컴퓨터 구조

디지털 논리회로(1)

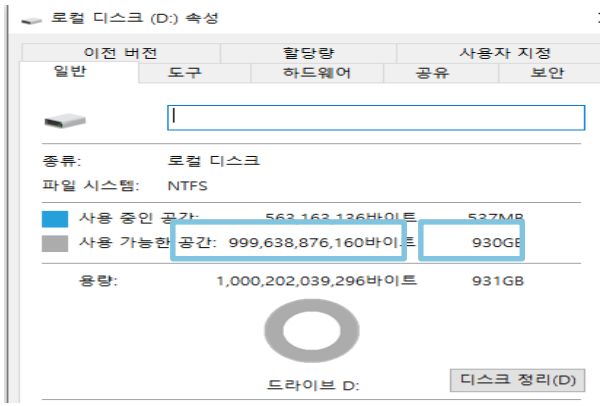
목 차

Unit

- Unit(단위)
- Bit(Binary digit)
  - 데이터를 나타내는 최소 단위 (0 또는 1)
- Byte
  - Bit들의 집합 (1byte = 8bit)
  - 대부분의 경우 데이터 크기를 표시하는 최소 단위
- Word
  - 다양하게 구성이 가능(2/4/8byte)
  - 대부분 4byte(=32bit)
  - Doubleword = 2word

SI 표준			IEC 표준
1000(10^3)B	1KB	1024(2^{10})B	1KiB
1000(10^6)B (=1000KB)	1MB	1048576 (2^{20})B (=1024KB)	1MiB
1000(10^9)B (=1000MB)	1GB	1073741824 (2^{30})B (=1024MB)	1GiB
1000(10^{12})B (=1000GB)	1TB	109951162777 (2^{40})B (=1024GB)	1TiB

< 단위 비교 >



Unit

표 2-1 SI 단위와 IEC 단위 비교

SI(10진수 단위)			IEC(2진수 단위)			
값	기호	이름	값	기호	이름	10진수 변환 크기
$(10^3)^1=10^3$	k, K	kilo–	$(2^{10})^1=2^{10}\cong10^{3.01}$	Ki	kibi–	1,024
$(10^3)^2=10^6$	M	mega–	$(2^{10})^2=2^{20}\cong10^{6.02}$	Mi	mebi–	1,048,576
$(10^3)^3=10^9$	G	giga–	$(2^{10})^3=2^{30}\cong10^{9.03}$	Gi	gibi–	1,073,741,824
$(10^3)^4=10^{12}$	T	tera–	$(2^{10})^4=2^{40}\cong10^{12.04}$	Ti	tebi–	1,099,511,627,776
$(10^3)^5=10^{15}$	P	peta–	$(2^{10})^5=2^{50}\cong10^{15.05}$	Pi	pebi–	1,125,899,906,842,624
$(10^3)^6=10^{18}$	E	exa–	$(2^{10})^6=2^{60}\cong10^{18.06}$	Ei	exbi–	1,152,921,504,606,846,976
$(10^3)^7=10^{21}$	Z	zetta–	$(2^{10})^7=2^{70}\cong10^{21.07}$	Zi	zebi–	1,180,591,620,717,411,303,424
$(10^3)^8=10^{24}$	Y	yotta–	$(2^{10})^8=2^{80}\cong10^{24.08}$	Yi	yobi–	1,208,925,819,614,629,174,706,176

kibi-: kilobinary, mebi-: megabinary, gibi-: gigabinary,
tebi-: terabinary, pebi-: petabinary, exbi-: exabinary,
zebi-: zettabinary, yobi-: yottabinary

Radix

- Radix(기수법)
- 수를 나타내는 방식
- N진법
  - Digits의 개수가 N인 진법
  - 숫자(N)으로 표시

Radix	Digits	표기
10진법	0 ~ 9	140
2진법	0, 1	$10001100_{(2)}$
8진법	0 ~ 7	$214_{(8)}$
16진법	0 ~ 9, A ~ F(10 ~ 15)	$8C_{(16)}$

< N 진법 >

- 10진수(Decimal)
  - $1x10^2 + 4x10^1 = 140$
- 2진수(Binary)
  - $10001100_{(2)} = 1x2^7 + 1x2^3 + 1x2^2 = 140$
- 16진수(Hexadecimal)
  - $8C_{(16)} = 8x16^1 + Cx16^0 = 8x16 + 12 = 140$

Radix

표 2-2 2진수에 해당하는 8진수, 16진수, 10진수 표현

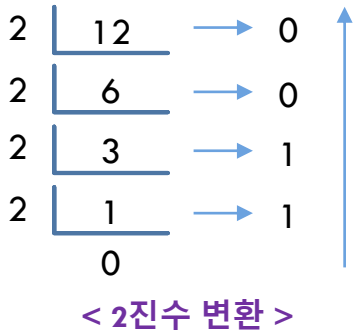
2진수	8진수	10진수
000	0	0
001	1	1
010	2	2
011	3	3
100	4	4
101	5	5
110	6	6
111	7	7

2진수	16진수	10진수	2진수	16진수	10진수
0000	0	0	1000	8	8
0001	1	1	1001	9	9
0010	2	2	1010	A	10
0011	3	3	1011	B	11
0100	4	4	1100	C	12
0101	5	5	1101	D	13
0110	6	6	1110	E	14
0111	7	7	1111	F	15

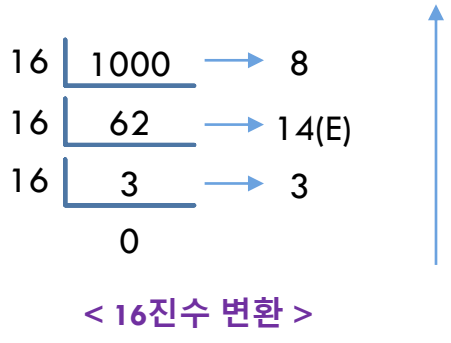
Radix Conversion(1)

- 2진수/16진수 → 10진수
  - 각 자리수에 Base의 거듭제곱을 곱해서 더함
  - 2진수 → 10진수
    - $1100.101_{(2)} = 1x2^3 + 1x2^2 + 1x2^{-1} + 1x2^{-3} = 12 + 0.5 + 0.125 = 12.625$
  - 16진수 → 10진수
    - $A3.D2_{(16)} = 10x16^1 + 3x16^0 + 13x16^{-1} + 2x16^{-2} = 163 + 13x0.0625 + 2x0.00390625 = 63.8203125$

