Digital Engineering
Spring

Unit 1
Number Systems and Conversion
(수의 체계와 변환)

Course Administration

- Instructor: Hong Jong-Phil jphong@cbnu.ac.kr

  Labs: ICAT Lab. 46-472
  (Integrated Circuits for Advanced Technology)

  TA: 01반 조현호 (cho50021@chungbuk.ac.kr)
  02반 송성윤 (shbssy0522a@chungbuk.ac.kr)

- Text: Fundamentals of Logic Design, 7e
  Charles H. Roth , Larry L. Kinney , Eugene B. John

- Slides: http://icat.cbnu.ac.kr/

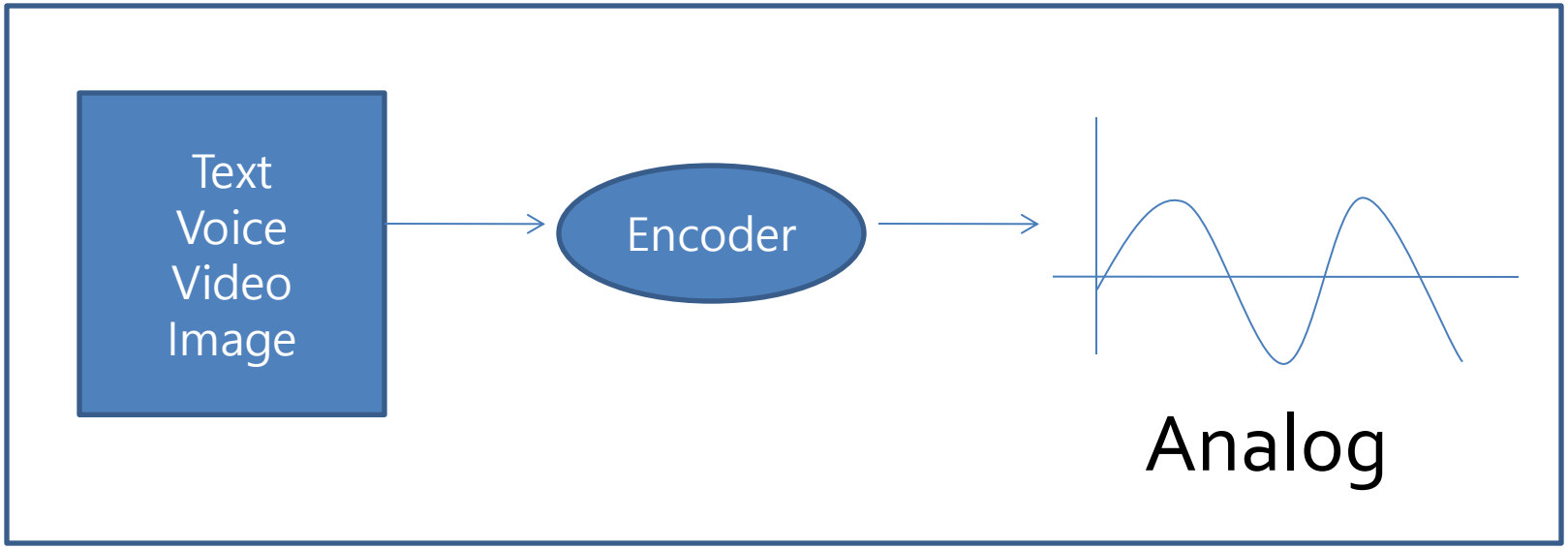
- Grade:

Midterm Exam	~35%
Final Exam	~40%
Homework	~15%
Class participation & Quizzes	~10%

강의 개요

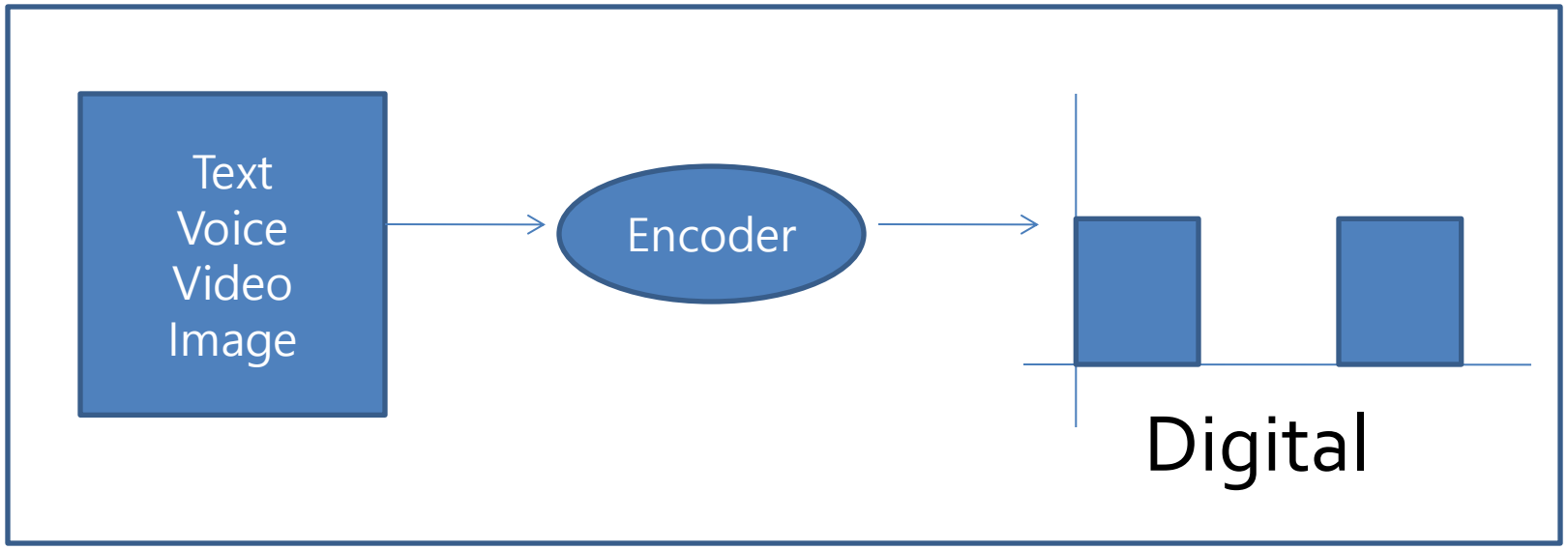
- 논리 회로의 개요
  - 논리란?
  - Digital이란?
  - 회로란?
  - 설계란?
  - 논리 설계란?
- 논리회로의 기본
  - 수 체계와 변환
  - 2진 산술 연산
  - 음수의 표현
  - 2진 코드

Analog란?



