Lecture 2 – Boolean Algebra and Logic Gates

Lan-Da Van (范倫達), *Ph. D.*
Department of Computer Science
National Chiao Tung University
Taiwan, R.O.C.
Fall, 2007

Source: Charles Kime & Thomas Kaminski

Outline

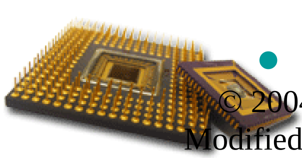
- **Binary Logic and Gates**
- **Boolean Algebra**
- **Standard Forms**

Binary Logic and Gates

- Binary variables take on one of two values.
- Basic logical operators are the logic functions AND, OR and NOT.
- Logic gates implement logic functions, where each logic gate represents a basic circuit of integrated circuits that is implemented using transistors and interconnection in complex semiconductor devices.
- Boolean Algebra: a useful mathematical system for specifying and transforming logic functions.
- We study Boolean algebra as foundation for designing and analyzing digital systems!

Binary Variables

- Recall that the two binary values have different names:
  - True/False
  - On/Off
  - Yes/No
  - 1/0
- We use 1 and 0 to denote the two values.
- Variable identifier examples:
  - A, B, y, z, or X_1 for now
  - RESET, START_IT, or ADD1 later



Logical Operations

- The three basic logical operations are:
  - AND
  - OR
  - NOT
- AND is denoted by a dot (·).
- OR is denoted by a plus (+).
- NOT is denoted by an overbar (¯), a single quote mark (') after, or (~) before the variable.

Notation Examples

- Examples:
  - $Y = A \cdot B$ is read "Y is equal to A AND B."
  - $z = x + y$ is read "z is equal to x OR y."
  - $X = \overline{A}$ is read "X is equal to NOT A."
- Note: The statement:

  1 + 1 = 2 (read "one plus one equals two")

  is not the same as

  1 + 1 = 1 (read "1 or 1 equals 1").

Operator Definitions

- Operations are defined on the values "0" and "1" for each operator:

AND

$$0 \cdot 0 = 0$$
$$0 \cdot 1 = 0$$
$$1 \cdot 0 = 0$$
$$1 \cdot 1 = 1$$

OR

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 1$$

NOT

$$\overline{0} = 1$$
$$\overline{1} = 0$$

Truth Tables

- *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	$Z = X \cdot Y$
0	0	0
0	1	0
1	0	0
1	1	1

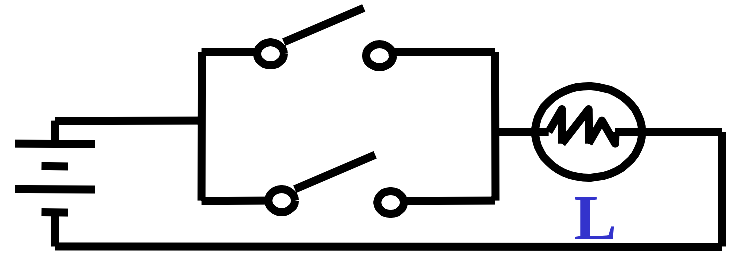
OR		
X	Y	$Z = X + Y$
0	0	0
0	1	1
1	0	1
1	1	1

NOT	
X	$Z = \overline{X}$
0	1
1	0

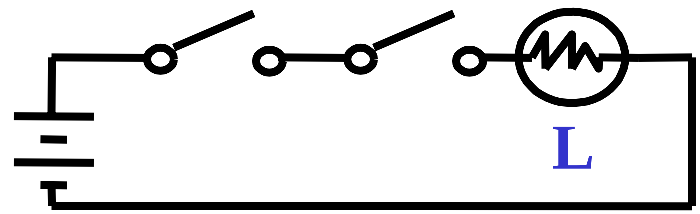
Logic Function Implementation (1/3)

- Using Switches
  - For inputs:
    - logic 1 is switch closed
    - logic 0 is switch open
  - For outputs:
    - logic 1 is light on
    - logic 0 is light off.
  - NOT uses a switch such that:
    - logic 1 is switch open
    - logic 0 is switch closed

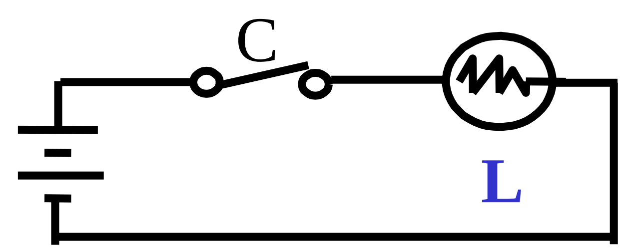
Switches in parallel => OR



Switches in series => AND

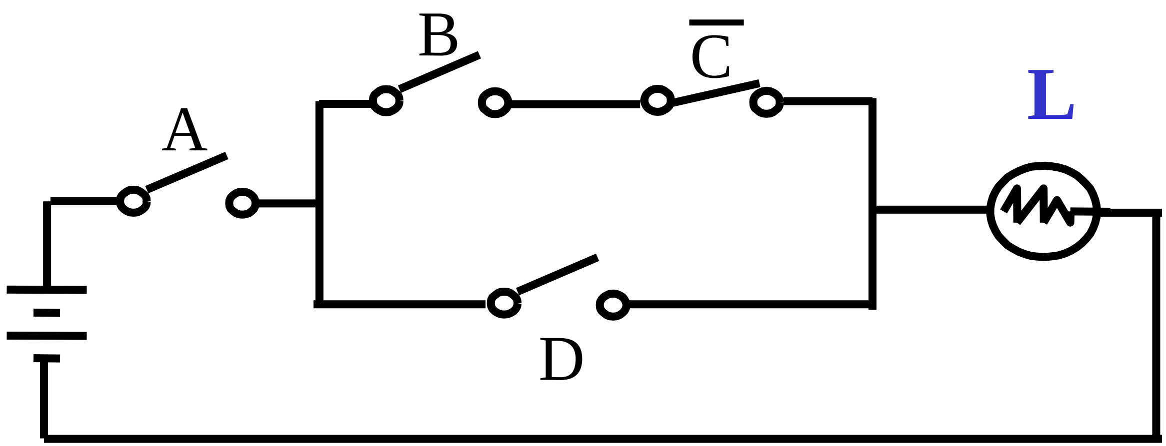


Normally-closed switch => NOT



Logic Function Implementation (2/3)

