Computer Logic Design Fundamentals

Chapter 2 – Combinational Logic Circuits

Part 1 – Gate Circuits and Boolean Equations

Prof. Yueming Wang

ymingwang@zju.edu.cn

College of Computer Science and Technology,
Zhejiang University

Overview

- **Part 1 – Gate Circuits and Boolean Equations**
  - **Binary Logic and Gates**
  - **Boolean Algebra**
  - **Standard Forms**
- **Part 2 – Circuit Optimization**
  - **Two-Level Optimization**
  - **Map Manipulation**
  - **Practical Optimization (Espresso)**
  - **Multi-Level Circuit Optimization**
- **Part 3 – Additional Gates and Circuits**
  - **Other Gate Types**
  - **Exclusive-OR Operator and Gates**
  - **High-Impedance Outputs**
  - **Propagation Delay**

Binary Logic and Gates

- Binary variables take on one of two values.
- Logical operators operate on binary values and binary variables.
- Basic logical operators are the logic functions AND, OR and NOT.
- Logic gates implement logic functions.
- Boolean Algebra: a useful mathematical system for specifying and transforming logic functions.
- We study Boolean algebra as a 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:**
  - **$Y = A \times B$ 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 <u>plus</u> one equals two")

  is not the same as

  1 + 1 = 1 (read "1 <u>or</u> 1 equals 1").

Operator Definitions

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

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 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

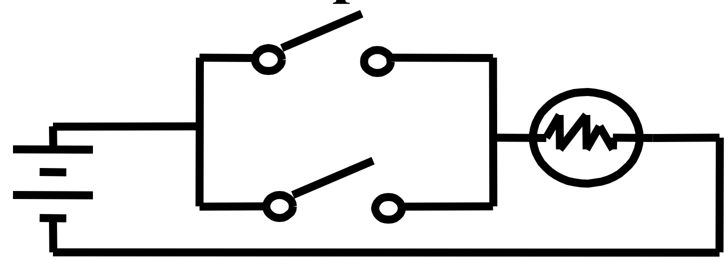
OR		
X	Y	$Z = X+Y$
0	0	0
0	1	1
1	0	1
1	1	1

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

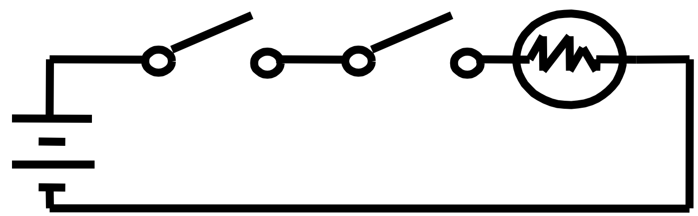
Logic Function Implementation

- 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

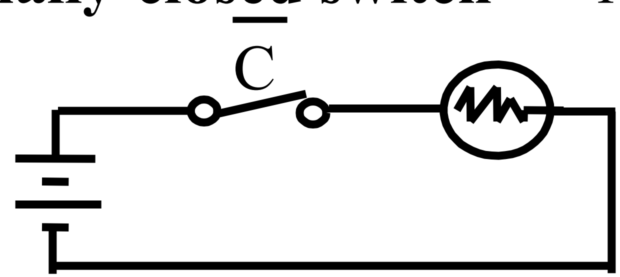
Switches in parallel => OR



Switches in series => AND

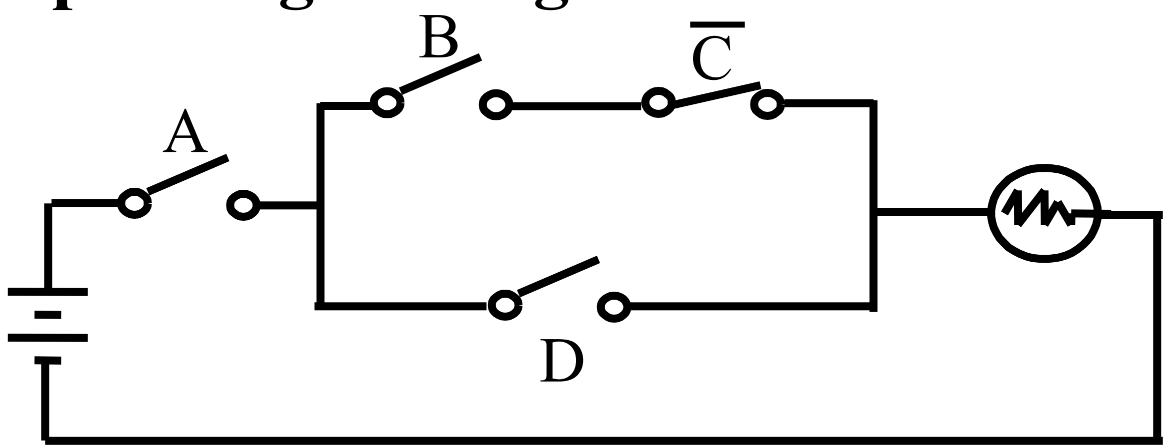


Normally-closed switch => NOT



Logic Function Implementation (Continued)