아날로그(analog)의 신호와 자료는 연속적인 물리량으로 나타낸 것이다..
우리가 거시적인 자연에서 얻는 신호는 대개 아날로그이다.
이를테면, 빛의 밝기, 소리의 높낮이나 크기, 바람의 세기 등이 있다.

Digital 란?

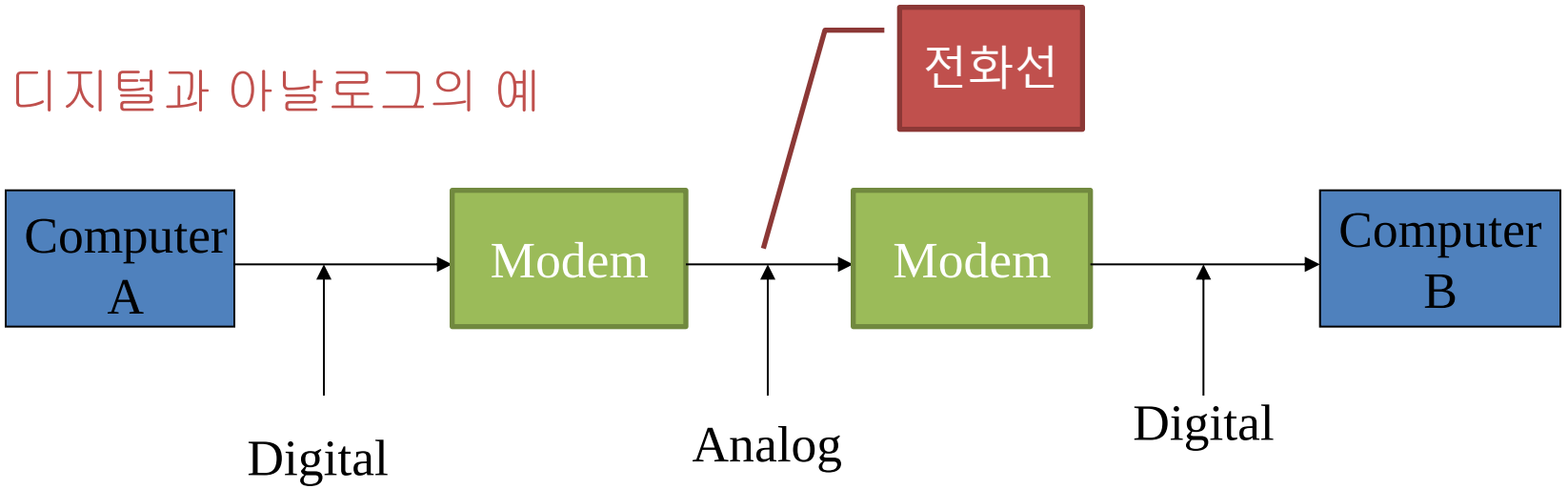


디지털(Digital)은 아날로그의 반대되는 개념으로 자료를 연속적인 실수가 아닌, 특정한 최소 단위를 갖는 이산적인 수치를 이용하여 처리하는 방법을 말한다.

디지털과 아날로그

- 아날로그
  - 모든 연속적인 값을 갖는다
  - 예: 음성, 음악, CRT모니터
- 디지탈
  - 0과 1 (이진수)
  - 컴퓨터가 처리하는 데이타

디지털과 아날로그의 예



모뎀을 이용한 두 컴퓨터 사이의 화일 전송

조합과 순차

- 조합 회로 (combinational circuit)
  - Memory가 없다.
  - 입력한 값에 따른 출력
    - 출력 = *f*(입력)

- 순차 회로 (sequential circuit)
  - Memory가 있다.
  - Memory에는 회로의 현 상태가 저장
  - 출력은 입력과 현 상태에 의해 결정
    - (출력,다음 상태) = *f*(입력, 상태)

논리회로 설계란?

- 기능이 주어질 때, 그 기능을 논리 소자 (하드웨어)로 구현하는 것
  - 기능의 표현 (뭘하는 회로인가?)
    - 문장, 말, pseudo code, program
    - 진리표
    - 카르노 맵
    - minterm의 합, maxterm의 곱
    - FSM
    - BDD, 등등등…
  - 구현 방법
    - 논리 회로 설계에서는 논리 게이트를 서로 연결하여 회로망을 형성하여 구현
    - 하나의 기능을 구현하는 논리 회로는 많이 있다. 그 중 최적을 구하는 것이 중요

설계의 단계

- 시스템 단계(board-level)
- 칩 단계 (chip-level)
- 모듈 단계 (module-level)
- 게이트 단계 (gate-level)
- 회로 단계 (circuit-level)
- 트랜지스터 단계 (transistor-level)

