# Ch.2 Boolean Algebra and Logic gates

## Basic Definitions

- mathematical system 만족한다고 가정.
    - ① **Closure** : 자연수의 집합 내에서 닫혀 있음을 의미
    - ② **Associative law**(결합법칙)
    - ③ **Commutative law**(교환법칙)
    - ④ **Identity element**(항등원) : e는 *연산자의 항등원, i) 0은 +연산자의 항등원 ii)
    - ⑤ **Inverse**(역원)
    - ⑥ **Distributive law**(분배법칙)

---

## Asiomatic Definition of Boolean Algebra

: Boolean 대수학적으로 정의 해보자면

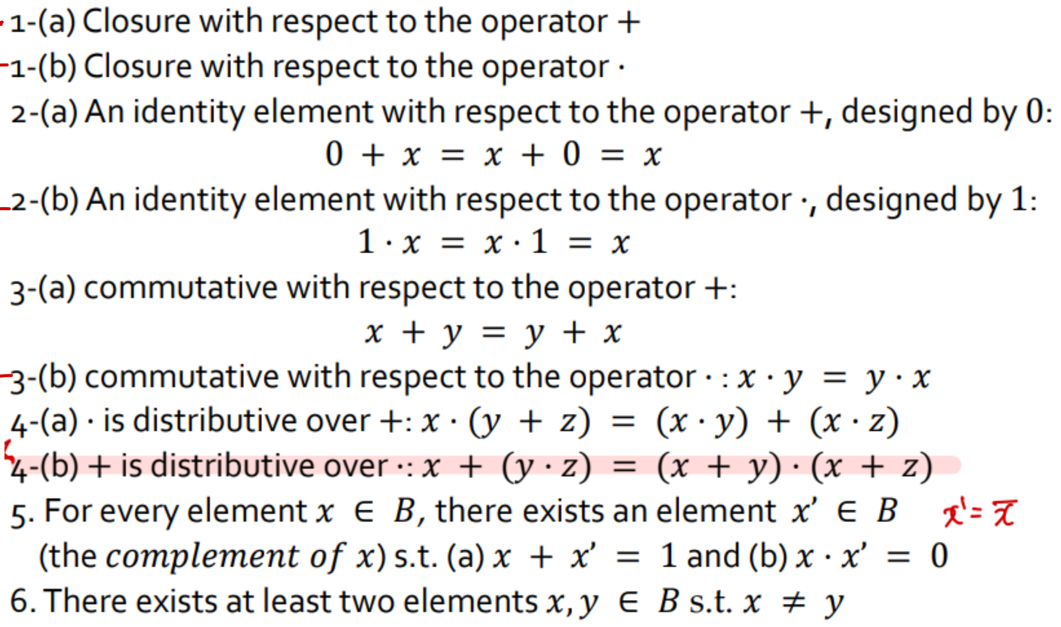
# Huntington의 가정

closure ①

항등원 ②

교환 ③

분배 ④



1-(a) Closure with respect to the operator +

1-(b) Closure with respect to the operator ·

2-(a) An identity element with respect to the operator +, designed by 0:

$$0 + x = x + 0 = x$$

2-(b) An identity element with respect to the operator ·, designed by 1:

$$1 \cdot x = x \cdot 1 = x$$

3-(a) commutative with respect to the operator +:

$$x + y = y + x$$

3-(b) commutative with respect to the operator · : $x \cdot y = y \cdot x$

4-(a) · is distributive over +: $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$

4-(b) + is distributive over ·: $x + (y \cdot z) = (x + y) \cdot (x + z)$

⑤ 5. For every element $x \in B$, there exists an element $x' \in B$ $x' = \bar{x}$ (the *complement of* $x$) s.t. (a) $x + x' = 1$ and (b) $x \cdot x' = 0$

⑥ 6. There exists at least two elements $x, y \in B$ s.t. $x \neq y$

⇒ do not include the associative law → this law holds for Boolean Algebra

→ 곱셈 분배도 되고 덧셈 분배도 가능!

- 4-(b) is valid for Boolean algebra, not for ordinary algebra
- Boolean algebra does not have additive or multiplicative inverse.

