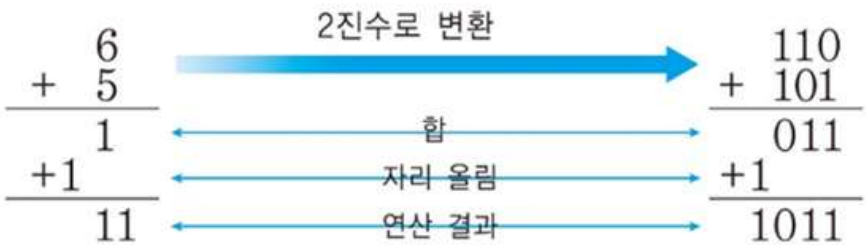
$(0.6875)_{10} = (0.1011)_2$

- 2진수,8진수,16진수의 상호 변환 관계



3) 2진수의 연산

(1) 덧셈



(2) 보수의 개념

- 보수에는 진수를 나타내는 수인 r의 보수와 (r-1)의 보수가 있음
- (r-1)의 보수: (r-1)의 값에서 수의 각 자리의 숫자를 빼면 (r-1)의 보수를 얻게 됨

$(835)_{10}$의 9의 보수는$(164)_{10}$ $(1010)_2$의 1의 보수는 $(0101)_2$

- r의 보수: (r-1)의 보수를 구하여 가장 낮은 자리에 1을 더함

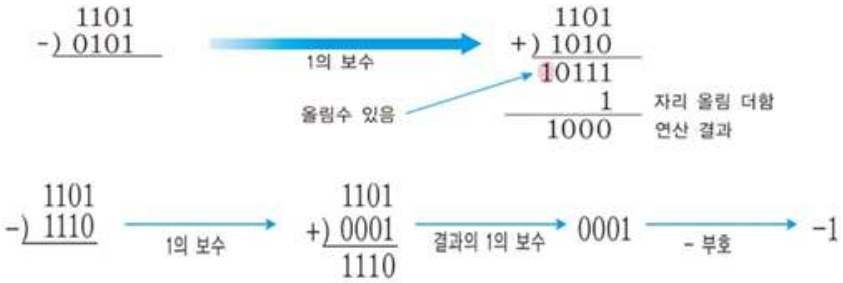
$(835)_{10}$의 10의 보수는 164+1=165 $(1010)_2$의 2의 보수는 0101+1=0110

(3) 1의 보수에 의한 뺄셈 처리 과정

- 컴퓨터에서는 덧셈만 가능하기 때문에 뺄셈의 경우 보수를 이용하여 덧셈으로 변환하여

결과를 얻음
- 피감수에 감수의 1의 보수를 취하여 더함
- 맨 윗자리에 자리 올림수가 있으면 최하위 비트에 1을 더하고 올림수가 없으면 결과에서 다시 1의 보수를 취하고 -를 붙임

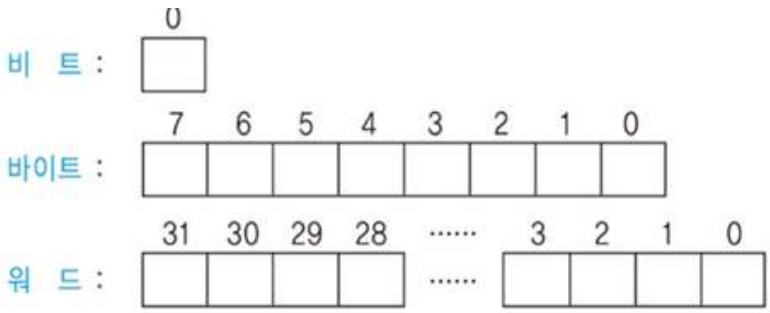
4) 1의 보수에 의한 뺄셈 처리 과정



2. 데이터의 표현

1) 수치 데이터 표현

(1) 비트, 바이트, 워드
- 비트(bit): 컴퓨터에서 사용하는 최소의 단위로서 0, 1을 나타냄
- 바이트(byte): 영문 1 글자를 나타내는 단위로 8비트로 이루어짐
- 워드(word): 워드의 크기는 컴퓨터의 종류에 따라 2바이트, 4바이트, 8바이트 등이 있는데 통상 4바이트를 말함



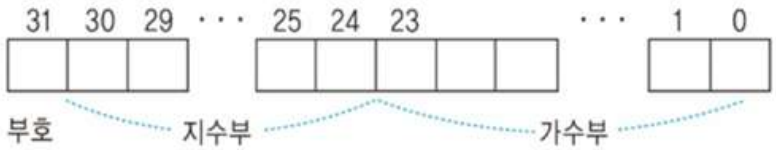
<기억 용량의 크기 단위>
- 1KB(Kilo Byte): 1,024바이트(210바이트)
- 1MB(Mega Byte): 1,048,576바이트 또는 1,024KB(220바이트), 약 백만 바이트
- 1GB(Giga Byte): 1,073,741,824바이트 또는 1,024MB(230바이트), 약 십억 바이트

(2) 고정 소수점 데이터 형식
- MSB(Most Significant Bit): 부호 비트(양수:0, 음수:1)
- 양수의 경우 정수 부분: 10진수를 2진수로 변환하여 표시함
- 음수의 경우 정수 부분: 부호와 절대값의 표현법

- 1의 보수법이나 2의 보수법 중 하나를 쓰는데 보통 2의 보수법을 많이 사용함

(3) 부동 소수점 데이터 형식

- MSB: 부호 비트(양수:0, 음수:1)
- 지수부: 지수를 2진수로 변환하여 표시함
- 가수부: 소수점 안의 유효 숫자를 2진수로 표현함
  - 이때 소수점은 지수부와 가수부 사이에 있는 것으로 가정함



