

디지털논리회로 (Digital Logic Circuit)

- Chapter 1

이 수업에서는 무엇을 배우나? - 교과서에서..

Chapter 1 Introduction

컴퓨터에서의 수 체계 (이진수 + alpha)

Chapter 2 Combinational Systems - 가위바위보

진리표로 구현하는 combinational systems

Bool 대수로 표현되는 진리표

게이트로 표현되는 진리표

입력 → Combinational systems → 출력

Chapter 3 The Karnaugh Map (K-map)

진리표를 map으로 표시 → 단순화, 최소화 → 최적의 시스템 만들기

Chapter 5 Designing Combinational Systems

Combinational system(=진리표)로 만들 수 있는 '그럴듯한' 실제 시스템들

- adders, comparators, decoders, encoders, multiplexers..

이 수업에서는 무엇을 배우나? - 교과서에서..

Chapter 6 Analysis of Sequential Systems - 숫자세기

Beyond 진리표

입력+현재 상태→Sequential systems→출력+다음 상태

Chapter 7 The design of Sequential Systems

Sequential system의 실례와 설계

실례1: Counter

설계 방법:

- 1)고객이 의뢰
- 2)우리가 FSM (Finite State Machine) 으로 설계 도면 작성
- 3)직접 만들거나, 컴퓨터보고 시킴

Chapter 8 Solving Larger Sequential Problems

Combinational + Sequential로 이루어진 실제 시스템 다루어 보기

Chapter 1 Introduction

1.1 논리 설계(Logic Design)

1.2 수 체계(Number Systems)

1.2.1 16진수 (Hexadecimal)

1.2.2 2진 덧셈(Binary Addition)

1.2.3 부호화 수(Signed Numbers)

1.2.4 2진 뺄셈(Binary Subtraction)

1.2.5 2진화 십진 코드(Binary Coded Decimal, BCD)

1.2.6 다른 코드들(Other Codes)

Chapter 1 Introduction

1.1 논리 설계(Logic Design)

1.2 수 체계(Number Systems)

1.2.1 16진수 (Hexadecimal)

1.2.2 2진 덧셈(Binary Addition)

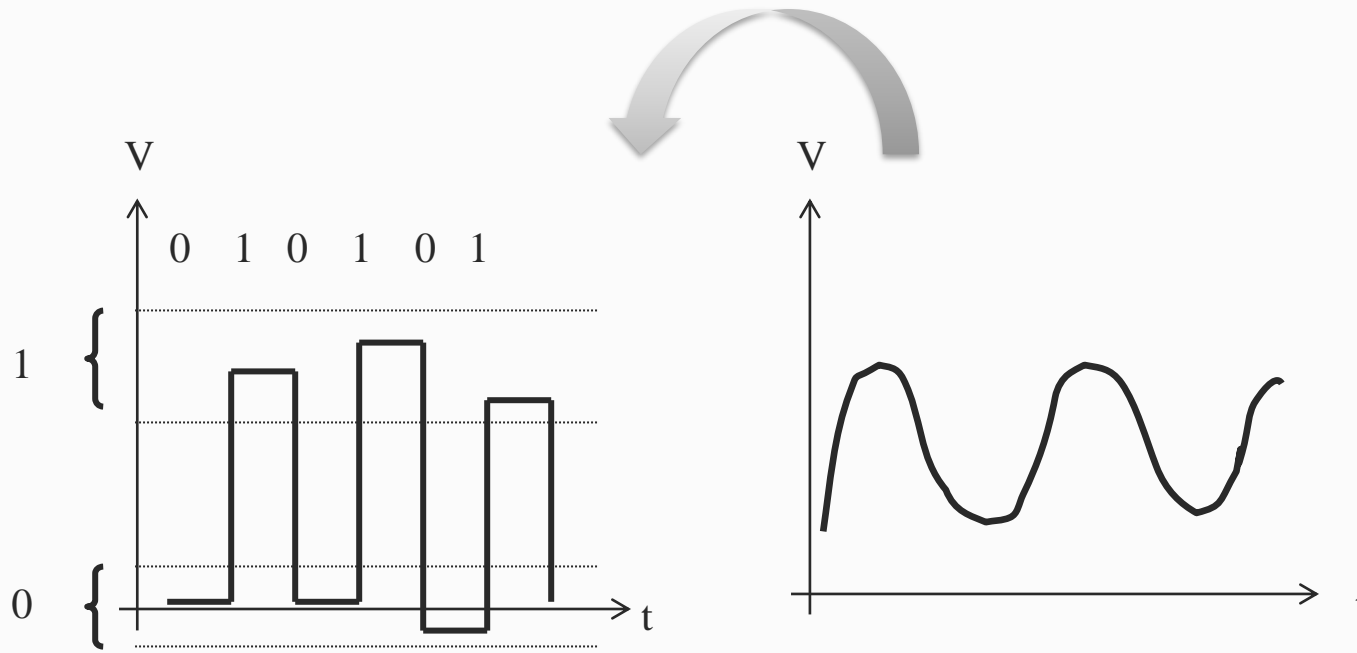
1.2.3 부호화 수(Signed Numbers)

1.2.4 2진 뺄셈(Binary Subtraction)

1.2.5 2진화 십진 코드(Binary Coded Decimal, BCD)

1.2.6 다른 코드들(Other Codes)

Digital vs. Analog

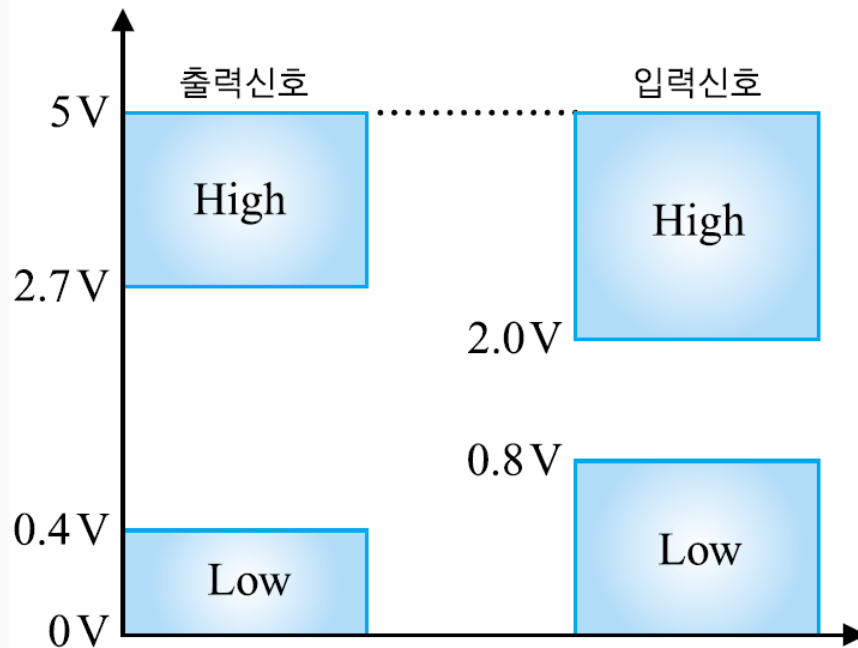


Digital: discrete values

Analog: continuous values

Digital vs. Analog

디지털 정보를 표현하기 위해 2진수 체계(binary system)를 사용
“0”과 “1”만의 2종류의 디지트(digit)를 사용



〈디지털 시스템의 전압레벨〉

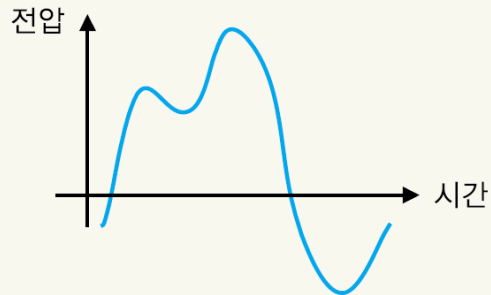
Digital vs. Analog

아날로그 신호(Analog Signal)

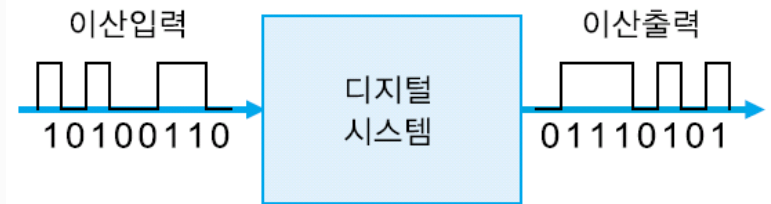
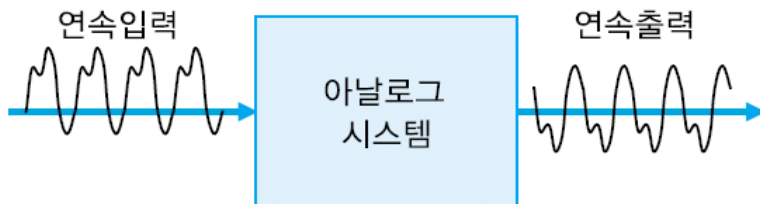
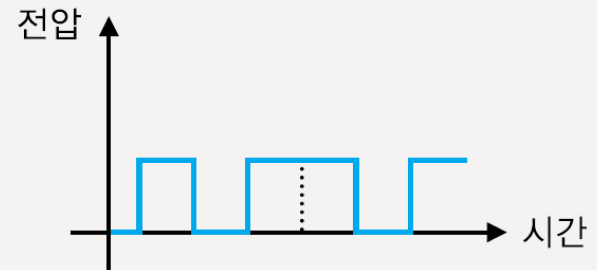
VS

디지털 신호(Digital Signal)

- 자연계에서 일어나는 물리적인 양은 시간에 따라 연속적으로 변화
- 온도, 습도, 소리, 빛 등은 시간에 따라 연속적인 값을 갖는다.