- **Example: Logic Using Switches**



- **Light is on (L = 1) for**

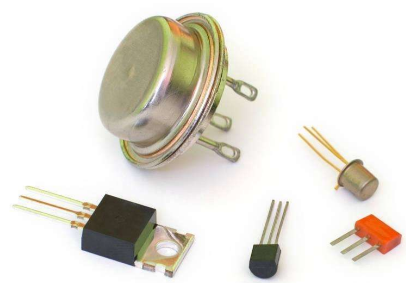
  L(A, B, C, D) =

  and off (L = 0), otherwise.

- **Useful model for relay circuits and for CMOS gate circuits, the foundation of current digital logic technology**

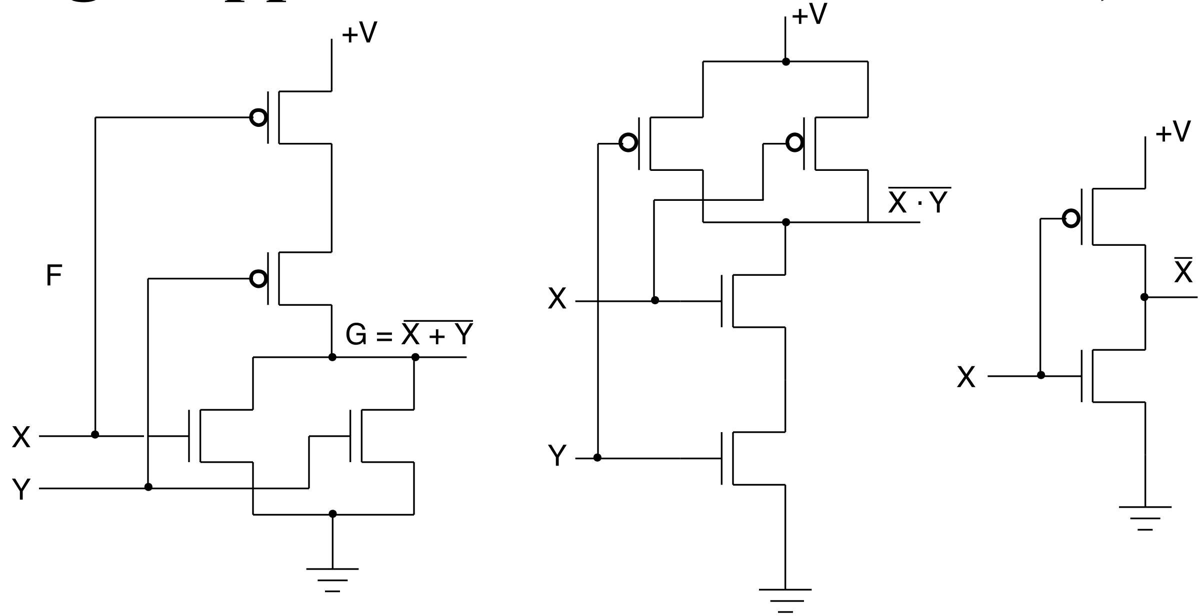
Logic Gates

- **In the earliest computers, switches were opened and closed by magnetic fields produced by energizing coils in *relays*. The switches in turn 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 Gates (Continued)

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

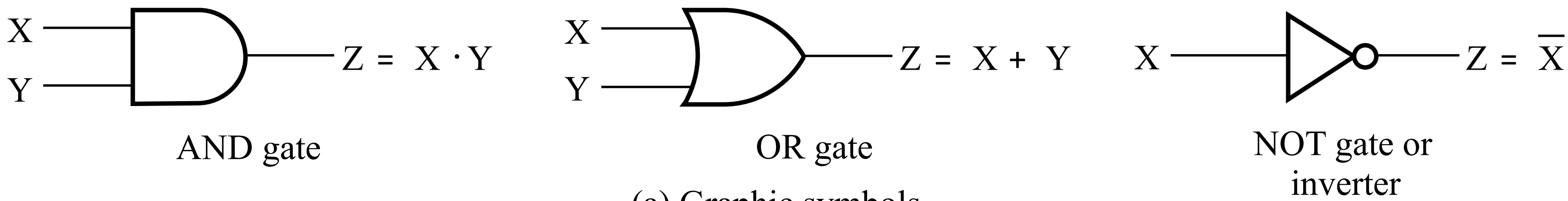


(a) NOR (b) NAND (c) NOT

- **Transistor or tube implementations of logic functions are called <u>logic gates</u> or just <u>gates</u>**
- **Transistor gate circuits can be modeled by switch circuits**

