

컴퓨터과학기초

4주차

논리게이트

인하공업전문대학 컴퓨터정보과

이수정 교수



차례

Ch.3 디지털 코드

- 4. 에러 검출 코드
- 5. 영숫자 코드



4. 에러 검출 코드

1) 패리티 비트

- 짝수패리티(even parity) : 데이터에서 1의 개수를 짝수 개로 맞춤
- 홀수패리티(odd parity) : 데이터에서 1의 개수를 홀수 개로 맞춤
- 패리티 비트는 데이터 전송과정에서 에러 검사를 위한 추가 비트
패리티는 단지 에러 검출만 가능하며, 여러 비트에 에러가 발생할 경우에는 검출이 안될 수도 있음
- 7비트 ASCII 코드에 패리티 비트를 추가한 코드

데이터	짝수패리티	홀수패리티
...
A	0 1000001	1 1000001
B	0 1000010	1 1000010
C	1 1000011	0 1000011
D	0 1000100	1 1000100
...

4. 에러 검출 코드

■ 병렬 패리티(parallel parity)

- 패리티를 블록 데이터에 적용해서 가로와 세로 데이터들에 대해서 패리티를 적용하면 에러를 검출하여 그 위치를 찾아 정정할 수 있다.

1	0	1	0	1	1	1	1	0
1	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	1
1	1	1	1	0	0	0	0	0
1	0	1	1	1	0	0	1	1
0	0	0	0	0	1	1	1	1
1	1	1	1	1	1	1	1	0
0	1	1	1	1	0	0	0	0
1	0	1	0	0	1	0	1	0

원래 데이터 블록

가로세로 모두 1
의 개수가 짝수임

1	0	1	0	1	1	1	1	0
1	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	1
1	1	1	1	0	0	0	0	0
1	0	1	1	0	0	0	1	1
0	0	0	0	0	1	1	1	1
1	1	1	1	1	1	1	1	0
0	1	1	1	1	0	0	0	0
1	0	1	0	0	1	0	1	0

에러가 발생한 블록

가로세로 회색 부분에 1의 개수
가 홀수임 : 검치는 부분 에러

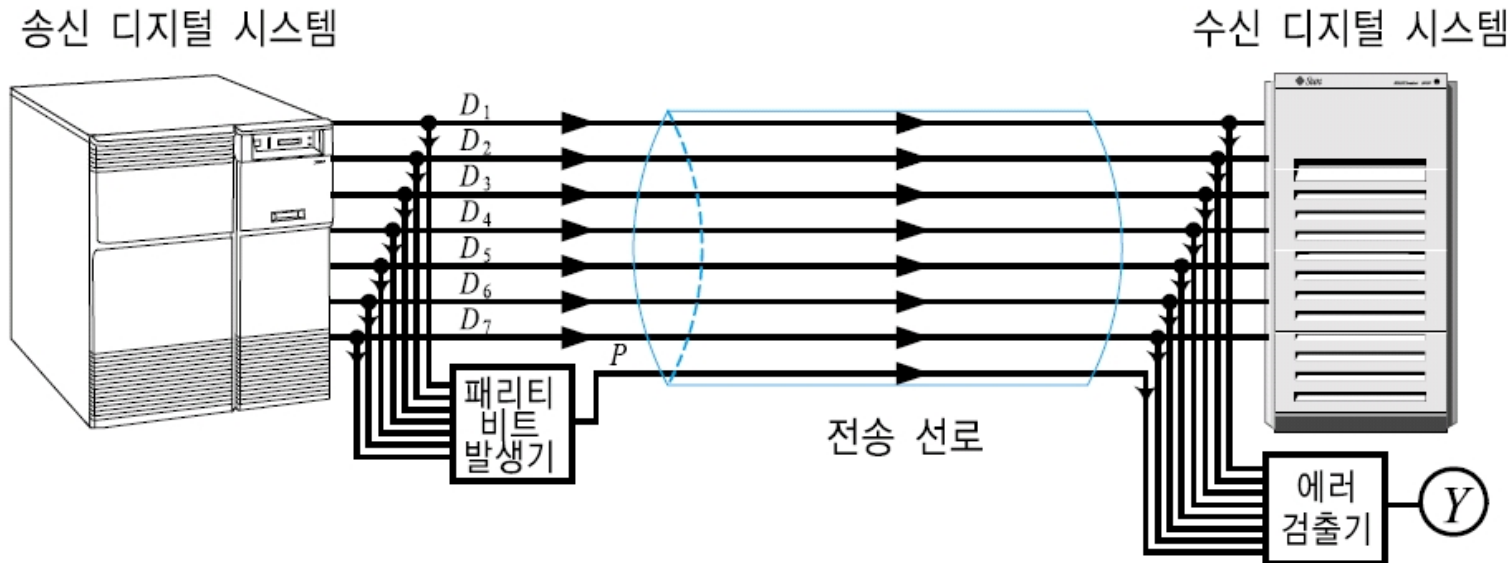
4. 에러 검출 코드

■ 데이터 전송 시스템에서 패리티 비트를 사용한 에러 검출

- 에러를 검출하기 위하여 송신측에 패리티 발생기를 구성하고 수신측에는 패리티 검출기를 구성하여 그 출력을 보고 에러 발생 여부를 판단

짝수 패리티 $Y=0$ (에러 없음), $Y=1$ (에러 발생)

홀수 패리티 $Y=1$ (에러 없음), $Y=0$ (에러 발생)



4. 에러 검출 코드

2) 에러 정정 코드: 해밍코드(Hamming Code)

- 에러를 정정할 수 있는 코드
- 추가적으로 많은 비트가 필요하므로 많은 양의 데이터 전달이 필요
- 데이터 비트와 패리티 비트와의 관계

$$2^{p-1} - p + 1 \leq d \leq 2^p - p - 1$$

p 는 패리티 비트의 수, d 는 데이터 비트의 수, $p \geq 2$

- $p=4$ 일 때, $2^{4-1} - 4 + 1 \leq d \leq 2^4 - 4 - 1$ 이므로 $5 \leq d \leq 11$ 이다.
- 따라서 데이터 비트수가 5개 이상 11개 이하일 때 패리티는 4개가 필요하다.