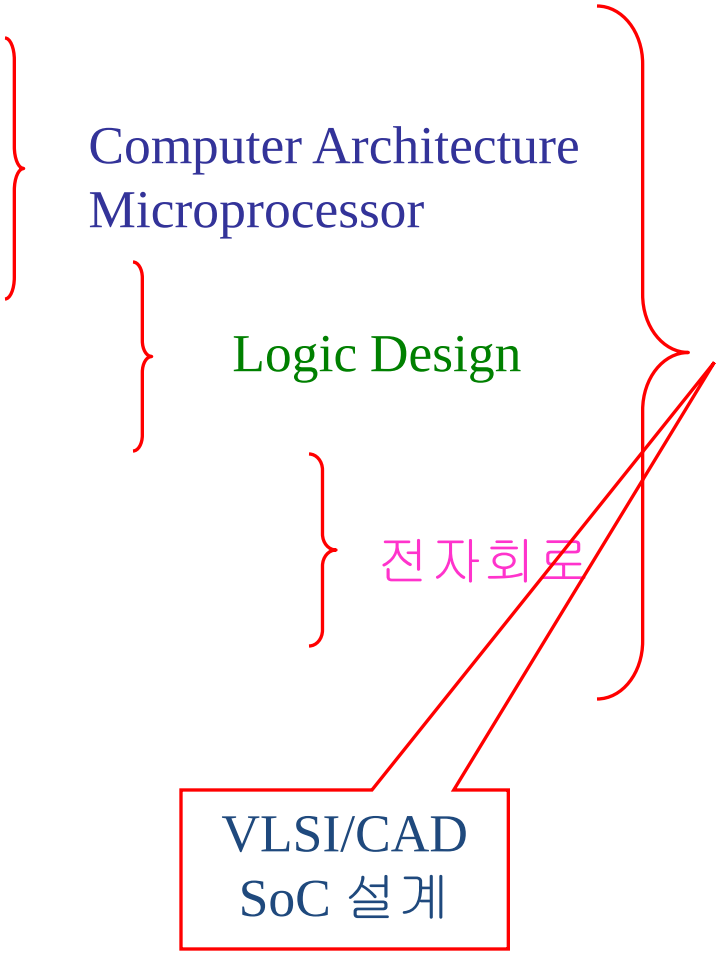
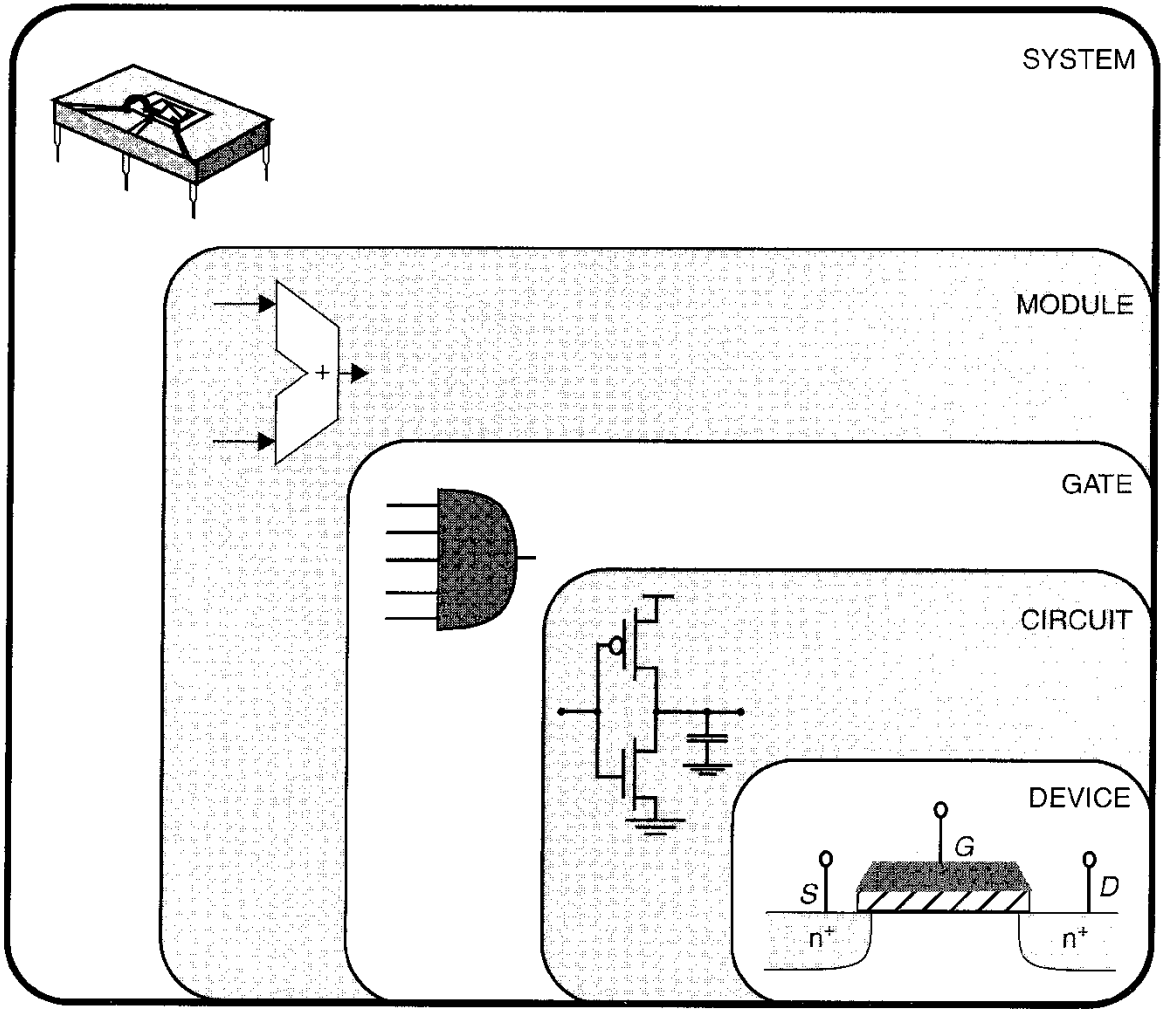


그림: 설계의 단계



[Jan Rabaey's Digital Circuit Design 중에서]

1.2 수의 체계와 변환

✓ 위치 표기법(positional notation)에 위한 10진수(decimal)의 표현.
각 자리수는 위치에 따라 적절한 10의 지수(power) 값이 곱해진다.

$$953.78_{10} = 9\times10^{2} + 5\times10^{1} + 3\times10^{0} + 7\times10^{-1} + 8\times10^{-2}$$

✓ 마찬가지로, 2진수(binary, 기저 2, radix or base 2)에 대해서도 다음과 같이 표현된다.

$$1011.11_{2} = 1\times2^{3} + 0\times2^{2} + 1\times2^{1} + 1\times2^{0} + 1\times2^{-1} + 1\times2^{-2}$$

$$= 8 + 0 + 2 + 1 + \frac{1}{2} + \frac{1}{4} = 11\frac{3}{4} = 11.75_{10}$$

✓ 양의 정수 R(R >1)도 수 체계(Number System)의 기저(Radix or Base)로 사용 가능.
기저가 R 인 수에서는 모두 R개(0, 1, 2, …,R-1)의 수가 사용된다.

✓ R = 8인 경우 필요한 기호 숫자는 0,1,2,3…6, 7이다.

1.2 수의 체계와 변환

✓ 따라서 위치 표기법으로 사용된 수는 R의 지수 급수적으로 전개될 수 있다.

$$N = (a_4 a_3 a_2 a_1 a_0 . a_{-1} a_{-2} a_{-3})_R$$
$$= a_4 \times R^4 + a_3 \times R^3 + a_2 \times R^2 + a_1 \times R^1 + a_0 \times R^0$$
$$+ a_{-1} \times R^{-1} + a_{-2} \times R^{-2} + a_{-3} \times R^{-3}$$

✓ 여기서 a_i 는 R^i 의 계수이고, $0 \le a_i \le R-1$ 의 범위를 갖는다.