- 10진수 → 2진수/16진수
  - 10진수 값을 몫이 0이 될 때 까지 Base로 나누고, 나머지 값을 역순으로 연결
  - 10진수 → 2진수
    - $12 = 1100_{(2)}$
  - 10진수 → 16진수
    - $1000 = 3E8_{(16)}$



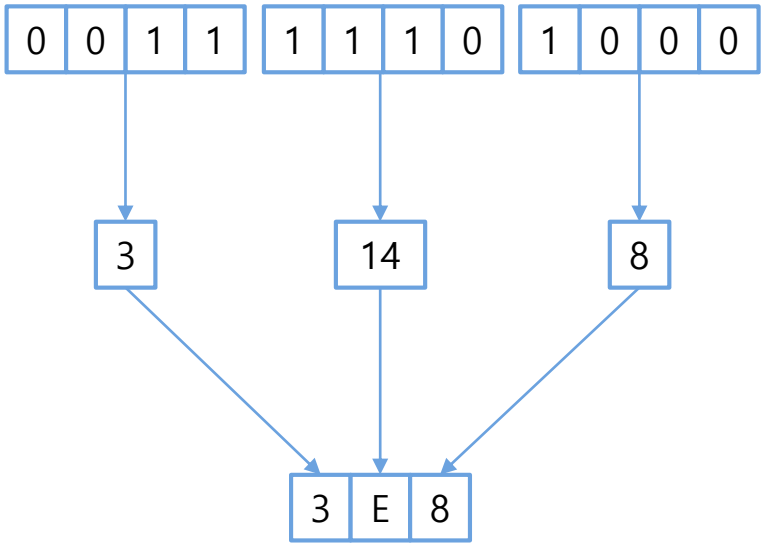
< 2진수 변환 >



< 16진수 변환 >

Radix Conversion(2)

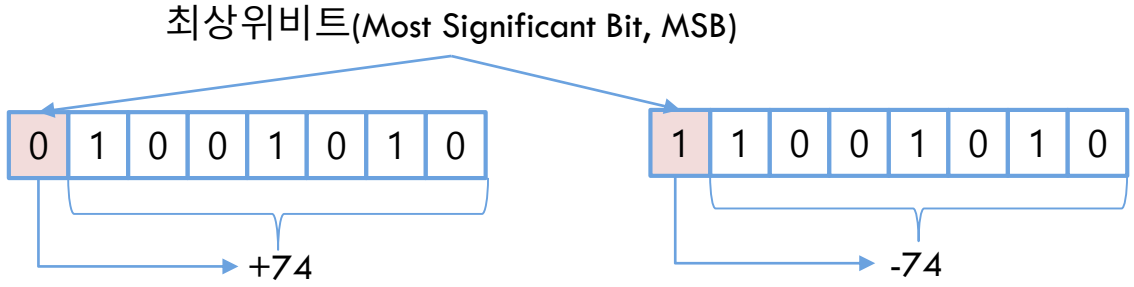
- 2진수 → 16진수
- 4bit식 묶어서 표현
  - $1000 = 0011\ 1110\ 1000_{(2)} = 3E8_{(16)}$



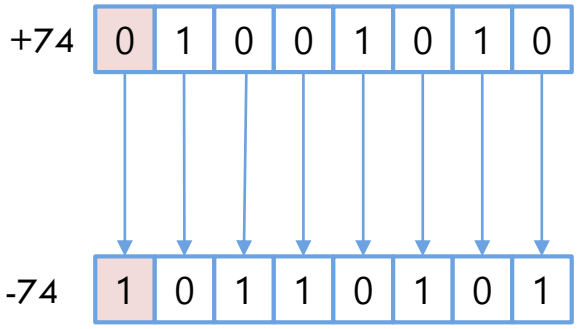
2진수 표현(1)

- 2진수의 음수 표현
- MSB를 부호 bit로 사용
  - 0이면 양수 / 1이면 음수

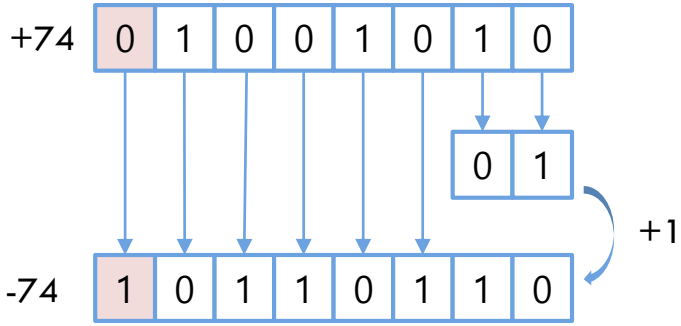
- 2진수의 음수 표현 방식
- 부호와 절대값(sign-magnitude)
  - MSB를 제외한 bit는 절대값을 표현
- 1의 보수(1's complement)
  - 양수 값의 0 → 1, 1 → 0으로 변환
- 2의 보수(2's complement)
  - 양수 값의 1의 보수 + 1



< 부호와 절대값 >



< 1의 보수 >



< 2의 보수 >

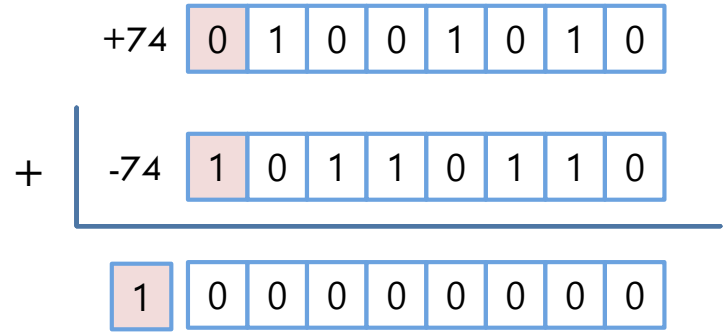
2진수 표현

표 2-3 수의 표현 방법에 따른 10진수 대응 값

4비트 2진수	부호 없는 수	부호와 절댓값	1의 보수	2의 보수
0000	0	0	0	0
0001	1	1	1	1
0010	2	2	2	2
0011	3	3	3	3
0100	4	4	4	4
0101	5	5	5	5
0110	6	6	6	6
0111	7	7	7	7
1000	8	-0	-7	-8
1001	9	-1	-6	-7
1010	10	-2	-5	-6
1011	11	-3	-4	-5
1100	12	-4	-3	-4
1101	13	-5	-2	-3
1110	14	-6	-1	-2
1111	15	-7	-0	-1