4. 에러 검출 코드

- 패리티 비트의 위치는 앞에서 부터 1, 2, 4, 8, 16, ... 번째이다.
- 데이터 비트는 나머지 위치에 순서대로 들어간다.
- 해밍코드에서는 짝수 패리티를 사용

$$P_1 = D_3 \oplus D_5 \oplus D_7 \oplus D_9 \oplus D_{11}$$

$$P_2 = D_3 \oplus D_6 \oplus D_7 \oplus D_{10} \oplus D_{11}$$

$$P_4 = D_5 \oplus D_6 \oplus D_7 \oplus D_{12}$$

$$P_8 = D_9 \oplus D_{10} \oplus D_{11} \oplus D_{12}$$

비트 위치	1	2	3	4	5	6	7	8	9	10	11	12
기호	P_1	P_2	D_3	P_4	D_5	D_6	D_7	P_8	D_9	D_{10}	D_{11}	D_{12}
P_1 영역	✓		✓		✓		✓		✓		✓	
P_2 영역		✓	✓			✓	✓			✓	✓	
P_4 영역				✓	✓	✓	✓					✓
P_8 영역								✓	✓	✓	✓	✓

4. 에러 검출 코드

- For Example

P_1	P_2	D_3	P_4	D_5	D_6	D_7	P_8	D_9	D_{10}	D_{11}	D_{12}
		0		0	1	0		1	1	1	0

$$P_1 = D_3 \oplus D_5 \oplus D_7 \oplus D_9 \oplus D_{11} = 0 \oplus 0 \oplus 0 \oplus 1 \oplus 1 = 0$$

$$P_2 = D_3 \oplus D_6 \oplus D_7 \oplus D_{10} \oplus D_{11} = 0 \oplus 1 \oplus 0 \oplus 1 \oplus 1 = 1$$

$$P_4 = D_5 \oplus D_6 \oplus D_7 \oplus D_{12} = 0 \oplus 1 \oplus 0 \oplus 0 = 1$$

$$P_8 = D_9 \oplus D_{10} \oplus D_{11} \oplus D_{12} = 1 \oplus 1 \oplus 1 \oplus 0 = 1$$

00101001100110000100

00100100100011000100

생성된 패리티

4. 에러 검출 코드

■ 해밍코드에서 패리티 비트 검사 과정

전송된 데이터: 010111011110

P_1	P_2	D_3	P_4	D_5	D_6	D_7	P_8	D_9	D_{10}	D_{11}	D_{12}
0	1	0	1	1	1	0	1	1	1	1	0

☞ 패리티들을 포함하여 검사

$$P_1 = P_1 \oplus D_3 \oplus D_5 \oplus D_7 \oplus D_9 \oplus D_{11} = 0 \oplus 0 \oplus 1 \oplus 0 \oplus 1 \oplus 1 = 1$$

$$P_2 = P_2 \oplus D_3 \oplus D_6 \oplus D_7 \oplus D_{10} \oplus D_{11} = 1 \oplus 0 \oplus 1 \oplus 0 \oplus 1 \oplus 1 = 0$$

$$P_4 = P_4 \oplus D_5 \oplus D_6 \oplus D_7 \oplus D_{12} = 1 \oplus 1 \oplus 1 \oplus 0 \oplus 0 = 1$$

$$P_8 = P_8 \oplus D_9 \oplus D_{10} \oplus D_{11} \oplus D_{12} = 1 \oplus 1 \oplus 1 \oplus 1 \oplus 0 = 0$$

- 검사된 패리티를 $P_8 P_4 P_2 P_1$ 순서대로 정렬
- 모든 패리티가 0이면 에러 없음
- 하나라도 1이 있으면 에러 발생: 결과가 0101이므로 에러 있음
- 0101을 10진수로 바꾸면 5이며, 수신된 데이터에서 앞에서 5번째 비트 0101**1**011110에 에러가 발생한 것이므로 0101**0**1011110으로 바꾸어 주면 에러가 정정된다.

4. 에러 검출 코드

■ 해밍코드에서 에러가 발생한 경우 교정

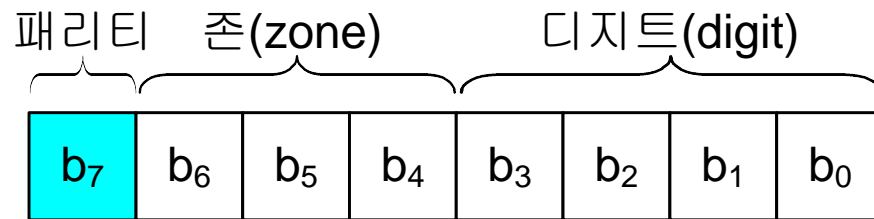
비트위치	1	2	3	4	5	6	7	8	9	10	11	12
기호	P_1	P_2	D_3	P_4	D_5	D_6	D_7	P_8	D_9	D_{10}	D_{11}	D_{12}
Error 해밍코드	0	1	0	1	1	1	0	1	1	1	1	0
P_1 계산	1	0	0	0	1	0	0	1	1	1	1	
P_2 계산	0		1	0		1	0			1	1	
P_4 계산	1				1	1	1	0				0
P_8 계산	0								1	1	1	0

$P_8P_4P_2P_1 = 0101 = 5$: 5번째 비트에 에러 발생, $1 \rightarrow 0$ 으로 교정

5. 영숫자 코드

1) ASCII(American Standard Code for Information Interchange) 코드

- 미국 국립 표준 연구소(ANSI)가 제정한 정보 교환용 미국 표준 코드
- 128가지의 문자를 표현 가능



- ASCII 코드의 구성

parity	zone bit			digit bit			
7	6	5	4	3	2	1	0
C	1	0	0	영문자 A~O(0001~1111)			
	1	0	1	영문자 P~Z(0000~1010)			
	0	1	1	숫자 0~9(0000~1001)			

5. 영숫자 코드

■ 표준 ASCII 코드표

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	TAB	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;		=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