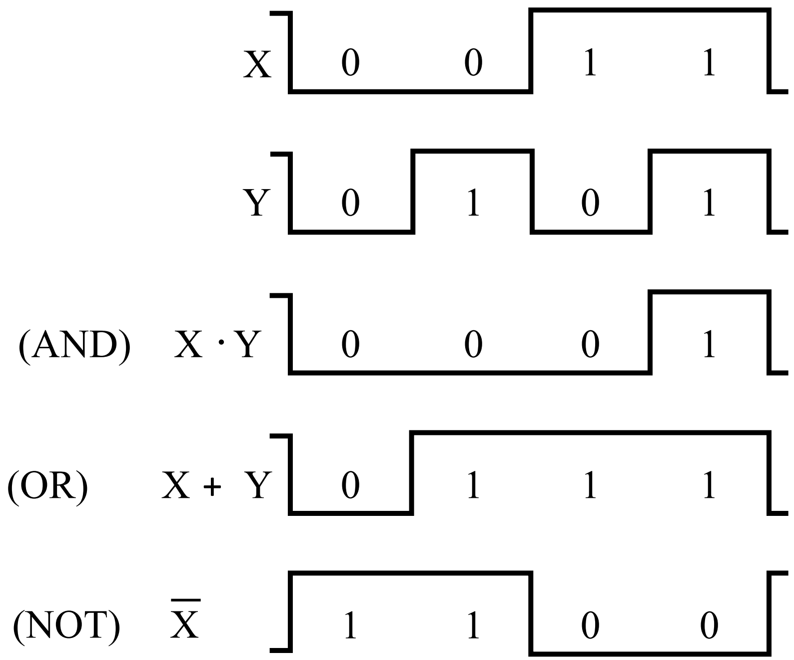
Logic Gate Symbols and Behavior

- **Logic gates have special symbols:**



(a) Graphic symbols

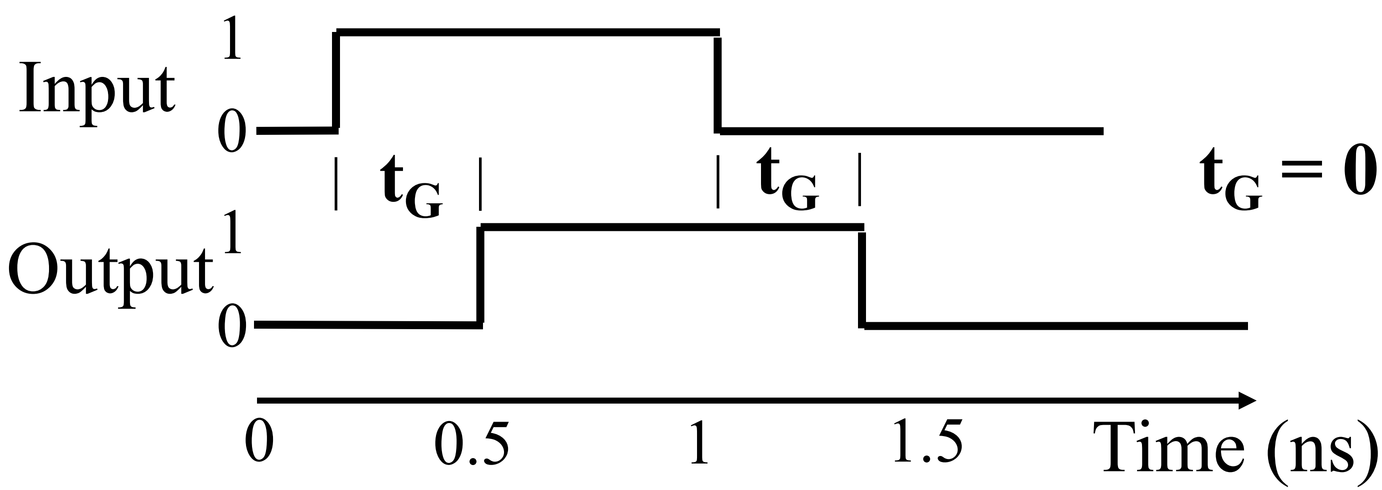
- **And waveform behavior in time as follows:**



(b) Timing diagram

Gate Delay

- **In actual physical gates, if one or more input changes causes the output to change, the output change does not occur instantaneously.**
- **The delay between an input change(s) and the resulting output change is the *gate delay* denoted by t_G:**