- Example: Logic Using Switches

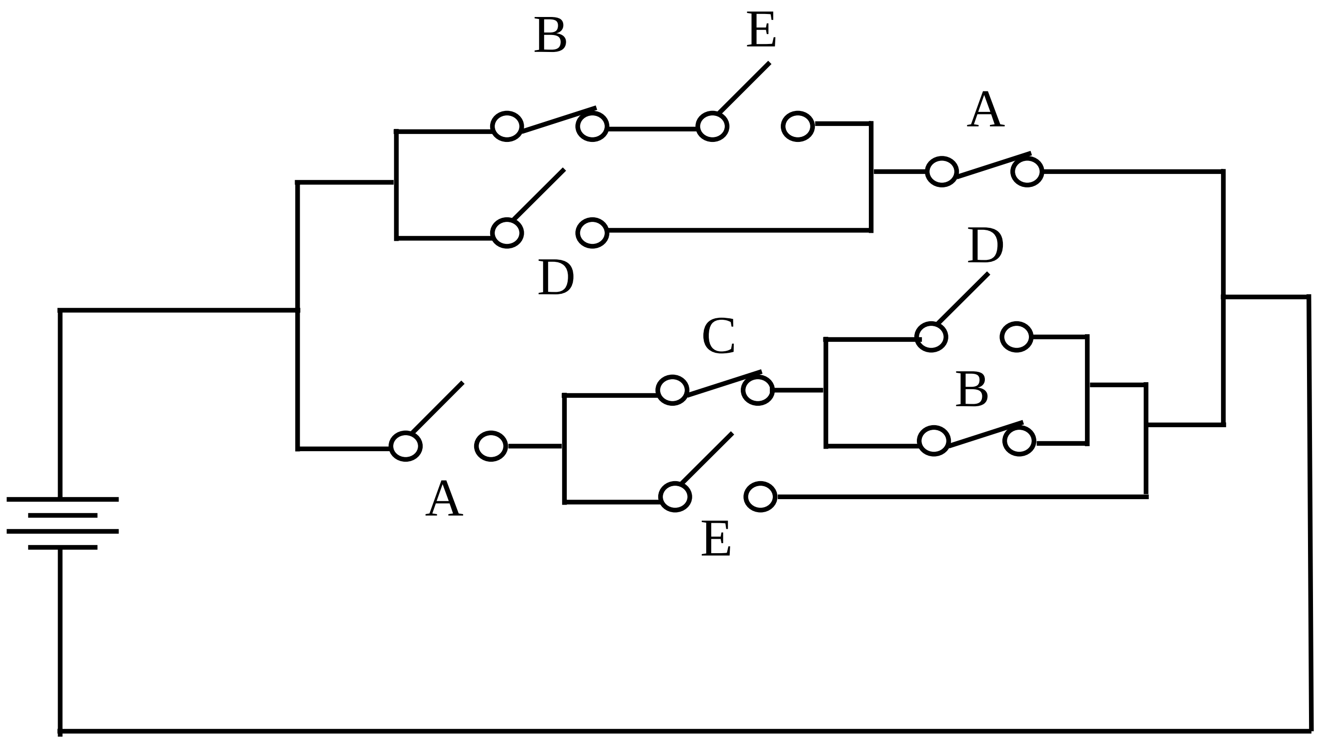


- Light is on (L = 1) for

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

  and off (L = 0), otherwise.
- Useful model for relay circuits and for CMOS gate circuits, where the latter one is the foundation of current digital logic technology

Logic Function Implementation (3/3)



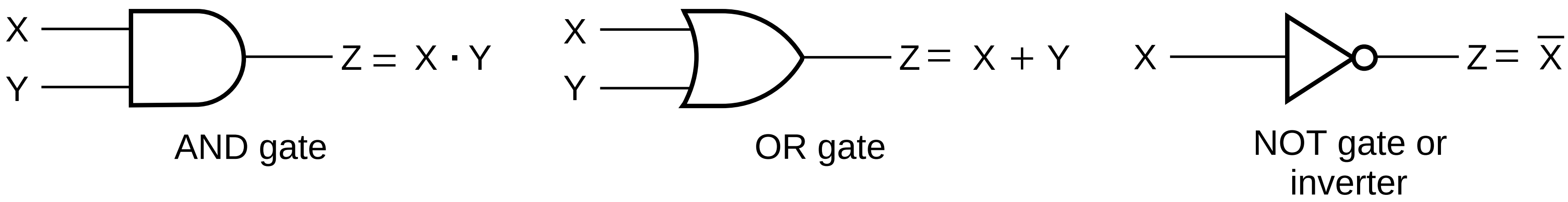
$$L = (\overline{B} \cdot E + D) \cdot \overline{A} + A \cdot (\overline{C} \cdot (\overline{B} + D) + E)$$

Logic Gates

- In the earliest computers, switches were opened and closed by magnetic fields produced by energizing coils in *relays*. The switches in turn are opened and closed the current paths.
- Later, *vacuum tubes* that open and close current paths electronically replaced relays.
- Today, *transistors* are used as electronic switches that open and close current paths.

