EECS1010 Logic Design
Lecture 2
Boolean Algebra and Logic Gates

Jenny Yi-Chun Liu
jennyliu@gapp.nthu.edu.tw

Outline

- Digital systems and information
- Boolean algebra and logic gates
- Gate-level minimization
- Combinational logic
- Sequential circuits
- Registers and counters
- Memory

Chapter Outline

the most primitive logic element

- Boolean algebra and logic gates
  - Binary logic
  - Basic theorems and properties of Boolean algebra
  - Normal and standard forms
  - Other logic functions

Binary Logic

Digital Circuits

- Digital circuits: hardware components that manipulate binary information.
- Each basic circuit is called a logic gate
- Each gate performs a specific logic function.

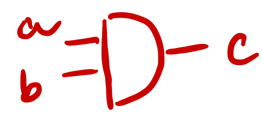
Binary Logic

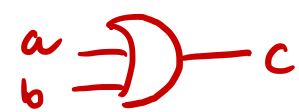
- Binary logic deals with 0's and 1's.

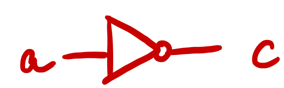
equation
Boolean expression

logic diagram

\#
- Basic logic functions:

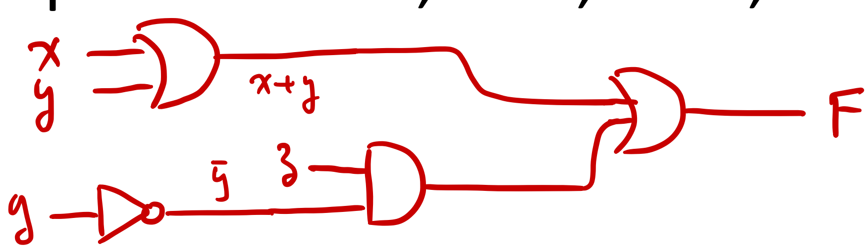
1. AND $\cdot$ $a \cdot b = c$


2. OR $+$ $a + b = c$


3. NOT $'$, $\overline{}$ a', $\bar{a} = c$


\#
- Operation order: parentheses, NOT, AND, OR

$F = (x+y) + z \cdot \bar{y}$



Operator Definitions

- Binary logic operation vs. binary arithmetic
- Definitions of AND, OR, and NOT:

AND	OR	NOT
$0 \cdot 0 = 0$	$0 + 0 = 0$	$\overline{0} = 1$
$0 \cdot 1 = 0$	$0 + 1 = 1$	$\overline{1} = 0$
$1 \cdot 0 = 0$	$1 + 0 = 1$	
$1 \cdot 1 = 1$	$1 + 1 = 1$	

Truth Table

- Truth table: a tabular listing of the values of a function for all possible combinations of values on its arguments
- Example: truth tables for the basic logic operations

AND

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

OR

x	y	$x + y$
0	0	0
0	1	1
1	0	1
1	1	1

NOT

x	$\bar{x}$
0	1
1	0

Example: Truth Table

Practice

- $F = (x+y) + z \cdot \bar{y}$. Truth table = ?

Boolean algebra

Logic Diagrams and Expressions

- Boolean algebra: algebra dealing with binary variables and logic operations.
- Boolean equations, truth tables and logic diagrams describe the same function!
- Truth tables are unique; expressions and logic diagrams are not. This gives flexibility in implementing functions.

Axioms of Boolean Algebra

- Axioms: a set of mathematical statements that we assert to be true.

- Identity $1 \cdot x = x$

  $0 + x = x$

- Annihilation $0 \cdot x = 0$

  $1 + x = 1$

- Negation $\bar{0} = 1$

  $\bar{1} = 0$

Basic Identities of Boolean Algebra (1/2)

1. Communicative $x \cdot y = y \cdot x$

   $x + y = y + x$

2. Associative, $x \cdot (y \cdot z) = (x \cdot y) \cdot z$

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

3. Distributive $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$

   $x + (y \cdot z) = (x + y) \cdot (x + z)$

4. Idempotence $x \cdot x = x$

   $x + x = x$

5. Absorption $x \cdot (x + y) = x$

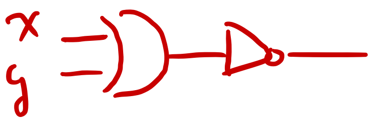
   $x + (x \cdot y) = x$

Basic Identities of Boolean Algebra (2/2)

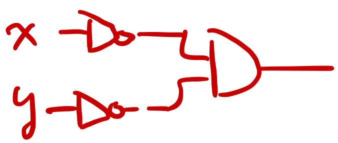
6. Combining, $(x\cdot y)+(x\cdot \bar{y})=x$

$$(x+y)\cdot(x+\bar{y})=x$$

7. DeMorgan's $(\overline{x\cdot y})=\bar{x}+\bar{y}$



$$(\overline{x+y})=\bar{x}\cdot\bar{y}$$



8. Complementation, $x+\bar{x}=1$

$$x\cdot\bar{x}=0$$

9. Involution, $\bar{\bar{x}}=x$

Duality

- Dual: interchanging AND and OR, 0's and 1's.

usually F^D (dual of F) $\neq F$

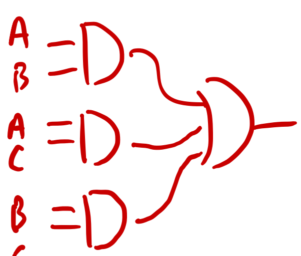
- Example:

1) $F = (A+\bar{C})\cdot B + 0 \Rightarrow F^D = \left[(A\cdot\bar{C}) + B\right]\cdot 1 = A\cdot\bar{C} + B$

$\neq F$

2) $G = \underbrace{AB}_{A\cdot B} + AC + BC$, $\Rightarrow G^D = G$

Practice



Summary of Useful Theorems

- Boolean algebra is applied to reduce an expression to obtain a simpler circuit.