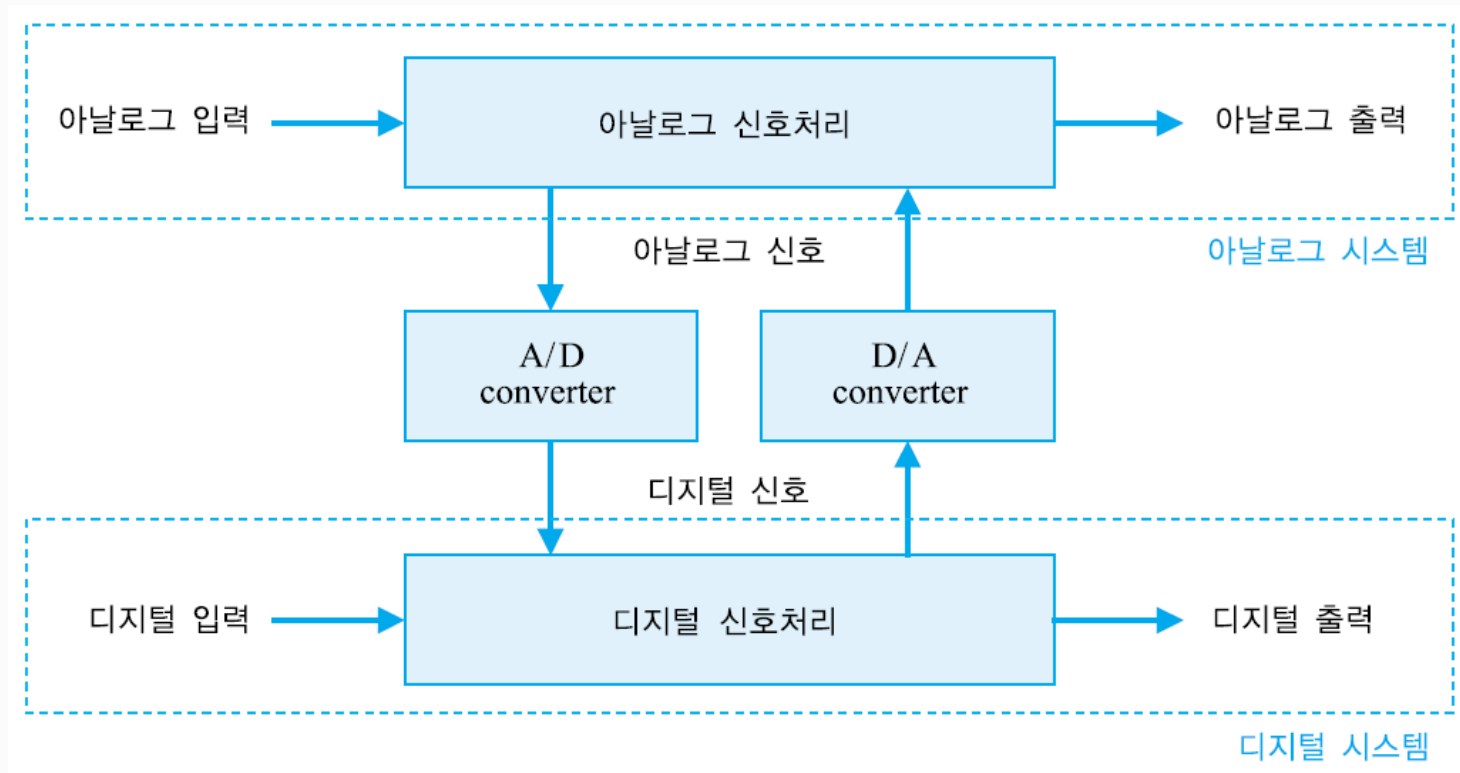


- 분명히 구별되는 두 레벨의 신호값만을 갖는다.



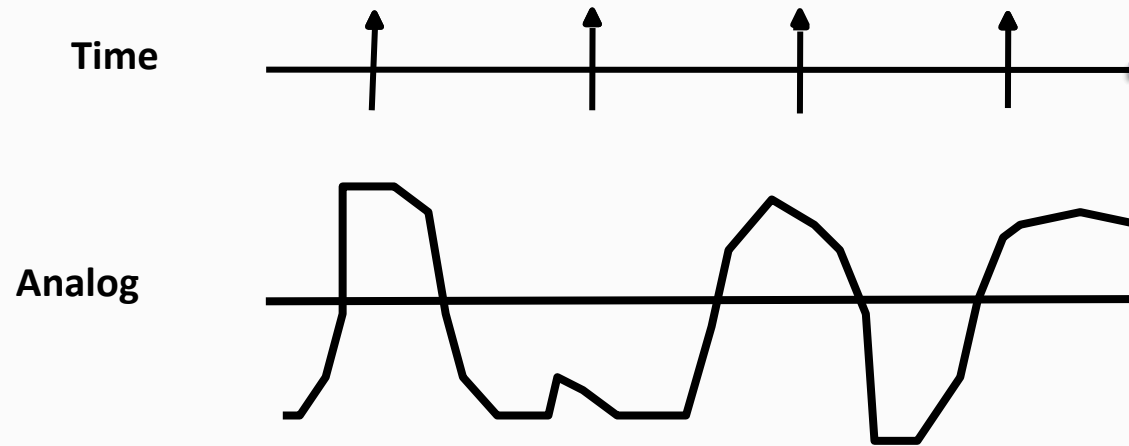
Digital vs. Analog

◇ 아날로그 회로와 디지털 회로의 상호 연결



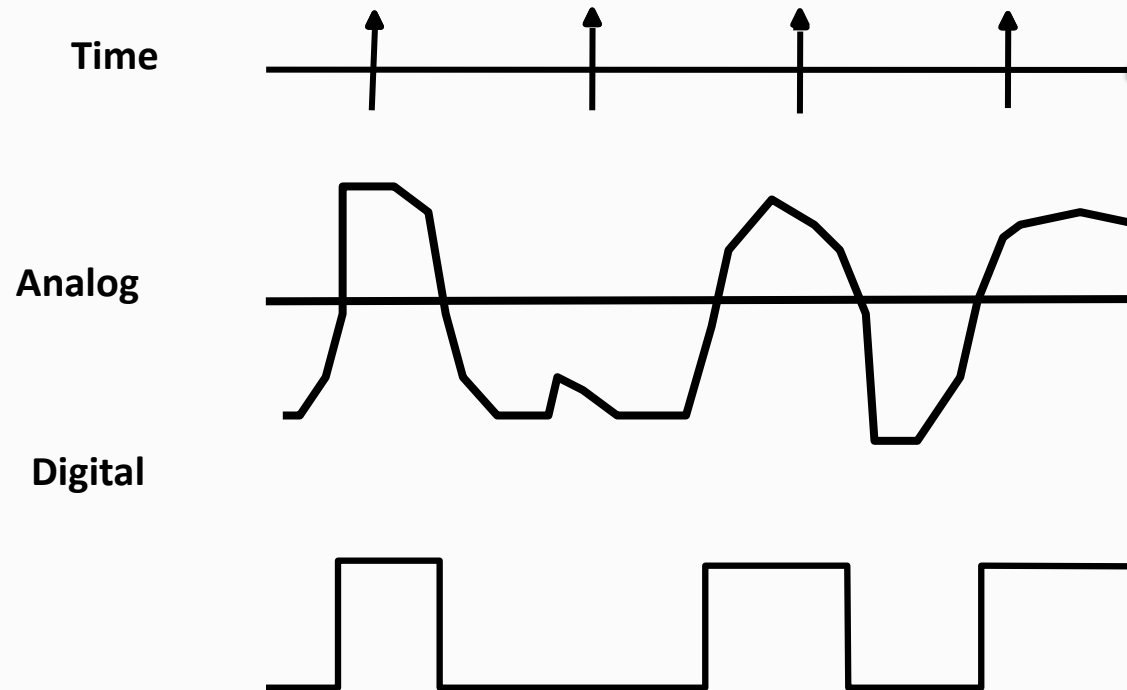
<CD 오디오 시스템에서의 신호처리과정>

Digital vs. Analog



Continuous
in value & time

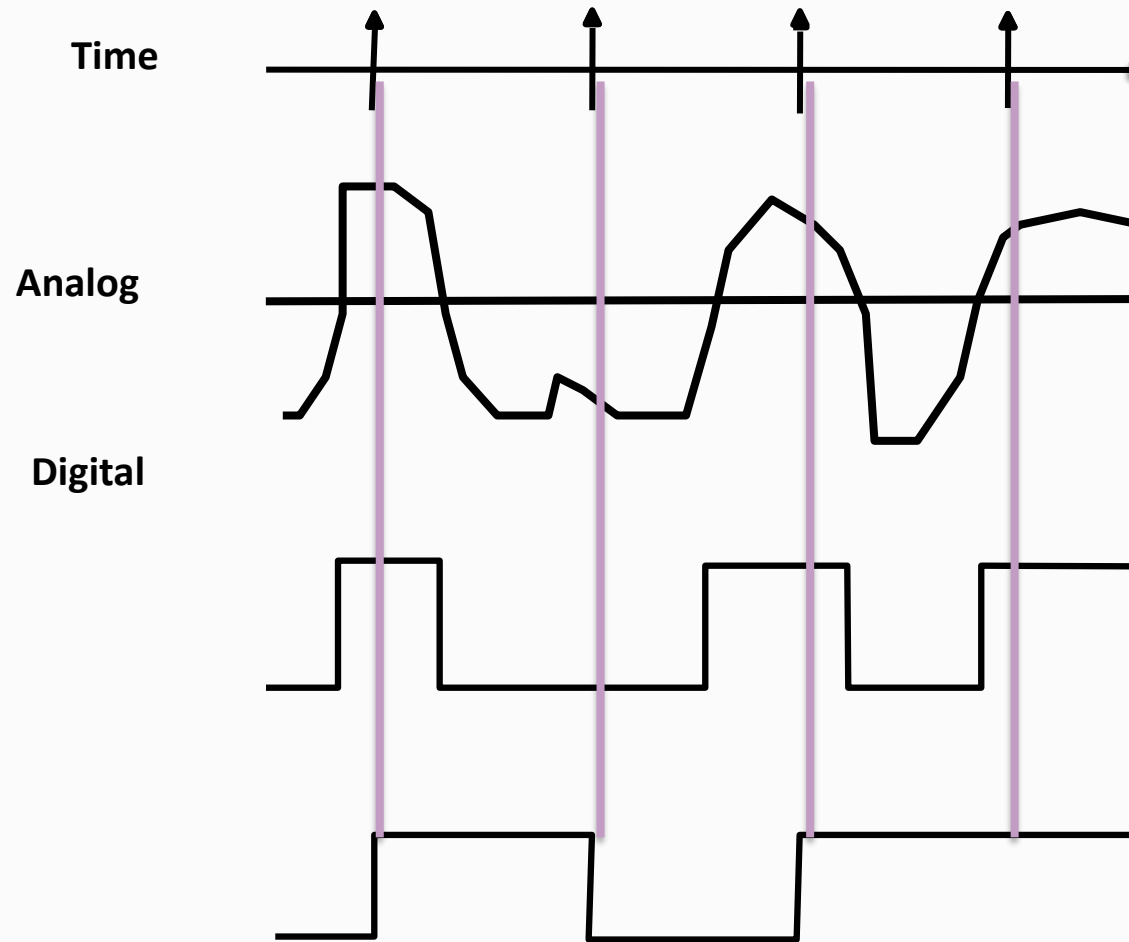
Digital vs. Analog



Continuous
in value & time

Discrete in value but
continuous in time

Digital vs. Analog



Continuous
in value & time

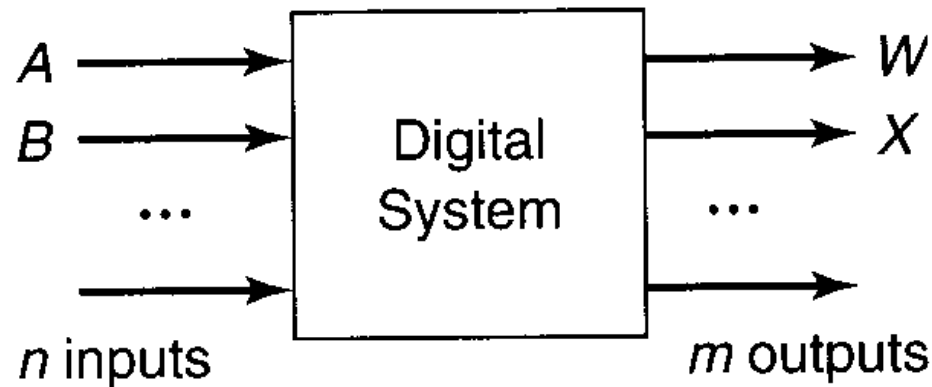
Discrete in value but
continuous in time

Discrete
in value & time

논리 회로 = 디지털 시스템

- 두 가지 신호 값: 0, 1
- 입력: (A, B, ...)
 - 클럭(clock): 특별한 입력. 규칙적으로 0과 1값을 반복적으로 발생
- 출력 : (W, X, ...)