## Boolean vs. Ordinary algebra

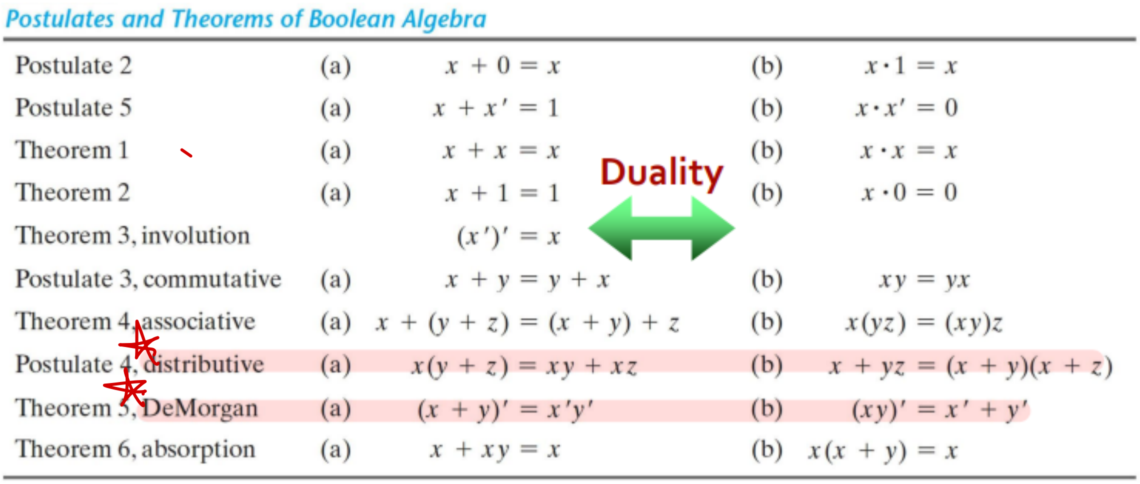
vs

- **Ordinary algebra** : real number → infinite
  - Complement(보수) is not available in ordinary algebra.
- **Two-valued Boolean algebra** : only two value {0,1}
  - Exatcly same as the AND, OR(Binary), and NOT(Unary)
  - Huntington postulates are valid for {0,1}
    1. closure
    2. 진리표에 의해서 증명 가능 → $0+0=0$ $1 \cdot 1=1$ $0+1=1+0=1$ $1 \cdot 0=0 \cdot 1=0$
    3. The commutative law is obvious
    4. The distributive law holds true for two operator : from the truth tables

5. Form the complement table
6. is satisfied cuz only two values exist. → 0, 1

## Basic Theorems and Properties of Boolean Algebra

- **Duality(쌍대성)** : the dual of any equation which is true is always true



*Postulates and Theorems of Boolean Algebra*

| | | | | |
|---|---|---|---|---|
| Postulate 2 | (a) | $x + 0 = x$ | (b) | $x \cdot 1 = x$ |
| Postulate 5 | (a) | $x + x' = 1$ | (b) | $x \cdot x' = 0$ |
| Theorem 1 | (a) | $x + x = x$ | (b) | $x \cdot x = x$ |
| Theorem 2 | (a) | $x + 1 = 1$ | (b) | $x \cdot 0 = 0$ |
| Theorem 3, involution | | $(x')' = x$ | | |
| Postulate 3, commutative | (a) | $x + y = y + x$ | (b) | $xy = yx$ |
| Theorem 4, associative | (a) | $x + (y + z) = (x + y) + z$ | (b) | $x(yz) = (xy)z$ |
| Postulate 4, distributive | (a) | $x(y + z) = xy + xz$ | (b) | $x + yz = (x + y)(x + z)$ |
| Theorem 5, DeMorgan | (a) | $(x + y)' = x'y'$ | (b) | $(xy)' = x' + y'$ |
| Theorem 6, absorption | (a) | $x + xy = x$ | (b) | $x(x + y) = x$ |

⇒ 진리표를 통해 증명도 가능

$x(y+z) = xy + xz$

$x + yz = (x+y)(x+z)$

- Some more laws

1. x + xy = x → $x(1+y) = x$
2. xy + xy' = x → $x(y+y') = x$
3. x + x'y = x + y → $(x+x')(x+y) = x+y$
4. x(x+y) = x → $xx + xy = x + xy = x$
5. (x+y)(x+y') = x → $x + (yy') = x$
6. x(x'+y) = xy → $xx' + xy = xy$
7. xy + x'z + yz = xy + x'z
8. (x+y)(x'+z)(y+z) = (x+y)(x'+z)

8. $(x+y)(z+x'y)$

$= xz + zy + x'$

$= z(x+y) + x'$

$= (x'+z)(x+y)$

7. $xy + x'z + yz = xy + x'z + yz(x+x')$

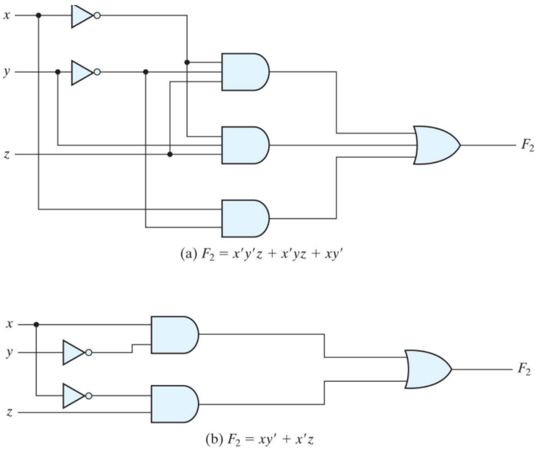
$= xy + x'z + yzx + yzx'$

$= xy(1+z) + x'z(y+1)$

$= xy + x'z$

## Simplication of Boolean Function

$$\begin{aligned} F_2 &= x'y'z + x'yz + xy' \\ &= x'z(y' + y) + xy' \\ &= x'z + xy' \end{aligned}$$



(a) $F_2 = x'y'z + x'yz + xy'$

(b) $F_2 = xy' + x'z$

등가

⇒ two diffrent but equivalent circuits

⇒ Fewer gates is “better”