1. Minimization = combining (p.14)

2. Consensus: $xy + \bar{x}z + yz = xy + \bar{x}z$

$$(x+y)(\bar{x}+z)(y+z) = (x+y)(\bar{x}+z)$$

3. Simplification: $x + \bar{x}y = x + y$

$$x(\bar{x}+y) = xy$$

Example: Boolean Function

- Truth table

$F_1 = x y \bar{z}$ $\quad x=1,\ y=1,\ \bar{z}=1,\ z=0$

$F_2 = \boxed{x} + \boxed{\bar{y} z}$ $\quad x=1,\ \bar{y}z=1,\ \bar{y}=1,\ z=1$

$F_3 = \bar{x}\bar{y}\bar{z} + \bar{x}yz + x\bar{y}$ $\quad x=1,\ y=0$

$F_4 = x\bar{y} + \bar{x}z$ $\quad \begin{cases} x=1 \\ y=0 \end{cases} \quad x=0,\ z=1$

$\bar{F_3}$

x	y	z	F1	F2	F3	F4
0	0	0	0	0	1	0
0	0	1	0	1	0	1
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	0

Practice. $\bar{F_3} = ?$, $\bar{F_4} = ?$

$\bar{F_1} = \overline{(x y \bar{z})} = \bar{x} + \bar{y} + z$

NOT

Complementing Functions

- Complement: interchanging 0's and 1's in the truth table
- General rules: use DeMorgan's theorem to complement a function
  - Interchange AND and OR
  - Complement each literal ($x \to \bar{x}$, $\bar{y} \to y$) and constant ($0 \leftrightarrow 1$)

∘ Examples:

1) $F = \bar{x}y\bar{z} + x\bar{y}\bar{z}$, $\quad \bar{F} = \overline{(\bar{x}y\bar{z} + x\bar{y}\bar{z})} = (x+\bar{y}+z)\cdot(\bar{x}+y+z)$

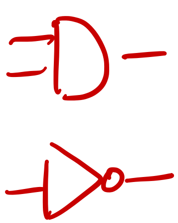
2) $G = (\bar{x}+yz)\cdot\bar{a}+b$, $\quad \bar{G} = [(x\cdot(\bar{y}+\bar{z}))+a]\cdot\bar{b}$

- Homework1: average 90.5
- Quiz 1: solutions posted
- Midterm1: 3/28 (Tue) 1:20pm. classroom to be announced. Lec1-Lec2.
- Homework2: due 3/25 (Sat) 6pm, online

Normal and standard forms

Boolean Function

- Can be represented by a truth table with 2^n rows. n is the number of variables in the function.
- There are many algebraic expressions to specify a given Boolean function. It is important to find the simplest one. $F(x,y,z) = xy + z$
- Any Boolean function can be transformed into a logic diagram with only AND, OR, and NOT gates.



Normal Forms

- It is useful to specify Boolean functions in a standard way.
- Common normal forms

$+$

1. Sum of minterms (SOM)

2. Product of maxterms (POM)

$\cdot$

Minterms

AND

- Minterm: a product term in which all of the variables appear exactly once, either complemented or uncomplemented.
- Minterm represents exactly one combination of variables in the truth table.

- 2^n minterms for n variables.
- Example: 2 variables, x and y.

Truth table

x	y	minterm	symbol
0	0	$\bar{x}\bar{y}$	m_0
0	1	$\bar{x}y$	m_1
1	0	$x\bar{y}$	m_2
1	1	xy	m_3

Maxterms

- Maxterm: a sum (OR) term in which all of the variables appear exactly once, either complemented or uncomplemented.
- 2^n maxterms for n variables.

Example: 2 variables, x and y

Truth table

x	y	maxterm	symbol
0	0	$x + y$	M_0
0	1	$x + \bar{y}$	M_1
1	0	$\bar{x} + y$	M_2
1	1	$\bar{x} + \bar{y}$	M_3

Standard Minterms and Maxterms

m_0 M_1

- Minterms and maxterms are designated with a subscript, corresponding to a binary pattern
- All variables are presented in a minterm or maxterm and will be listed in the same order (usually alphabetically)
- Example: For variables a, b, c:
  - Maxterm $a+\bar{b}+c$

    $b+a+c$ X X $ab+c$ X $b+$
  - Minterm $a\bar{b}c$

    x bac x ba

Minterm and Maxterm Relationship

- Each maxterm is the complement of its corresponding minterm and vice versa.
- Use DeMorgan's Theorem