5. 영숫자 코드

■ 확장 ASCII 코드표

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
9	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	ç	£	¥	Pt	f
A	á	í	ó	ú	ñ	Ñ	ª	º	¿	「	」	½	¼	¡	«	»
B	⌘	⌘	⌘		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬
C	⌚	⌚	⌚	¡	—	+	¢	£	¤	¥	¦	§	¨	©	ª	±
D	⌚	⌚	⌚	⌚	⌚	⌚	⌚	⌚	⌚	⌚	⌚	⌚	⌚	⌚	⌚	⌚
E	α	β	Γ	π	Σ	σ	μ	τ	Φ	Θ	Ω	δ	∞	∅	ε	∩
F	≡	±	≥	≤	∫	∫	÷	≈	°	•	·	√	n	2	■	

5. 영숫자 코드

2) 표준 BCD(Binary Code Decimal) 코드

- 6비트로 하나의 문자를 표현
- 최대 64문자까지 표현 가능한 코드

코드의 구성

parity	zone bit		digit bit			
6	5	4	3	2	1	0
C	1	1	영문자 A~I(0001~1001)			
	1	0	영문자 J~R(0001~1001)			
	0	1	영문자 S~Z(0010~1001)			
	0	0	숫자 0~9(0001~1010)			
	혼용		특수문자 및 기타문자			

5. 영숫자 코드

■ 표준 BCD 코드표

문자	C ZZ8421	문자	C ZZ8421	문자	C ZZ8421	문자	C ZZ8421	문자	C ZZ8421
A	0 110001	J	1 100001	S	1 010010	1	0 000001	=	0 001011
B	0 110010	K	1 100010	T	0 010011	2	0 000010	>	1 001100
C	1 110011	L	0 100011	U	1 010100	3	1 000011	+	0 010000
D	0 110100	M	1 100100	V	0 010101	4	0 000100	,	1 011011
E	1 110101	N	0 100101	W	0 010110	5	1 000101)	0 011100
F	1 110110	O	0 100110	X	1 010111	6	1 000110	%	1 011101
G	0 110111	P	1 100111	Y	1 011000	7	0 000111	?	0 011111
H	0 111000	Q	1 101000	Z	0 011001	8	0 001000	-	1 100001
I	1 111001	R	0 101001			9	1 001001	@	1 111010
						0	1 001010	\$	1 111111

5. 영숫자 코드

3) EBCDIC(Extended Binary Coded Decimal Interchange Code) 코드

- 대형 컴퓨터와 IBM 계열 컴퓨터에서 많이 사용되고 있는 8비트 코드(IBM에서 개발)
- 256종류의 문자 코드를 표현할 수 있는 영숫자 코드

코드의 구성

b ₉	b ₈ b ₇ b ₆ b ₅	b ₄ b ₃ b ₂ b ₁
패리티	존(zone)	디지트(digit)
1	4	4

b ₈ b ₇		b ₆ b ₅	
0 0	통신제어문자		
0 1	특수문자		
1 0	소문자	0 0	a ~i
		0 1	j~r
		1 0	s~z
		1 1	
1 1	대문자/숫자	0 0	A~I
		0 1	J~R
		1 0	S~Z
		1 1	0~9

5. 영숫자 코드

■ EBCDIC 코드표

16진		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	2진	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	0000	NUL	SOH	STX	ETX		HT		DEL				VT	FF	CR	SO	SI
1	0001	DLE						BS		CAN	EM			IFS	IGS	IRS	IUS
2	0010						LF	ETB	ESC						ENQ	ACK	BEL
3	0011			SYN					EOT						NAK		SUB
4	0100	space										[.		(+	
5	0101	&										!	\$	*)		^
6	0110	-	/										,	%	_	>	?
7	0111										`	:	#	@	'	=	"
8	1000		a	b	c	d	e	f	g	h	i						
9	1001		j	k	l	m	n	o	p	q	r						
A	1010		~	s	t	u	v	w	x	y	z						
B	1011																
C	1100	{	A	B	C	D	E	F	G	H	I						
D	1101	}	J	K	L	M	N	O	P	Q	R						
E	1110	\		S	T	U	V	W	X	Y	Z						
F	1111	0	1	2	3	4	5	6	7	8	9						

5. 영숫자 코드

4) 유니코드(Unicode)

- ASCII 코드의 한계성을 극복하기 위하여 개발된 인터넷 시대의 표준
- 유니코드 컨소시엄(IBM, Novell, Microsoft, DEC, Apple 등)에 의해서 32(UTF-32), 16(UTF-16), 8bit(UTF-8)의 세 가지 기본 코드
- 미국, 유럽, 동아시아, 아프리카, 아시아 태평양 지역 등의 주요 언어들에 적용될 수 있다.
- 유니코드는 유럽, 중동, 아시아 등 거의 대부분의 문자를 포함하고 있으며, 10만개 이상의 문자로 구성되어 있다.
- 특히 아시아의 중국, 일본, 한국, 타이완, 베트남, 싱가포르에서 사용하는 표의 문자(한자) 70,207개를 나타낼 수 있다.
- 구두표시, 수학기호, 전문기호, 기하학적 모양, 덩벙 기호 등을 포함
- 앞으로도 계속해서 산업계의 요구나 새로운 문자들을 추가하여 나갈 것이다.

5. 영숫자 코드

5) 한글코드

- 한글은 ASCI코드를 기반으로 16비트를 사용하여 하나의 문자를 표현
- 조합형과 완성형으로 분류

조합형

- 조합형으로 표현된 한글은 때에 따라서 다른 응용프로그램에서는 사용할 수 없는 문자들이 많다.
- 조합형은 자음과 모음으로 조합 가능한 모든 한글을 사용할 수 있으며, 심지어 우리나라 고어(古語)까지 취급할 수 있는 장점이 있으나, 출력 시 다시 모아 써야 하는 불편이 있다는 것이 단점이다.