2진수 표현(2)

- 2진수의 뺄셈
  - 양수 값과 2의 보수를 더하면 유효 자리수는 항상 0
  - X + Y = 0 ↔ X = -Y
  - X – Y = X + (X의 2의 보수)
    - 뺄셈 연산기를 직접 구현하는 대신 2의 보수를 더하여 뺄셈을 구현



< +74 + -74 = 0 >

- 정수의 범위
  - n bit이 표시할 수 있는 정수 값의 개수는 2^n
  - 2의 보수는 0이 양수로 취급되어, 표시 범위 $-2^{n-1} \sim 2^{n-1}-1$

크기	범위
1byte	-2^7(-128) ~ 2^7-1(127)
2byte	-2^{15}(-32,768)~ 2^{15}-1(32,767)
4byte	-2^{31}(-2,147,483,648) ~ 2^{31}-1(2,147,483,647)
8byte	-2^{63} ~ 2^{63}-1

< 정수 값의 범위 >

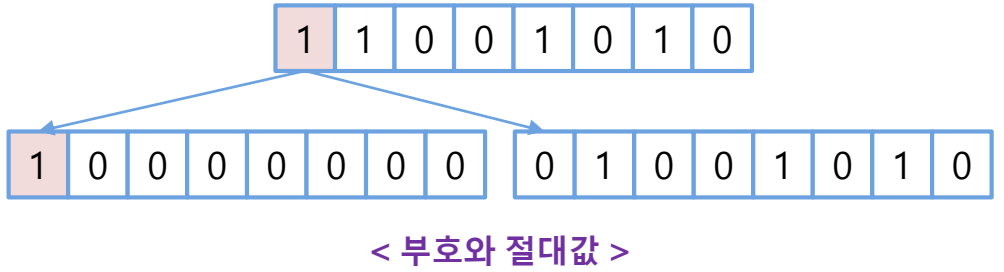
2진수	부호 없는 정수	2의 보수
000	0	0
001	1	1
010	2	2
011	3	3
100	4	-4
101	5	-3
110	6	-2
111	7	-1

< 3bit 2진수의 표현 >

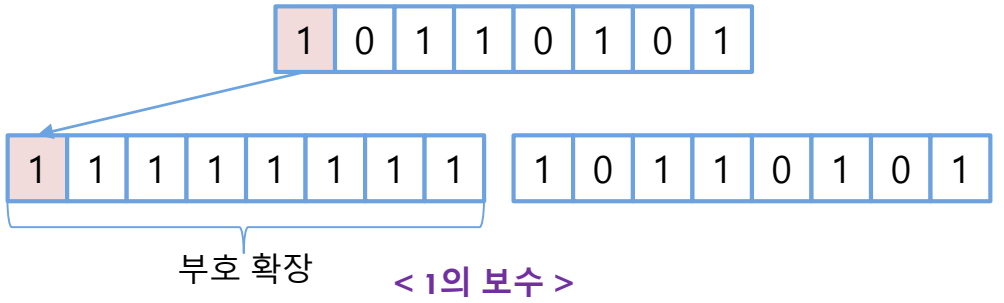
2진수 표현(3)

- 부호 확장
- bit수가 늘어날 때, 부호 bit를 처리하는 방법

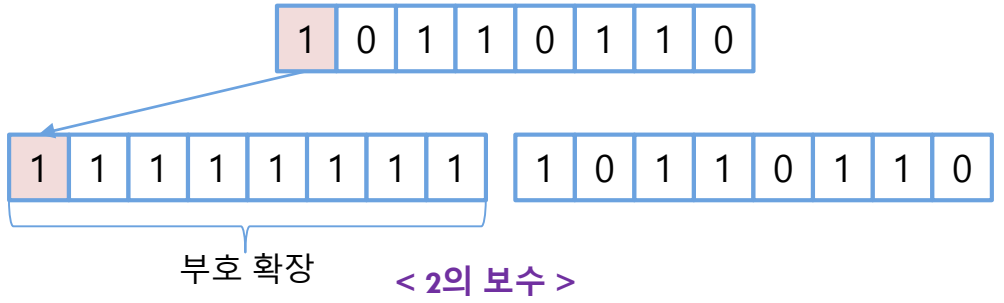
- 부호와 절대값(sign-magnitude)
  - 부호만 MSB로 옮기고, 나머지는 0으로 채움
- 1의 보수(1's complement)
  - 늘어난 bit를 부호와 같은 bit로 채움
- 2의 보수(2's complement)
  - 늘어난 bit를 부호와 같은 bit로 채움



< 부호와 절대값 >



< 1의 보수 >



< 2의 보수 >

Real Number

- Real Number(실수) 표현
- 고정 소수점(Fixed point number)
  - 정수 값과 소수 값을 분리해서 표현
- 부동 소수점(Floating point number)
  - 부호(Sign), 지수(Exponent), 가수(Mantissa)로 표현
  - $N = (-1)^S \times M \times 2^E$
  - 정밀도에 따라 Single precision(단정도) / Double precision(배정도)로 구분

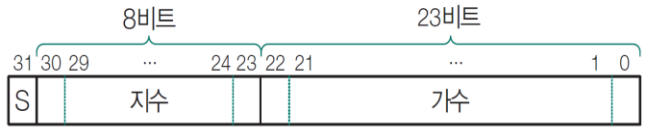
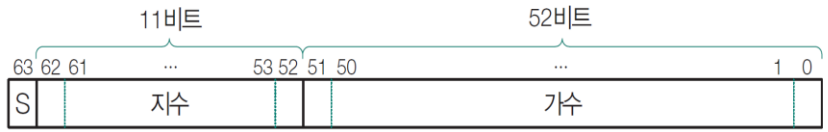
구분	IEEE 754 표준 부동 소수점 수의 비트 할당	바이어스
단정도 부동 소수점 수		127
배정도 부동 소수점 수		1023

그림 2-5 단정도 및 배정도 부동 소수점 수에 할당된 비트 수

단정도(단일정밀도): 32비트, 배정도(2배정밀도): 64비트, 4배정도(4배정밀도): 128비트

목 차

Code

❖ Code(코드 or 부호)

- 문자를 컴퓨터에 저장하는 방식
- Ex) Binary Coded Decima(BCD) / Excess-3 Code / etc...