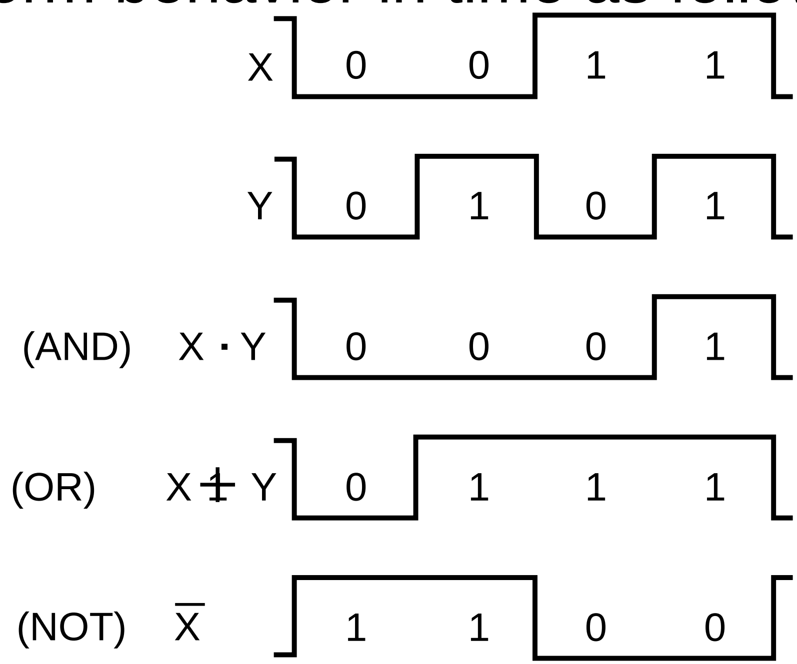
Logic Gate Symbols and Behavior

- Logic gates have special symbols:

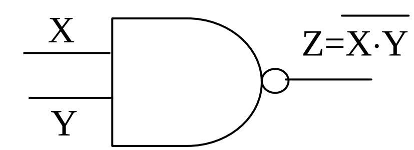


(a) Graphic symbols

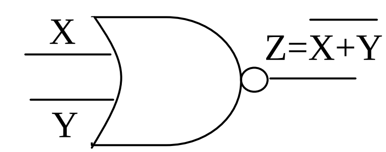
- And waveform behavior in time as follows:



(b) Timing diagram



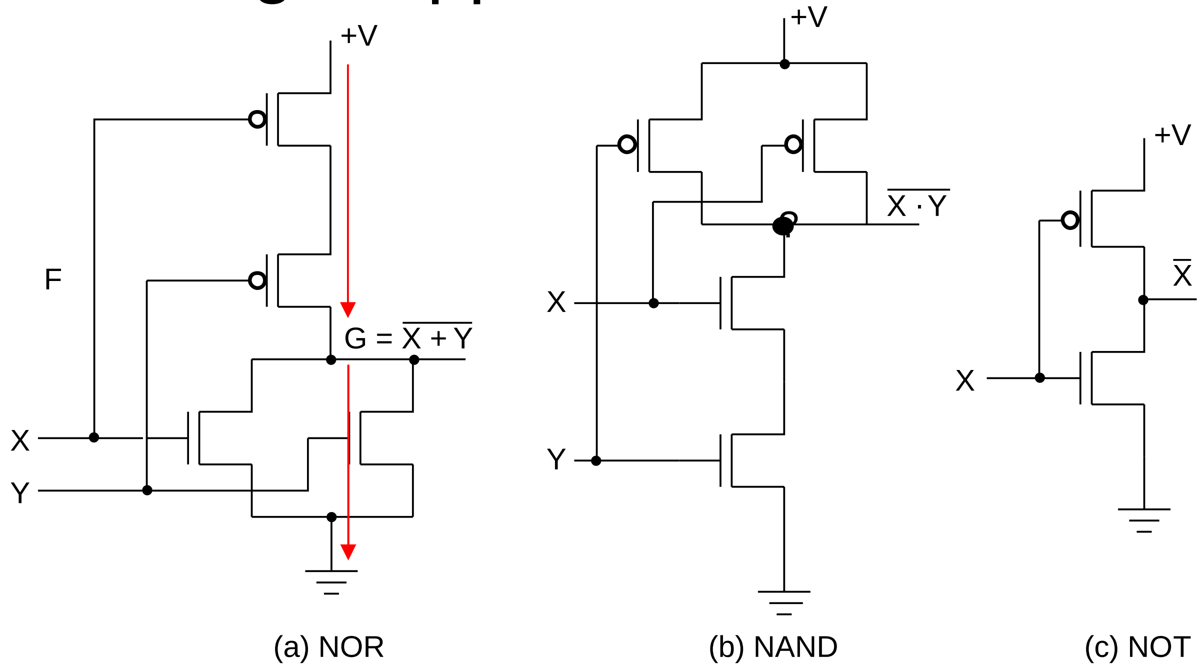
NAND gate



NOR gate

Logic Gates – Complementary Metal-Oxide-Semiconductor Circuits

- Implementation of logic gates with transistors (See Reading Supplement – CMOS Circuits)



- Transistor or tube implementations of logic functions are called logic gates or just gates
- Transistor gate circuits can be modeled by switch circuits