Figure 1.1 A digital system.



논리 회로 = 디지털 시스템

- 두 가지 신호 값: 0, 1
 - 두 가지의 전압(0V, 5V)
 - 스위치의 위와 아래 방향
 - 전등의 on과 off
 - 자계의 두 방향

디지털 시스템의 동작 표현 방법

- 진리표 (Truth Table) 형태
- 대수식 (Algebra) 형태
- 회로도 (Circuit)

예제 1.1

세 개의 입력 A, B, C와 한 개의 출력 Z를 갖는 시스템은 두 개의 입력이 1일 때만 $Z=1$ 이다.

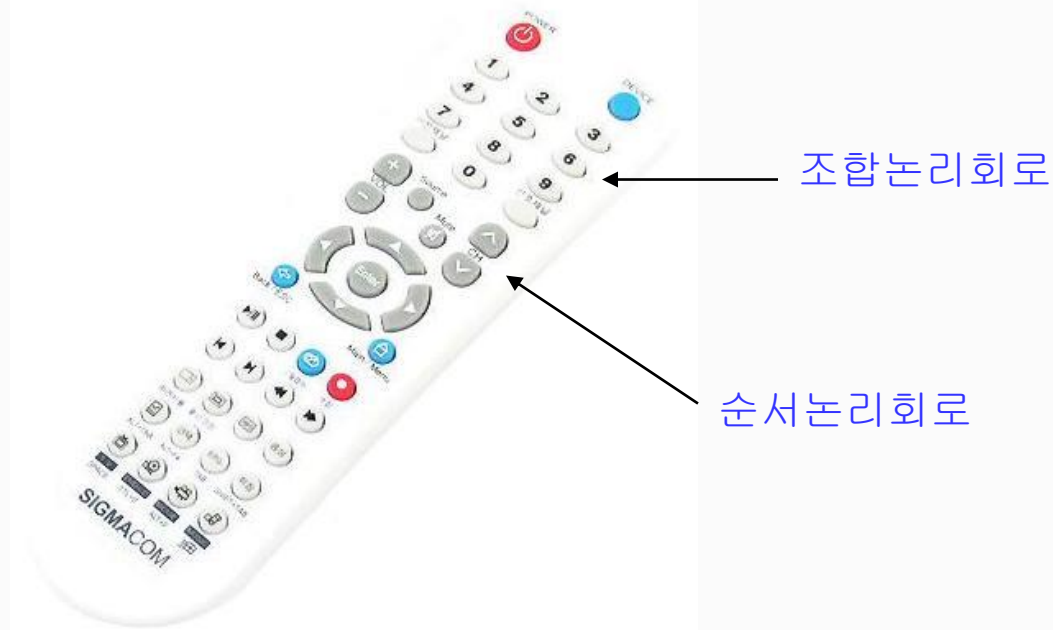
A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

진리표

디지털 시스템

Digital Systems

- 조합 회로(Combinational Circuits)
- 순서 회로 (Sequential Circuits)



디지털 시스템 - *combinational*

조합 회로: 출력이 **입력의 현재 값에만 의존**한다

예제 1.1에서 A, B, C의 현재 값을 안다면, Z가 지금 어떤 값을 갖는 지를 결정할 수 있다

예제 1.1

세 개의 입력 A, B, C와 한 개의 출력 Z를 갖는 시스템은 두 개의 입력이 1일 때만 $Z=1$ 이다.

디지털 시스템 - *sequential*

순차 회로: 출력이 **현재 입력만이 아니고 과거의 입력에도 영향을 받는다.**

즉, 과거의 입력 상태들을 알아야 하기 때문에 메모리(memory)가 필요하게 된다.

Memory + Combinational Circuit

예제 1.3

하나의 입력 A와 클럭, 그리고 하나의 출력 Z로 구성된 시스템에서

마지막 3개의 연속된 클럭에서 **입력이 1일 때만 출력이 1**이 된다.

Memory

Combinational Circuit

Chapter 1 Introduction

1.1 논리 설계(Logic Design)

1.2 수 체계(Number Systems)

1.2.1 16진수 (Hexadecimal)

1.2.2 2진 덧셈(Binary Addition)

1.2.3 부호화 수(Signed Numbers)

1.2.4 2진 뺄셈(Binary Subtraction)

1.2.5 2진화 십진 코드(Binary Coded Decimal, BCD)

1.2.6 다른 코드들(Other Codes)

수 체계

◇ 정수 표현(Integers using positional number systems)

$$N = a_{n-1}r^{n-1} + a_{n-2}r^{n-2} + \dots + a_2r^2 + a_1r + a_0$$

n : number of digits, (정수)

r : radix(base) (진수)

a_i : coefficients $0 \leq a_i < r$ (계수)

$$7642_{10} = 7 \times 10^3 + 6 \times 10^2 + 4 \times 10 + 2$$

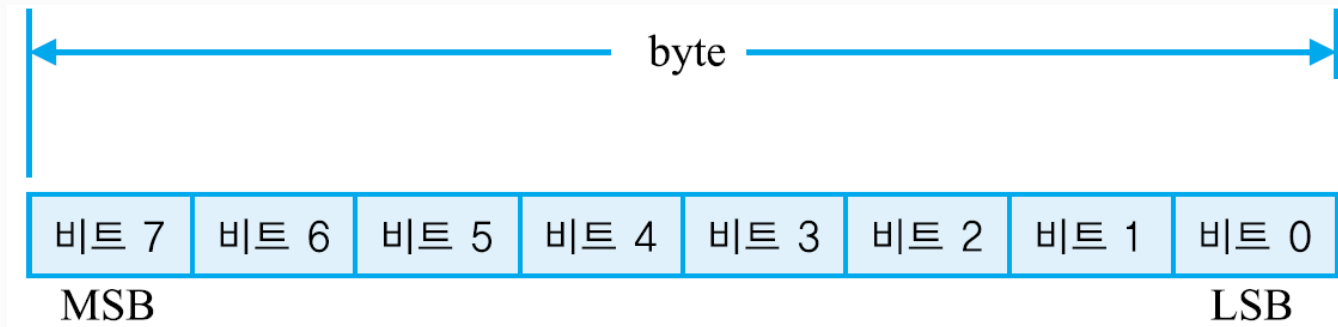
➔ 10진수

$$\begin{aligned} 10111_2 &= 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2 + 1 \\ &= 32 + 8 + 4 + 2 + 1 = 47_{10} \end{aligned}$$

➔ 2진수

수 체계

- ◆ 1byte = 8bit
- ◆ 1byte = 1character
- ◆ 영어는 1byte로 1 문자 표현, 한글은 2byte가 필요
- ◆ 1word : 특정 CPU에서 취급하는 명령어나 데이터의 길이에 해당하는 비트 수



MSB(most significant bit) : 최상위비트

LSB(least significant bit) : 최하위비트

수 체계

Table 1.3 First 32 binary integers.

Decimal	Binary	4-bit	Decimal	Binary
0	0	0000	16	10000
1	1	0001	17	10001
2	10	0010	18	10010
3	11	0011	19	10011
4	100	0100	20	10100
5	101	0101	21	10101
6	110	0110	22	10110
7	111	0111	23	10111
8	1000	1000	24	11000
9	1001	1001	25	11001
10	1010	1010	26	11010
11	1011	1011	27	11011
12	1100	1100	28	11100
13	1101	1101	29	11101
14	1110	1110	30	11110
15	1111	1111	31	11111

Note:

bits = binary+digits

- N bit 양의 정수의 범위: $0 \dots 2^n - 1$
- $2^n - 1 = 111\dots1$

Powers of Two

Table 1.1
Powers of Two

<i>n</i>	<i>2ⁿ</i>	<i>n</i>	<i>2ⁿ</i>	<i>n</i>	<i>2ⁿ</i>
0	1	8	256	16	65,536
1	2	9	512	17	131,072
2	4	10	1,024 (1K)	18	262,144
3	8	11	2,048	19	524,288
4	16	12	4,096 (4K)	20	1,048,576 (1M)
5	32	13	8,192	21	2,097,152
6	64	14	16,384	22	4,194,304
7	128	15	32,768	23	8,388,608

수 체계

Convert $(857)_{10}$

			Remainder	
$857 \div 2$	$=$	428	1	LSB
$428 \div 2$	$=$	214	0	
$214 \div 2$	$=$	107	0	
$107 \div 2$	$=$	53	1	
$53 \div 2$	$=$	26	1	
$26 \div 2$	$=$	13	0	
$13 \div 2$	$=$	6	1	
$6 \div 2$	$=$	3	0	
$3 \div 2$	$=$	1	1	
$1 \div 2$	$=$	0	1	MSB

Result is $(1101011001)_2$

Figure 5.1. Conversion from decimal to binary.

Chapter 1 Introduction

1.1 논리 설계(Logic Design)

1.2 수 체계(Number Systems)

1.2.1 16진수 (Hexadecimal)

1.2.2 2진 덧셈(Binary Addition)

1.2.3 부호화 수(Signed Numbers)

1.2.4 2진 뺄셈(Binary Subtraction)

1.2.5 2진화 십진 코드(Binary Coded Decimal, BCD)

1.2.6 다른 코드들(Other Codes)

16진수 (Hexadecimal)



※ 10이상의 16진법수

10 = A

11 = B

12 = C

13 = D

14 = E

15 = F

◇ 왜 16진수 필요?

◆ 2진수가 너무 기니까~ 짧게 나타낼 수 있는 방법

예제 1.9: 2 진수에서 16진수로 변환

$$\begin{aligned} 1011101010_2 &= 0010\ 1110\ 1010_2 \\ &= 2EA_{16} \end{aligned}$$

Chapter 1 Introduction

1.1 논리 설계(Logic Design)

1.2 수 체계(Number Systems)

1.2.1 16진수 (Hexadecimal)

1.2.2 2진 덧셈(Binary Addition)

1.2.3 부호화 수(Signed Numbers)

1.2.4 2진 뺄셈(Binary Subtraction)

1.2.5 2진화 십진 코드(Binary Coded Decimal, BCD)

1.2.6 다른 코드들(Other Codes)

2진 덧셈 (Binary Addition)

예제 1.12

Carry bit	→	1 1		
Input bit	[0 1 1 0	6	
		0 1 1 1	7	
<hr/>				
Sum bit	→	1 1 0 1	13	

4 bit로는
자리가 부족하다!
→ Overflow

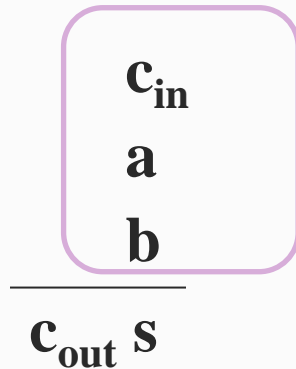
	1 1			
	0 1 1 0	6		
	1 1 1 1	15		
<hr/>				
1	0 1 0 1	21		

→ 2개의 4bit 입력의 합은
4bit 또는 5bit

각 자리 수마다 () 개의 숫자를 더하고 있다

2진 덧셈 (Binary Addition)

1 bit 덧셈 회로 표현:



일반적으로 표현하면 3개의 1bit (c_{in} , a , b)를 더하는 회로가 필요

Table 1.5 One-bit adder.

a	b	c_{in}	c_{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

<입력>

<출력>

Addition은 combinational logic

2진 덧셈 (Binary Addition)

4bit 덧셈회로

Figure 1.2 A 4-bit adder.