$m_2 = x\bar{y}$

$M_2 = \bar{x} + y$

$\bar{m}_2 = (\overline{x\bar{y}}) = \bar{x} + y = M_2$

- For n variables, $m_i = \overline{M_i}$, $i = 0, 1, \cdots 2^n - 1$

Minterms and Maxterms with Three Variables

	x y z	minterms	notation	maxterms	notation
0	0 0 0	$x'y'z'$	m_0	$x+y+z$	M_0
1	0 0 1	$x'y'z$	m_1	$x+y+z'$	M_1
2	0 1 0	$x'yz'$	m_2	$x+y'+z$	M_2
3	0 1 1	$x'yz$	m_3	$x+y'+z'$	M_3
4	1 0 0	$xy'z'$	m_4	$x'+y+z$	M_4
5	1 0 1	$xy'z$	m_5	$x'+y+z'$	M_5
6	1 1 0	xyz'	m_6	$x'+y'+z$	M_6
7	1 1 1	xyz	m_7	$x'+y'+z'$	M_7

Sum of Minterms (SOM)

- A Boolean function can be represented algebraically from a given truth table by forming the logical sum of all the minterms that produce 1 in the function.

SOM Example 1

- Given a truth table.

  Find som .

x	y	z	$F(x,y,z)$	minterm
0	0	0	1	m_0
0	0	1	0	m_1
0	1	0	1	m_2
0	1	1	0	m_3
1	0	0	0	m_4
1	0	1	1	m_5
1	1	0	0	m_6
1	1	1	1	m_7

$$F(x,y,z) = m_0 + m_2 + m_5 + m_7$$

$$= \bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}z + xyz$$

$$= \sum m(0,2,5,7)$$

$$= \sum (0,2,5,7)$$

SOM Example 2

- Given $F(x,y,z) = m_1 + m_4 + m_7$.
  Find truth table.

	x y z	$F(x,y,z)$	$\bar{F}$
m_0	0 0 0	0	1
m_1	0 0 1	1	0
m_2	0 1 0	0	1
m_3	0 1 1	0	1
m_4	1 0 0	1	0
m_5	1 0 1	0	1
m_6	1 1 0	0	1
m_7	1 1 1	1	0

$\bar{F}(x,y,z) = \Sigma(0,2,3,5,6)$

$F + \bar{F} = 1$

SOM Example 3

Practice

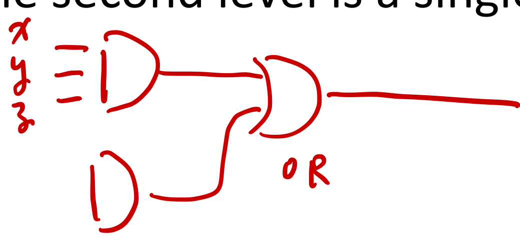
Given $F(A,B,C,D,E) = m_2 + m_9 + m_{17} + m_{23}$

1) Derive its truth table

2) Find the SOM Boolean expression (with literals)

Properties of SOM

- There are 2^n minterms for n Boolean variables.
- Any Boolean function can be expressed as a logical sum of minterms.
- The complement of a function contains those minterms not included in the original function.
- A function that include all the 2^n minterms is equal to 1.
- A function not in the sum-of-minterms form can be converted to that form by a truth table. $F(x,y,z) = x + \bar{y}z$
- Implementation of this form is a two-level network of gates such that: $xyz + x\bar{y}\bar{z}$
  - The first level consists of *n*-input AND gates
  - The second level is a single OR gate (with fewer than 2^n inputs)



Product of Maxterms (POM)

- A Boolean function can be represented algebraically from a given truth table by forming the logical product of all the maxterms that produce 0 in the function =>

Practice

- Example.

Given $F(x,y,z) = x + \bar{x}\bar{y}$. Find its POM expression

1) By truth table

2) By Boolean algebra

POM Example 1

- Given truth table. Find its POM.

x	y	z	$F(x,y,z)$	maxterm
0	0	0	0	M_0
0	0	1	1	M_1
0	1	0	0	M_2
0	1	1	0	M_3
1	0	0	1	M_4
1	0	1	0	M_5
1	1	0	0	M_6
1	1	1	1	M_7

$$F(x,y,z) = M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6$$

$$= (x+y+z)\cdot(x+\bar{y}+z)\cdot(x+\bar{y}+\bar{z})\cdot(\bar{x}+y+\bar{z})\cdot(\bar{x}+\bar{y}+z)$$

$$= \Pi M(0,2,3,5,6)$$

$$= \Pi(0,2,3,5,6)$$

$$\bar{F}(x,y,z) = \Pi(1,4,7) \quad \text{POM}$$

SOM $F(x,y,z) = \Sigma(1,4,7)$

POM Example 2

$x=1$
$z=0$

- $F(x,y,z) = x\bar{z} + yz + \bar{x}\bar{y}$. Find its POM.

① By truth table

x	y	z	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

M_2

M_5

$F = M_2 \cdot M_5$

② By Boolean algebra

SOM, $F = \Sigma(0,1,3,4,6,7)$

Use Boolean algebra

(p.11)

SOM.

$x \cdot 1 = x$

- $F(x,y,z) = x\bar{z} + yz + \bar{x}\bar{y}$.