Logic Diagrams and Expressions

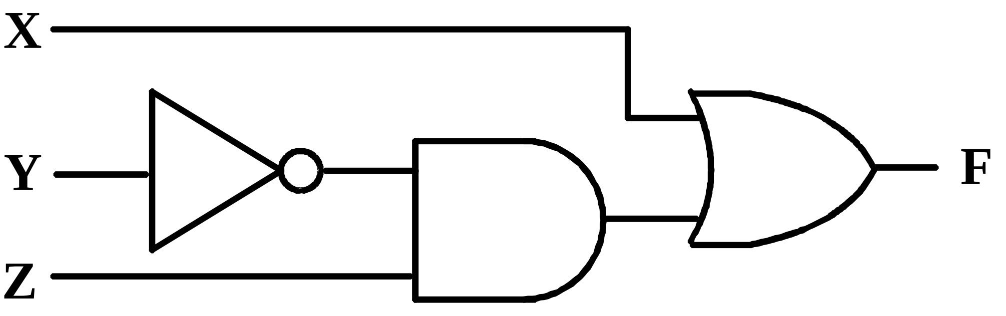
Truth Table

X Y Z	$F = X + \overline{Y} \cdot Z$
0 0 0	0
0 0 1	1
0 1 0	0
0 1 1	0
1 0 0	1
1 0 1	1
1 1 0	1
1 1 1	1

Equation

$$F = X + \overline{Y}\, Z$$

Logic Diagram



- Boolean equations, truth tables and logic diagrams describe the same function!
- Truth tables are unique; Boolean equation and logic diagrams are not. This gives flexibility in implementing functions.
- Different expressions require different hardware resources, i.e., devices and interconnections.

Boolean Algebra

- An algebraic structure defined on a set of at least two elements together with three binary operators (denoted +, · and ‾) that satisfies the following basic identities:

1. $X + 0 = X$	2. $X \cdot 1 = X$	
3. $X + 1 = 1$	4. $X \cdot 0 = 0$	
5. $X + X = X$	6. $X \cdot X = X$	
7. $X + \overline{X} = 1$	8. $X \cdot \overline{X} = 0$	
9. $\overline{\overline{X}} = X$		
10. $X + Y = Y + X$	11. $XY = YX$	Commutative
12. $(X + Y) + Z = X + (Y + Z)$	13. $(XY)Z = X(YZ)$	Associative
14. $X(Y + Z) = XY + XZ$	15. $X + YZ = (X + Y)(X + Z)$	Distributive
16. $\overline{X + Y} = \overline{X} \cdot \overline{Y}$	17. $\overline{X \cdot Y} = \overline{X} + \overline{Y}$	DeMorgan's

dual ⇔ (between the left and right columns)

The **dual** of an algebraic expression: interchange • and +, 0 and 1

Properties of Boolean Algebra (1/2)

- If the meaning is unambiguous, we leave out the symbol "·"
- The identities above are organized into pairs. These pairs have names as follows:

  1-4 Existence of 0 and 1   5-6 Idempotence

  7-8 Existence of complement   9 Involution

  10-11 Commutative Laws   12-13 Associative Laws

  14-15 Distributive Laws   16-17 DeMorgan's Laws

- Dual Rule: The dual of an algebraic expression is obtained by interchanging + and · and interchanging 0's and 1's.
- The identities appear in dual pairs. When there is only one identity on a line, the identity is self-dual, i. e., the dual expression = the original expression.

Properties of Boolean Algebra (2/2)

- Unless it happens to be self-dual, the dual of an expression does not equal the expression itself.
- Example: $F = (A + \overline{C}) \cdot B + 0$

  dual $F = (A \cdot \overline{C} + B) \cdot 1 = A \cdot \overline{C} + B$
- Example: $G = X \cdot Y + (\overline{W + Z})$

  dual $G = (X + Y) \cdot (\overline{W \cdot Z})$
- Example: $H = A \cdot B + A \cdot C + B \cdot C$

  dual $H = (A + B) \cdot (A + C) \cdot (B + C)$
- Are any of these functions self-dual?

Boolean Operator Precedence

- The order of evaluation in a Boolean expression is:
  1. Parentheses
  2. NOT
  3. AND
  4. OR

  Order!

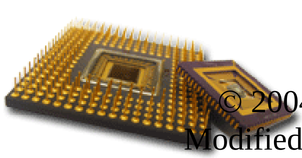
- Consequence: parentheses appear around OR expressions
- Example: $F = A(B + C)(C + \overline{D})$

Example 1: Boolean Algebraic Proof

- $A + A \cdot B = A$ (Absorption Theorem)

Proof Steps	Justification (identity or theorem)
$A + A \cdot B$	
$= A \cdot 1 + A \cdot B$	$X = X \cdot 1$
$= A \cdot (1 + B)$	$X \cdot Y + X \cdot Z = X \cdot (Y + Z)$ (Distributive Law)
$= A \cdot 1$	$1 + X = 1$
$= A$	$X \cdot 1 = X$

- Our primary reason for doing proofs is to learn:
  - Careful and efficient use of the identities and theorems of Boolean algebra.
  - How to choose the appropriate identity or theorem to apply to make forward progress, irrespective of the application.