- cost less
- less power

⇒ fewer gate 찾기 위해 work 필요

## Algebraic Manipulation

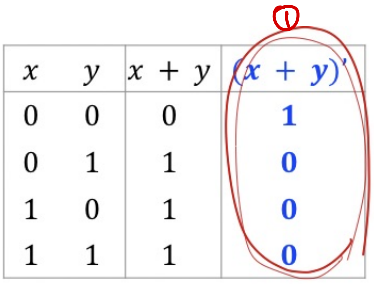
- $x(x' + y) = xx' + xy = 0 + xy = xy$
- $x + x'y = (x + x')(x + y) = 1(x + y) = x + y$
- $(x + y)(x + y') = x + xy + xy' + yy' = x(1 + y + y') = x$
- $xy + x'z + yz = xy + x'z + yz(x + x')$
  $= xy + x'z + xyz + x'yz$
  $= xy(1 + z) + x'z(1 + y)$
  $= xy + x'z$
- $(x + y)(x' + z)(y + z) = (x + y)(x' + z)$, by duality from function 4

## Complement of  a Function

- f(x, y, z) = x(y’z + yz)
- **De Morgan’s low**

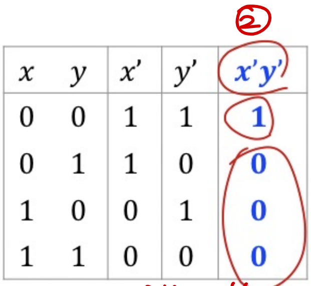
$$\begin{aligned} f'(x,y,z) &= (x(y'z' + yz))' && [\text{ complement both sides }] \\ &= x' + (y'z' + yz)' && [\text{ because } (xy)' = x' + y' ] \\ &= x' + (y'z')'(yz)' && [\text{ because } (x + y)' = x'y' ] \\ &= x' + (y + z)(y' + z') && [\text{ because } (xy)' = x' + y', \text{twice}] \end{aligned}$$

⇒ (x+y)’ = x’y’ (cuz the truth table)



| $x$ | $y$ | $x+y$ | $(x+y)'$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 |

①

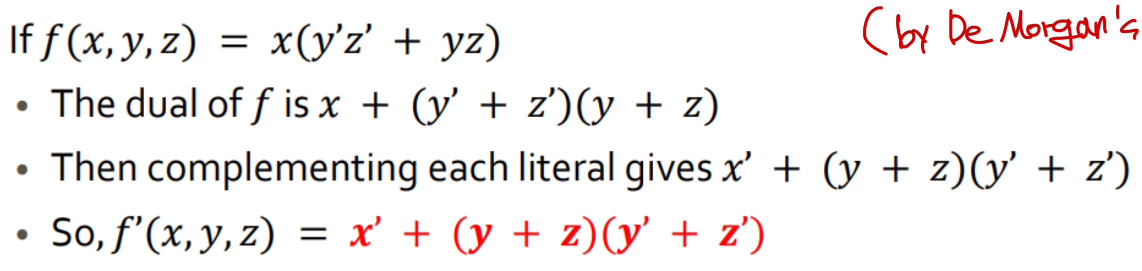


| $x$ | $y$ | $x'$ | $y'$ | $x'y'$ |
|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |

②

↳ 이와 같은 결과가 나온다면 ①,②로 작성 가능 (by De Morgan's Law)

- **dual of the func**



If $f(x,y,z) = x(y'z' + yz)$

- The dual of $f$ is $x + (y' + z')(y + z)$
- Then complementing each literal gives $x' + (y + z)(y' + z')$
- So, $f'(x,y,z) = \mathbf{x' + (y + z)(y' + z')}$

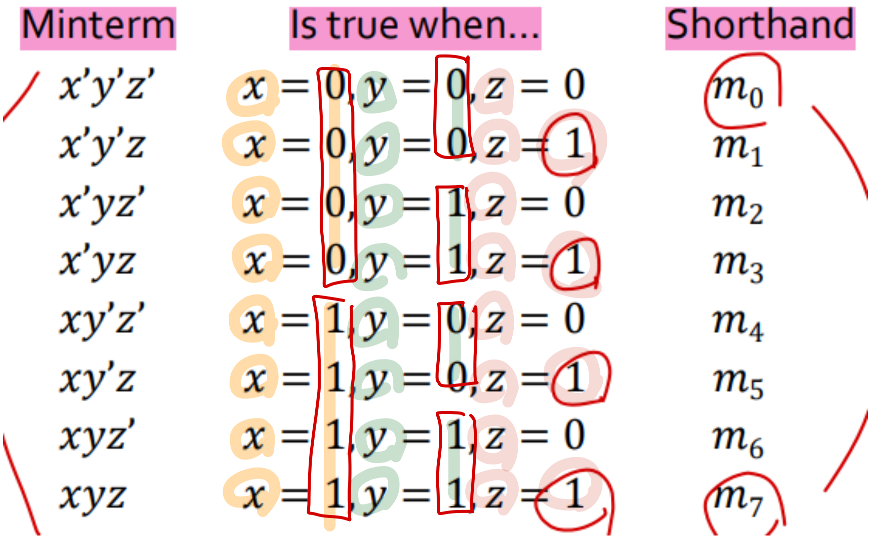
$f(x,y,z) = x(y'z + yz)$

- In general,
  - (A+B+C+ … + F)’ = A’B’C’ … F’
  - (ABCD… F)’ = A’+B’+C’ … + F’