$(y+\bar{y})$

$= x\bar{z}\cdot(y+\bar{y}) + yz(x+\bar{x}) + \bar{x}\bar{y}\cdot(z+\bar{z})$

$= xy\bar{z} + x\bar{y}\bar{z} + xyz + \bar{x}yz + \bar{x}\bar{y}z + \bar{x}\bar{y}\bar{z}$

Practice

distributive

POM. $F(x,y,z) = x\bar{z} + yz = (x\bar{z}+y)(x\bar{z}+z)$

$= (x+y)\cdot(\bar{z}+y)\cdot(x+z)\cdot(\bar{z}+z)$

$= (x+y)\cdot(y+\bar{z})\cdot(x+z)$

$x+0=x$

$= (x+y+z\cdot\bar{z})\cdot(y+\bar{z}+x\bar{x})\cdot(x+z+y\bar{y})$

$= (x+y+z)\cdot(x+y+\bar{z})\cdot(y+\bar{z}+x)(y+\bar{z}+\bar{x})\cdot(x+z+y)$

$(x+\bar{y}+z)$

$= (x+y+z)\cdot(x+y+\bar{z})\cdot(\bar{x}+y+\bar{z})\cdot(x+\bar{y}+z)$

POM Example 3

Practice

- $F(A,B,C,D,E) = \Pi M(3, 8, 11, 14)$

Find its truth table and SOM.

SOM and POM Conversion

- Any Boolean function can be represented by either SOM or POM normal forms.
  - To convert from one normal form to another, interchange Σ and π, and list the index that are excluded from the original form.

∘ Example.

$$F(x,y,z) = \underset{\text{SOM}}{\Sigma(1,4,7)} = \underset{\text{POM}}{\pi(0,2,3,5,6)}$$

$$\bar{F}(x,y,z) = \underset{\text{SOM}}{\Sigma(0,2,3,5,6)} = \underset{\text{POM}}{\pi(1,4,7)}$$

Conversion to Normal Form

- Convert any Boolean expression into normal form
  - By truth table
  - By expanding the missing variables in each term using x + x' = 1, xx' = 0

$xy(z+\bar{z}) = xyz + xy\bar{z}$

$(x+y) = (x+y+z\cdot\bar{z}) = (x+y+z)\cdot(x+y+\bar{z})$

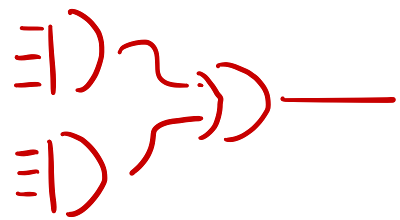
Complement of Function

- The complement of a function expressed as a sum of minterms is constructed by selecting the minterms missing in the sum-of-minterms normal forms.
- Alternatively, the complement of a function expressed by a Sum of Minterms form is simply the Product of Maxterms with the same indices.

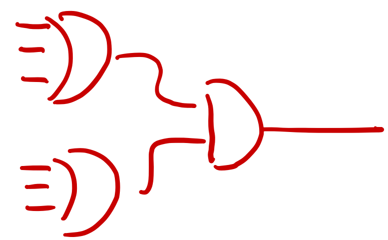
Standard Forms

2 - level

1. • Sum-of-products (SOP)
   - Has fewer literals than SOM



2. • Product-of-sums (POS)
   - Has fewer literals than POM



- The corresponding circuit may be simpler than that of the normal form.
- Standard forms are not unique.

$(x+y)\cdot(\bar{y}+z)$

POS

Standard Sum-of-Products (SOP)

- Standard Sum-of-Products (SOP) form: equations are written as an OR of AND (product) terms.

SOM implementation

- Example:

SOM

$F(x,y,z) = \Sigma m(1,4,5,6,7)$

$= \bar{x}\bar{y}z + x\bar{y}\bar{z} + xy\bar{z} + x\bar{y}z + xyz$

$= \bar{x}\bar{y}z + x(\bar{y}\bar{z} + y\bar{z} + \bar{y}z + yz)$

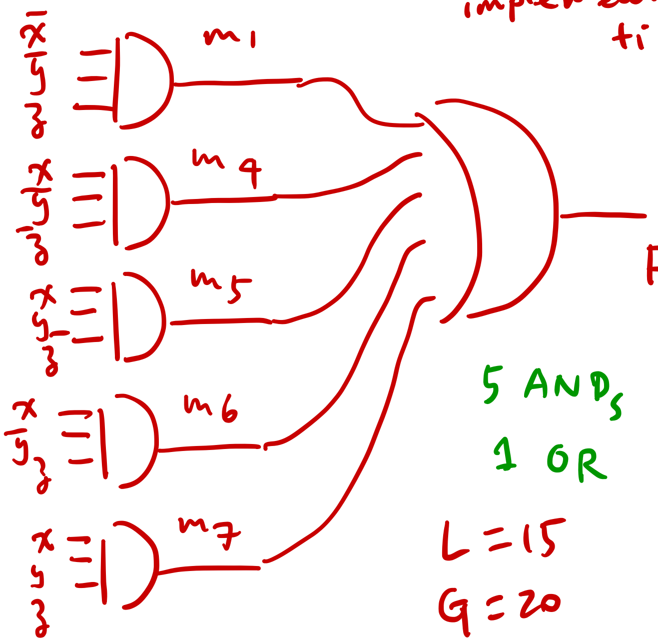
L = 8

G = 12

$= \bar{x}\bar{y}z + x(y+\bar{y})(z+\bar{z})$

$= \bar{x}\bar{y}z + x$

$= \bar{y}z + x$



5 ANDs

1 OR

L = 15

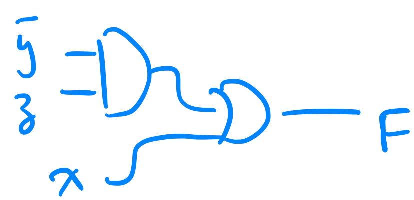
G = 20

GN = 23

SOP

L = 3

G = 4



1 AND

1 OR

Optimization

Circuit Optimization

- The complexity of a logic circuit directly related to the algebraic expression (literal count).
- Goal: to obtain the simplest implementation for a given function.
- Optimization is a formal approach to simplification that is performed using a specific procedure or algorithm.
- Simplest form is not necessarily unique.
- Optimization requires a cost criterion to measure the simplicity of a circuit.