두번째 바이트								첫번째 바이트							
1															
	초성				중성				종성						

완성형

- 1987년 정부가 한국표준으로 정한 것으로 가장 많이 사용되는 한글 음절을 2 바이트의 2진수와 1 대 1로 대응하여 표현하는 방법

차례 : 논리 게이트

1. 논리 레벨
2. NOT 게이트와 버퍼 게이트
3. AND 게이트
4. OR 게이트
5. NAND 게이트
6. NOR 게이트
7. XOR 게이트
8. XNOR 게이트
9. 정논리와 부논리
10. 게이트의 전기적 특성

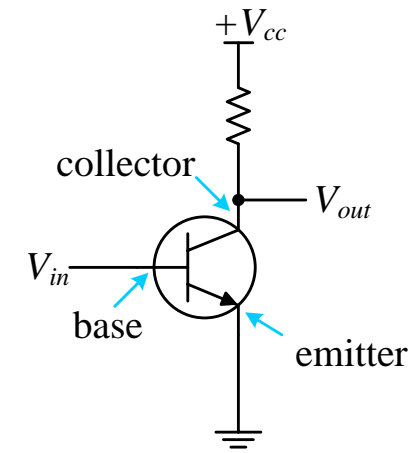
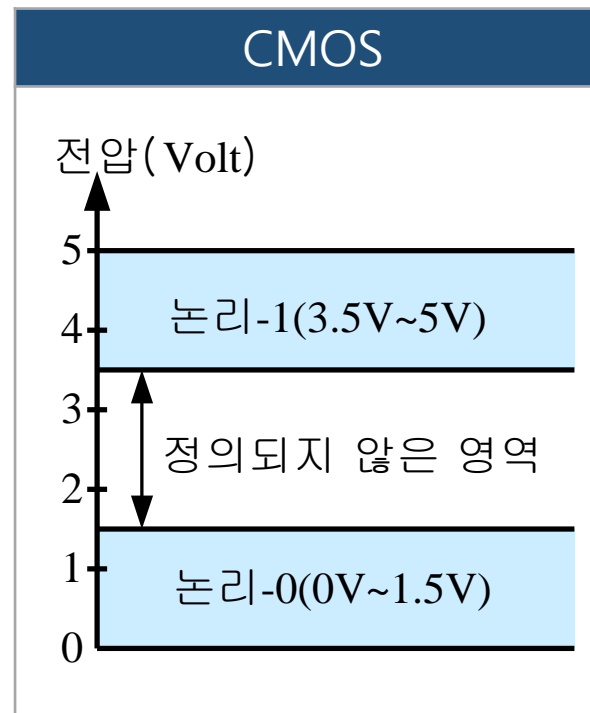
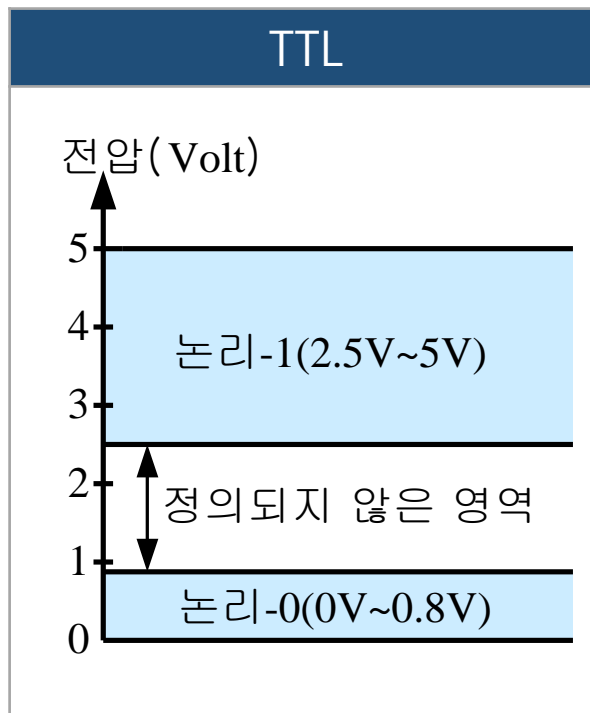
1. 논리 레벨

- 논리회로 : 하드웨어를 구성하는 기본 요소인 논리게이트로 구성
- 논리게이트
 - 한 개 이상의 입력과 하나의 출력으로 구성되는 전자회로
 - 두 가지(0, 1) 신호를 발생하는 기능을 가진 장치
- 디지털 시스템 : 여러 가지 논리게이트가 모여 구성
 - 기본이 되는 논리게이트 : AND, OR, NOT
 - 조합으로 만든 논리게이트 : NAND, NOR, XOR, XNOR

1. 논리 레벨

■ TTL과 CMOS 논리 레벨 정의 영역

- 논리 1 : high 전압
- 논리 0 : low 전압



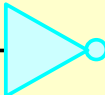
Transistor

디지털 회로에서 전자스위치로 사용되는 반도체 소자. 베이스에 적절한 전압을 인가하여 컬렉터-에미터 접합이 개방 또는 단락된 스위치처럼 동작

2. NOT 게이트와 버퍼 게이트

1) NOT 게이트

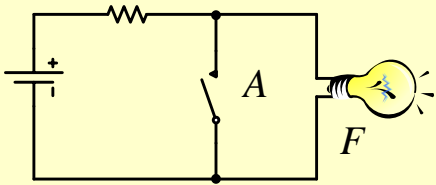
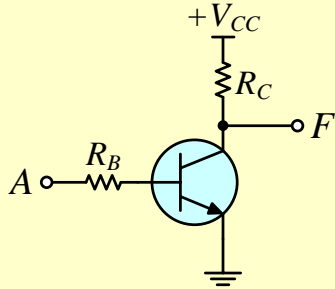
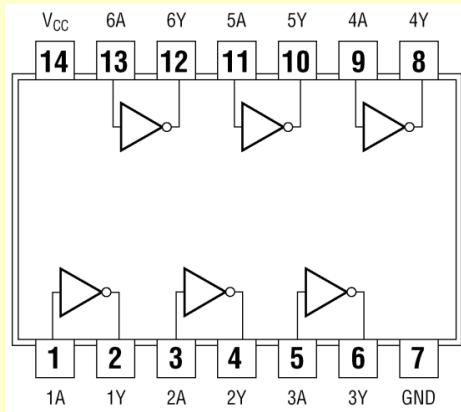
- 한 개의 입력과 한 개의 출력을 갖는 게이트
- 논리 부정
- 인버터(inverter)