## Minterms

최소항

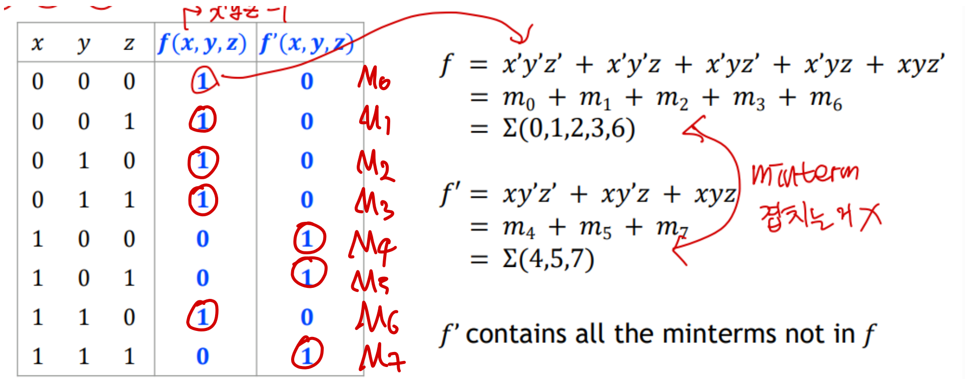
- **a special product(AND) of literals (논리곱)**
- n개의 variable → 2의 n승 개의 minterms



| Minterm | Is true when… | Shorthand |
|---|---|---|
| $x'y'z'$ | $x=0, y=0, z=0$ | $m_0$ |
| $x'y'z$ | $x=0, y=0, z=1$ | $m_1$ |
| $x'yz'$ | $x=0, y=1, z=0$ | $m_2$ |
| $x'yz$ | $x=0, y=1, z=1$ | $m_3$ |
| $xy'z'$ | $x=1, y=0, z=0$ | $m_4$ |
| $xy'z$ | $x=1, y=0, z=1$ | $m_5$ |
| $xyz'$ | $x=1, y=1, z=0$ | $m_6$ |
| $xyz$ | $x=1, y=1, z=1$ | $m_7$ |

↳ 순서 바뀌지 X

- **Sum of Minterms Form**
  - Every function can be written as a sum of minterms
  - function output is 1 되는 minterm들의 합으로 function을 표현 가능



→ x'y'z =1

| $x$ | $y$ | $z$ | $f(x,y,z)$ | $f'(x,y,z)$ | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | $M_0$ |
| 0 | 0 | 1 | 1 | 0 | $M_1$ |
| 0 | 1 | 0 | 1 | 0 | $M_2$ |
| 0 | 1 | 1 | 1 | 0 | $M_3$ |
| 1 | 0 | 0 | 0 | 1 | $M_4$ |
| 1 | 0 | 1 | 0 | 1 | $M_5$ |
| 1 | 1 | 0 | 1 | 0 | $M_6$ |
| 1 | 1 | 1 | 0 | 1 | $M_7$ |

$$f = x'y'z' + x'y'z + x'yz' + x'yz + xyz'$$
$$= m_0 + m_1 + m_2 + m_3 + m_6$$
$$= \Sigma(0,1,2,3,6)$$

$$f' = xy'z' + xy'z + xyz$$
$$= m_4 + m_5 + m_7$$
$$= \Sigma(4,5,7)$$

minterm
겹치는거 X

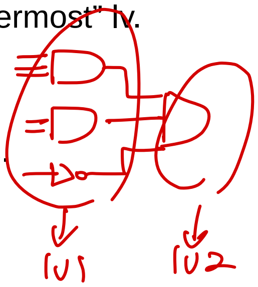
$f'$ contains all the minterms not in $f$

  - **dual idea(P → S ,** two level circuits)
    - POS expression contains : only AND operations at the “outermost” lv.

      ex). $f(x,y,z) = y'(x'+y+z')(x+z)$ ↪

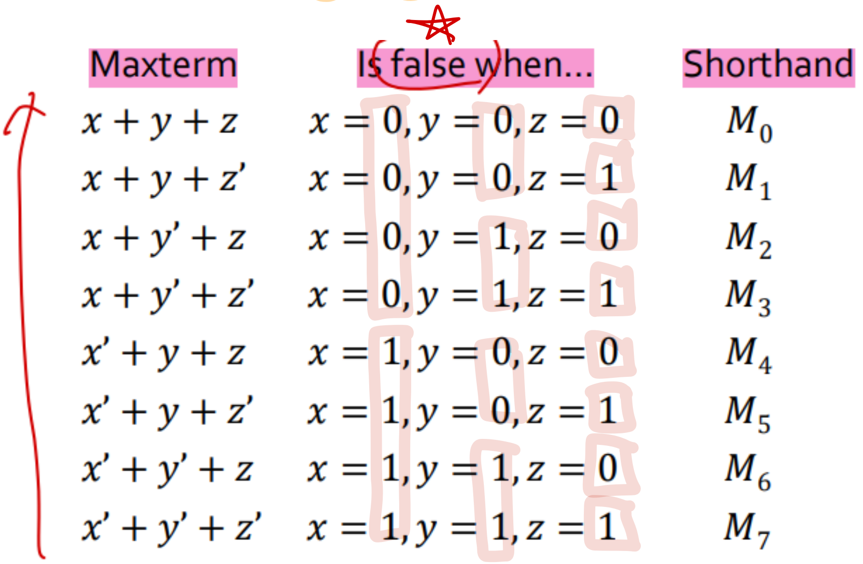
    - POS expression can be implemented with two-level circuits.