Computer aided design
- CAD tools are commonly used today for optimization.
- Cost criteria we will use:

1. Literal cost (L)

* 2. Gate input cost (G)

* 3. Gate input cost with NOTs (GN)

Literal Cost

- Literal: a variable or its complement
- Literal cost: the number of the literals appeared in a Boolean expression.
  - Simple to count.
  - Not accurate in some cases.

Example.

$F = ABC + AC\bar{D} + B, \quad L = 7 \quad G = 9 \quad GN = 9 + 1 = 10$

$G = (A+B)(C+D)(\bar{B}+C+\bar{D}), \quad L = 7 \quad G = 10, \quad GN = 10 + 2 = 12$

Gate Input Cost

- Gate input costs: the number of the inputs to the gates corresponding to a given Boolean expression.
  - Obtain from the logic diagram.
- For SOP and POS equations, it can be found from the equations by finding the sum of:
  - All literal appearances = L
  - The number of terms excluding single literal terms
  - The number of distinct complemented single literals (GN)

  = G

  GN
- Gate input cost is a good measure of logic implementation.
  - It is proportional to the number of transistors and wires in implementation.
  - It considers the internal inputs, particular important for circuits more than two levels.

Cost Criteria Example 1

Practice

- $F = AB' + BC + B'C'$

L =

G =

GN =

- $G = (AC' + B)(B + C + D')(B' + C' + D)$

L =

G =

GN =

Cost Criteria Example 2

- F = ABC + A’B’C’, G = (A + C’)(B’ + C)(A’ + B)

SOP, SOM

POS

L = 6

G = 6 + 2 = 8

GN = 8 + 3 = 11

L = 6

G = 6 + 3 = 9

GN = 9 + 3 = 12

F = G same function, same L.

Function F has better G and GN than function G.

Other gates

Other Gate Types

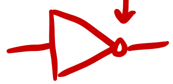
- Why?
  - Implementation feasibility and low cost

NAND
NOR
easier to fabricate than AND, OR

- Gate classifications
  - Primitive gate - a gate that can be described using a single primitive operation type (AND or OR) plus inversions. AND, OR, NOT, NAND, NOR, buffer
  - Complex gate - a gate that requires more than one primitive operation type for its description.

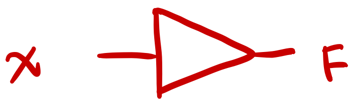
XOR, XNOR, AOI, OAI, AO, OA

NOT (complement)



Buffer

- A buffer is a gate with the function: F = x



- In terms of Boolean function, a buffer is the same as a connection!
- So why use it?
  - A buffer is an electronic amplifier used to improve circuit voltage levels and increase the speed of circuit operation.

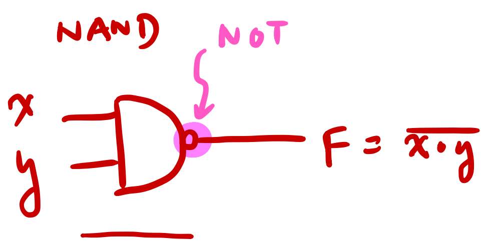
NAND Gate

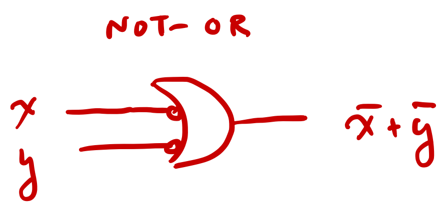
- Digital circuits are more common to implement with NAND/NOR gates rather than AND/OR/NOT gates due to simple fabrication.
- NAND is an universal gate that any operation can be implemented by NAND gates.

- NAND : AND + NOT = NOT + OR

x	y	$\overline{x\cdot y} = \bar{x} + \bar{y}$
0	0	1
0	1	1
1	0	1
1	1	0

Symbol



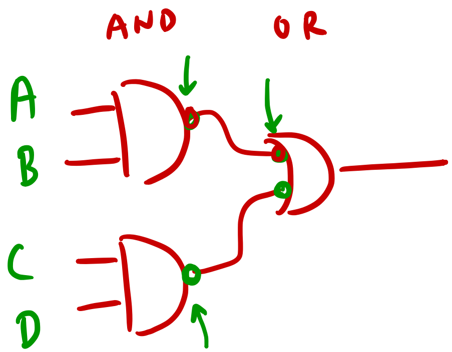


NAND-NAND Implementation

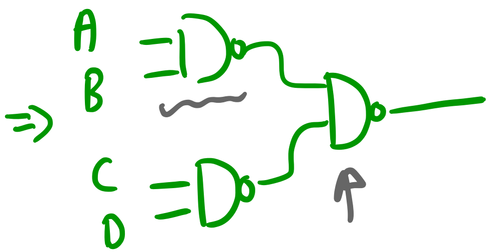
- NAND-NAND implementation procedure:
  - Simplify the function in the sum-of-products form. (AND - OR)
  - Transfer it to a 2-level NAND-NAND expression by DeMorgan's Theorem.
  - Draw the corresponding NAND-NAND implementation. An 1-input NAND gate can be replaced by an inverter.

