2장 데이터 표현과 컴퓨터 연산

2진수의 연산

- 정수의 산술
  - 음수화(2의 보수 변환) ; A = A'+ 1 (A ': 1의 보수)
  - 덧셈 ; C = A + B
  - 뺄셈 ; C = A – B
  - 곱셈 ; C = A× B
  - 나눗셈 ; C = A / B

음수화(negation)

- 2의 보수를 사용
  - -19를 2의 보수로 만드는 예

+19 :	00010011
1의 보수 :	11101100
	+ 1
-19 :	11101101

덧셈

- 양수와 양수의 덧셈
  (+2) + (+3) = (+5)

```
    0010
+   0011
  ------
    0101
```

- 음수와 양수의 덧셈
  (-3) + (+5) = (+2)

```
       1101
    +  0101
     ------
    1  0010
  버림
```

- 음수와 양수의 덧셈
  (-6) + (+3) = (-3)

```
    1010
+   0011
  ------
    1101
```

- 음수와 음수의 덧셈
  (-2) + (-4) = (-6)

```
       1110
    +  1100
     ------
    1  1010
  버림
```

뺄셈

- 빼는 수를 보수화하여 결과적으로 덧셈을 이용

A – (+B) = A + (–B)

A – (–B) = A + (+B)

- 자리 올림수가 발생하지 않는 경우

  (+2)–(+5)=(+2)+(-5)=(-3)

```
     0010
+    1011
---------
     1101   = -3
```

- 자리 올림수가 발생하는 경우
  - (+5)–(+2)=(+5)+(-2)=(+3)

```
     0101
+    1110
---------
1    0011   = +3
버림
```

곱셈

- 곱하는 수(B)를 승수(multiplier)라고 하며, 곱하여 지는 수(A)를 피승수(multiplicand)

$$A \times B = C$$

- 부호가 없는 경우의 곱셈의 예

			×	1	1	0	1	피승수(13)
				1	0	1	1	승수(11)
				1	1	0	1	부분합
			1	1	0	1		(partial product)
		0	0	0	0			
	1	1	0	1				
1	0	0	0	1	1	1	1	곱의 결과(143)

  - 4비트의 두 수가 서로 곱셈을 수행하면, 2배인 8비트의 길이의 결과를 출력

나눗셈

- 나누어지는 수 D를 피제수(dividend), 나누는 수 V를 제수(divisor), 나눗셈의 결과 몫(quotient) Q와, 나머지 수(remainder) R

$$D \div V = Q \ldots R$$

```
                         00001101        ← 몫
                        ___________
제수 →            1011 | 10010011        ← 피제수
                          1011
                        ___________
부분나머지        →      001110
                           1011
                        ___________
부분나머지        →        001111
                             1011
                        ___________
                              100        ← 나머지
```

부동소수점 수의 산술

- 가수와 지수의 연산을 분리해서 수행

- 덧셈과 뺄셈의 경우:
  1. 지수를 같은 값으로 조정
  2. 가수들에 대하여 덧셈과 뺄셈을 수행

- 곱셈과 나눗셈의 경우:
  1. 가수끼리는 곱셈과 나눗셈을 수행
  2. 지수의 연산에서는 곱셈의 경우는 덧셈을 수행하고 나눗셈의 경우에는 뺄셈을 한다

부동소수점 수의 덧셈과 뺄셈

- 지수들이 동일한 값을 갖도록 일치
- 가수들 간의 덧셈과 혹은 뺄셈을 수행
- 이진수의 경우 결과를 정규화 (normalization)
- 십진수의 부동소수점의 수의 덧셈과 뺄셈

$$A = 0.3 \times 10^2, \ B = 0.2 \times 10^3$$

  - 덧셈 연산

$$A + B = 0.3 \times 10^2 + 0.2 \times 10^3 = 0.3 \times 10^2 + 2 \times 10^2 = 2.3 \times 10^2$$

  - 뺄셈 연산

$$A - B = 0.3 \times 10^2 - 0.2 \times 10^3 = 0.3 \times 10^2 - 2 \times 10^2 = -1.7 \times 10^2$$

지수조정

$$\begin{array}{lcll} 0.110010 \times 2^2 & \Rightarrow & 0.011001 \times 2^3 & \\ +0.111011 \times 2^3 & & +\ 0.111011 \times 2^3 & \text{정규화} \\ \hline & & 1.010100 \times 2^3 & \Rightarrow \quad 0.101010 \times 2^4 \end{array}$$

부동소수점 수의 곱셈과 나눗셈

- 부동소수점 수의 곱셈
  - 가수끼리는 곱셈 연산을 수행하고 지수끼리는 덧셈을 수행
- 십진 부동소수점 수의 곱셈 예

$$A = 0.3 \times 10^2,\ B = 0.2 \times 10^3$$

$$A \times B = (0.3 \times 10^2) \times (0.2 \times 10^3)$$
$$= (0.3 \times 0.2) \times 10^{2+3} = 0.06 \times 10^5$$

- 이진 부동소수점 수의 곱셈 예

$$(0.1011 \times 2^3) \times (0.1001 \times 2^5)$$

  - 가수 곱하기: $1011 \times 1001 = 01100011$
  - 지수 더하기: $3 + 5 = 8$
  - 정규화: $0.01100011 \times 2^8 = 0.1100011 \times 2^7$