❖ Gray Code

- 연속된 값들 간의 code 차이는 1bit
- Data 전송시 오류 발생 확률 감소

정수	Binary Code	Gray Code
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

< Gray Code >

Character Code(1)

- Character Code(문자 코드)
- 모든 문자는 숫자로 변환되어 저장됨
  - Character Set(문자 집합)과 Encoding(인코딩)방식이 필요

- ASCII(American Standard Code for Information Interchange) Code
- 미국 국립 표준 연구소(ANSI)가 제정한 정보 교환용 미국 표준 코드
- 영문 Alphabet을 사용하는 문자 처리 방식
- 1Byte로 구성
  - 많은 글자를 표현할 수 없음

- 한글 Code
- Character Set - KS X 1001, KS X 1002, …
- Encoding 방식 - EUC-KR, CP949, MS949 …

0x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
B0A0		가	각	간	갇	갈	갉	갊	감	갑	값	갓	갔	강	갖	갗
B0B0	같	갚	갛	개	객	갠	갤	갬	갭	갯	갰	갱	갸	갹	갼	걀
B0C0	걋	걍	걔	걘	걜	거	걱	건	걷	걸	걺	검	겁	것	겄	겅
B0D0	겆	겉	겊	겋	게	겐	겔	겜	겝	겟	겠	겡	겨	격	겪	견
B0E0	겯	결	겸	겹	겻	겼	경	곁	계	곈	곌	곕	곗	고	곡	곤
B0F0	곧	골	곪	곬	곯	곰	곱	곳	공	곶	과	곽	관	괄	괆	

< KS X 1001 >

32	space	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

< ASCII Code >

Character Code(2)

- Unicode(유니코드)
- Unicode Consortium에서 제정
- 전 세계의 모든 문자를 다루도록 설계된 문자 처리 방식
- Plain(평면) 단위로 구별
  - Unicode 한 글자의 크기 ≠ 16bit
  - 가장 많이 쓰이는 BMP(다국어 기본 평면)은 16bit
- 한글은 U+로 시작 (U+AC00 : “가”)
  - 주로 U+AC00 ~ U+DA7F에 존재

다국어 기본 평면 BMP		다국어 보충 평면 SMP		상형 문자 보충 평면 SIP		상형 문자 제3 평면 TIP	특수 목적 보충 평면 SSP
U+0000	U+8000	U+10000	U+18000	U+20000	U+28000	U+30000	U+E0000
U+1000	U+9000	U+11000		U+21000	U+29000	U+31000	
U+2000	U+A000	U+12000		U+22000	U+2A000		
U+3000	U+B000	U+13000	U+1B000	U+23000	U+2B000		
U+4000	U+C000	U+14000		U+24000	U+2C000		
U+5000	U+D000		U+1D000	U+25000	U+2D000		
U+6000	U+E000	U+16000	U+1E000	U+26000	U+2E000		
U+7000	U+F000	U+17000	U+1F000	U+27000	U+2F000		

< Unicode Plain >

U+	0	1	2	3	4	5
AC00	가	각	갂	갃	간	갅
AC10	감	갑	값	갓	갔	강
AC20	갠	갡	갢	갣	갤	갥
AC30	갰	갱	갲	갳	갴	갵
AC40	갶	갷	갸	갹	갺	갻
AC50	갼	갽	갾	갿	걀	걁
AC60	걂	걃	걄	걅	걆	걇

< Unicode >

Character Code(3)

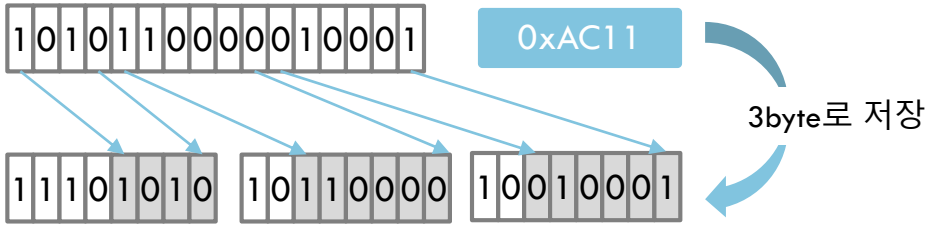
- Encoding(인코딩 방식)
  - Unicode를 Memory에 저장하는 방식
    - 대표적으로 UTF-8, UTF-16

Code 범위	UTF-8	설명
0x0 ~ 0x7F	0xxxxxxx	ASCII와 동일
0x80 ~ 0x7FF	110xxxxx 10xxxxxx	첫 byte는 110 or 1110 / 나머지는 10으로 시작
0x800 ~ 0xFFFF	1110xxxx 10xxxxxx 10xxxxxx	

< UTF-8 >

- UTF-8
  - 가장 많이 쓰이는 Encoding 방식
    - 기존 ASCII Code와 호환성 유지(ASCII 문자의 MSB가 0)

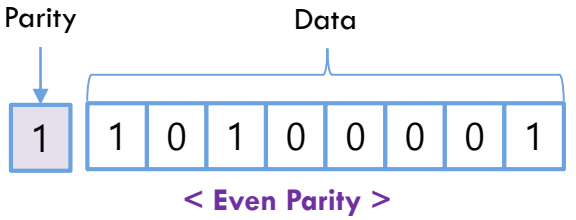
- UTF-16
  - BMP(기본 평면) 문자는 16bit 단위로 저장
    - Ex) Java의 기본 Encoding 방식