NAND-NAND Example

- Given $F(A,B,C,D) = \overset{\text{SOP}}{AB + CD}$. use NAND



NAND-NAND



AND-OR (SOP) $\longrightarrow$ NAND-NAND

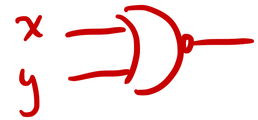
$$F(A,B,C,D) = AB + CD = \left(\overline{\overline{AB+CD}}\right) = \left(\overline{\overline{AB} \cdot \overline{CD}}\right)$$

NOR Gate

- NOR : OR – NOT = NOT – AND

x	y	$\overline{x+y} = \bar{x} \cdot \bar{y}$
0	0	1
0	1	0
1	0	0
1	1	0

Symbol



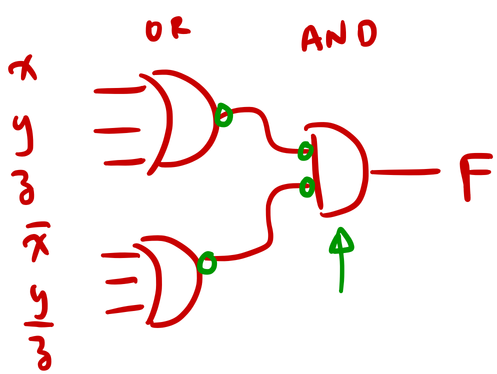
$F = \overline{x+y}$

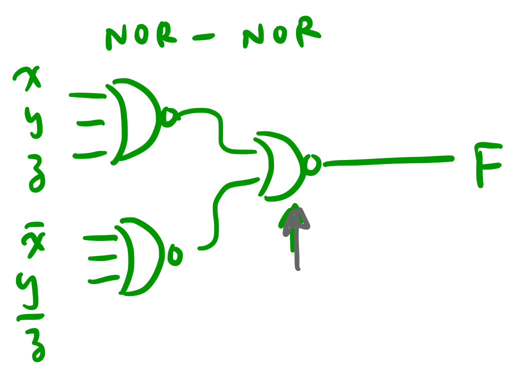
- POS (OR – AND) → NOR – NOR

NOR-NOR

- NOR-NOR is the dual of the NAND-NAND implementation
  - AND-OR => NAND-NAND
  - OR-AND => NOR-NOR

- Example: $F(x,y,z) = (x+y+z)(\bar{x}+y+\bar{z})$. use NOR



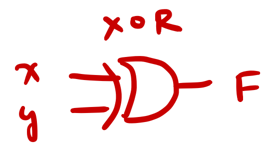


$$F = \overline{\overline{(x+y+z)(\bar{x}+y+\bar{z})}} = \overline{\underbrace{\overline{(x+y+z)}}_{} + \underbrace{\overline{(\bar{x}+y+\bar{z})}}_{}}$$

NOR - NOR

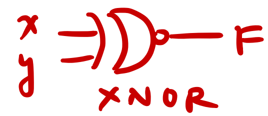
exclusive

XOR/XNOR (1/3)



x	y	$x \oplus y$ (XOR)	$\overline{x \oplus y}$ (XNOR)
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

- Definitions
  - The XOR function: $x \oplus y = x\bar{y} + \bar{x}y$
  - The eXclusive NOR (XNOR): $\overline{x \oplus y} = xy + \bar{x}\bar{y}$



- The eXclusive OR (XOR) function may be:
  - Implemented directly as an electronic circuit.
  - Implemented by interconnecting other gate types.
- The eXclusive NOR (XNOR) function is the complement of the XOR function.

XOR/XNOR (2/3)

- Uses for the XOR and XNOR gates:
  - Adders, subtractors, multipliers
  - Counters, incrementers, decrementers
  - Parity checkers, parity generators

- XOR and XNOR do not exist for more than two inputs. Instead, they are replaced by odd and even functions for more than two inputs.

x	y	z	$x \oplus y \oplus z$ (odd function)
1	0	0	1
0	1	0	1
0	0	1	1
1	1	1	1

奇數個"1"

XOR/XNOR (3/3)

- XOR identities:

$$x \oplus 0 = x$$

$$x \oplus 1 = \overline{x}$$

$$x \oplus x = 0$$

$$x \oplus \overline{x} = 1$$

$$x \oplus y = y \oplus x$$

$$x \oplus y \oplus z = (x \oplus y) \oplus z = x \oplus (y \oplus z)$$

Parity Generators and Checkers (1/2)

- A parity bit is an extra bit added to n-bit code to produce and produce an (n+1) bit code for error detection and correction.
- Example:

$n=3$, generate a parity bit for even parity codewords.
(4 bits)

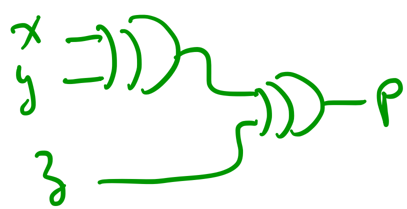
x	y	z	p
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$\left(\begin{array}{cccc} p & x & y & z \\ 1 & 0 & 0 & 1 \end{array}\right)$