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>입력 A 1 0 1 0</div> <div>출력 F 0 1 0 1</div>	<div>AF</div>
A	F							
0	1							
1	0							

논리식

$$F = \overline{A} = A'$$

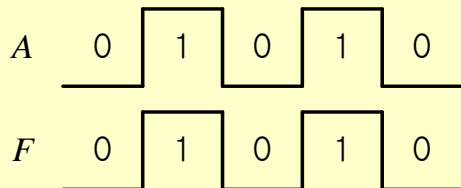
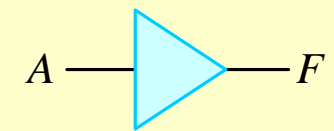
2. NOT 게이트와 버퍼 게이트

스위칭 회로	트랜지스터 회로	IC 7404 핀 배치도
		

2. NOT 게이트와 버퍼 게이트

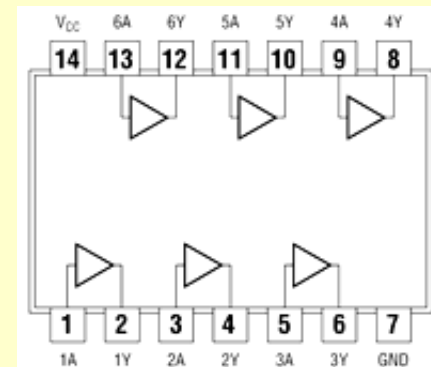
2) 버퍼(buffer)

- 입력된 신호를 변경하지 않고, 입력된 신호 그대로를 출력하는 게이트로 단순한 전송을 의미
- 입력 신호가 1인 경우에는 출력 신호는 1, 입력 신호가 0인 경우에는 출력 신호는 0

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

논리식 $F = A$

IC 7407 핀 배치도



2. NOT 게이트와 버퍼 게이트

■ 3상태(tri-state) 버퍼

- 출력이 3개 레벨
(High, Low, 하이 임피던스) 중의 하나를 갖는 논리소자
- 하이 임피던스: 입력과 출력이 연결되어 있지 않은 상태

	진리표	논리기호	핀 배치도															
제어 입력 이 Low 일 때	<table><tr><th>A</th><th>\bar{E}</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>Hi-Z</td></tr><tr><td>1</td><td>1</td><td>Hi-Z</td></tr></table>	A	\bar{E}	F	0	0	0	1	0	1	0	1	Hi-Z	1	1	Hi-Z		<p>74125</p>
A	\bar{E}	F																
0	0	0																
1	0	1																
0	1	Hi-Z																
1	1	Hi-Z																
제어 입력 이 High 일 때	<table><tr><th>A</th><th>E</th><th>F</th></tr><tr><td>0</td><td>0</td><td>Hi-Z</td></tr><tr><td>1</td><td>0</td><td>Hi-Z</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	E	F	0	0	Hi-Z	1	0	Hi-Z	0	1	0	1	1	1		<p>74126</p>
A	E	F																
0	0	Hi-Z																
1	0	Hi-Z																
0	1	0																
1	1	1																

3. AND 게이트

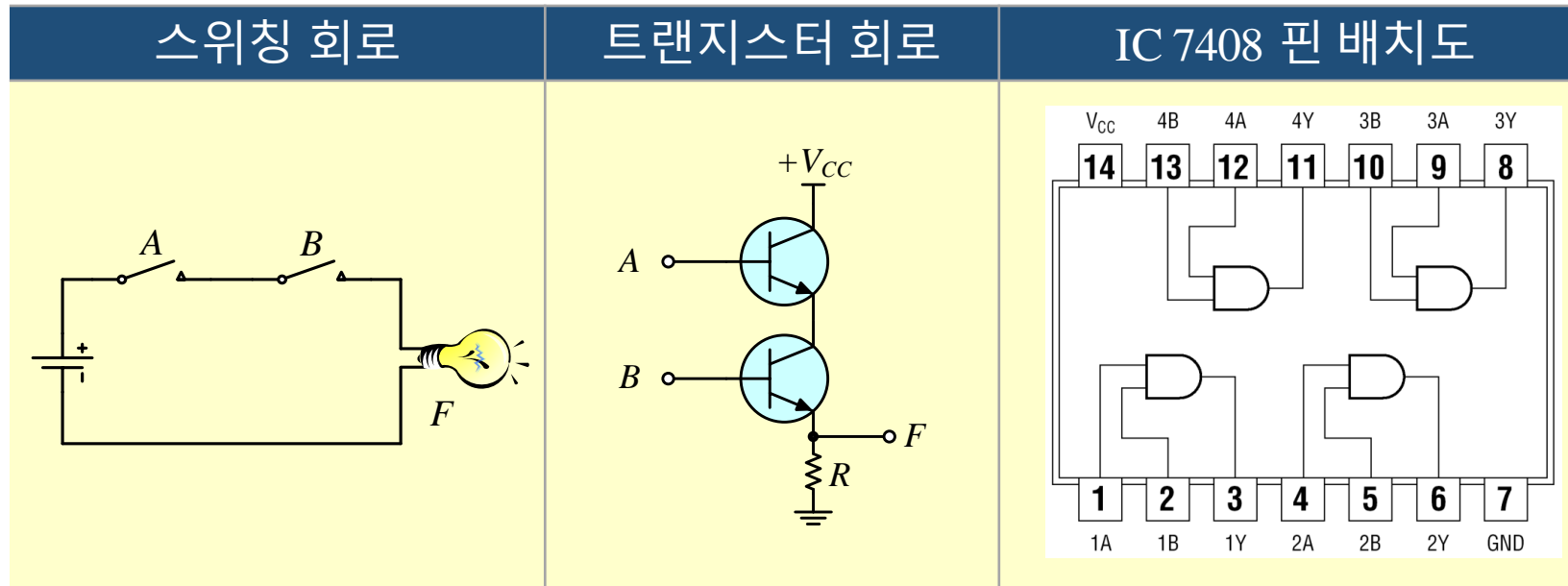
■ AND 게이트의 기본 개념(2입력)