✓ 기저가 8인 8진수의 표기법

$$147.3_8 = 1 \times 8^2 + 4 \times 8^1 + 7 \times 8^0 + 3 \times 8^{-1}$$
$$= 64 + 32 + 7 + \frac{3}{8} = 103.375_{10}$$

1.2 수의 체계와 변환

- **10진 정수(Integer)의 R 진수 변환**

✓ **R = 2 인 경우**

$$
\begin{array}{lll}
2 & \underline{53} & \\
2 & \underline{26} & rem. = 1 \rightarrow a_0 \\
2 & \underline{13} & rem. = 0 \rightarrow a_1 \\
2 & \underline{06} & rem. = 1 \rightarrow a_2 \\
2 & \underline{03} & rem. = 0 \rightarrow a_3 \\
 & \underline{01} & rem. = 1 \rightarrow a_4 \\
 & a_5 &
\end{array}
$$

$$53_{10} = a_5 a_4 a_3 a_2 a_1 a_0 = 110101_{\ 2}$$

✓ **R = 8 인 경우**

$$
\begin{array}{lll}
8 & \underline{53} & \\
 & \underline{6} & rem. = 5 \rightarrow a_0 \\
 & a_1 &
\end{array}
$$

$$53_{10} = a_1 a_0 = 65_8$$

1.2 수의 체계와 변환

- **10진 정수(Integer)의 R 진수 변환**

  - ✓ **R = 16 인 경우**

$$
\begin{array}{lll}
16 & \underline{534} & \\
16 & \underline{33} & rem. = 6 \rightarrow a_0 \\
16 & \underline{2} & rem. = 1 \rightarrow a_1 \\
 & a_2 &
\end{array}
$$

$$534_{10} = a_2 a_1 a_0 = 216_{16}$$

$$534_{10} = 216_{16} = 1000010110_{\ 2}$$

  - ✓ **R = 2 인 경우**

$$
\begin{array}{lll}
2 & \underline{534} & \\
2 & \underline{267} & rem. = 0 \rightarrow a_0 \\
2 & \underline{133} & rem. = 1 \rightarrow a_1 \\
2 & \underline{66} & rem. = 1 \rightarrow a_2 \\
2 & \underline{33} & rem. = 0 \rightarrow a_3 \\
2 & \underline{16} & rem. = 1 \rightarrow a_4 \\
2 & \underline{08} & rem. = 0 \rightarrow a_5 \\
2 & \underline{04} & rem. = 0 \rightarrow a_6 \\
2 & \underline{02} & rem. = 0 \rightarrow a_7 \\
2 & \underline{1} & rem. = 0 \rightarrow a_8 \\
 & a_9 &
\end{array}
$$

1.2 수의 체계와 변환

✓ 기저가 16인 16진수[hexadecimal] 시스템에 대한 기호표현.

$10_{10} \rightarrow \mathrm{A} \quad 11_{10} \rightarrow \mathrm{B} \quad 12_{10} \rightarrow \mathrm{C}$

$13_{10} \rightarrow \mathrm{D} \quad 14_{10} \rightarrow \mathrm{E} \quad 15_{10} \rightarrow \mathrm{F}$

$$\mathrm{A2F}_{16} = 10\times16^2 + 2\times16^1 + 15\times16^0$$
$$= 2560 + 32 + 15 = 2607_{10}$$

✓ 기저가 16인 16진수와 기저가 2인 2진수의 변환

$$\mathrm{A2F}_{16} = 1010 \quad 0010 \quad 1111$$
$$= 1\times2^{11} + 1\times2^9 + 1\times2^5 + 1\times2^3 + 1\times2^2 + 1\times2^1 + 1\times2^0$$
$$= 2048 + 512 + 32 + 8 + 4 + 2 + 1 = 2607_{10}$$
$$= 10\times16^2 + 2\times16^1 + 15\times16^0$$
$$= 2560 + 32 + 15 = 2607_{10}$$

$$101101101011_2 = 1011 \quad 0110 \quad 1011 = 11 \quad 6 \quad 11$$
$$= \mathrm{B6B}_{16}$$

1.2 수의 체계와 변환

✓ 10진 소수(fraction)의 R 진수 변환

$$F = \left(.a_{-1}a_{-2}a_{-3}\cdots a_{-m}\right)_R$$
$$= a_{-1}R^{-1} + a_{-2}R^{-2} + a_{-3}R^{-3} + \cdots + a_{-m}R^{-m}$$

✓ R을 곱하면,

$$FR = a_{-1} + a_{-2}R^{-1} + a_{-3}R^{-2} + \cdots + a_{-m}R^{-m+1} = a_{-1} + F_1$$

여기서 F_1 은 결과값의 소수부분을 나타내며, a_{-1} 은 정수부이다.

✓ F_1 에 R을 다시 곱하면

$$F_1R = a_{-2} + a_{-3}R^{-1} + \cdots + a_{-m}R^{-m+2} = a_{-2} + F_2$$

✓ F_2 에 R을 다시 곱하면

$$F_2R = a_{-3} + \cdots + a_{-m}R^{-m+3} = a_{-3} + F_3$$

1.2 수의 체계와 변환

➢ 10진 소수(Fraction)의 R 진수 변환

✓ R = 2 인 경우

$$\begin{array}{rrr} .625 & .250 & .500 \\ \times\quad 2 & \times\quad 2 & \times\quad 2 \\ \hline 1.250 & 0.500 & 1.000 \end{array} \qquad 0.625_{10} = .101_{2}$$

$$a_{-1} = 1 \qquad a_{-2} = 0 \qquad a_{-3} = 1$$

$$\begin{array}{rrrrrr} .7 & .4 & .8 & .6 & .2 & .4 \\ \times\quad 2 & \times\quad 2 & \times\quad 2 & \times\quad 2 & \times\quad 2 & \times\quad 2 \\ \hline 1.4 & 0.8 & 1.6 & 1.2 & 0.4 & 0.8 \end{array}$$

$$a_{-1} = 1 \quad a_{-2} = 0 \quad a_{-3} = 1 \quad a_{-4} = 1 \quad a_{-5} = 0 \quad a_{-6} = 0$$

$$0.7_{10} = .101100110\ \cdots_{2}$$

0.4는 앞에서 얻어진 수이며, 여기서부터 과정이 반복된다.

1.2 수의 체계와 변환

➢ 10진 수가 아닌 두 기저 사이의 변환

일반적으로는 우선 10진수로 변환한 다음, 이 변환된 10진수를 새로운 기저로 변환하는 것이 용이하다.