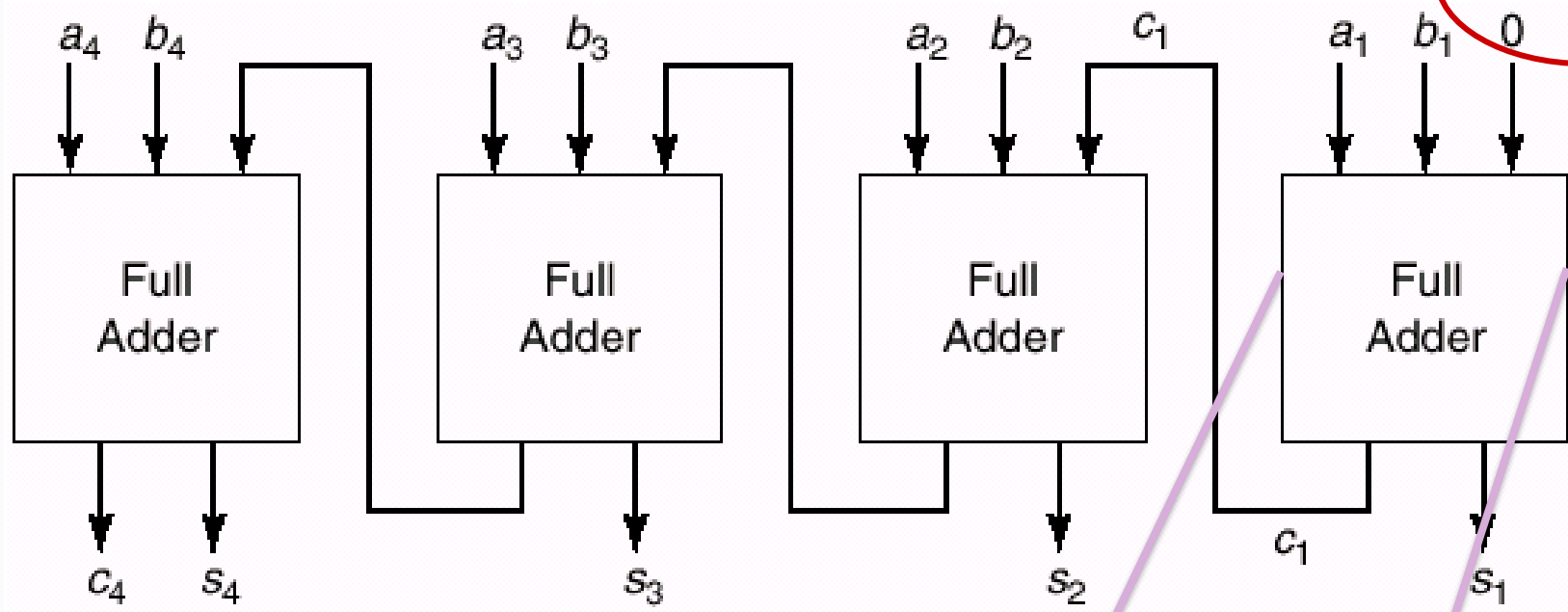
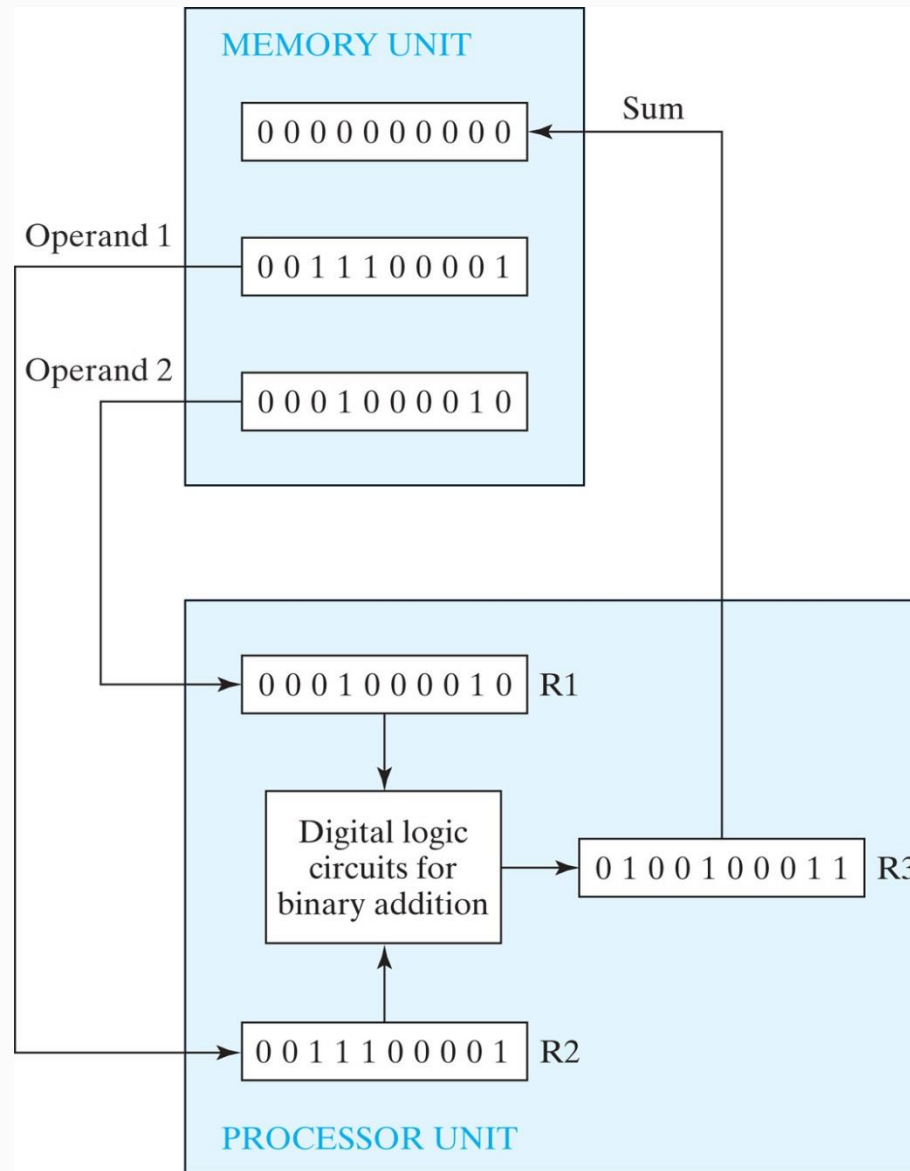


Table 1.5 One-bit adder.

a	b	c_{in}	c_{out}	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Example of binary information processing



Chapter 1 Introduction

1.1 논리 설계(Logic Design)

1.2 수 체계(Number Systems)

1.2.1 16진수 (Hexadecimal)

1.2.2 2진 덧셈(Binary Addition)

1.2.3 부호화 수(Signed Numbers)

1.2.4 2진 뺄셈(Binary Subtraction)

1.2.5 2진화 십진 코드(Binary Coded Decimal, BCD)

1.2.6 다른 코드들(Other Codes)

부호화 수 (Signed Numbers)

- 무부호화 수(*unsigned numbers*) : 양의 정수 (positive integers)
- 부호화 수 (*signed numbers*) : 양수 와 음수 (positive and negative numbers)

부호화 수 (Signed Numbers)

+5
-5



+101
-101

부호를 어떻게 표현할 것인가!

방법 1 : 부호(Sign) + 크기(Magnitude)

+5
-5



+101
-101

0101

1101

방법 1 : 부호(Sign) + 크기(Magnitude)

$$\begin{array}{r} +5 \\ -5 \end{array} \quad \longrightarrow \quad \begin{array}{r} 0101 \\ 1101 \end{array}$$

➔ 직관적이고 좋긴 한데.. 불편할 때도 있다.

$\begin{array}{r} +5 \\ +3 \\ \hline +8 \end{array}$	$\begin{array}{r} -5 \\ -3 \\ \hline -8 \end{array}$	$\begin{array}{r} +5 \\ -3 \\ \hline +2 \end{array}$	$\begin{array}{r} -5 \\ +3 \\ \hline -2 \end{array}$	$\begin{array}{r} -3 \\ +5 \\ \hline +2 \end{array}$	$\begin{array}{r} +3 \\ -5 \\ \hline -2 \end{array}$
--	--	--	--	--	--

- 큰 수 작은 수 파악 해야 함
- 그에 따라 결과 값에 적절한 부호 붙이기

방법 2 : 2의 보수 (2's complement)

Table 1.6 Signed and unsigned 4-bit numbers.

Binary	Positive	Signed (two's complement)
0000	0	0
0001	1	+ 1
0010	2	+ 2
0011	3	+ 3
0100	4	+ 4
0101	5	+ 5
0110	6	+ 6
0111	7	+ 7
1000	8	- 8
1001	9	- 7
1010	10	- 6
1011	11	- 5
1100	12	- 4
1101	13	- 3
1110	14	- 2
1111	15	- 1

방법 2 : 2의 보수 (2's complement)

음수 만드는 법

1. 크기에 해당하는 2 진수 값을 찾는다.
2. 각 비트에 대한 보수를 취한다.
3. 1을 더한다.

예제 1.13

-5?	-1?	-0?
1. 5: 0101	1: 0001	0: 0000
2. 1010	1110	1111
3. +1	+1	+1
-5: 1011	-1: 1111	0000

음수에서 양수로,
양수에서 음수로
부호를 바꿀 때도
동일한 방법을
쓸 수 있습니다!