- 2개 이상의 입력에 1개의 출력. 논리곱(logical product)
- 입력이 모두 1(on)인 경우에만 출력은 1(on)
- 입력 중에 0(off)인 것이 하나라도 있을 경우에는 출력은 0(off)

진리표	동작파형	논리기호															
<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>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></table>	A	B	F	0	0	0	0	1	0	1	0	0	1	1	1		
A	B	F															
0	0	0															
0	1	0															
1	0	0															
1	1	1															
		논리식															
		$F = AB = A \cdot B$															

3. AND 게이트

■ AND 게이트의 회로 표현과 IC



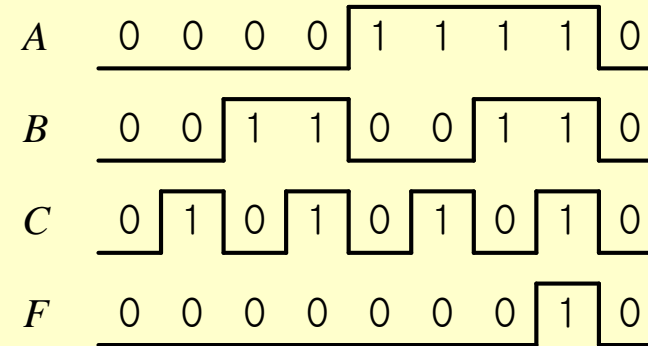
3. AND 게이트

■ AND 게이트의 기본 개념(3입력)

진리표

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

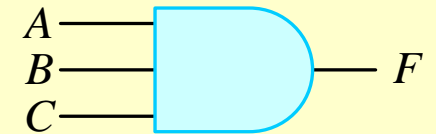
동작파형



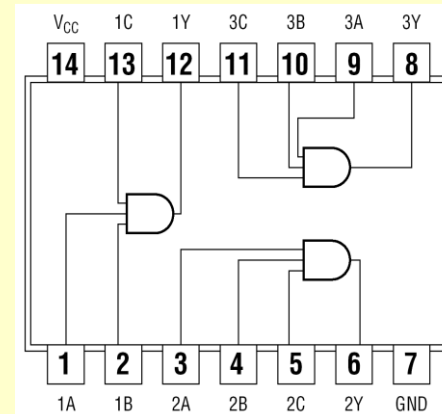
논리식

$$F = ABC = A \cdot B \cdot C$$

논리기호



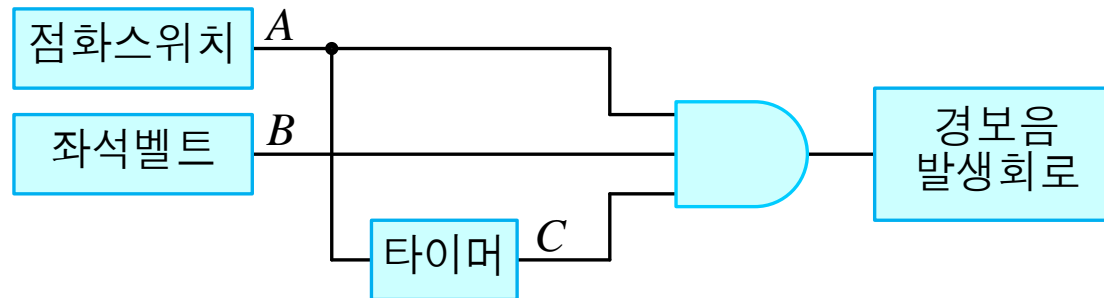
IC 7411 핀 배치도



3. AND 게이트

■ AND 게이트를 이용한 자동차 좌석벨트 경보 시스템

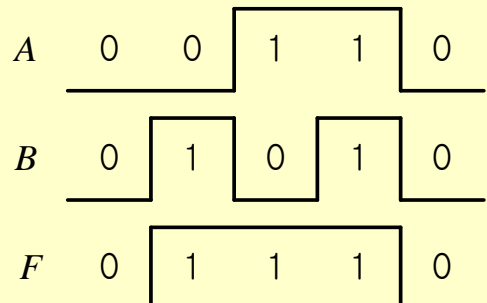
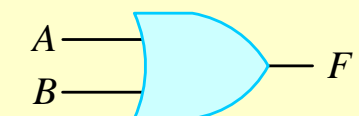
- 점화스위치(A)가 켜지고(High) 좌석벨트(B)가 풀려있는 상태(High)를 감지
- 점화스위치가 켜지면 타이머가 작동되어 타이머 C가 30초 동안 High로 유지
- 점화 스위치가 켜지고, 좌석벨트가 풀려있고, 타이머가 작동하는 3가지 조건하에서 AND 게이트의 출력은 High. 운전자에게 주의를 환기시키는 경보음이 울리게 된다.
- 30초간 경보음 동작 후에는 경보음은 울리지 않으며, 처음부터 좌석벨트가 채워져 있으면 경보음은 울리지 않는다.