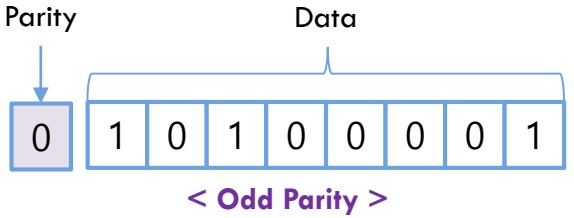
< UTF-8 예제 >

Error Detecting Code

- Parity Bit(패리티 비트)
  - Data에 오류가 있는지 확인하기 위해 추가된 bit
  - 오류 검출만 가능
  - Even(짝수) parity
    - Data의 1의 개수를 짝수로 맞춤
  - Odd(홀수) parity
    - Data의 1의 개수를 홀수로 맞춤



< Even Parity >



< Odd Parity >

- Hamming Code
  - Data의 오류를 검출 및 정정을 위해 parity bit들이 추가
  - $2^P \geq D + P + 1$
    - D : bit 수 / P : 추가되는 bit 수
    - Ex) D = 4 → P = 3 / D = 8 → P = 4 / D = 16 → P = 5

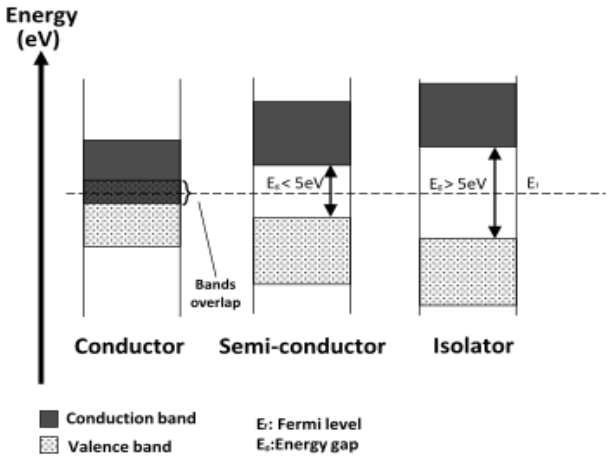
표 2-16 해밍 코드에서 패리티 비트의 위치와 패리티 생성 영역

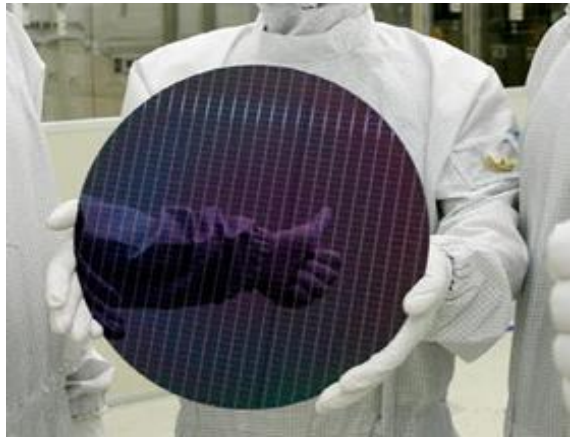
비트 위치	1	2	3	4	5	6	7	8	9	10	11	12
기호	P_1	P_2	D_3	P_4	D_5	D_6	D_7	P_8	D_9	D_{10}	D_{11}	D_{12}
P_1 영역	✓		✓		✓		✓		✓		✓	
P_2 영역		✓	✓			✓	✓			✓	✓	
P_4 영역				✓	✓	✓	✓					✓
P_8 영역								✓	✓	✓	✓	✓

목 차

Physical Device(1)

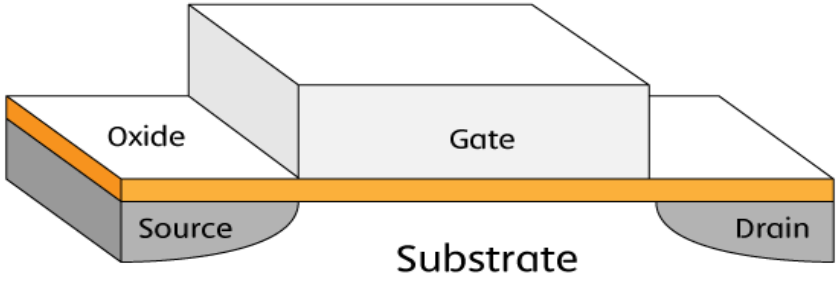
- Semiconductor
  - Conductor / Insulator / Semiconductor?

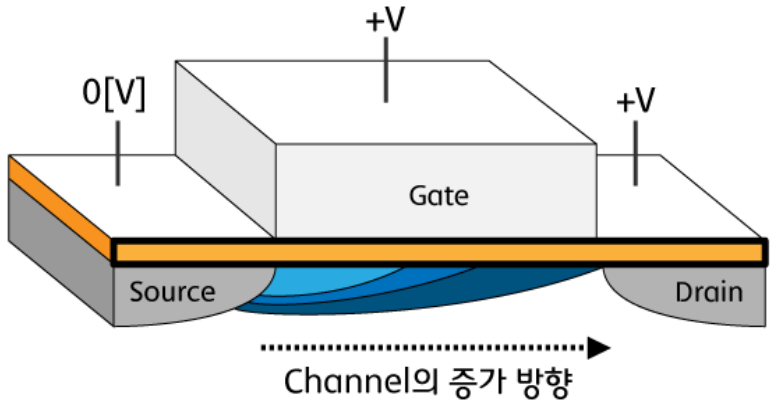




Physical Device(2)