      Lv 1 : OR Gate
      Lv 2 : A single AND gate.



# Maxterms ( ↔ Minterms)

- **a sum of literals**
- n개의 variable → 2의 n승 개의 maxterms

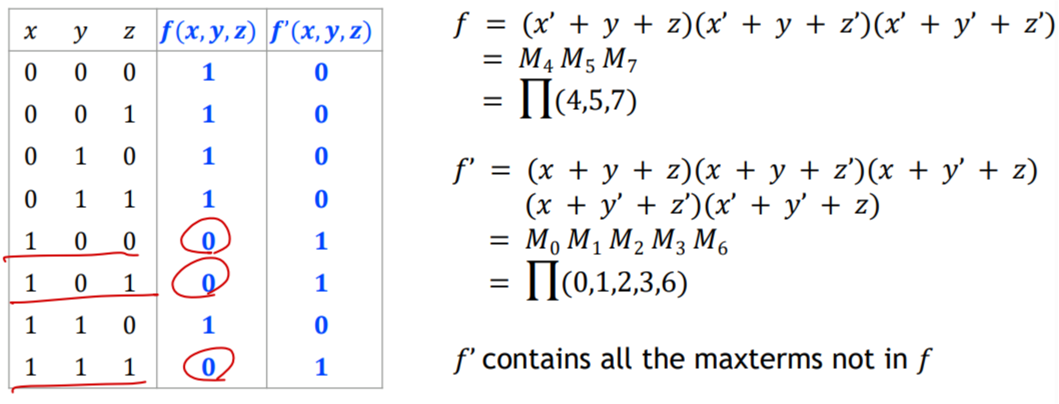


| Maxterm | Is false when... | Shorthand |
|---|---|---|
| $x + y + z$ | $x = 0, y = 0, z = 0$ | $M_0$ |
| $x + y + z'$ | $x = 0, y = 0, z = 1$ | $M_1$ |
| $x + y' + z$ | $x = 0, y = 1, z = 0$ | $M_2$ |
| $x + y' + z'$ | $x = 0, y = 1, z = 1$ | $M_3$ |
| $x' + y + z$ | $x = 1, y = 0, z = 0$ | $M_4$ |
| $x' + y + z'$ | $x = 1, y = 0, z = 1$ | $M_5$ |
| $x' + y' + z$ | $x = 1, y = 1, z = 0$ | $M_6$ |
| $x' + y' + z'$ | $x = 1, y = 1, z = 1$ | $M_7$ |

min

- **Product of maxterms form**
  - Every function can be written as a unique product of maxterms.
  - function output is 0 되는 maxterm들의 합으로 function을 표현 가능

minterm과 반대!!



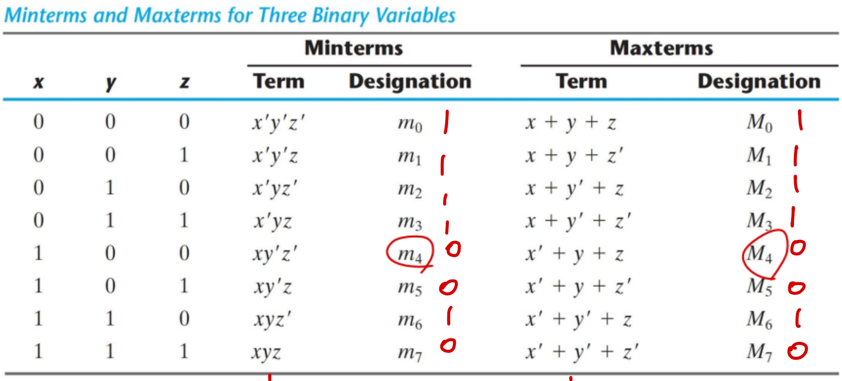
| $x$ | $y$ | $z$ | $f(x,y,z)$ | $f'(x,y,z)$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |

$$f = (x' + y + z)(x' + y + z')(x' + y' + z')$$
$$= M_4 M_5 M_7$$
$$= \prod(4,5,7)$$

$$f' = (x + y + z)(x + y + z')(x + y' + z)(x + y' + z')(x' + y' + z)$$
$$= M_0 M_1 M_2 M_3 M_6$$
$$= \prod(0,1,2,3,6)$$