방법 2 : 2의 보수 (2's complement)

이제 더하기/빼기 연산의 불편한 점이 해결 되었을까?

$$\begin{array}{r}
 -5 \\
 +7 \\
 \hline
 +2
 \end{array}
 \quad
 \begin{array}{r}
 1011 \\
 0111 \\
 \hline
 (1) 0010
 \end{array}$$

무조건 더하면 됨

$$\begin{array}{r}
 -5 \\
 +3 \\
 \hline
 -2
 \end{array}
 \quad
 \begin{array}{r}
 1011 \\
 0011 \\
 \hline
 (0) 1110
 \end{array}$$

Carry는 무시

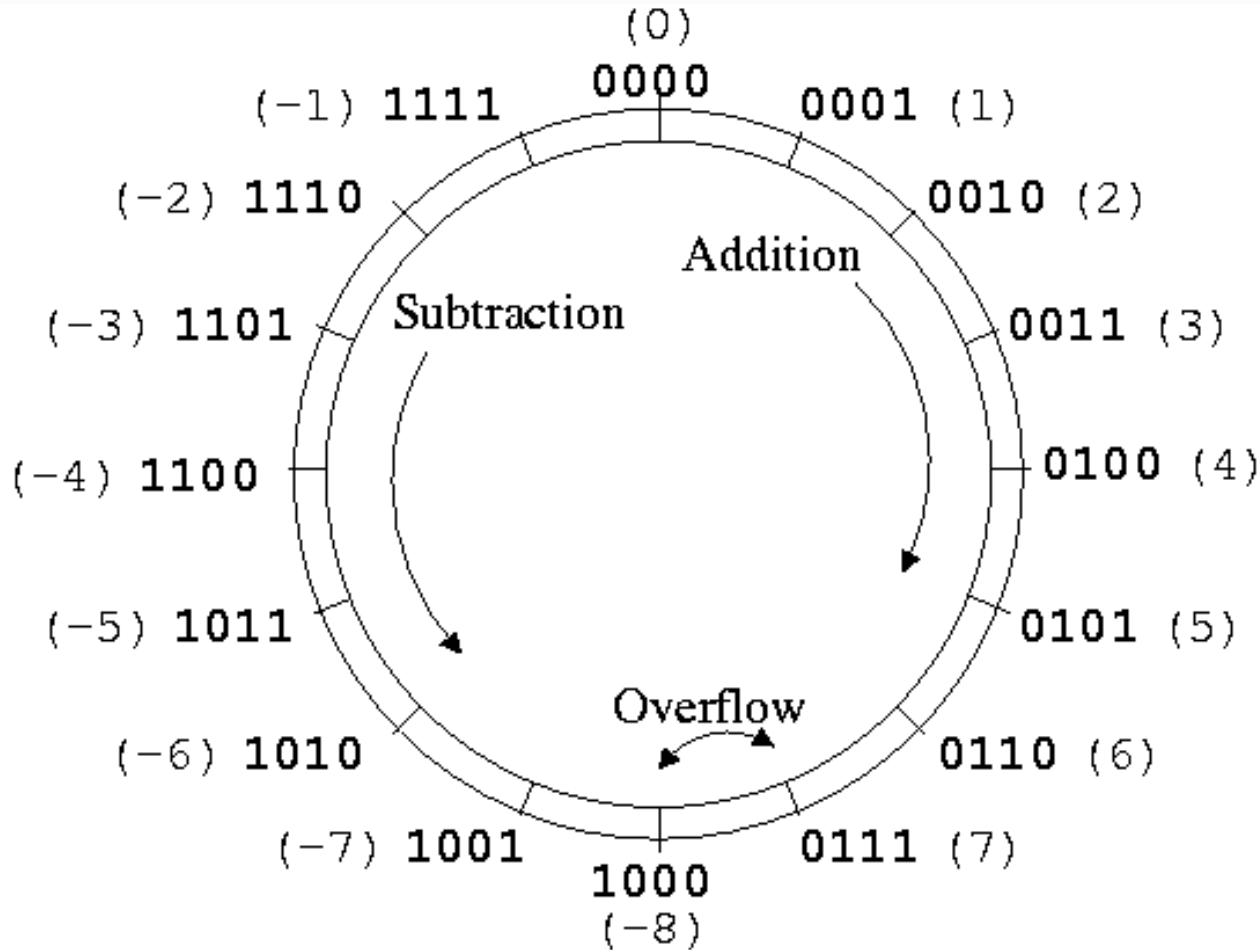
Table 1.6 Signed and unsigned 4-bit numbers.

Binary	Positive	Signed (two's complement)
0000	0	0
0001	1	+ 1
0010	2	+ 2
0011	3	+ 3
0100	4	+ 4
0101	5	+ 5
0110	6	+ 6
0111	7	+ 7
1000	8	- 8
1001	9	- 7
1010	10	- 6
1011	11	- 5
1100	12	- 4
1101	13	- 3
1110	14	- 2
1111	15	- 1

부호가 **다른** 숫자 계산이 편리 해졌음!

방법 2 : 2의 보수 (2's complement)

그러나
주의할 점!!
Overflow!



부호가 **같은** 계산에서는 overflow 특히 조심해야 함

방법 2 : 2의 보수 (2's complement)

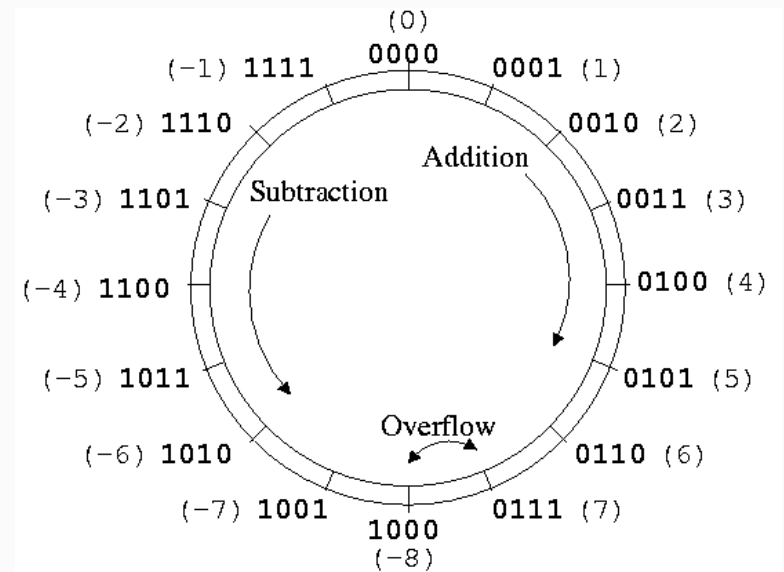
그러나
주의할 점!!
Overflow!

예제 1.16: overflow

$$\begin{array}{r} +5 \quad \quad 0101 \\ +4 \quad \quad 0100 \\ \hline (0) \quad 1001 \end{array} \quad (\text{looks like } -7)$$

예제 1.17: overflow

$$\begin{array}{r} -5 \quad \quad 1011 \\ -4 \quad \quad 1100 \\ \hline (1) \quad 0111 \end{array} \quad (\text{looks like } +7)$$



방법 2 : 2의 보수 (2's complement)

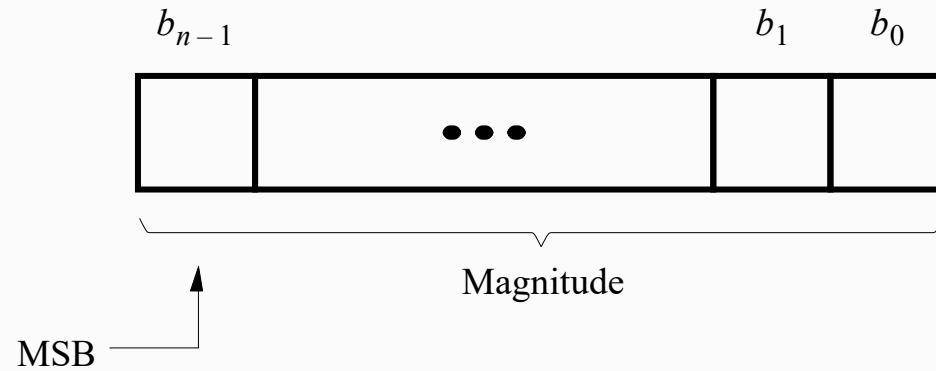
그러나
주의할 점!!
Overflow!

예제 1.16: overflow

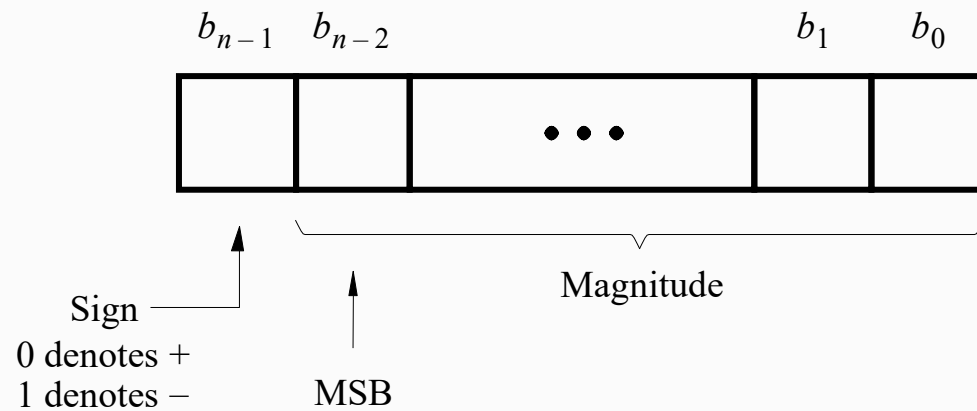
$$\begin{array}{r} +5 \qquad 0101 \\ +4 \qquad 0100 \\ \hline (0) 1001 \end{array} \quad (\text{looks like } -7)$$

답이 9 맞지 않나요?
왜 틀렸지요?

9인지 -7인지 어떻게
구별하지?



(a) Unsigned number



(b) Signed number

Formats for representation of integers.

Chapter 1 Introduction

1.1 논리 설계(Logic Design)

1.2 수 체계(Number Systems)

1.2.1 16진수 (Hexadecimal)

1.2.2 2진 덧셈(Binary Addition)

1.2.3 부호화 수(Signed Numbers)

1.2.4 2진 뺄셈(Binary Subtraction)

1.2.5 2진화 십진 코드(Binary Coded Decimal, BCD)

1.2.6 다른 코드들(Other Codes)

2진수 뺄셈

- 뺄셈: 2의 보수를 취하여 더한다.

$$a - b = a + (-b)$$

예제 1.18 : $7 - 5 = 7 + (-5)$

5:	0101	7:	0111
	1010	-5:	+1011
+	1	<hr/>	<hr/>
		2	(1) 0010
<hr/>			
-5:	1011		

5의 2의 보수를 구하고

더해준다.