2) 문제 데이터 형식

(1) 아스키 코드

- 미국 정보 교환 표준 코드로서 미국 표준 협회가 제정한 데이터 처리 및 통신시스템 상호 간의 정보 교환용 표준 코드
  - 구성: 패리티 비트:1개
  - 존(zone) 비트: 3개(001: 숫자, 100: A~O, 101: P~Z)
  - 디지트(digit) 비트: 4개

(2) BCD 코드

- 6비트를 사용하여 하나의 문자를 표시하는 방식으로 기억 장치의 단어 길이가 6의 배수로 설계된 컴퓨터에 적합함
- 자료 구조는 존 필드와 디지트 필드로 나뉘어 있으며 하나의 문자를 표현함

(3) EBCDIC 코드

- 한 문자를 8비트로 나타내며 기존의 BCD코드를 8비트로 확장한 코드로 256개의 문자까지 표현 가능함

(4) 한글과 한자 데이터의 표현

완성형 (KSC5601)	- 한글, 특수문자, 숫자 한글 낱자, 한자, 외국문자 등의 모양을 미리 만들어 놓고 표현하는 코드 - 메모리를 많이 차지함 - 글자 정렬과 글자체의 모양을 좋게 할 수 있음
조합형	- 현대 한글 음절 11,172개 모두를 표현할 수 있는 방식으로 초성, 중성,

	종성을 각각 별도로 처리하여 모든 글자를 조합해서 만들 수 있음
유니코드	- 한글만을 위한 코드 체계가 아닌 전세계 언어를 하나의 코드 체계 안으로 통합하려는 컴퓨터 업체들의 합의에 의해 만들어진 코드임 - 2바이트를 사용하여 각 국가의 언어를 표시할 수 있으므로 유니코드를 지원하는 프로그램이면 프로그램 상에서 한글이나 일본어 등에 대한 별도의 처리 없이 자유롭게 볼 수 있음

3. 논리회로

1) 부울대수
- 영국의 조지 부울이 제창
- 0과 1의 2진수 표현으로 명제의 참과 거짓, 전기 신호의 유와 무, 스위치의 ON과 OFF 등을 표현함
- 논리합, 논리곱, 논리부정 등 3가지 연산 기호를 사용하여 논리식 표현에 사용됨

2) 기본 논리회로
- 부울 대수의 기본 연산인 논리합, 논리곱, 논리부정 등의 연산을 실행하기 위한 회로로서 논리 게이트(Logic Gate)라고도 함
- 2진 정보를 취급하며 보통 2개 이상의 입력 단자와 하나의 출력단자로 구성됨

(1) 논리합 회로
- 논리합(OR) 조건을 만족시키는 회로로서 다음과 같이 2개의 조건이 있을 때 이 중 하나 이상을 만족하는 조건인데, 입력 A와 B 중 적어도 한쪽이 1이면 출력 Y가 1이 되는 논리회로
- 논리합 연산자는‘+’로 표현함

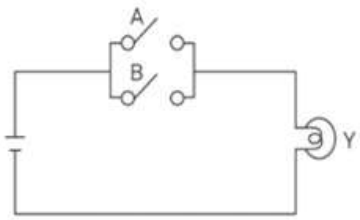
논리합 회로

입력		출력
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

[진리표]

[기호]

Y = A+B
[논리식]



(2) 논리곱 회로
- 논리곱(AND) 조건을 만족시키는 회로로서 다음과 같이 2개의 조건이 있을 때 모든 조건을 만족해야 되는 경우인데, 입력 A와 B가 모두 1인 경우에만 출력 Y가 1이 됨
- 논리곱 연산자는‘·’로 표현됨

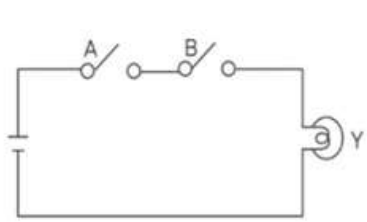
논리곱 회로

입력		출력
A	B	A·B
0	0	0
0	1	0
1	0	0
1	1	1

[진리표]

[기호]

Y = A·B
[논리식]



(3) 논리부정 회로

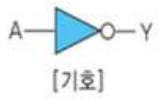
논리부정(NOT) 조건을 만족시키는 회로로서 다음과 같이 출력 조건이 입력 조건의 반대가 되는 경우인데, 입력 A가 1이면 출력 Y는 0, 입력 A가 0이면 출력 Y는 1이 됨

논리부정 연산자는‘ ¯’또는‘’로 표현됨

논리 부정 회로

입력	출력
A	A'
0	1
1	0

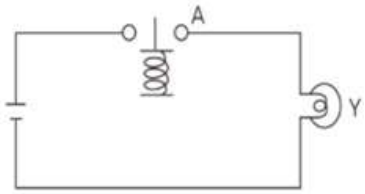
[진리표]



[기호]

$Y = \overline{A}$ 또는 $Y = A'$

[논리식]



<주요 논리 회로>

게이트	기호	수식	진리표
AND		x = A · B	A B | x 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OR		x = A+B	A B | x 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1
NOT(inverter)		x = A'	A | x 0 | 1 1 | 0
NAND		x = (A · B)'	A B | x 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0
NOR		x = (A+B)'	A B | x 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0
exclusive-OR(XOR)		x = (A⊕B) or x = A'B+AB'	A B | x 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0
exclusive-NOR		x = (A⊕B)' or x = AB+A'B'	A B | x 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 1