$f'$ contains all the maxterms not in $f$

- **Minterms and maxterms are related**
  - Any minterm m_i 은 maxterm M_i의 complement



*Minterms and Maxterms for Three Binary Variables*

| | | | Minterms | | Maxterms | |
|---|---|---|---|---|---|---|
| $x$ | $y$ | $z$ | Term | Designation | Term | Designation |
| 0 | 0 | 0 | $x'y'z'$ | $m_0$ 1 | $x + y + z$ | $M_0$ 1 |
| 0 | 0 | 1 | $x'y'z$ | $m_1$ 1 | $x + y + z'$ | $M_1$ 1 |
| 0 | 1 | 0 | $x'yz'$ | $m_2$ 1 | $x + y' + z$ | $M_2$ 1 |
| 0 | 1 | 1 | $x'yz$ | $m_3$ 1 | $x + y' + z'$ | $M_3$ 1 |
| 1 | 0 | 0 | $xy'z'$ | $m_4$ 0 | $x' + y + z$ | $M_4$ 0 |
| 1 | 0 | 1 | $xy'z$ | $m_5$ 0 | $x' + y + z'$ | $M_5$ 0 |
| 1 | 1 | 0 | $xyz'$ | $m_6$ 1 | $x' + y' + z$ | $M_6$ 1 |
| 1 | 1 | 1 | $xyz$ | $m_7$ 0 | $x' + y' + z'$ | $M_7$ 0 |

↓ True ↓ False

ex) m_4' = M_4

## Coverting between standard forms(POS , SOP)

$$f = \Sigma(0,1,2,3,6)$$
$$= \prod(4,5,7)$$

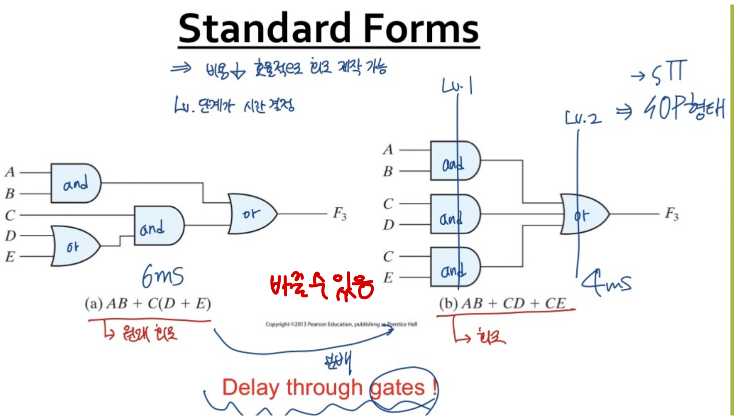
→ minterm에서 안 나온 number을 maxterm으로 converting

**proof :**

| | | | |
|---|---|---|---|
| From before | $f$ | $= \Sigma(0,1,2,3,6)$ | |
| and | $f'$ | $= \Sigma(4,5,7)$ | |
| | | $= m_4 + m_5 + m_7$ | |
| complementing | $(f')'$ | $= (m_4 + m_5 + m_7)'$ | |
| so | $f$ | $= m_4' m_5' m_7'$ | [ DeMorgan's law ] |
| | | $= M_4 M_5 M_7$ | [ By the previous page ] |
| | | $= \prod(4,5,7)$ | |

## Standard forms of expressions

- 비용 적게 효율적으로 회로 제작 가능(lv 단계의 수가 시간을 결정)
- 항상 Delay through gates.
- SOP expression contains
  - only OR operations at the "outermost" level.
  - using a two-level circuit 사용 (lv1. and + lv2. or)
  - NOT gates are implicit
  - literals are reused
- POS expression contains → duality