Chapter 1 Introduction

1.1 논리 설계(Logic Design)

1.2 수 체계(Number Systems)

1.2.1 16진수 (Hexadecimal)

1.2.2 2진 덧셈(Binary Addition)

1.2.3 부호화 수(Signed Numbers)

1.2.4 2진 뺄셈(Binary Subtraction)

1.2.5 2진화 십진 코드(Binary Coded Decimal, BCD)

1.2.6 다른 코드들(Other Codes)

BCD: Binary Coded Decimal

BCD코드는 10진수 0(0000)부터 9(1001)까지를 2진화한 코드
표기는 2진수이지만 의미는 10진수

8421code+3

Decimal digit	8421 code	5421 code	2421 code	Excess 3 code	2 of 5 code
0	0000	0000	0000	0011	11000
1	0001	0001	0001	0100	10100
2	0010	0010	0010	0101	10010
3	0011	0011	0011	0110	10001
4	0100	0100	0100	0111	01100
5	0101	1000	1011	1000	01010
6	0110	1001	1100	1001	01001
7	0111	1010	1101	1010	00110
8	1000	1011	1110	1011	00101
9	1001	1100	1111	1100	00011
unused	1010	0101	0101	0000	any of the 22 patterns with 0, 1, 3, 4, or 5 1's
	1011	0110	0110	0001	
	1100	0111	0111	0010	
	1101	1101	1000	1101	
	1110	1110	1001	1110	
	1111	1111	1010	1111	

5개 중 2개
만 항상 1,
오류 검출
용이

*가중치 (weighted) 코드: 8421, 5421, 2421 코드

ASCII 코드

zone bit			digit bit			
6	5	4	3	2	1	0
1	0	0	영문자 A~O(0001~1111)			
1	0	1	영문자 P~Z(0000~1010)			
0	1	1	숫자 0~9(0000~1001)			

$a_3a_2a_1a_0$	$a_6a_5a_4$					
	010	011	100	101	110	111
0000	space	0	@	P	`	p
0001	!	1	A	Q	a	q
0010	"	2	B	R	b	r
0011	#	3	C	S	c	s
0100	\$	4	D	T	d	t
0101	%	5	E	U	e	u
0110	&	6	F	V	f	v
0111	'	7	G	W	g	w
1000	(8	H	X	h	x
1001)	9	I	Y	i	y
1010	*	:	J	Z	j	z
1011	+	;	K	[k	{
1100	,	<	L	\	l	
1101		=	M]	m	}
1110	.	>	N	^	n	~
1111	/	?	O	_	o	delete

Gray 코드

가중치가 없는 코드이기 때문에 연산에는 부적당하지만, 아날로그-디지털 변환기나 입출력 장치 코드로 주로 쓰인다.

연속되는 코드들 간에 하나의 비트만 변화하여 새로운 코드가 된다.

Number	Gray code	Number	Gray code
0	0000	8	1100
1	0001	9	1101
2	0011	10	1111
3	0010	11	1110
4	0110	12	1010
5	0111	13	1011
6	0101	14	1001
7	0100	15	1000

값이 1 증가할 때 마다 1개 자리수만 변경되는 코드

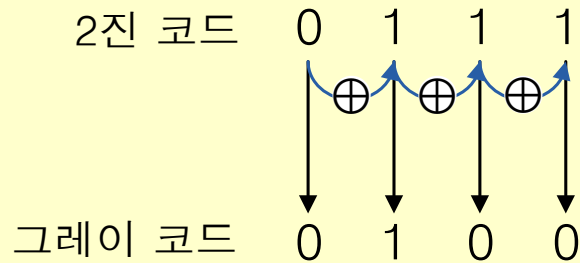
7→8

0100→1000 vs. 0100→1100

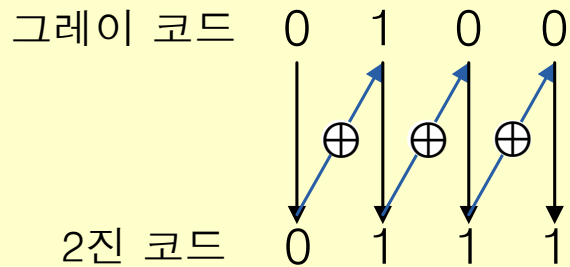
산술 연산이 아닌 입출력 장치와 A/D 변환기 등의 응용 장치에서 사용

Gray 코드

2진 코드를 그레이 코드로 변환하는 방법



그레이 코드를 2진 코드로 변환하는 방법



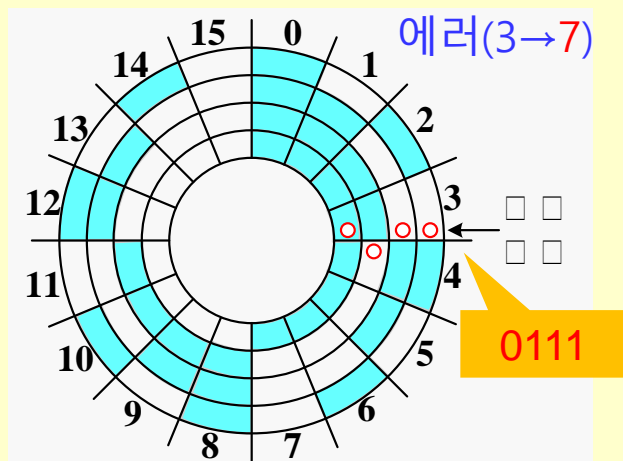
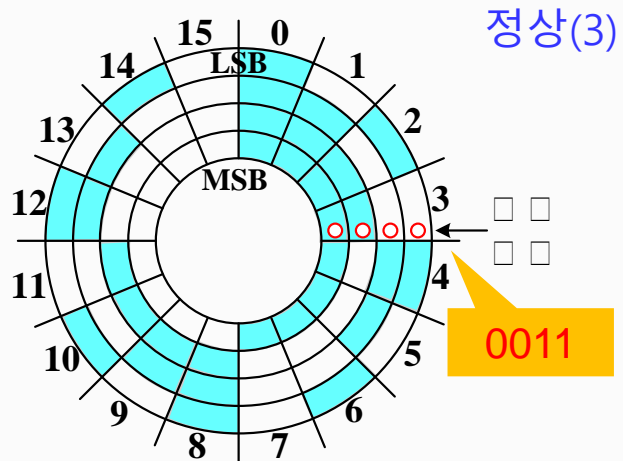
<XOR 진리표>

입력		출력
A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

$$F = A \oplus B$$

Gray 코드

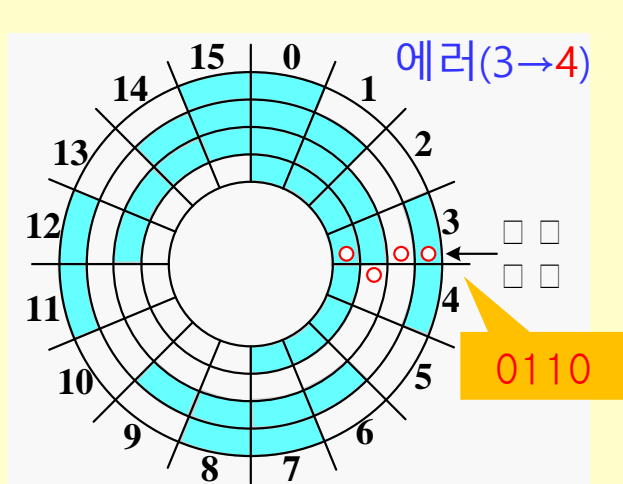
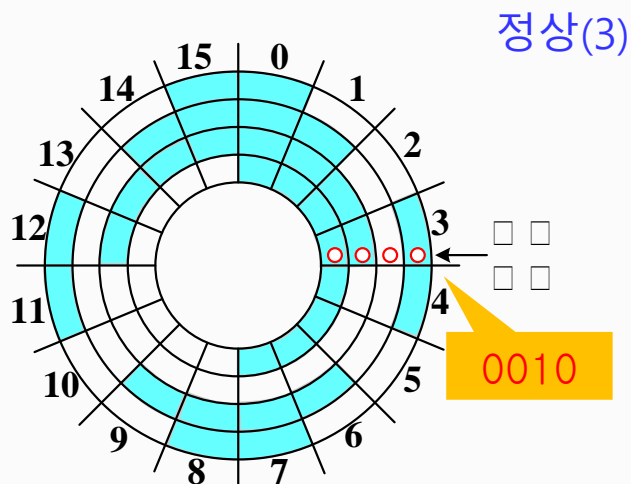
2진 코드



0

1

그레이 코드

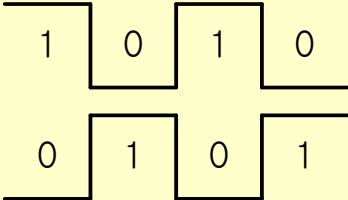
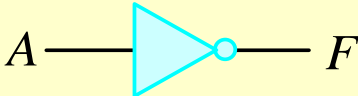


그레이
코드는
오차가
적음

Logic gates

NOT gate

한 개의 입력과 한 개의 출력을 갖는 게이트로 논리 부정을 나타낸다.

진리표	동작파형	논리기호						
<table><tr><th>A</th><th>F</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	F	0	1	1	0	<div><div>입력 A</div><div>출력 F</div></div> 	
A	F							
0	1							
1	0							

논리식
$F = \overline{A} = A'$

AND gate

입력이 모두 1(on)인 경우에만 출력은 1(on)이 되고, 입력 중에 0(off)인 것이 하나라도 있을 경우에는 출력은 0(off)이 된다.

진리표	동작파형	논리기호															
<table border="1"> <thead> <tr> <th><i>A</i></th><th><i>B</i></th><th><i>F</i></th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>0</td></tr> <tr> <td>0</td><td>1</td><td>0</td></tr> <tr> <td>1</td><td>0</td><td>0</td></tr> <tr> <td>1</td><td>1</td><td>1</td></tr> </tbody> </table>	<i>A</i>	<i>B</i>	<i>F</i>	0	0	0	0	1	0	1	0	0	1	1	1	<p>The timing diagram shows three signals over five time intervals. Signal A is 0, 0, 1, 1, 0. Signal B is 0, 1, 0, 1, 0. The output signal F is 0, 0, 0, 1, 0, which matches the truth table where F is 1 only when both A and B are 1.</p>	
<i>A</i>	<i>B</i>	<i>F</i>															
0	0	0															
0	1	0															
1	0	0															
1	1	1															
		논리식															
		$F = AB = A \cdot B$															

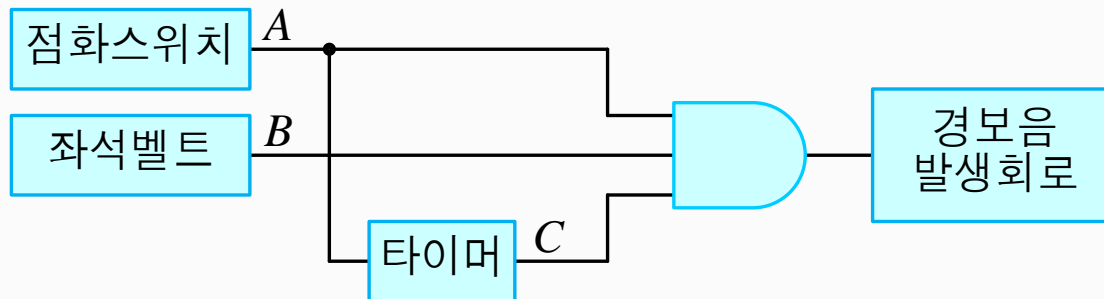
AND gate

진리표	동작파형	논리식																																				
<table><tr><th>A</th><th>B</th><th>C</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	A	B	C	F	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0	0	1	0	1	0	1	1	0	0	1	1	1	1	<div><div>A</div><div>B</div><div>C</div><div>F</div></div>	<div>$F = ABC = A \cdot B \cdot C$</div> <div>논리기호</div> <div><div>A</div><div>B</div><div>C</div><div>F</div></div>
A	B	C	F																																			
0	0	0	0																																			
0	0	1	0																																			
0	1	0	0																																			
0	1	1	0																																			
1	0	0	0																																			
1	0	1	0																																			
1	1	0	0																																			
1	1	1	1																																			
IC 7411 핀 배치도																																						
<div><div>VCC</div><div>1C</div><div>1Y</div><div>3C</div><div>3B</div><div>3A</div><div>3Y</div><div>14</div><div>13</div><div>12</div><div>11</div><div>10</div><div>9</div><div>8</div><div>1</div><div>2</div><div>3</div><div>4</div><div>5</div><div>6</div><div>7</div><div>1A</div><div>1B</div><div>2A</div><div>2B</div><div>2C</div><div>2Y</div><div>GND</div></div>																																						