✓ 231.3_4 을 7진수로 변환하라.

$$231.3_4 = 2\times4^2 + 3\times4^1 + 1\times4^0 + 3\times4^{-1}$$

$$= 32 + 12 + 4 + \frac{3}{4} = 45.75_{10}$$

$$\begin{array}{rl} 7 & \underline{45} \\ 7 & \underline{6} \quad rem. = 3 \rightarrow a_0 \\ & \underline{0} \quad rem. = 6 \rightarrow a_1 \end{array}$$

$$\begin{array}{rrr} .75 & .25 & .75 \\ \times\quad 7 & \times\quad 7 & \times\quad 7 \\ \hline 5.25 & 1.75 & 5.25 \\ a_{-1} = 5 & a_{-2} = 1 & a_{-3} = 5 \end{array}$$

$$231.3_4 = 63.5151\cdots_7$$

1.3 2진 산술연산

- 산술연산: ＋, －, ×, ÷
- 디지털 시스템의 산술 연산은 2진 연산으로 수행
- 2진 산술연산 : 0과 1로만 이루어진

  ＋, －, ×, ÷

1.3 2진 산술연산

✓ 2진수에 대한 덧셈표

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 0$$
$$10$$

다음 열로 자리 올림(carry) 1

1.3 2진 산술연산

예제) 2진수를 이용하여 13_{10} 과 11_{10} 을 더하라.

$$
\begin{array}{rcl}
 & & \boxed{1\ \ 1\ \ 1\ \ 1} \quad \text{carry} \\
13_{10} & = & 1\ \ 1\ \ 0\ \ 1 \\
11_{10} & = & 1\ \ 0\ \ 1\ \ 1 \\
\hline
 & = & 1\ \ 1\ \ 0\ \ 0\ \ 0
\end{array}
$$

1.3 2진 산술연산

- 2진수의 뺄셈

  - ✓ 2진수에 대한 뺄셈표

$$0 - 0 = 0$$
$$0 - 1 = 1$$
$$1 - 0 = 1$$
$$1 - 1 = 0$$

앞 열에서 자리 내림(borrow) 1

1.3 2진 산술연산

예제 a)

borrow

```
      1
    1 1 1 0 1
  - 1 0 0 1 1
-------------
    0 1 0 1 0
```

예제 b)

borrow

```
    1 1 1 1
    1 0 0 0 0
  -       1 1
-------------
      1 1 0 1
```

예제 c)

borrow

```
    1 1 1
1 1 1 0 0 1
    1 0 1 1
-----------
1 0 1 1 1 0
```

2진 뺄셈에 대한 다른 방법으로는 1.4절에서 논의할 2의 보수를 이용한 산술연산이 있다.

1.3 2진 산술연산

✓ 2진수에 대한 곱셈표

$$
\begin{aligned}
0 \times 0 &= 0 \\
0 \times 1 &= 0 \\
1 \times 0 &= 0 \\
1 \times 1 &= 1
\end{aligned}
$$

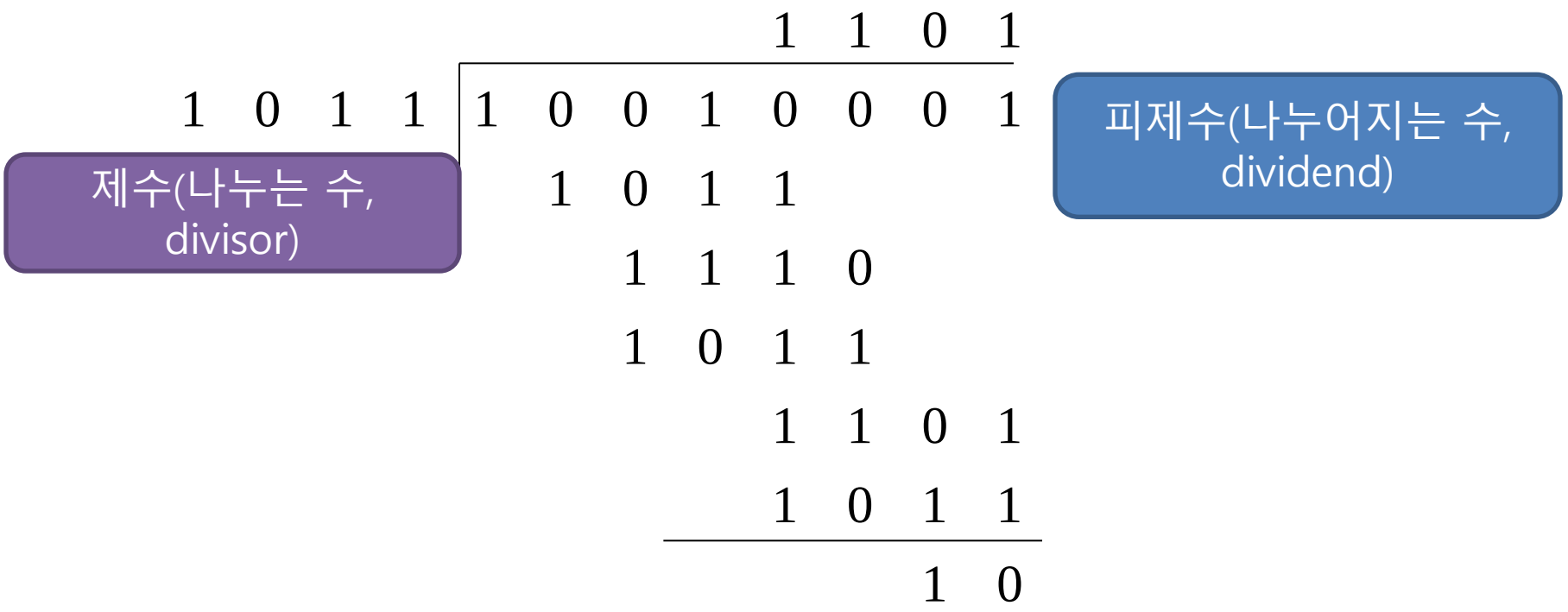
1.3 2진 산술연산

예제) 2진수를 이용하여 13_{10} 과 11_{10} 을 곱하라.

				1	1	0	1	피승수(곱하여지는 수,Multiplicand)
				1	0	1	1	승수(곱하는 수,Multiplier)
				1	1	0	1	$101\underline{1}$
			1	1	0	1		$10\underline{1}1$
		0	0	0	0			$1\underline{0}11$
	1	1	0	1				$\underline{1}011$
1	0	0	0	1	1	1	1	= 143_{10}

1.3 2진 산술연산

예제) 2진수를 이용하여 145_{10} 과 11_{10} 의 나눗셈을 하여라.