**Standard Forms**

⇒ 비용↓ 효율적으로 회로 제작 가능

Lv. 단계가 시간 결정

(a) $AB + C(D + E)$

↳ 원래 회로

(b) $AB + CD + CE$

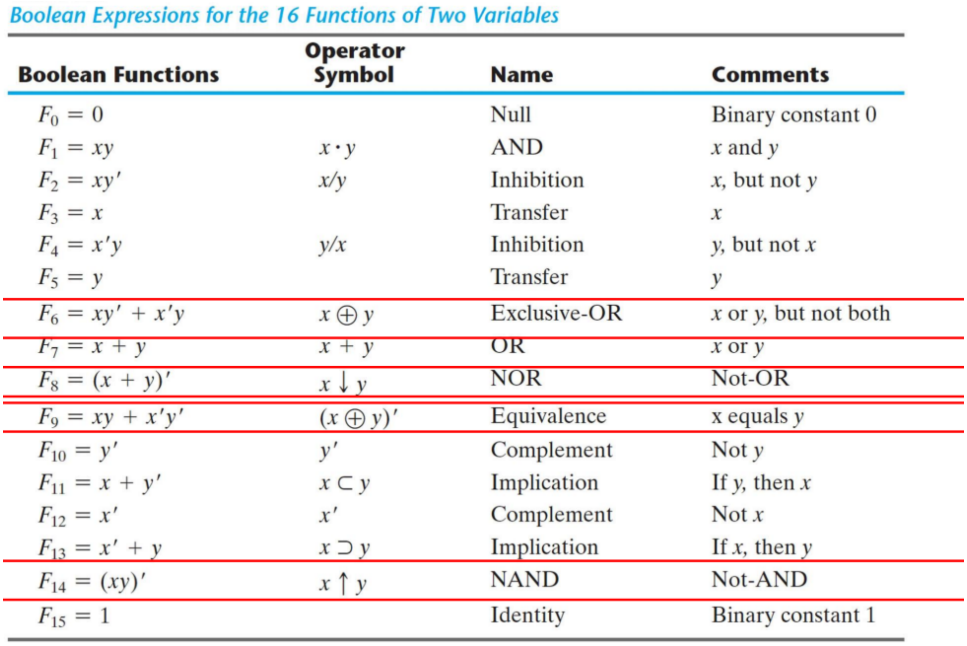
↳ 회로

Copyright ©2013 Pearson Education, publishing as Prentice Hall

분해

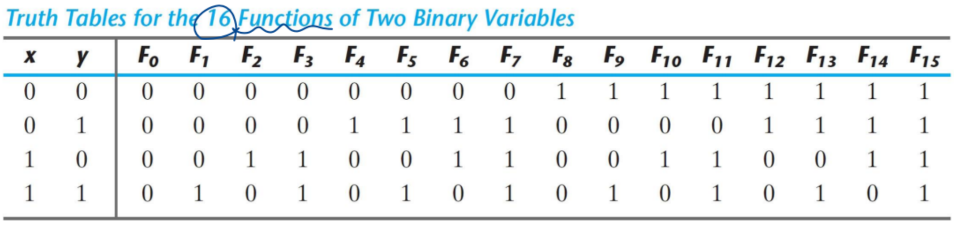
Delay through gates !

## Other Logic Operations



Boolean Expressions for the 16 Functions of Two Variables

| Boolean Functions | Operator Symbol | Name | Comments |
|---|---|---|---|
| $F_0 = 0$ | | Null | Binary constant 0 |
| $F_1 = xy$ | $x \cdot y$ | AND | $x$ and $y$ |
| $F_2 = xy'$ | $x/y$ | Inhibition | $x$, but not $y$ |
| $F_3 = x$ | | Transfer | $x$ |
| $F_4 = x'y$ | $y/x$ | Inhibition | $y$, but not $x$ |
| $F_5 = y$ | | Transfer | $y$ |
| $F_6 = xy' + x'y$ | $x \oplus y$ | Exclusive-OR | $x$ or $y$, but not both |
| $F_7 = x + y$ | $x + y$ | OR | $x$ or $y$ |
| $F_8 = (x + y)'$ | $x \downarrow y$ | NOR | Not-OR |
| $F_9 = xy + x'y'$ | $(x \oplus y)'$ | Equivalence | x equals $y$ |
| $F_{10} = y'$ | $y'$ | Complement | Not $y$ |
| $F_{11} = x + y'$ | $x \subset y$ | Implication | If $y$, then $x$ |
| $F_{12} = x'$ | $x'$ | Complement | Not $x$ |
| $F_{13} = x' + y$ | $x \supset y$ | Implication | If $x$, then $y$ |
| $F_{14} = (xy)'$ | $x \uparrow y$ | NAND | Not-AND |
| $F_{15} = 1$ | | Identity | Binary constant 1 |

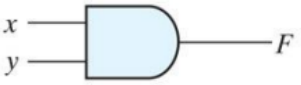
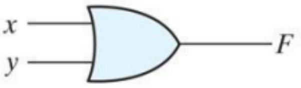
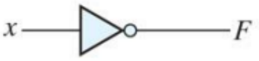
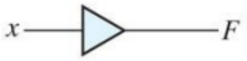
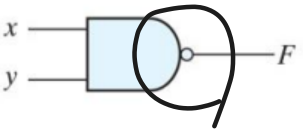
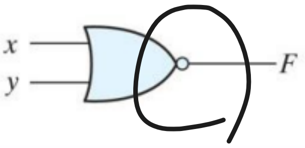
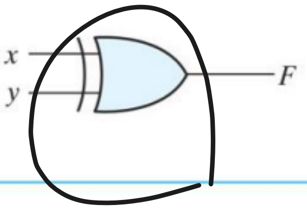
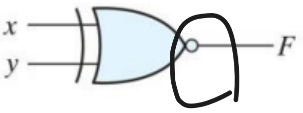


Truth Tables for the 16 Functions of Two Binary Variables

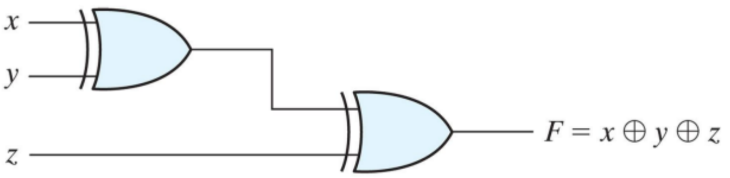
| $x$ | $y$ | $F_0$ | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ | $F_6$ | $F_7$ | $F_8$ | $F_9$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_{13}$ | $F_{14}$ | $F_{15}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