부동소수점 수의 곱셈과 나눗셈

- 부동소수점 수의 나눗셈
  - 가수부분은 나눗셈 연산 수행, 지수부분은 뺄셈 연산 수행
- 십진수의 부동소수점 수의 나눗셈

$$A \div B = 0.3 \times 10^{2} \div 0.2 \times 10^{3}$$
$$= (0.3 \div 0.2) \times 10^{2-3} = 1.5 \times 10^{-1}$$

- 이진수 부동소수점 수의 나눗셈
  - $(0.1100 \times 2^{5}) \div (0.1100 \times 2^{3})$
  - 가수 나누기: $1100 \div 1100 = 1$
  - 지수 뺄셈: $5 - 3 = 2$
  - 정규화: 0.1×2^{2}

2진수의 논리 연산

- AND, OR, NOT, XOR(Exclusive-OR)

기본적인 논리 연산의 진리표

입력 X	입력 Y	X AND Y	X OR Y	X XOR Y	NOT X	NOT Y
0	0	0	1	0	1	1
0	1	0	1	1	1	0
1	0	0	1	1	0	1
1	1	1	0	0	0	0

컴퓨터 응용 논리연산

- Selective-set 연산
  - 특정 비트를 1로 세트시키는 연산
- Selective-보수 연산
  - 특정 비트를 반전시키는 연산
- Selective-set (mask) 연산
  - 특정 비트를 0으로 clear시키는 연산
- 삽입 연산
  - 원하는 위치에 새로운 비트값을 삽입하는 연산
- 비교 연산
  - 두 데이터를 비교해서 같으면 0, 다르면 1로 세트

컴퓨터 응용 논리연산

- Selective-set 연산
  - 원하는 비트들을 선택적으로 세트하는데 사용하는 연산
  - A 레지스터의 하위 4비트를 1로 세트하는 경우의 예

A = 1 0 1 1 0 1 0 1	**(연산 전)**
B = 0 0 0 0 1 1 1 1	**선택적 세트 연산**
A = 1 0 1 1 1 1 1 1	**(연산 후)**

컴퓨터 응용 논리연산

- Selective-보수 연산
  - 원하는 비트들을 선택적으로 반전하는데 사용하는 연산
  - A 레지스터의 하위 4비트를 반전하는 경우의 예

A = 1 0 1 1 0 1 0 1	**(연산 전)**
B = 0 0 0 0 1 1 1 1	**선택적 보수 연산**
A = 1 0 1 1 1 0 1 0	**(연산 후)**

컴퓨터 응용 논리연산

- 마스크(mask) 연산
  - 원하는 비트들을 선택적으로 clear하는 데 사용하는 연산
  - A 레지스터의 상위 4비트를 0으로 clear하는 경우의 예

A = 1 0 1 1 0 1 0 1	**(연산 전)**
B = 0 0 0 0 1 1 1 1	**마스크 (AND)연산**
A = 0 0 0 0 0 1 0 1	**(연산 후)**

컴퓨터 응용 논리연산

- 삽입 연산 – 마스킹 후, OR 연산
  - 원하는 위치에 새로운 비트값을 삽입하는 연산
  - A 레지스터의 상위 4비트에 1100을 삽입하는 경우의 예

A = 1 0 1 1 0 1 0 1	**(연산 전)**
B = 0 0 0 0 1 1 1 1	**마스크 (AND)연산**
A = 0 0 0 0 0 1 0 1	**(연산 후)**
B = 1 1 0 0 0 0 0 0	**OR연산**
A = 1 1 0 0 0 1 0 1	

컴퓨터 응용 논리 연산

- 비교(compare) 연산
  - 두 데이터를 비교하는 연산으로 대응되는 비트들의 값이 같으면, 해당 비트를 0으로 세트하고 서로 다르면, 해당 비트를 1 세트

A = 1 1 1 0 0 0 0 1	**(연산 전)**
B = 0 1 0 1 1 0 0 1	**비교(XOR)연산**
A = 1 0 1 1 1 0 0 0	**(연산 후)**

컴퓨터 응용 논리 연산

- 산술적 쉬프트(arithmetic shift)
  - 쉬프트 과정에서 부호 비트는 유지하고, 수의 크기를 나타내는 비트들만 쉬프트한다.
  - 산술적 좌측-쉬프트
    - D4 (불변), D3 ← D2, D2 ← D1, D1 ← 0
  - 산술적 우측-쉬프트
    - D4 (불변), D4 → D3 , D3 → D2, D2 → D1
  - 다음은 산술적 쉬프트의 예이다.

A = 1 0 1 0 1 1 1 0 ; 초기 상태
1 1 0 1 1 1 0 0 ; A의 산술적 좌측-쉬프트 결과
1 1 0 1 0 1 1 1 ; A의 산술적 우측-쉬프트 결과

문자 데이터의 표현

- 표준 BCD 코드
- ASCII code- 미국 국립 표준 연구소에서 제정한 정보교환용 표준 코드
  - 0-9, A-Z, a-z, 이외 키보드의 기호들
  - 106쪽 표 2-7