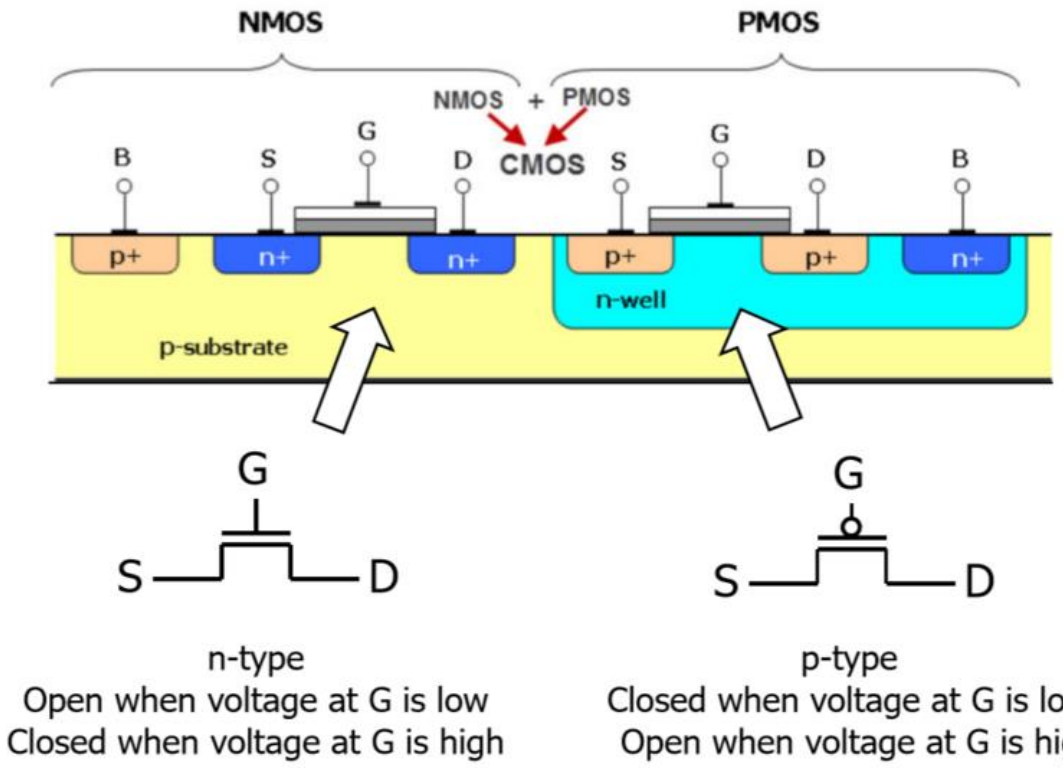
- Semiconductor
  - Conductor / Insulator / Semiconductor?
- Fabrication process
  - Channel length - 3nm?





Circuit Level

- nmos / pmos / cmos
  - Conductor / Insulator / Semiconductor?
- Fabrication process
  - Channel length - 3nm?



Logic Gate(1)

❖ Logic Level(논리 레벨)

- 연속적인 값을 가지는 전압을 불연속적인 Logic 값으로 표시
  - Analog 값을 Digital 값으로 변환

❖ Logic Gate(논리 게이트)

- 한 개 이상의 Input과 한 개 이상의 Output이 존재하는 회로
- Hardware 구성의 기본 요소
  - 0과 1의 신호를 사용

❖ Truth Table(진리표)

- 모든 가능한 입력에 대한 출력을 기록한 표

Input	Output
0	1
1	0

< Truth Table >

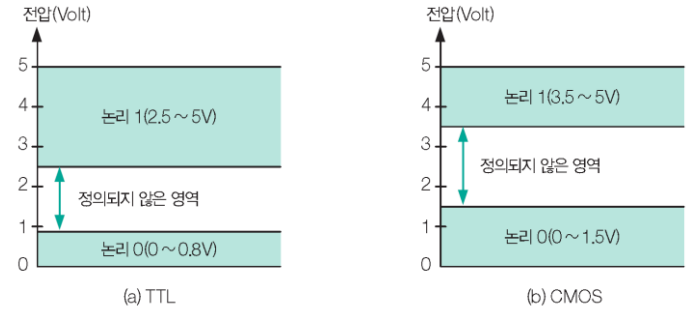
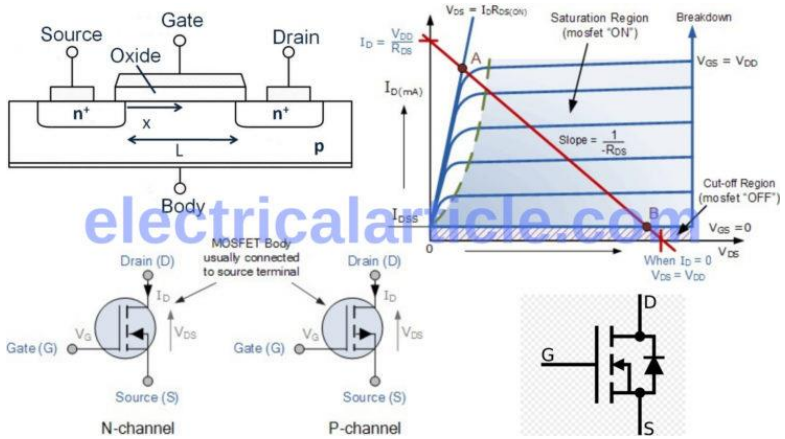


그림 3-1 TTL과 CMOS 논리 레벨 정의 영역

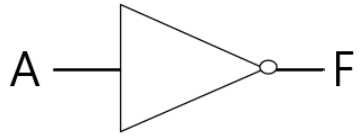


Logic Gate(2)

- NOT Gate(Inverter)
  - 논리 부정
  - Input과 Output의 값이 다름

A	F
0	1
1	0

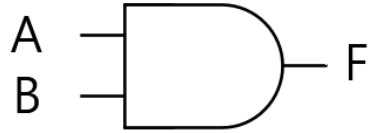
< NOT Gate >



- AND Gate
  - Input이 모두 1이면 Output도 1, 아니면 0

A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

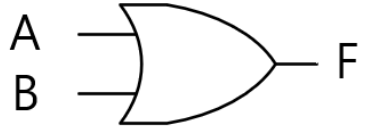
< AND Gate >



- OR Gate
  - Input이 모두 0이면 Output도 0, 아니면 1

A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

< OR Gate >

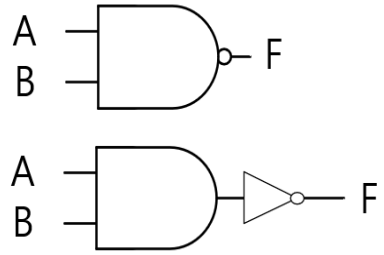


Logic Gate(3)

- NAND Gate
  - Input이 모두 1이면 Output이 0, 아니면 1
  - AND Gate의 Output의 논리 부정
    - NOT-AND