1.4 음수의 표현

✓ 지금까지 사용한 2진수는 양수만 취급되었다.(unsigned)

✓ 음수가 포함된 2진수(signed)는 다음과 같이 여러 가지 방법으로 표현된다.

- Signed Magnitude(부호와 크기 표현 방법)
- 1’ s Complement(1의 보수 표현 방법)
- 2’ s Complement(2의 보수 표현 방법)

1.4 음수의 표현

	Positive		Negative Integers		
$+N$	Integers (all systems)	$-N$	Sign and Magnitude	2's Complement N^*	1's Complement $\bar{N}$
+0	0000	−0	1000	——	1111
+1	0001	−1	1001	1111	1110
+2	0010	−2	1010	1110	1101
+3	0011	−3	1011	1101	1100
+4	0100	−4	1100	1100	1011
+5	0101	−5	1101	1011	1010
+6	0110	−6	1110	1010	1001
+7	0111	−7	1111	1001	1000
		−8	——	1000	——

1.4 음수의 표현 (Unsigned Magnitude Number)

✓ *n* 비트로 구성된 word의 경우 모두 2^n 의 수를 표현할 수 있다.

✓ *n* 비트로 구성된 word를 unsigned magnitude 표현으로 사용하는 경우,

0에서 2^n-1 까지의 양수를 표현할 수 있다.

예) 3 비트로 구성된 word를 unsigned magnitude 표현을 사용하는 경우

〔양수의 표현〕

binary number	*number*
111	+7
110	+6
101	+5
100	+4
011	+3
010	+2
001	+1
000	0

1.4 음수의 표현 (Signed Magnitude Number)

✓ n 비트로 구성된 word를 signed magnitude 표현으로 사용하는 경우,

+0과 -0 그리고 $2^{n-1}-1$개의 양수와 $2^{n-1}-1$ 개의 음수를 표현할 수 있다.

예) 3 비트로 구성된 word를 signed magnitude 표현을 사용하는 경우

〔양수와 음수의 표현〕

binary number	*number*
111	−3
110	−2
101	−1
100	−0
011	+3
010	+2
001	+1
000	+0

1.4 음수의 표현 (2’ s Complement Number)

✓ 2’ s Complement 의 표현 방법에서 0과 양수 N 은 signed magnitude 표현과 같이 0 뒤에 크기를 붙여 사용한다.

Positive number	*2's Complement Positive number*
0	0000
+1	0001
+2	0010
+3	0011
+4	0100
+5	0101
+6	0110
+7	0111

2’ s Complement 표현방법에서 양수의 표현 방법은 Signed magnitude 표현방법에서의 양수와 동일하게 표현됨

1.4 음수의 표현 (2’ s Complement Number)

✓ 2’ s Complement 의 표현 방법에서 음수 -N 의 표현은 2의 보수 N*으로 표시된다. *n* 비트 word에서 양의 정수 N에 대한 2의 보수 N* 은 다음과 같이 정의된다.

$$N^* = 2^n - N$$

$$(-4) = 4^* = 2^4 - 4$$
$$= 16 - 4 = 12 = 1100_2$$

$$(-0) = 0^* = 2^4 - 0$$
$$= 16 = 10000_2$$

Negative number	*2's Complement Positive number*
−1	1111
−2	1110
−3	1101
−4	1100
−5	1011
−6	1010
−7	1001
−8	1000

1.4 음수의 표현 (2’s Complement Number)

✓ 2’s Complement 와 Signed magnitude 표현의 비교

$+N$	*Signed Magnitude*	*2's Complement*	$-N$	*Signed Magnitude*	*2's Complement*
0	0000	0000	−0	1000	−−
+1	0001	0001	−1	1001	1111
+2	0010	0010	−2	1010	1110
+3	0011	0011	−3	1011	1101
+4	0100	0100	−4	1100	1100
+5	0101	0101	−5	1101	1011
+6	0110	0110	−6	1110	1010
+7	0111	0111	−7	1111	1001
			−8	−−	1000

1.4 음수의 표현 (1’ s Complement Number)

✓ 1’ s Complement 의 표현 방법에서 0과 양수 N 은 signed magnitude 표현과 같이 0 뒤에 크기를 붙여 사용한다.

Positive number	*1's Complement Positive number*
0	0000
+1	0001
+2	0010
+3	0011
+4	0100
+5	0101
+6	0110
+7	0111

1’ s Complement 표현방법에서 양수의 표현 방법은 Signed magnitude 표현방법에서의 양수와 동일하게 표현됨

1.4 음수의 표현 (1’s Complement Number)

✓ 1’s Complement 의 표현 방법에서 음수 -N 의 표현은 1의 보수 $\overline{N}$ 으로 표시된다. *n* 비트 word에서 양의 정수 N에 대한 2의 보수 $\overline{N}$ 은 다음과 같이 정의된다.

$$\overline{N} = (2^n - 1) - N$$

$$(-4) = \overline{4} = (2^4 - 1) - 4$$
$$= 15 - 4 = 11 = 1011_2$$

$$(-0) = \overline{0} = (2^4 - 1) - 0$$
$$= 15 = 1111_2$$

Negative number	*1's Complement Positive number*
-0	1111
-1	1110
-2	1101
-3	1100
-4	1011
-5	1010
-6	1001
-7	1000

1111은 -0을 나타내고 -8은 4비트 워드의 경우에서는 나타낼 수 없다.

1.4 음수의 표현 (1’s Complement Number)

✓ 1’s Complement 와 Signed magnitude 표현의 비교

$+N$	*Signed Magnitude*	*1's Complement*	$\overline{N}$	*Signed Magnitude*	*1's Complement*
0	0000	0000	-0	1000	1111
$+1$	0001	0001	-1	1001	1110
$+2$	0010	0010	-2	1010	1101
$+3$	0011	0011	-3	1011	1100
$+4$	0100	0100	-4	1100	1011
$+5$	0101	0101	-5	1101	1010
$+6$	0110	0110	-6	1110	1001
$+7$	0111	0111	-7	1111	1000
			-8	$--$	$--$

1.4 음수의 표현 (1’s Complement Number)

✓ 1’s Complement 수를 만드는 또 다른 방법은 단순히 N을 비트 별로(bit by bit) 0은 1로 1은 0으로 대체하여 보수를 취하는 것이다.