Example 2: Boolean Algebraic Proofs

- $AB + \overline{A}C + BC = AB + \overline{A}C$ (Consensus Theorem)

Proof Steps	Justification (identity or theorem)
$AB + \overline{A}C + BC$	
$= AB + \overline{A}C + 1 \cdot BC$	$X = X \cdot 1$
$= AB + \overline{A}C + (A + \overline{A}) \cdot BC$	$X + \overline{X} = 1$
$= AB + ABC + \overline{A}C + \overline{A}BC$	Distributive & Commutative
$= AB(1+C) + \overline{A}C(1+B)$	$1 + X = 1$
$= AB + \overline{A}C$	$X \cdot 1 = X$

Example 3: Boolean Algebraic Proofs

- $(\overline{X+Y})Z + X\overline{Y} = \overline{Y}(X+Z)$

Proof Steps	Justification (identity or theorem)
$(\overline{X+Y})Z + X\overline{Y}$	
$= X\overline{Y} + (\overline{X+Y})Z$	commutative
$= (X\overline{Y} + (\overline{X+Y}))\,(X\overline{Y} + Z)$	distributive
$= (X\overline{Y} + \overline{X}\,\overline{Y})\,(X\overline{Y} + Z)$	DeMorgan's
$= \overline{Y}(X + \overline{X})\,(X\overline{Y} + Z)$	distributive
$= \overline{Y}\,(X\overline{Y} + Z)$	$X + X = 1$
$= X\overline{Y} + \overline{Y}Z$	$X \cdot X = X$
$= \overline{Y}\,(X + Z)$	distributive

- Duality holds? $(\overline{X}Y + Z)(X + \overline{Y}) =? \overline{Y} + XZ$

Example 3: Boolean Algebraic Proofs

Proof Steps	Proof Steps
$x \cdot y + \overline{x} \cdot y = y$	$(x + y)(\overline{x} + y) = y$
$y(x + \overline{x})$	$(y + x)(y + \overline{x})$
$= y \cdot 1 = y$	$= y + x \cdot \overline{x}$
	$= y + 0 = y$

Truth Table of DeMorgan's Laws

$$\overline{x+y} = \overline{x} \cdot \overline{y} \qquad \overline{x \cdot y} = \overline{x} + \overline{y}$$

Truth Tables to Verify DeMorgan's Theorem

A) X	Y	X+Y	$\overline{X+Y}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

B) X	Y	$\overline{X}$	$\overline{Y}$	$\overline{X}\cdot\overline{Y}$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

Useful Theorems

- $x \cdot y + \overline{x} \cdot y = y \qquad (x + y)(\overline{x} + y) = y$ Minimization
- $x + x \cdot y = x \qquad x \cdot (x + y) = x$ Absorption
- $x + \overline{x} \cdot y = x + y \qquad x \cdot (\overline{x} + y) = x \cdot y$ Simplification
- $x \cdot y + \overline{x} \cdot z + y \cdot z = x \cdot y + \overline{x} \cdot z$ Consensus

  $(x + y) \cdot (\overline{x} + z) \cdot (y + z) = (x + y) \cdot (\overline{x} + z)$
- $\overline{x + y} = \overline{x} \cdot \overline{y} \qquad \overline{x \cdot y} = \overline{x} + \overline{y}$ DeMorgan's Laws

Boolean Function Evaluation

$F1 = xy\overline{z}$

$F2 = x + \overline{y}z$

$F3 = \overline{x}\,\overline{y}\,\overline{z} + \overline{x}yz + x\overline{y}$

$F4 = x\overline{y} + \overline{x}z$

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

Simplified Expression

- An application of Boolean algebra
- Simplify to contain the smallest number of literals (complemented and uncomplemented variables):

$$AB + \overline{A}CD + \overline{A}BD + \overline{A}C\overline{D} + ABCD$$

$$= AB + ABCD + \overline{A}CD + \overline{A}C\overline{D} + \overline{A}BD$$

$$= AB + AB(CD) + \overline{A}C(D + \overline{D}) + \overline{A}BD$$

$$= AB + \overline{A}C + \overline{A}BD = B(A + \overline{A}D) + \overline{A}C$$

$$= B(A + D) + \overline{A}C$$ <5 literals>

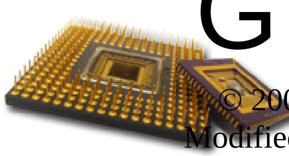
Complementing Functions

- Use DeMorgan's Theorem to complement a function:
  1. Interchange AND and OR operators
  2. Complement each constant value and literal
- Example: Complement F= $\bar{x}y\bar{z} + x\bar{y}\bar{z}$

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

- Example: Complement G = $(\bar{a} + bc)\bar{d}+e$

$$\bar{G} = (a \cdot (\bar{b}+\bar{c})+d) \cdot \bar{e}$$



Overview – Canonical Forms

- What are Canonical Forms?
- Minterms and Maxterms
- Index Representation of Minterms and Maxterms
- Sum-of-Minterm (SOM) Representations
- Product-of-Maxterm (POM) Representations
- Representation of Complements of Functions
- Conversions between Representations

Canonical Forms

- It is useful to specify Boolean functions in a form that:
  - Allows comparison for equality.
  - Has a correspondence to the truth tables
- Canonical Forms in common usage:
  - Sum of Minterms (SOM)
  - Product of Maxterms (POM)

Minterms

- Minterms are AND terms with every variable present in either true or complemented form.
- Given that each binary variable may appear normal (e.g., x) or complemented (e.g., $\overline{x}$), there are 2^n minterms for *n* variables.
- Example: Two variables (X and Y) produce 2 x 2 = 4 combinations:
  - XY (both normal)
  - $X\overline{Y}$ (X normal, Y complemented)
  - $\overline{X}Y$ (X complemented, Y normal)
  - $\overline{X}\,\overline{Y}$ (both complemented)
- Thus there are four minterms of two variables.