A	B	F
0	0	1
0	1	1
1	0	1
1	1	0

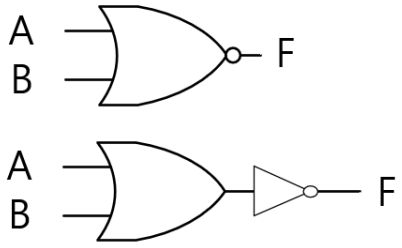
< NAND Gate >



- NOR Gate
  - Input이 모두 0이면 Output이 0, 아니면 1
  - OR Gate의 Output의 논리 부정
    - NOT-OR

A	B	F
0	0	1
0	1	0
1	0	0
1	1	0

< NOR Gate >



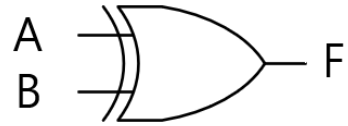
Logic Gate(4)

- XOR(Exclusive-OR) Gate
  - Input 1의 개수가 홀수이면 Output이 0

- XNOR Gate
  - Input 1의 개수가 짝수이면 Output이 0
  - XOR Gate의 Output의 논리 부정
    - NOT-XOR

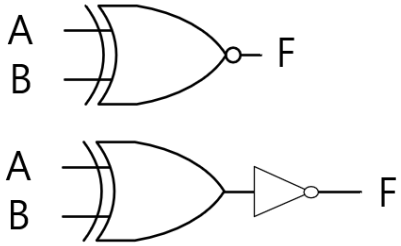
A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

< XOR Gate >



A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

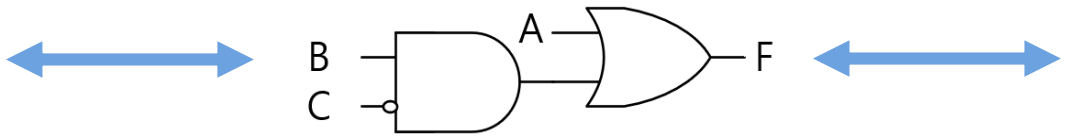
< NOR Gate >



목 차

Boolean Algebra

- Boolean Algebra(불 대수)
- 1854년 영국의 수학자 Geroge Boole이 제안
- 논리식을 형식화하여 표현하는 방식
- 기본적으로 AND, OR, NOT를 이용하여 표현
  - A AND B → AB
  - A OR B → $A + B$
  - NOT A → $\bar{A}$

- Ex) $F = A + B\bar{C}$ 

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

< F = A + B$\bar{C}$>

공리 및 법칙(1)

❖ Boolean Algebra 공리 및 법칙

공리	내용
P1	A = 0 or A = 1
P2	0·0 = 0
P3	1·1 = 1
P4	0 + 0 = 0
P5	1 + 1 = 1
P6	0·1 = 1·0 = 0
P7	0 + 1 = 1 + 0 = 1

< Boolean Algegra 공리 >

법칙	내용
기본 법칙	$A+0=0+A=A$
	$A+1=1+A=1$
	$A\cdot 1=1\cdot A=A$
	$A\cdot 0=0\cdot A=0$
	$A+A=A\cdot A=A$
	$A+\bar{A}=1$
	$A\cdot\bar{A}=0$
교환 법칙	$A+B=B+A$
	$A\cdot B=B\cdot A$

법칙	내용
결합 법칙	$(A+B)+C=A+(B+C)$
	$(A\cdot B)\cdot C=A\cdot(B\cdot C)$
분배 법칙	$A\cdot(B+C)=A\cdot B+A\cdot C$
	$A+B\cdot C=(A+B)\cdot(A+C)$
드모르간의 정리	$\overline{A+B}=\bar{A}\cdot\bar{B}$
	$\overline{A\cdot B}=\bar{A}+\bar{B}$
흡수 법칙	$A+A\cdot B=A$
	$A\cdot(A+B)=A$

< Boolean Algegra법칙 >

공리 및 법칙(2)

- 분배 법칙
- $A + B \cdot C = (A + B) \cdot (A + C)$

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

< F = A + BC >

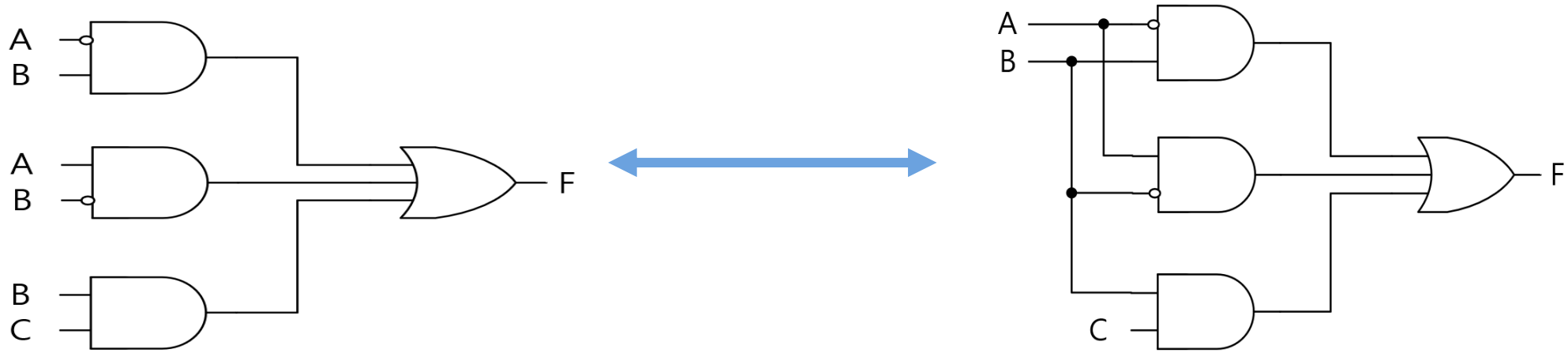
A	B	C	A+B	A+C	F
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	1	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

< F = (A + B)(A + C) >

논리 회로 변환(1)