예를 들어 n = 6이고, N=010101(21)인 경우

$$(-21) = \overline{21} = \overline{010101} = 101010$$

✓ 2’s Complement 수를 만드는 또 다른 방법은 1’s Complement 를 취한 후 1을 더해 구할 수 있다.

$$N^* = 2^n - N = (2^n - 1 - N) + 1 = \overline{N} + 1$$

예를 들어 N=0101100인 경우

$$N = 0101100 \rightarrow 1010011 \rightarrow 1010100 = N^*$$

1.4 음수의 표현 (1's Complement Number)

✓ 1's Complement로 표현되는 음수 $\overline{N}$ (-N)의 크기를 구하는 방법은 다음 식과 같이 음수에 대한 1의 보수를 취하여 구할 수 있다.

$$\overline{N} = (2^n - 1) - N$$

$$N = (2^n - 1) - \overline{N}$$

✓ 2's Complement로 표현되는 음수 N^* (-N)의 크기를 구하는 방법은 다음 식과 같이 음수에 대한 2의 보수를 취하여 구할 수 있다.

$$N^* = 2^n - N$$

$$N = 2^n - N^*$$

보수로 표현되는 음수 -N 이 주어지면, 이 음수에 대한 보수를 다시 취하면 그 음수에 대한 크기를 구할 수 있다.

1.4 음수의 표현 (2's Complement Number의 덧셈)

✓ *n* 비트의 signed binary number에 대한 덧셈은 2의 보수체계를 이용해 구할 수 있다. 모든 수를 양수라고 하여 덧셈을 하고, 부호 비트에서 발생하는 자리올림은 무시하면 된다. 이것은 overflow가 발생하는 경우를 제외하고는 항상 옳은 결과가 된다.

✓ 워드길이가 *n* 비트일 때, 합을 옳게 표현하기 위해서 n 비트 이상이 필요하면 overflow가 발생되었다고 한다.

1.4 음수의 표현 (2's Complement Number의 덧셈)

1. 두 양수의 덧셈, 합 $< 2^{n-1}$

+	2	0	0	1	0
+	5	0	1	0	1
+	7	0	1	1	1

옳은 답

2. 두 양수의 덧셈, 합 $\geq 2^{n-1}$

+	5	0	1	0	1
+	6	0	1	1	0
		1	0	1	1

Overflow로 인해 틀린 답(+11은 부호를 포함하여 5비트로 표현가능)

1.4 음수의 표현 (2’ s Complement Number의 덧셈)

3. 양수와 음수의 덧셈(음수의 크기가 더 큰 경우)

+	5		0	1	0	1
–	6		1	0	1	0
–	1		1	1	1	1

옳은 답

4. 양수와 음수의 덧셈(양수의 크기가 더 큰 경우)

–	5		1	0	1	1
+	6		0	1	1	0
+	1	1	0	0	0	1

부호 비트에서 발생한 자리올림은 무시하면 옳은 답(이것은 overflow가 아님)

1.4 음수의 표현 (2’ s Complement Number의 덧셈)

5. 두 음수의 덧셈 |합| $\leq 2^{n-1}$

–	3		1	1	0	1
–	5		1	0	1	1
–	8	1	1	0	0	0

부호 비트에서 발생한 자리올림은 무시하면 옳은 답(이것은 overflow가 아님)

6. 두 음수의 덧셈 |합| $> 2^{n-1}$

–	5		1	0	1	1
–	6		1	0	1	0
–	11	1	0	1	0	1

Overflow로 인해 틀린 답(-11은 부호를 포함하여 5비트로 표현가능)

1.4 음수의 표현 (1’s Complement Number의 덧셈)

✓1의 보수 덧셈은 2의 보수 덧셈과 유사하나, 다만 마지막 자리올림은 버리지 않고 *n* 비트의 합에 최 우측 자리로 하여 더한다. 이것을 끝단 돌림(end-round) 자리올림이라 한다.

✓ 양수의 덧셈에 대해서는 2의 보수 덧셈에서의 경우 1과 경우 2와 같다.

1.4 음수의 표현 (1의 보수의 덧셈)

3. 양수와 음수의 덧셈(음수의 크기가 더 큰 경우)

+	5		0	1	0	1
−	6		1	0	0	1
−	1		1	1	1	0

옳은 답

4. 양수와 음수의 덧셈(양수의 크기가 더 큰 경우)

−	5		1	0	1	0
+	6		0	1	1	0
		1	0	0	0	0
						1
			0	0	0	1

부호 비트에서 발생한 자리올림을 끝단 돌림 하면 옳은 답
(이것은 overflow가 아님)

1.4 음수의 표현 (1의 보수의 덧셈)

5. 두 음수의 덧셈, |합| < 2^{n-1}

끝단 돌림 자리올림
옳은 답(이것은 overflow가 아님)

–	3		1	1	0	0
–	4		1	0	1	1
		1	0	1	1	1
						1
			1	0	0	0

6. 두 음수의 덧셈, |합| ≥ 2^{n-1}

끝단 돌림 자리올림
Overflow로 인해 틀린 답

–	5		1	0	1	0
–	6		1	0	0	1
		1	0	0	1	1
						1
			0	1	0	0

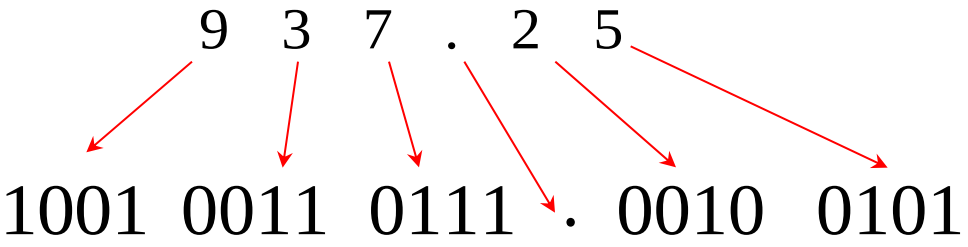
1.5 2진 코드

✓ 대부분의 디지털 시스템은 2진수로 동작

✓ 입출력 장치는10 진수로 동작. 10진수를 2진수로 표현하는 방법이 필요하다.

✓ 이와 같이 10진수를 2진수로 표현하는 것을 부호화 (coding)이라고 한다.

✓ 가장 간단한 2진 부호화의 방법은 다음과 같이 각 10진 숫자를 동등한 2진수로 대체시키는 것이다.



BCD(Binary Coded Decimal)

✓ 이 결과는 전체를 2진으로 변환하는 경우와 매우 다르다는 점을 주목.

✓ 오직 10개의 10진 수가 있기 때문에 1011에서 1111까지는 존재 하지 않는다.

1.5 2진 코드

✓ 유효한 2진 코드가 되기 위한 요구조건은 각 10 개의 10진 수에 대해 서로 다른 2진 숫자들의 조합으로 표현되기만 하면 되는 것이기 때문에, 많은 수의 2진 코드가 존재한다.