# Digital Logic Gate

# Digital Logic Gate

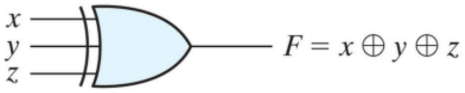
| Name | Graphic symbol | Algebraic function | Truth table | |
|---|---|---|---|---|
| AND |  | $F = x \cdot y$ | x y F<br>0 0 0<br>0 1 0<br>1 0 0<br>1 1 1 | |
| OR |  | $F = x + y$ | x y F<br>0 0 0<br>0 1 1<br>1 0 1<br>1 1 1 | |
| Inverter |  | $F = x'$ | x F<br>0 1<br>1 0 | |
| Buffer |  | $F = x$ | x F<br>0 0<br>1 1 | |
| ① | ② | ③ | ④ | |
| NAND |  | $F = (xy)'$ | x y F<br>0 0 1<br>0 1 1<br>1 0 1<br>1 1 0 | |
| NOR |  | $F = (x + y)'$ | x y F<br>0 0 1<br>0 1 0<br>1 0 0<br>1 1 0 | |
| Exclusive-OR<br>(XOR) |  | $F = xy' + x'y$<br>$= x \oplus y$ | x y F<br>0 0 0<br>0 1 1<br>1 0 1<br>1 1 0 | 다르면 1 |
| Exclusive-NOR<br>or<br>equivalence |  | $F = xy + x'y'$<br>$= (x \oplus y)'$ | x y F<br>0 0 1<br>0 1 0<br>1 0 0<br>1 1 1 | 같으면 1 |

## XOR gates ⇒ using 2 input gates



(a) Using 2-input gates

↳ 일반적으로는 2-input 사용

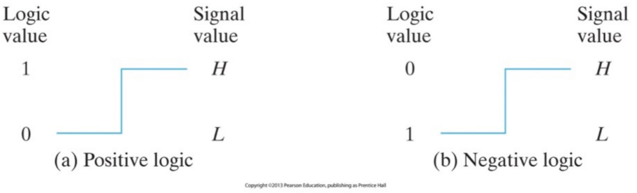


(b) 3-input gate

⇒ 짝수 1이면 0, 홀수 1이면 1

## positive and negative logic

- positive logic
  - low = 0, high = 1
- negative logic
  - low = 1, high = 0



(a) Positive logic

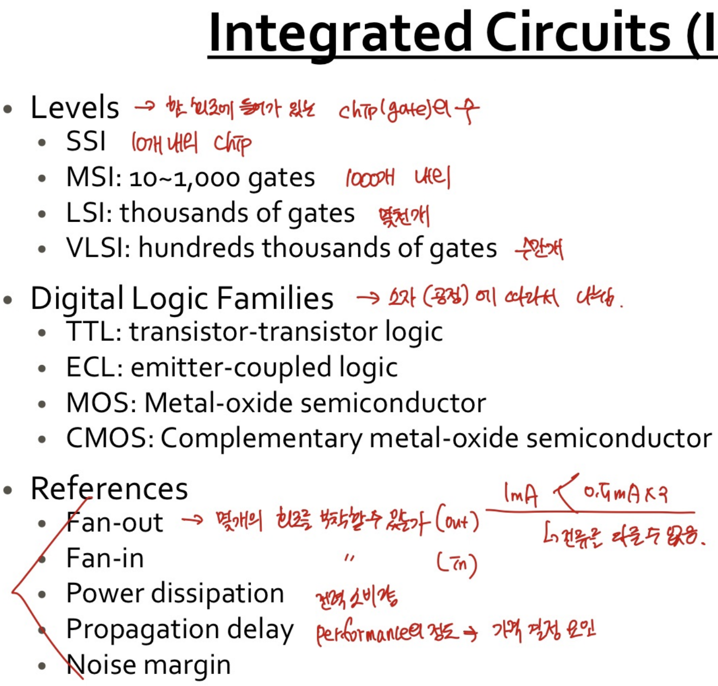
(b) Negative logic

Copyright ©2013 Pearson Education, publishing as Prentice Hall

⇒ positive logic으로 수업.

## Integrated Circuits(IC)

: 집적회로 → 어떻게 분류?



**Integrated Circuits (IC)** 집적 회로

- Levels → 한 회로에 들어가 있는 chip(gate)의 수
  - SSI 10개 내의 chip
  - MSI: 10~1,000 gates 1000개 내의
  - LSI: thousands of gates 몇천개
  - VLSI: hundreds thousands of gates 수만개
- Digital Logic Families → 소자 (공정)에 따라서 나눔.
  - TTL: transistor-transistor logic
  - ECL: emitter-coupled logic
  - MOS: Metal-oxide semiconductor
  - CMOS: Complementary metal-oxide semiconductor
- References
  - Fan-out → 몇개의 회로를 부착할 수 있는가 (out)
  - Fan-in " (in)
  - Power dissipation 전력 소비량
  - Propagation delay performance의 정도 → 가격 결정 요인
  - Noise margin

1mA < 0.5mA×?

↳ 전류로 따질 수 있음.