4. OR 게이트

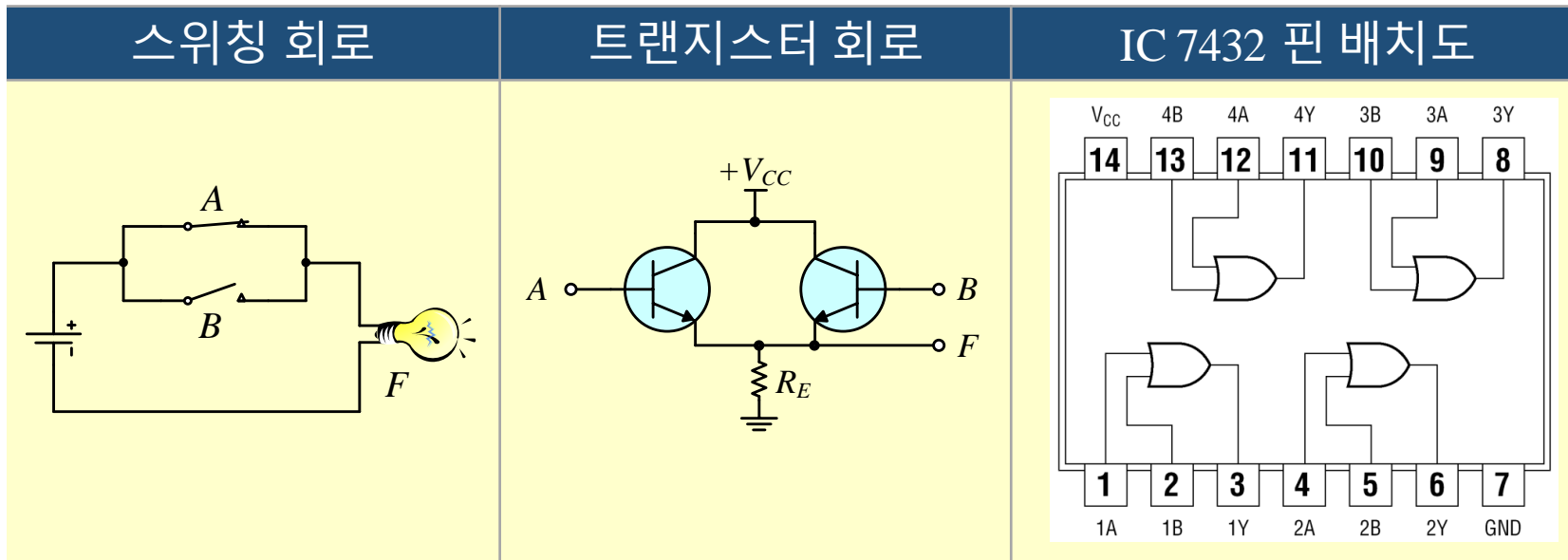
■ OR 게이트의 기본 개념(2입력)

- 입력이 모두 0인 경우에만 출력은 0
- 입력 중에 1이 하나라도 있으면, 출력은 1

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>1</td></tr></table>	A	B	F	0	0	0	0	1	1	1	0	1	1	1	1		$F = A + B$
A	B	F															
0	0	0															
0	1	1															
1	0	1															
1	1	1															
		논리기호															
																	

4. OR 게이트

■ OR 게이트의 회로 표현과 IC



4. OR 게이트

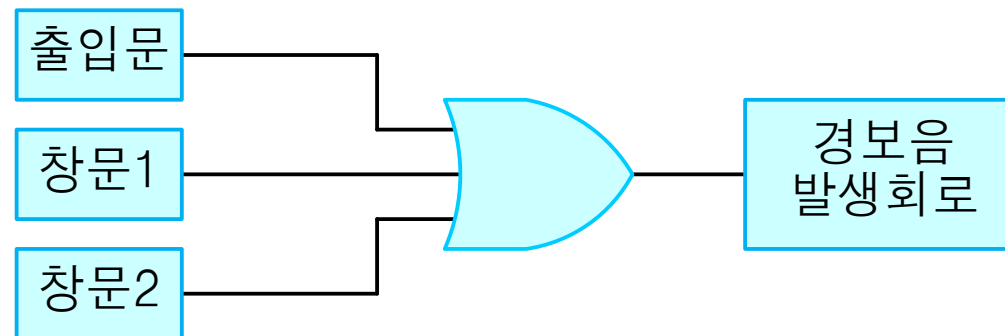
■ OR 게이트의 회로 표현과 IC

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><div>$F = A + B + C$</div><div>논리기호</div><div><div><div>A</div><div>B</div><div>C</div></div><div><div></div><div></div><div></div></div><div>F</div></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																																			

4. OR 게이트

■ OR 게이트를 이용한 침입 탐지 시스템

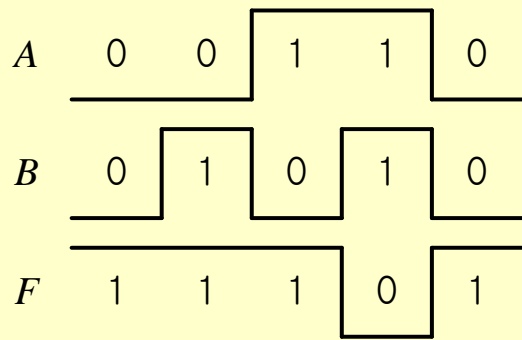
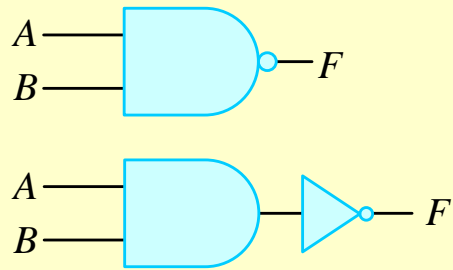
- 일반 가정에서 출입문 1개와 창문 2개가 있다고 가정
- 출입문과 창문에 설치된 각 센서는 자기 스위치(magnetic switch)로서 문이 열려 있을 때 High를 출력하고, 닫혀 있을 때에는 Low를 출력



5. NAND 게이트

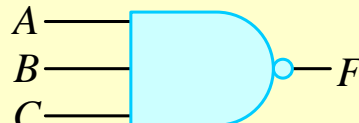
■ NAND 게이트의 기본 개념(2입력)

- 입력이 모두 1인 경우에만 출력은 0, 그렇지 않을 경우에는 출력은 1
- AND 게이트와는 반대로 작동하는 게이트
- NOT AND의 의미로 'NAND 게이트'라고 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>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	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															
																	