Maxterms

- Maxterms are OR terms with every variable in true or complemented form.
- Given that each binary variable may appear normal (e.g., x) or complemented (e.g., $\overline{x}$), there are 2^n maxterms for *n* variables.
- Example: Two variables (X and Y) produce 2 x 2 = 4 combinations:

$X + Y$ (both normal)

$X + \overline{Y}$ (x normal, y complemented)

$\overline{X} + Y$ (x complemented, y normal)

$\overline{X} + \overline{Y}$ (both complemented)

Maxterms and Minterms

- Examples: Two variable minterms and maxterms.

Complemented

true

Index	Minterm	Maxterm
0	$\bar{x}\,\bar{y}$	$x + y$
1	$\bar{x}\,y$	$x + \bar{y}$
2	$x\,\bar{y}$	$\bar{x} + y$
3	$x\,y$	$\bar{x} + \bar{y}$

- The index above is important for describing which variables in the terms are true and which are complemented – the variable order is the bit order of the index binary number.

Standard Order

- Minterms and maxterms are designated with a subscript.
- The subscript is a number, corresponding to a binary pattern
- The bits in the pattern represent the complemented or normal state of each variable listed in a standard order.
- All variables will be presented in a minterm or maxterm and will be listed in the same order (usually alphabetically)
- Example: For variables a, b, c:
  - Maxterms: $(a + b + c)$, $(a + b + c)$
  - Minterms: $a \bar{b} c$, $\bar{a} b c$, $\bar{a} \bar{b} c$
  - Terms: $(b + a + c)$, $a \bar{c} b$, and $(c + b + a)$ are NOT in standard order.
  - Terms: $(a + c)$, $\bar{b} c$, and $(\bar{a} + b)$ do not contain all variables

Purpose of the Index

- The index for the minterm or maxterm, expressed as a binary number, is used to determine whether the variable is shown in the true form or complemented form.
- For Minterms:
  - "1" means the variable is "Not Complemented" and
  - "0" means the variable is "Complemented".
- For Maxterms:
  - "0" means the variable is "Not Complemented" and
  - "1" means the variable is "Complemented".

Standard order

	X	Y	Z
index	0	1	1

$\overline{X} \cdot Y \cdot Z$

$X + \overline{Y} + \overline{Z}$

Index Example in Three Variables

- Assume the variables are called X, Y, and Z.
- The standard order is X, then Y, then Z.
- The Index 0 (base 10) = 000 (base 2) for three variables). All three variables are complemented for <u>minterm 0</u> ($\overline{X}, \overline{Y}, \overline{Z}$) and no variables are complemented for <u>Maxterm 0</u> (X,Y,Z).
  - Minterm 0, called m_0 is $\overline{X}\,\overline{Y}\,\overline{Z}$.
  - Maxterm 0, called M_0 is (X + Y + Z).
  - Minterm 6 ? $XY\overline{Z}$
  - Maxterm 6 ? $(\overline{X} + \overline{Y} + Z)$

Index Examples in Four Variables

Index i	Binary Pattern	Minterm m_i	Maxterm M_i
0	0000	$\bar{a}\,\bar{b}\,\bar{c}\,\bar{d}$	$a + b + c + d$
1	0001	$\bar{a}\,\bar{b}\,\bar{c}\,d$	?
3	0011	?	$a + b + \bar{c} + \bar{d}$
5	0101	$\bar{a}\,b\,\bar{c}\,d$	$a + \bar{b} + c + \bar{d}$
7	0111	?	$a + \bar{b} + \bar{c} + \bar{d}$
10	1010	$a\,\bar{b}\,c\,\bar{d}$	$\bar{a} + b + \bar{c} + d$
13	1101	$a\,b\,\bar{c}\,d$	?
15	1111	$a\,b\,c\,d$	$\bar{a} + \bar{b} + \bar{c} + \bar{d}$

Minterm and Maxterm Relationship

- Review: DeMorgan's Theorem
  $\overline{x \cdot y} = \overline{x} + \overline{y}$ and $\overline{x + y} = \overline{x} \cdot \overline{y}$
- Two-variable example:
  $M_2 = \overline{x} + y$ and $m_2 = x \cdot \overline{y}$
  Thus M_2 is the complement of m_2 and vice-versa.
- Since DeMorgan's Theorem holds for *n* variables, the above holds for terms of *n* variables
- giving:
  $$M_i = \overline{m}_i \text{ and } m_i = \overline{M}_i$$
  Thus M_i is the complement of m_i.

Function Tables for Both

- Minterms of 2 variables　　Maxterms of 2 variables

x y	m_0	m_1	m_2	m_3
0 0	1	0	0	0
0 1	0	1	0	0
1 0	0	0	1	0
1 1	0	0	0	1

x y	M_0	M_1	M_2	M_3
0 0	0	1	1	1
0 1	1	0	1	1
1 0	1	1	0	1
1 1	1	1	1	0

- Each column in the maxterm function table is the complement of the column in the minterm function table since M_i is the complement of m_i.

Observations

- In the function tables:
  - Each minterm has one and only one 1 present in the 2^n terms (a minimum of 1s). All other entries are 0.
  - Each maxterm has one and only one 0 present in the 2^n terms. All other entries are 1 (a maximum of 1s).
- We can implement any function by "ORing" the minterms corresponding to "1" entries in the function table. These are called the minterms of the function.
- We can implement any function by "ANDing" the maxterms corresponding to "0" entries in the function table. These are called the maxterms of the function.
- This gives us two canonical forms:
  - Sum of Minterms (SOM)
  - Product of Maxterms (POM)

for stating any Boolean function.