5

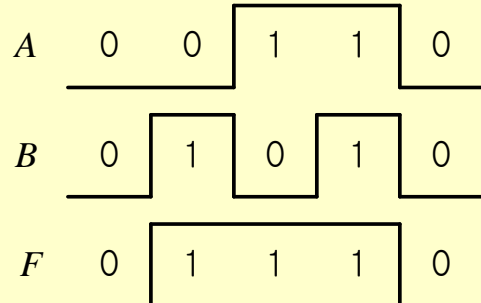
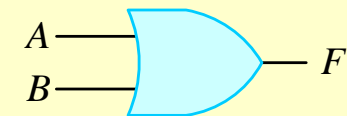
AND gate

- ◇ AND 게이트를 이용한 자동차 좌석벨트 경보 시스템
 - ◆ 점화스위치(A)가 켜지고(High) 좌석벨트(B)가 풀려있는 상태(High)를 감지
 - ◆ 점화스위치가 켜지면 타이머가 작동되어 타이머 C가 30초 동안 High로 유지
 - ◆ 점화 스위치가 켜지고, 좌석벨트가 풀려있고, 타이머가 작동하는 3가지 조건 하에서 AND 게이트의 출력은 High가 되며, 운전자에게 주의를 환기시키는 경보음이 울리게 된다.
 - ◆ 30초간 경보음 동작 후에는 경보음은 울리지 않으며, 처음부터 좌석벨트가 채워져 있으면 경보음은 울리지 않는다.



OR gate

입력이 모두 0인 경우에만 출력은 0이 되고, 입력 중에 1이 하나라도 있으면, 출력은 1이 된다.

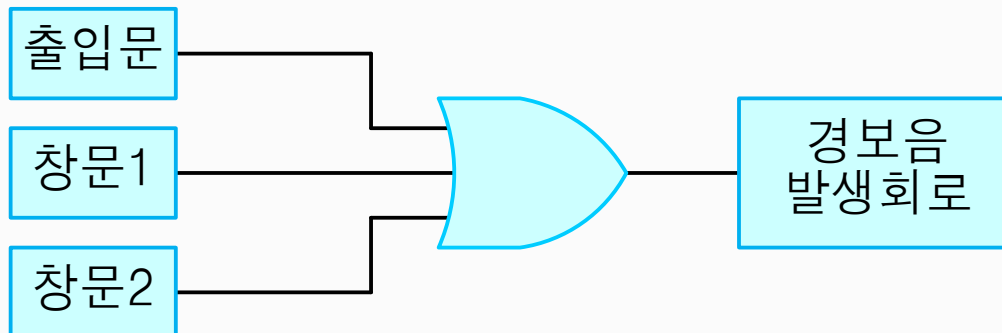
진리표	동작파형	논리식															
<table><tr><th><i>A</i></th><th><i>B</i></th><th><i>F</i></th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	<i>A</i>	<i>B</i>	<i>F</i>	0	0	0	0	1	1	1	0	1	1	1	1		$F = A + B$
<i>A</i>	<i>B</i>	<i>F</i>															
0	0	0															
0	1	1															
1	0	1															
1	1	1															
		논리기호															
																	

OR gate

진리표	동작파형	논리식																																				
<table><tr><th>A</th><th>B</th><th>C</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	A	B	C	F	0	0	0	0	0	0	1	1	0	1	0	1	0	1	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	1	1	1	<div><div>A</div><div>0 0 0 0 1 1 1 1 0</div></div> <div><div>B</div><div>0 0 1 1 0 0 1 1 0</div></div> <div><div>C</div><div>0 1 0 1 0 1 0 1 0</div></div> <div><div>F</div><div>0 1 1 1 1 1 1 1 0</div></div>	<div>$F = A + B + C$</div> <div>논리기호</div> <div><div>A</div><div>B</div><div>C</div><div></div><div>F</div></div>
A	B	C	F																																			
0	0	0	0																																			
0	0	1	1																																			
0	1	0	1																																			
0	1	1	1																																			
1	0	0	1																																			
1	0	1	1																																			
1	1	0	1																																			
1	1	1	1																																			

OR gate

- ◇ OR 게이트를 이용한 침입 탐지 시스템
 - ◆ 일반 가정에서 출입문 1개와 창문 2개가 있다고 가정
 - ◆ 출입문과 창문에 설치된 각 센서는 자기 스위치(magnetic switch)로서 문이 열려 있을 때 High를 출력하고, 닫혀있을 때에는 Low를 출력한다.




NAND gate

- ◆ 입력이 모두 1인 경우에만 출력은 0이 되고, 그렇지 않을 경우에는 출력은 1이 된다.
- ◆ 이 게이트는 AND 게이트와는 반대로 작동하는 게이트로서, NOT AND의 의미로 NAND 게이트라고 부른다.

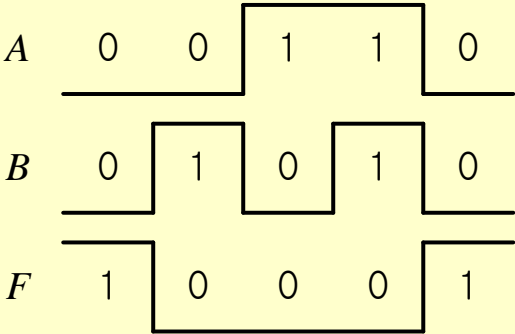
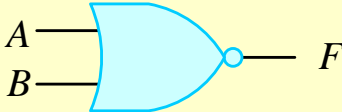
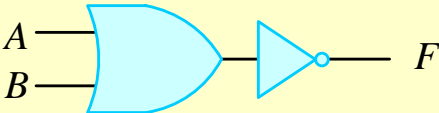
진리표	동작파형	논리식															
<table border="1"> <thead> <tr> <th>A</th><th>B</th><th>F</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>1</td></tr> <tr> <td>0</td><td>1</td><td>1</td></tr> <tr> <td>1</td><td>0</td><td>1</td></tr> <tr> <td>1</td><td>1</td><td>0</td></tr> </tbody> </table>	A	B	F	0	0	1	0	1	1	1	0	1	1	1	0		$F = \overline{AB} = \overline{A} \cdot \overline{B}$
A	B	F															
0	0	1															
0	1	1															
1	0	1															
1	1	0															
		논리기호															

NAND gate

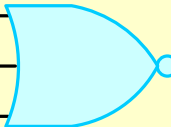
진리표	동작파형	논리식																																				
<table><tr><th>A</th><th>B</th><th>C</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	A	B	C	F	0	0	0	1	0	0	1	1	0	1	0	1	0	1	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	1	1	0	<div><div>A</div><div>0 0 0 0 1 1 1 1 0</div><div>B</div><div>0 0 1 1 0 0 1 1 0</div><div>C</div><div>0 1 0 1 0 1 0 1 0</div><div>F</div><div>1 1 1 1 1 1 1 0 1</div></div>	<div>$F = \overline{ABC} = \overline{A \cdot B \cdot C}$</div> <div>논리기호</div> <div><div>A</div><div>B</div><div>C</div><div></div><div>F</div></div>
A	B	C	F																																			
0	0	0	1																																			
0	0	1	1																																			
0	1	0	1																																			
0	1	1	1																																			
1	0	0	1																																			
1	0	1	1																																			
1	1	0	1																																			
1	1	1	0																																			

NOR gate

- ◆ 입력이 모두 0인 경우에만 출력은 1이 되고, 입력 중에 하나라도 1이 있는 경우는 출력은 0이 된다.
- ◆ 이 게이트는 OR 게이트와는 반대로 작동하는 게이트로, NOT OR의 의미로 NOR 게이트라고 부른다.

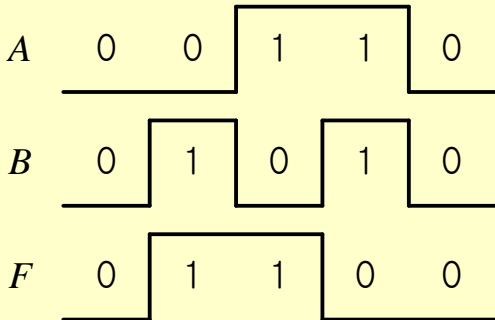
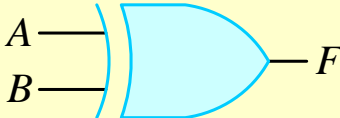
진리표	동작파형	논리식															
<table><tr><th>A</th><th>B</th><th>F</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	F	0	0	1	0	1	0	1	0	0	1	1	0		$F = \overline{A + B}$
A	B	F															
0	0	1															
0	1	0															
1	0	0															
1	1	0															
		논리기호															
																	
																	

NOR gate

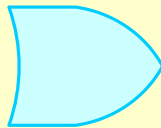
진리표	동작파형	논리식																																				
<table><tr><th>A</th><th>B</th><th>C</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	A	B	C	F	0	0	0	1	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0	0	1	0	1	0	1	1	0	0	1	1	1	0	<div><div>A</div><div>000011110</div><div>B</div><div>001100110</div><div>C</div><div>010101010</div><div>F</div><div>1000000001</div></div>	$F = \overline{A + B + C}$
	A	B	C	F																																		
	0	0	0	1																																		
	0	0	1	0																																		
	0	1	0	0																																		
	0	1	1	0																																		
	1	0	0	0																																		
	1	0	1	0																																		
	1	1	0	0																																		
	1	1	1	0																																		
		논리기호																																				
		<div><div>A</div><div>B</div><div>C</div><div></div><div>F</div></div>																																				

XOR gate

- ◆ 입력 중 홀수 개의 1이 입력된 경우에 출력은 1이 되고 그렇지 않은 경우에는 출력은 0이 된다.
- ◆ 2입력 XOR 게이트의 경우, 두 개의 입력 중 하나가 1이면 출력이 1이 되고, 두 개의 입력 모두가 0이거나 또는 두 개의 입력 모두가 1이면 출력은 0이 된다.