✓ 2진 코드의 예

- 8-4-2-1 BCD Code
- 6-3-1-1 Code
- Excess-3 Code
- 2-out-of-5 Code
- Gray Code

1.5 2진 코드

✓가중화 코드 예

8-4-2-1 BCD Code

6-3-1-1 Code

✓4bit 가중화 코드의 가중치가 w_3, w_2, w_1, w_0 일때, 코드 a_3, a_2, a_1, a_0 는 N을 나타낸다

$$N = w_3a_3 + w_2a_2 + w_1a_1 + w_0a_0$$

✓예로 6-3-1-1 코드 가중치는 $w_3 = 6, w_2 = 3, w_1 = 1, w_0 = 1$ 일 때, 2진 코드 1011은 8이 된다.

$$8 = 6 \cdot 1 + 3 \cdot 0 + 1 \cdot 1 + 1 \cdot 1$$

1.5 2진 코드

✓오류 검출 가능 코드

✓2-out-of-5 Code

모든 코드 조합에 대해서 2비트만 1을 갖는다.

✓Gray Code

연속된 10진 숫자에 대한 코드 구성이 1비트씩만 차이가 난다.

Decimal Digit	2-out-of-5 Code	Gray Code
0	00011	0000
1	00101	0001
2	00110	0011
3	01001	0010
4	01010	0110
5	01100	1110
6	10001	1010
7	10010	1011
8	10100	1001
9	11000	1000

1.5 2진 코드

Decimal Digit	8-4-2-1 Code (BCD)	6-3-1-1 Code	Excess-3 Code	2-out-of-5 Code	Gray Code
0	0000	0000	0011	00011	0000
1	0001	0001	0100	00101	0001
2	0010	0011	0101	00110	0011
3	0011	0100	0110	01001	0010
4	0100	0101	0111	01010	0110
5	0101	0111	1000	01100	1110
6	0110	1000	1001	10001	1010
7	0111	1001	1010	10010	1011
8	1000	1011	1011	10100	1001
9	1001	1100	1100	11000	1000

1.5 2진 코드

	ASCII Code								ASCII Code								ASCII Code							
Character	A_6	A_5	A_4	A_3	A_2	A_1	A_0		Character	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Character	A_6	A_5	A_4	A_3	A_2	A_1	A_0
space	0	1	0	0	0	0	0		@	1	0	0	0	0	0	0	‘	1	1	0	0	0	0	0
!	0	1	0	0	0	0	1		A	1	0	0	0	0	0	1	a	1	1	0	0	0	0	1
“	0	1	0	0	0	1	0		B	1	0	0	0	0	1	0	b	1	1	0	0	0	1	0
#	0	1	0	0	0	1	1		C	1	0	0	0	0	1	1	c	1	1	0	0	0	1	1
$	0	1	0	0	1	0	0		D	1	0	0	0	1	0	0	d	1	1	0	0	1	0	0
%	0	1	0	0	1	0	1		E	1	0	0	0	1	0	1	e	1	1	0	0	1	0	1
&	0	1	0	0	1	1	0		F	1	0	0	0	1	1	0	f	1	1	0	0	1	1	0
’	0	1	0	0	1	1	1		G	1	0	0	0	1	1	1	g	1	1	0	0	1	1	1
(	0	1	0	1	0	0	0		H	1	0	0	1	0	0	0	h	1	1	0	1	0	0	0
)	0	1	0	1	0	0	1		I	1	0	0	1	0	0	1	i	1	1	0	1	0	0	1
*	0	1	0	1	0	1	0		J	1	0	0	1	0	1	0	j	1	1	0	1	0	1	0
+	0	1	0	1	0	1	1		K	1	0	0	1	0	1	1	k	1	1	0	1	0	1	1
,	0	1	0	1	1	0	0		L	1	0	0	1	1	0	0	l	1	1	0	1	1	0	0
−	0	1	0	1	1	0	1		M	1	0	0	1	1	0	1	m	1	1	0	1	1	0	1
.	0	1	0	1	1	1	0		N	1	0	0	1	1	1	0	n	1	1	0	1	1	1	0
/	0	1	0	1	1	1	1		O	1	0	0	1	1	1	1	o	1	1	0	1	1	1	1
0	0	1	1	0	0	0	0		P	1	0	1	0	0	0	0	p	1	1	1	0	0	0	0
1	0	1	1	0	0	0	1		Q	1	0	1	0	0	0	1	q	1	1	1	0	0	0	1
2	0	1	1	0	0	1	0		R	1	0	1	0	0	1	0	r	1	1	1	0	0	1	0
3	0	1	1	0	0	1	1		S	1	0	1	0	0	1	1	s	1	1	1	0	0	1	1
4	0	1	1	0	1	0	0		T	1	0	1	0	1	0	0	t	1	1	1	0	1	0	0
5	0	1	1	0	1	0	1		U	1	0	1	0	1	0	1	u	1	1	1	0	1	0	1
6	0	1	1	0	1	1	0		V	1	0	1	0	1	1	0	v	1	1	1	0	1	1	0
7	0	1	1	0	1	1	1		W	1	0	1	0	1	1	1	w	1	1	1	0	1	1	1
8	0	1	1	1	0	0	0		X	1	0	1	1	0	0	0	x	1	1	1	1	0	0	0
9	0	1	1	1	0	0	1		Y	1	0	1	1	0	0	1	y	1	1	1	1	0	0	1
:	0	1	1	1	0	1	0		Z	1	0	1	1	0	1	0	z	1	1	1	1	0	1	0
;	0	1	1	1	0	1	1		[	1	0	1	1	0	1	1	{	1	1	1	1	0	1	1
<	0	1	1	1	1	0	0		\	1	0	1	1	1	0	0	|	1	1	1	1	1	0	0
=	0	1	1	1	1	0	1		]	1	0	1	1	1	0	1	}	1	1	1	1	1	0	1
>	0	1	1	1	1	1	0		^	1	0	1	1	1	1	0	~	1	1	1	1	1	1	0
?	0	1	1	1	1	1	1		—	1	0	1	1	1	1	1	delete	1	1	1	1	1	1	1

요약

- 논리설계 기본
  - Analog 와 Digital의 차이
  - Digital 시스템의 장점
  - 설계 방법(흐름)
- 수 체계의 변환 및 2진 코드
  - 수 체계의 변환 방법
  - 2진 부호 다양한 표현 방법 (음수)
  - 2진 산술연산
  - 2진 코드의 종류