Minterm Function Example in Three Variables

- Example: Find $F_1 = m_1 + m_4 + m_7$
- $F1 = \overline{x}\ \overline{y}\ z + x\ \overline{y}\ \overline{z} + x\ y\ z$

x y z	index	m_1	+	m_4	+	m_7	$= F_1$
0 0 0	0	0	+	0	+	0	= 0
0 0 1	1	1	+	0	+	0	= 1
0 1 0	2	0	+	0	+	0	= 0
0 1 1	3	0	+	0	+	0	= 0
1 0 0	4	0	+	1	+	0	= 1
1 0 1	5	0	+	0	+	0	= 0
1 1 0	6	0	+	0	+	0	= 0
1 1 1	7	0	+	0	+	1	= 1

Maxterm Function Example in Three Variables

- Example: Implement F1 in maxterms:

$$F_1 = M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6$$

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

x y z	i	$M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6$	= F1
0 0 0	0	0 · 1 · 1 · 1 · 1	= 0
0 0 1	1	1 · 1 · 1 · 1 · 1	= 1
0 1 0	2	1 · 0 · 1 · 1 · 1	= 0
0 1 1	3	1 · 1 · 0 · 1 · 1	= 0
1 0 0	4	1 · 1 · 1 · 1 · 1	= 1
1 0 1	5	1 · 1 · 1 · 0 · 1	= 0
1 1 0	6	1 · 1 · 1 · 1 · 0	= 0
1 1 1	7	1 · 1 · 1 · 1 · 1	= 1

Canonical Sum of Minterms

- Any Boolean function can be expressed as a Sum of Minterms.
  - For the function table – the minterms used are the terms corresponding to the 1's
  - For expressions – expand all terms first to explicitly list all minterms. Do this by "ANDing" any term missing a variable v with a term ($v + \overline{v}$).
- Example: Implement $f = x + \overline{x}\,\overline{y}$ as a sum of minterms.

First expand terms: $f = x(y + \overline{y}) + \overline{x}\ \overline{y}$

Then distribute terms: $f = xy + x\overline{y} + \overline{x}\,\overline{y}$

Express as sum of minterms: $f = m_3 + m_2 + m_0$

Another SOM Example

- Example: $F = A + \overline{B}C$
- There are three variables, A, B, and C which we take to be the standard order.
- Expanding the terms with missing variables:

$$F = A + \overline{B}C = A(B + \overline{B})(C + \overline{C}) + (A + \overline{A})\overline{B}C$$
$$= (AB + A\overline{B})(C + \overline{C}) + A\overline{B}C + \overline{A}\overline{B}C$$
$$= ABC + AB\overline{C} + A\overline{B}C + A\overline{B}\overline{C} + A\overline{B}C + \overline{A}\overline{B}C$$

- Collect terms (removing all but one of duplicate terms): terms 1, 4, 5, 6, 7
- Express as SOM: $ABC + AB\overline{C} + A\overline{B}C + A\overline{B}\overline{C} + \overline{A}\overline{B}C$

Shorthand SOM Form

- From the previous example, we started with:

$$F = A + \overline{B}\,C$$

- We ended up with:

$F = m_1 + m_4 + m_5 + m_6 + m_7$

- This can be denoted in the formal shorthand:

$$F(A, B, C) = \Sigma_m(1,4,5,6,7)$$

- Note that we explicitly show the standard variables in order and drop the "m" designators.

Canonical Product of Maxterms

- Any Boolean Function can be expressed as a <u>Product of Maxterms (POM)</u>.
  - For the function table – the maxterms used are the terms corresponding to the 0's.
  - For an expression – expand all terms first to explicitly list all maxterms. Do this by first applying the second distributive law , "ORing" terms missing variable v with a term equal to $v \cdot \bar{v}$ and then applying the distributive law again.
- Example: Convert to product of maxterms:

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

Apply the distributive law:

$$x + \bar{x}\,\bar{y} = (x + \bar{x})(x + \bar{y}) = 1 \cdot (x + \bar{y}) = x + \bar{y}$$

Add missing variable z:

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

Express as POM: $f = M_2 \cdot M_3$

Another POM Example

- Convert to Product of Maxterms:

$$f(A,B,C) = A\overline{C} + BC + \overline{A}\,\overline{B}$$

- Use $x + yz = (x+y)\cdot(x+z)$ with $x = (A\overline{C} + BC)$, $y = \overline{A}$, and $z = \overline{B}$ to get:

$$f = (A\overline{C} + BC + \overline{A})(A\overline{C} + BC + \overline{B})$$

- Then use $x + \overline{x}y = x + y$ to get:

$$f = (\overline{C} + BC + \overline{A})(A\overline{C} + C + \overline{B})$$

and a second time to get:

$$f = (\overline{C} + B + \overline{A})(A + C + \overline{B})$$

- Rearrange to standard order,

$$f = (\overline{A} + B + \overline{C})(A + \overline{B} + C)$$ to give $f = M_5 \cdot M_2$

Function Complements

- The complement of a function expressed as a sum of minterms is constructed by selecting the minterms missing in the sum-of-minterms canonical forms.
- Alternatively, the complement of a function expressed by a Sum of Minterms form is simply the Product of Maxterms with the same indices. $F(x, y, z) = \Sigma_m(1,3,5,7)$
- Example: Given

$$\overline{F}(x, y, z) = \Sigma_m(0,2,4,6)$$
$$\overline{F}(x, y, z) = \Pi_M(1,3,5,7)$$