5. NAND 게이트

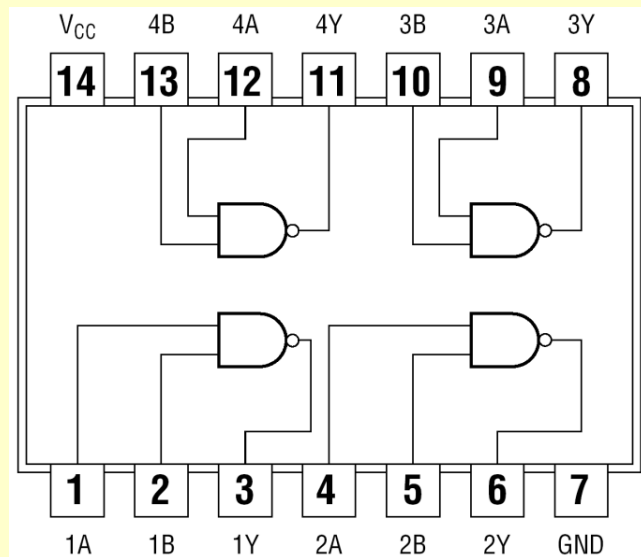
■ NAND 게이트의 기본 개념(3입력)

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

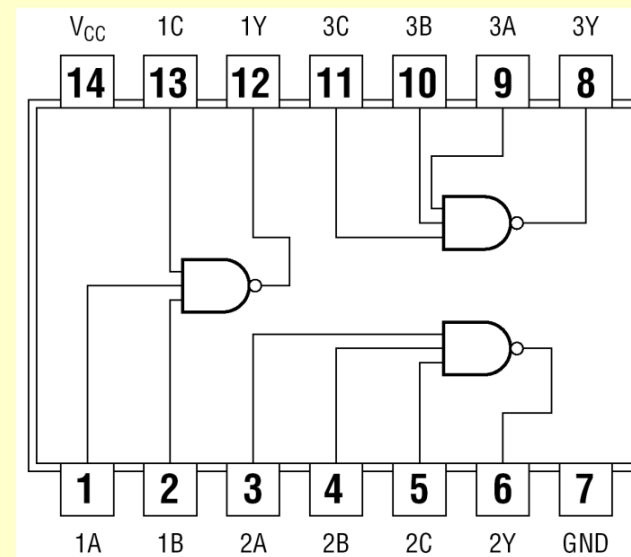
5. NAND 게이트

■ NAND 게이트의 IC

IC 7400 핀 배치도



IC 7410 핀 배치도



6. NOR 게이트

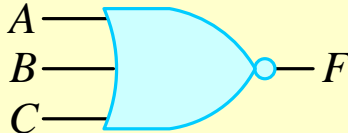
■ NOR 게이트의 기본 개념(2입력)

- 입력이 모두 0인 경우에만 출력은 1, 입력 중에 하나라도 1이 있는 경우는 출력은 0
- OR 게이트와는 반대로 작동하는 게이트로, NOT OR의 의미로 'NOR 게이트'라고 부른다.

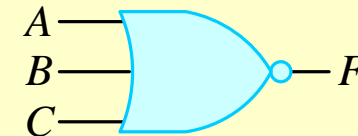
진리표	동작파형	논리식															
<table><tr><th>A</th><th>B</th><th>F</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	F	0	0	1	0	1	0	1	0	0	1	1	0		$F = \overline{A + B}$
A	B	F		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

6. NOR 게이트

■ NOR 게이트의 기본 개념(3입력)

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	<table><tr><td>A</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr><tr><td>B</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>C</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>F</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	A	0	0	0	0	1	1	1	1	0	B	0	0	1	1	0	0	1	1	0	C	0	1	0	1	0	1	0	1	0	F	1	0	0	0	0	0	0	0	1	$F = \overline{A + B + C}$
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																																																																											
A	0	0	0	0	1	1	1	1	0																																																																					
B	0	0	1	1	0	0	1	1	0																																																																					
C	0	1	0	1	0	1	0	1	0																																																																					
F	1	0	0	0	0	0	0	0	1																																																																					
		논리기호																																																																												
																																																																														

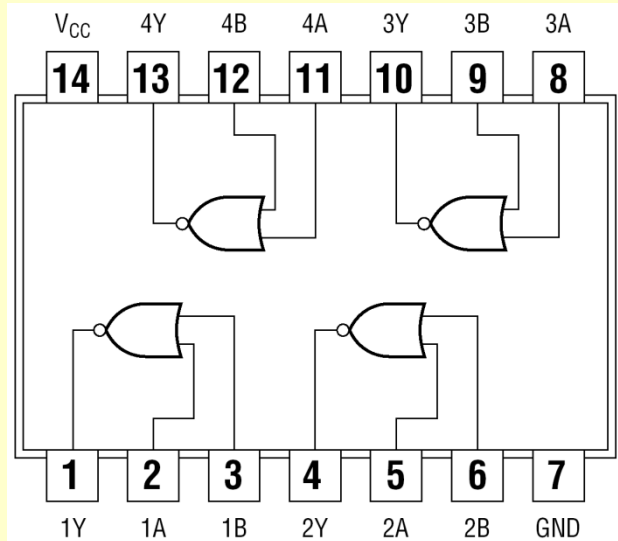
논리기호



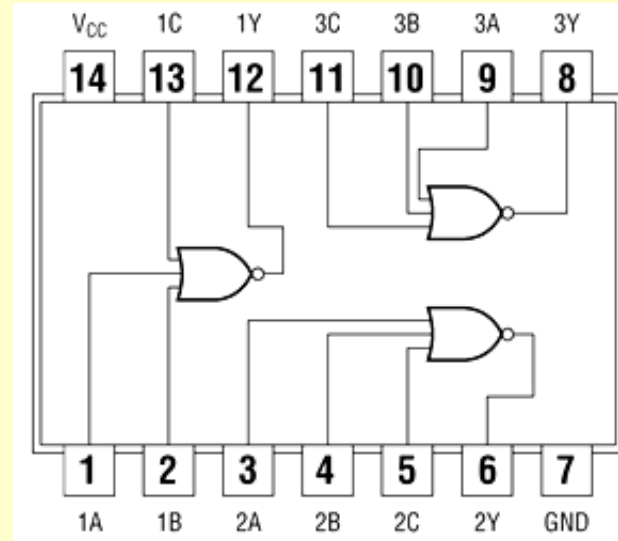
6. NOR 게이트

■ NOR 게이트 IC

IC 7402 핀 배치도



IC 7427 핀 배치도



다음 시간

5주차 : 부울 대수