진리표	동작파형	논리식															
<table><tr><th>A</th><th>B</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	F	0	0	0	0	1	1	1	0	1	1	1	0		$F = A \oplus B = \overline{A}B + A\overline{B}$
A	B	F															
0	0	0															
0	1	1															
1	0	1															
1	1	0															
		논리기호															
																	

XOR gate

진리표	동작파형	논리식																																				
<table><tr><th>A</th><th>B</th><th>C</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	A	B	C	F	0	0	0	0	0	0	1	1	0	1	0	1	0	1	1	0	1	0	0	1	1	0	1	0	1	1	0	0	1	1	1	1	<div><div>A</div><div>0 0 0 0 1 1 1 1 0</div></div> <div><div>B</div><div>0 0 1 1 0 0 1 1 0</div></div> <div><div>C</div><div>0 1 0 1 0 1 0 1 0</div></div> <div><div>F</div><div>0 1 1 0 1 0 0 1 0</div></div>	<div>$F = A \oplus B \oplus C$</div> <div>논리기호</div> <div><div>A</div><div>B</div><div>C</div><div></div><div>F</div></div>
A	B	C	F																																			
0	0	0	0																																			
0	0	1	1																																			
0	1	0	1																																			
0	1	1	0																																			
1	0	0	1																																			
1	0	1	0																																			
1	1	0	0																																			
1	1	1	1																																			

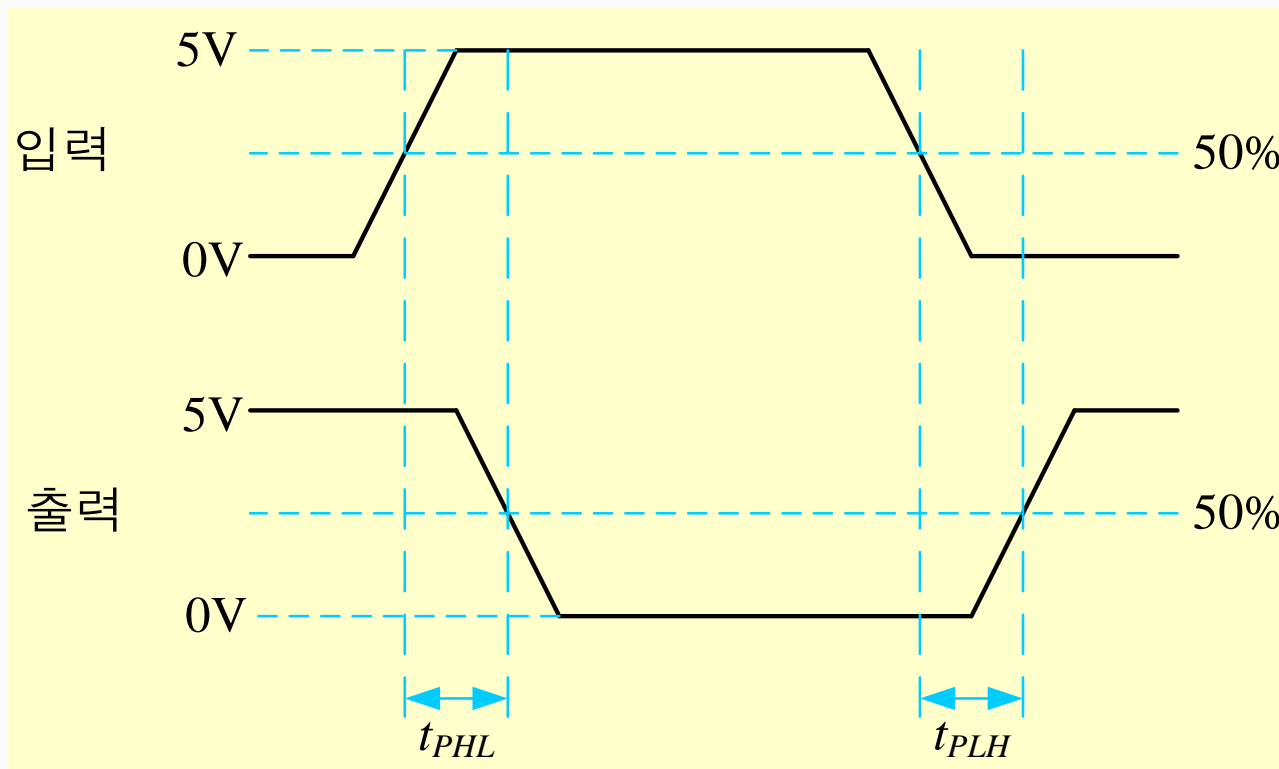
Gate의 전기적 특성

전파지연시간	<ul style="list-style-type: none">신호가 입력되어서 출력될 때까지의 시간을 말하며, 게이트의 동작 속도이다.
전력소모	<ul style="list-style-type: none">게이트가 동작할 때 소모되는 전력량
잡음여유도	<ul style="list-style-type: none">최대로 허용된 잡음 마진
팬-아웃	<ul style="list-style-type: none">하나의 게이트의 출력으로부터 다른 여러 개의 입력들로 공급되는 전류정상적인 동작으로 하나의 출력이 최대 몇 개의 입력으로 연결되는가를 나타낸다.

Gate의 전기적 특성

1. 전파지연시간(gate propagation delay time)

- ◆ 신호가 입력되어서 출력될 때까지의 시간을 말하며, 게이트의 동작 속도를 나타낸다.



Gate의 전기적 특성

2. 전력소모(power dissipation)

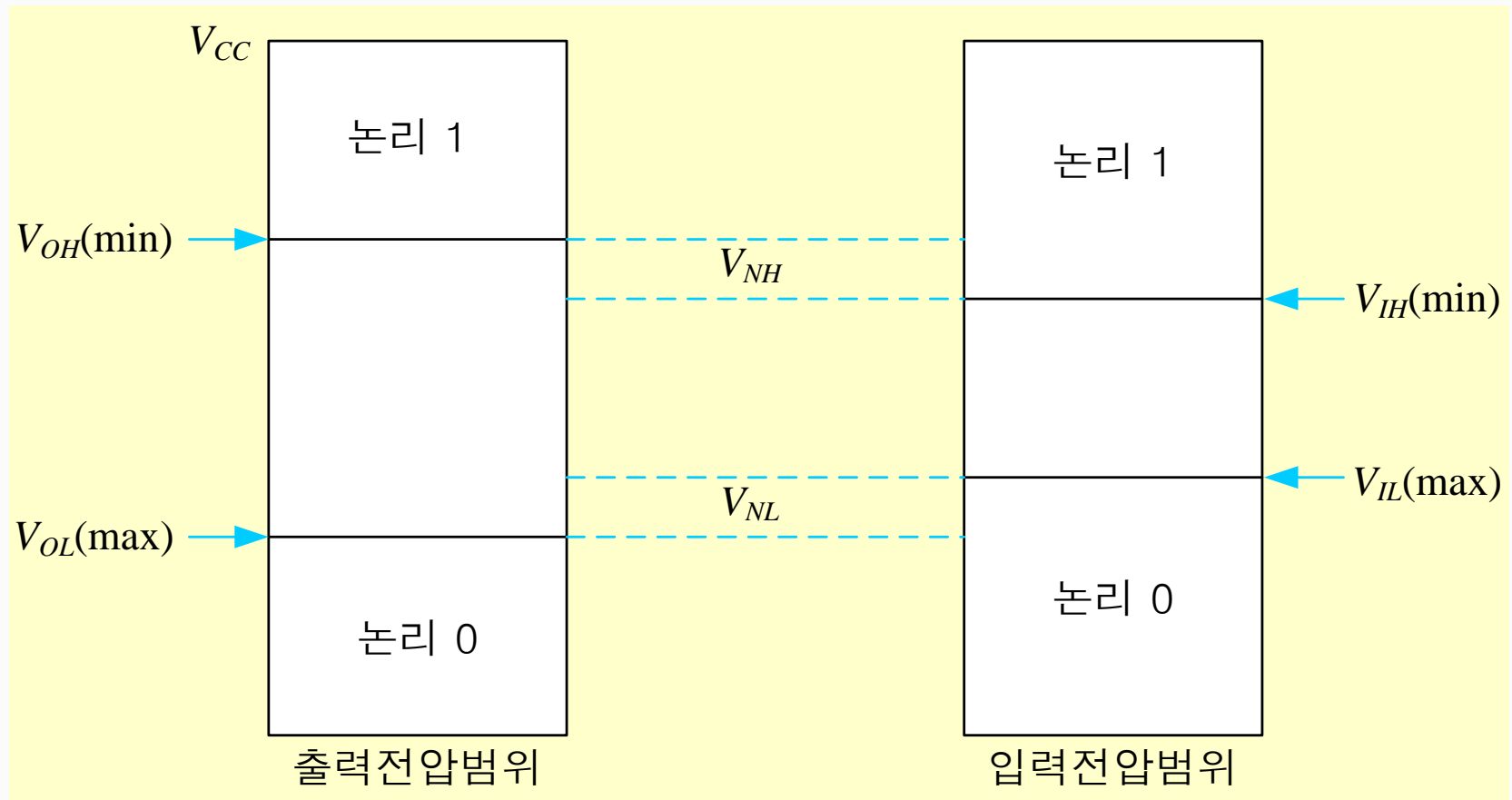
- ◆ 게이트가 동작할 때 소모되는 전력

$$P_{CC} = V_{CC} \times I_{CC}$$

3. 잡음여유도(noise margin)

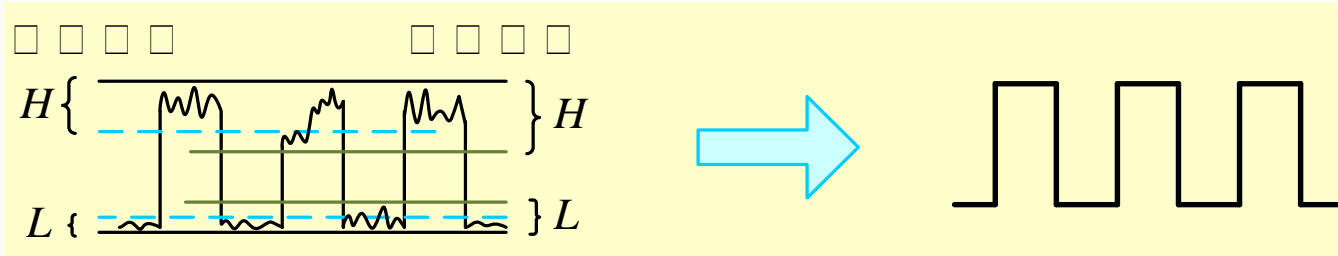
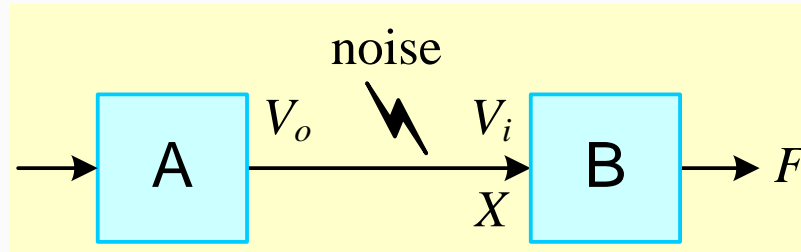
- ◆ 디지털 회로에서 데이터의 값에 변경을 주지 않는 범위 내에서 최대 허용된 Noise Margin을 의미

Gate의 전기적 특성



<입출력 전압 범위>

Gate의 전기적 특성



<입력신호 X (신호+잡음)>

<출력신호 F >

Gate의 전기적 특성

4. 팬-인(fan-in)과 팬-아웃(fan-out)

- ◆ 팬-아웃은 1 개의 게이트에서 다른 게이트의 입력으로 연결 가능한 최대 출력단의 수를 의미
- ◆ 팬-인은 1 개의 게이트에 입력으로 접속할 수 있는 단수를 의미