Conversion Between Two Forms

- To convert between sum-of-minterms and product-of-maxterms form (or vice-versa) we follow these steps:
  - Find the function complement by swapping terms in the list with terms not in the list.
  - Change from products to sums, or vice versa.
- Example:Given F as before: $F(x,y,z) = \Sigma_m(1,3,5,7)$
- Form the Complement: $\overline{F}(x,y,z) = \Sigma_m(0,2,4,6)$
- Then use the other form with the same indices:

$$F(x,y,z) = \Pi_M(0,2,4,6)$$

$$F(x,y,z) = \overline{\sum{}_m (0,2,4,6)} = \overline{m_0 + m_2 + m_4 + m_6}$$

$$= \overline{m_0} \bullet \overline{m_2} \bullet \overline{m_4} \bullet \overline{m_6} = M_0 \bullet M_2 \bullet M_4 \bullet M_6$$

$$= \Pi_M (0,2,4,6)$$

Standard Forms

- Standard form of Sum-of-Products (SOP): equations are written as an OR of AND terms
- Standard form of Product-of-Sums (POS): equations are written as an AND of OR terms
- Examples:
  - SOP: $A\,B\,C + \overline{A}\,\overline{B}\,C + B$
  - POS: $(A + B)\cdot(A + \overline{B} + \overline{C})\cdot C$
- These "mixed" forms are neither SOP nor POS
  - $(A\,B + C)\,(A + C)$
  - $A\,B\,\overline{C} + A\,C\,(A + B)$

Sum-of-Products (SOP)

- A sum of minterms form for *n* variables can be written down directly from a truth table.
  - Implementation of this form is a two-level network of gates such that:
    - The first level consists of *n*-input AND gates, and
    - The second level is a single OR gate (with fewer than 2^n inputs).
- This form often can be simplified so that the corresponding circuit is simpler.

Sum-of-Products (SOP)

- A Simplification Example:
- $F(A,B,C) = \Sigma m(1,4,5,6,7)$
- Writing the minterm expression:

  $F = \bar{A}\,\bar{B}\,C + A\,\bar{B}\,\bar{C} + A\,\bar{B}\,C + AB\bar{C} + ABC$
- Simplifying:

  duplicate

  $F = (\bar{A}\bar{B}C + A\bar{B}C) + (A\bar{B}\bar{C} + A\bar{B}C) + (AB\bar{C} + ABC)$

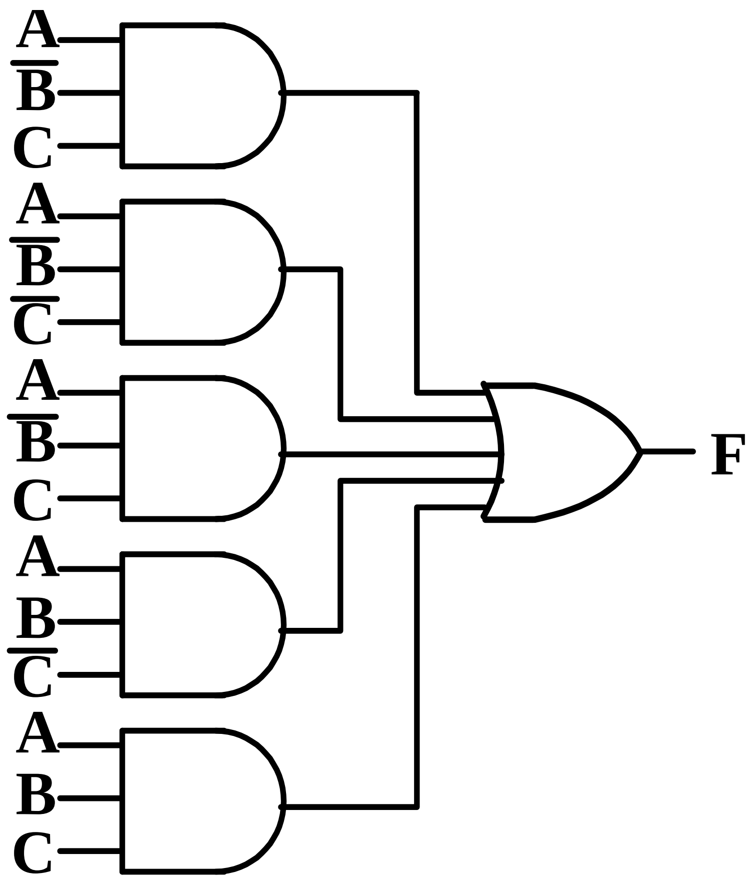
  $= \bar{B}C(\bar{A} + A) + A\bar{B}(\bar{C} + C) + AB(\bar{C} + C)$

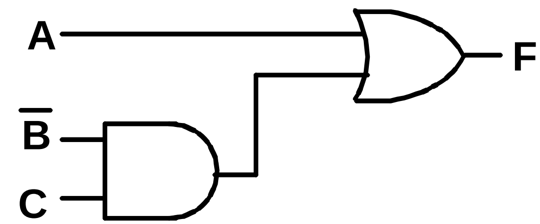
  $= \bar{B}C + A\bar{B} + AB$

  $= A + \bar{B}C$
- Simplified F contains 3 literals compared to 15 in minterm F

AND/OR Two-level Implementation of SOP Expression

- The two implementations for F are shown below – it is quite apparent which is simpler!





SOP and POS Observations

- The previous examples show that:
  - Canonical Forms (Sum-of-minterms, Product-of-Maxterms), or other standard forms (SOP, POS) differ in complexity
  - Boolean algebra can be used to manipulate equations into simpler forms.
  - Simpler equations lead to simpler two-level implementations
- Questions:
  - How can we attain a “simplest” expression?
  - Is there only one minimum cost circuit?
  - The next lecture will deal with these issues.

Terms of Use

- Return to Title Page