SOM, $p = \Sigma(1,2,4,7)$
$= \Pi(0,3,5,6)$

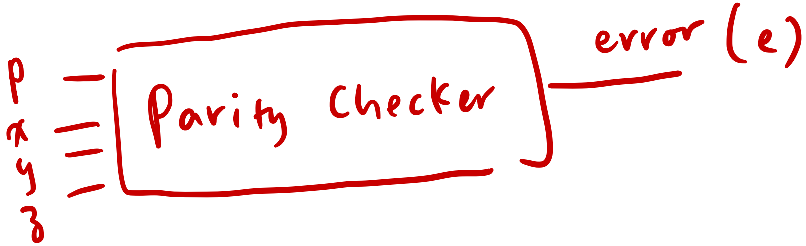
$x, y, z \rightarrow \boxed{\text{Generator}} \rightarrow p$

$p = x \oplus y \oplus z$



Parity generator.

Parity Generators and Checkers (2/2)

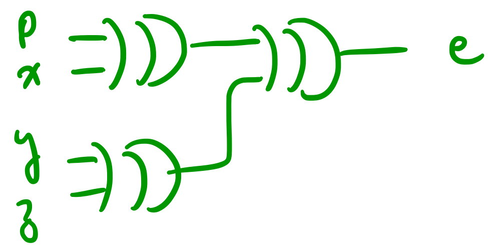


e=0 no error (even "1"s)

e=1 error (odd "1"s)

output

p	x	y	z	e
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0



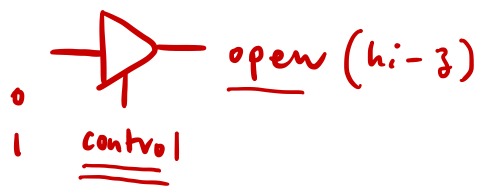
阻抗

High-Impedance Outputs (1/2)

- Logic gates introduced thus far
  - have 1 and 0 output values.
  - cannot have their outputs connected together.
  - transmit signals on connections in only one direction.
- Three-state logic adds a third logic value, High-Impedance (Hi-Z), giving three states: 0, 1, and Hi-Z on the outputs.

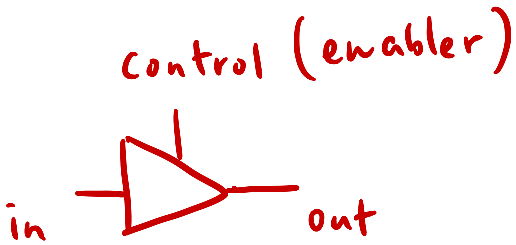
Hi-Impedance Outputs (2/2)

- What is a Hi-Z value?
  - The Hi-Z value behaves as an open circuit
  - Looking into the circuit, the output appears to be disconnected
- Hi-Z may appear on the output of any gate, but we restrict gates to:
  - ✓ A 3-state buffer
  - ✓ A transmission gate



Each of which has one data input and one control input.