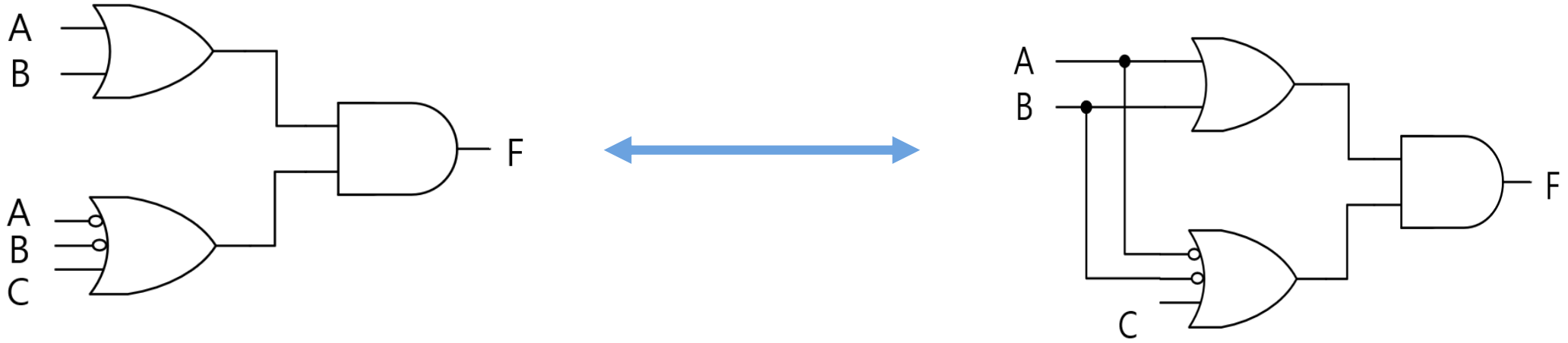
- 논리 회로 변환
  - Boolean Algebra 식은 논리 회로로 표현이 가능

- AND-OR 논리 회로(Sum of Product, SOP)
  - 주 연산자가 OR인 경우
    - 우선, AND 대수식을 AND Gate로 변환하고, OR Gate의 입력으로 연결
    - Ex) $F(A,B,C) = \bar{A}B + A\bar{B} + BC$



논리 회로 변환(2)

- OR-AND 논리 회로(Product of Sum, POS)
- 주 연산자가 AND인 경우
  - 우선, OR 대수식을 OR Gate로 변환하고, AND Gate의 입력으로 연결
  - Ex) $F(A,B,C) = (A+B)(\bar{A}+\bar{B}+C)$



논리식의 간소화

- Boolean Algebra 법칙을 이용한 논리식의 간소화
- $\bar{A}B\bar{C} + \bar{A}BC + A\bar{B}\bar{C} + A\bar{B}C + ABC$
- $= \bar{A}B(\bar{C} + C) + A\bar{B}(\bar{C} + C) + ABC$
- $= \bar{A}B + A\bar{B} + ABC$

- $\bar{A}B\bar{C} + \bar{A}BC + A\bar{B}\bar{C} + A\bar{B}C + ABC$
- $= \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}\bar{C} + A\bar{B}C + ABC + A\bar{B}C$
- $= \bar{A}B(\bar{C} + C) + A\bar{B}(\bar{C} + C) + AC(B + \bar{B})$
- $= \bar{A}B + A\bar{B} + AC$

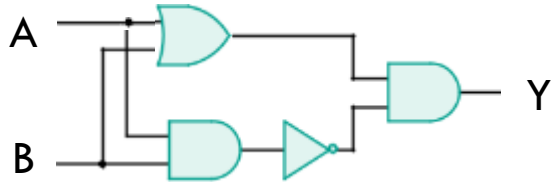
- $A + \bar{A}B$(분배 법칙)
- $= (A + \bar{A})(A + B) = A + B$

법칙	내용
결합 법칙	$(A + B) + C = A + (B + C)$
	$(A \cdot B) \cdot C = A \cdot (B \cdot C)$
분배 법칙	$A \cdot (B + C) = A \cdot B + A \cdot C$
	$A + B \cdot C = (A + B) \cdot (A + C)$
드모르간의 정리	$\overline{A + B} = \bar{A} \cdot \bar{B}$
	$\overline{A \cdot B} = \bar{A} + \bar{B}$
흡수 법칙	$A + A \cdot B = A$
	$A \cdot (A + B) = A$

법칙	내용
기본 법칙	$A + 0 = 0 + A = A$
	$A + 1 = 1 + A = 1$
	$A \cdot 1 = 1 \cdot A = A$
	$A \cdot 0 = 0 \cdot A = 0$
	$A + A = A \cdot A = A$
	$A + \bar{A} = 1$
	$A \cdot \bar{A} = 0$
교환 법칙	$A + B = B + A$
	$A \cdot B = B \cdot A$

기출문제

- 기출문제
- 5. 010110과 01110의 XOR결과는?
- 10. A=1101, B=0111이 입력되면, Y값은?

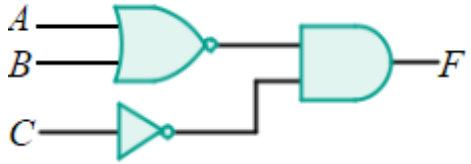


- 12. 불 대수식의정리 중 옳지않은것은?

① $A+AB=A$ ② $A+\overline{A}B=A+B$

③ $A+0=A$ ④ $A(\overline{A}+AB)=A+B$

- 23. 논리회로의 출력함수식 F를 구하시오.



- 28. $F=(\bar{A}+\bar{B}+\bar{C})(\bar{A}+B+\bar{C})$일때, $\bar{F}$식을 간소화하시오.

Homework

- 다음의 식이 성립하는지 Truth Table을 구해서 증명하시오
  - 1) $\overline{A + B + C} = \bar{A} \cdot \bar{B} \cdot \bar{C}$
  - 2) $A \cdot \bar{B} + \bar{B} \cdot \bar{C} + \bar{A} \cdot \bar{C} = A \cdot \bar{B} + \bar{A} \cdot \bar{C}$

- 다음 Boolean Algebra식을 논리 회로로 표시하시오
  - 5) $F = B\bar{C} + AB + ABD$
  - 6) $F = (A + B)(C + D)(\bar{A} + B + D)$

- 다음 논리식을 간소화하고, Truth Table을 구해서 증명하시오
  - 7) $(A\bar{B}(C + BD) + \bar{A}\bar{B})C$
  - 8) $\overline{AB + AC} + \bar{A}\bar{B}C$