The 3-State Buffer



Control = 0, out = high-Z regardless of input

Control = 1, out = in

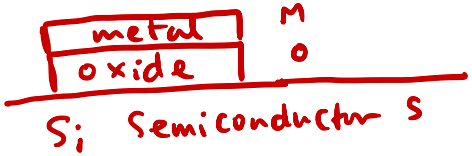
不考

Gate implementation

MOSFET: metal-oxide-semiconductor field effect transistor

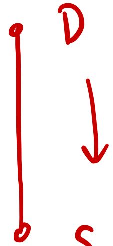
MOS Switches

cross section



G=0

G=1

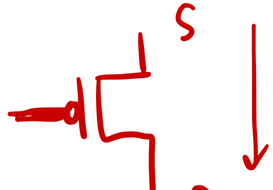


NMOS

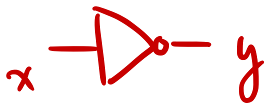
G=0

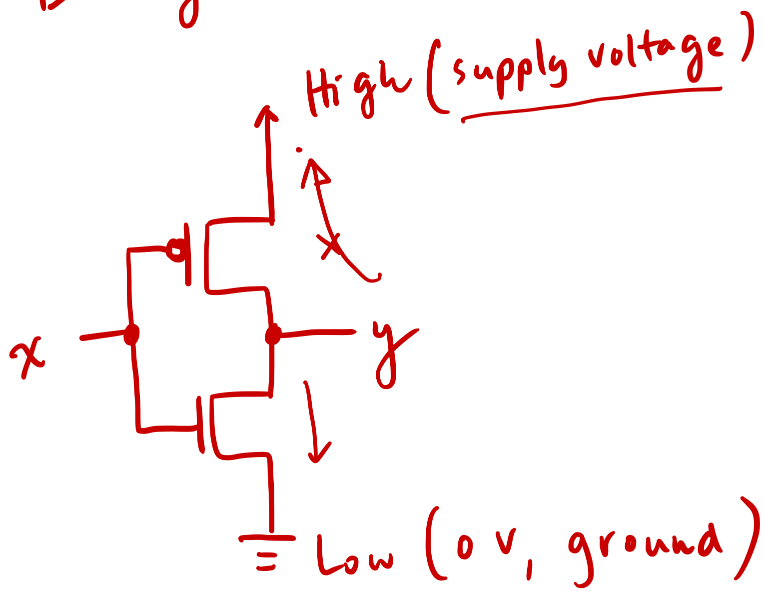
G=1

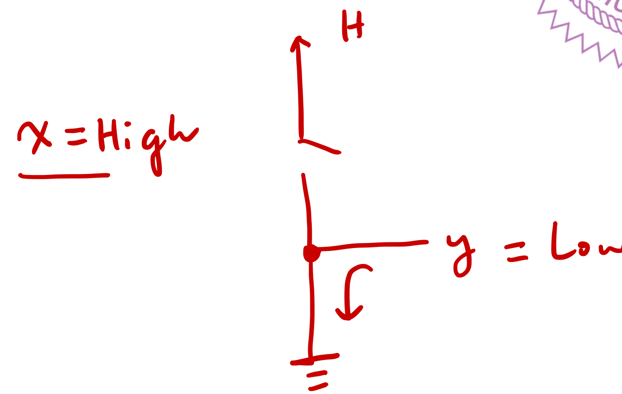
PMOS

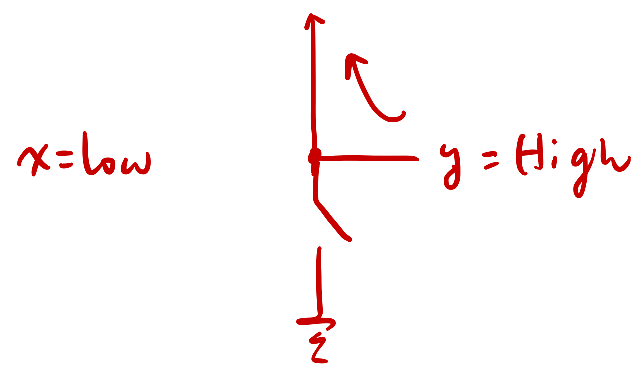


NOT Gate Implementation



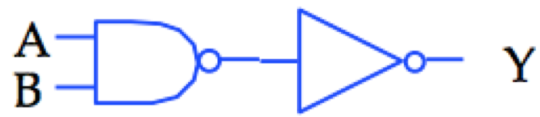






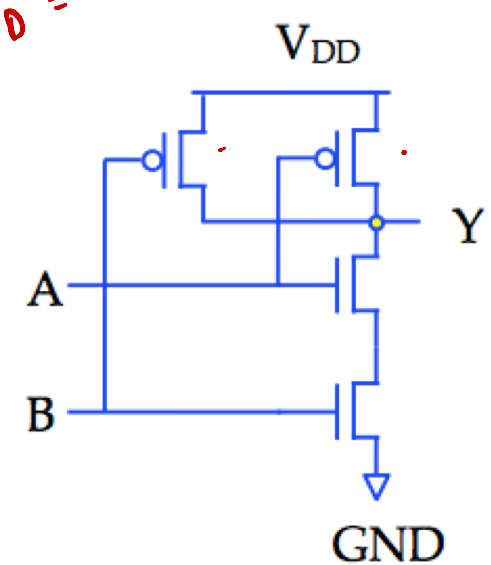
NAND and AND Implementations

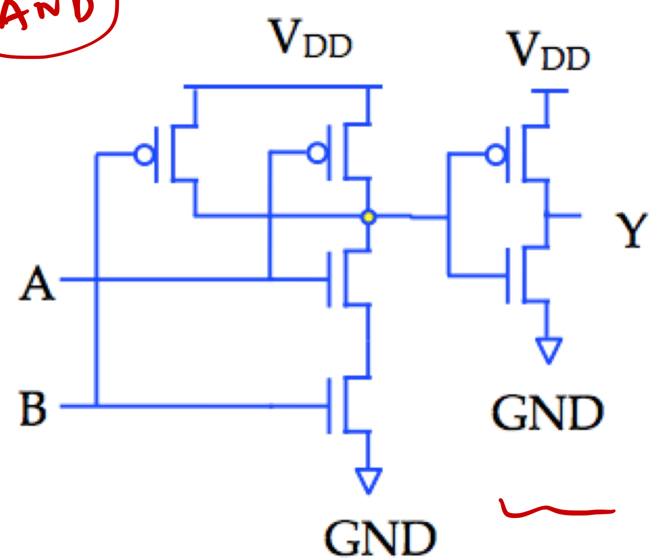
NAND - NOT



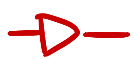
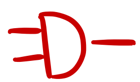
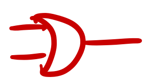
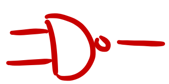
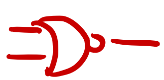
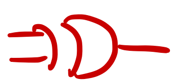
NAND = AND-NOT

AND





Basic Gate Implementation

Type	Symbol	Function	# of transistors	Gate delay (ns)
NOT		$F = \bar{x}$	2	1
Buffer		$F = x$	4	2
AND		$F = x \cdot y$	6	2.4
OR		$F = x + y$	6	2.4
NAND		$F = \overline{x \cdot y}$	4	1.4
NOR		$F = \overline{(x+y)}$	4	1.4
XOR		$F = x \oplus y$	14	4.2
XNOR		$F = \overline{x \oplus y}$	12	3.2

[Gajski]