$t_G = 0.3$ ns

Logic Diagrams and Expressions

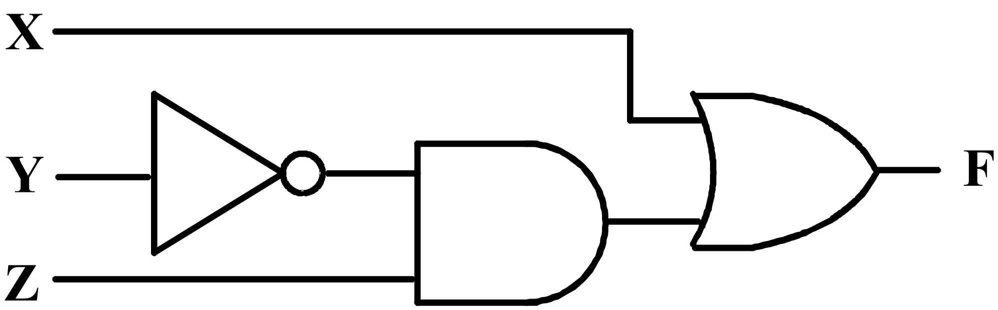
Truth Table

X Y Z	$F = X + \overline{Y} \times Z$
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; expressions and logic diagrams are not. This gives flexibility in implementing functions.**

Boolean Algebra

- **An algebraic structure defined on a set of at least two elements, B, 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

Boolean Operator Precedence

- **The order of evaluation in a Boolean expression is:**
  1. Parentheses
  2. NOT
  3. AND
  4. OR
- **Consequence: Parentheses appear around OR expressions**
- **Example:** $F = A(B + C)(C + \overline{D})$

Some Properties of Identities & the Algebra

- 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

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

Some Properties of Identities & the Algebra (Continued)

- **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 $G =$
- **Example:** $H = A \cdot B + A \cdot C + B \cdot C$

  dual $H =$
- **Are any of these functions self-dual?**

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, and
  - 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)**
$AB + \overline{A}C + BC$	
$= AB + \overline{A}C + 1 \cdot BC$	**?**
$= AB + \overline{A}C + (A + \overline{A}) \cdot BC$	**?**
$=$	

Example 3: Boolean Algebraic Proofs

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

Proof Steps **Justification (identity or theorem)**

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

$=$

Useful Theorems

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

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

Proof of Simplification

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

Boolean Function Evaluation

$F1 = xy\bar{z}$
$F2 = x + \bar{y}z$
$F3 = \bar{x}\bar{y}\bar{z} + \bar{x}\,y\,z + x\bar{y}$
$F4 = x\bar{y} + \bar{x}\,z$

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

Expression Simplification

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

$$A\,B + \overline{A}\,C\,D + \overline{A}\,B\,D + \overline{A}\,C\,\overline{D} + A\,B\,C\,D$$

$$= AB + ABCD + \overline{A}\,C\,D + \overline{A}\,C\,\overline{D} + \overline{A}\,B\,D$$

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

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

$$= B\,(A + D) + \overline{A}\,C \quad \text{5 literals}$$

Example: Simplify Expression

$$L = AB + A\overline{C} + \overline{B}C + \overline{C}B + \overline{B}D + \overline{D}B + ADE(F+G)$$

$$L = A\overline{\overline{B}C} + \overline{B}C + \overline{C}B + \overline{B}D + \overline{D}B + ADE(F+G)$$ **DeMorgan Laws**

$$= A + \overline{B}C + \overline{C}B + \overline{B}D + \overline{D}B + ADE(F+G)$$ $A + \overline{A}B = A + B$

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

$$= A + \overline{B}C(D+\overline{D}) + \overline{C}B + \overline{B}D + \overline{D}B(C+\overline{C})$$ $A + \overline{A} = 1$

$$= A + \overline{B}CD + \overline{B}C\overline{D} + B\overline{C} + \overline{B}D + \overline{D}BC + \overline{D}B\overline{C}$$ **Distributive Laws**

$$= A + \overline{B}C\overline{D} + B\overline{C} + \overline{B}D + \overline{D}BC$$ $A+AB=A$

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

$$= A + C\overline{D} + B\overline{C} + \overline{B}D$$ $A + \overline{A} = 1$

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 $G = (\bar{a} + bc)\bar{d} + e$

$\bar{G} =$

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 $2 \times 2 = 4$ 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.**

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

- **The index above is important for describing which variables in the terms are true and which are complemented.**

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 present in a minterm or maxterm and will be listed in the <u>same order</u> (usually alphabetically)**
- **Example: For variables a, b, c:**
  - **Maxterms: $(a + b + \bar{c})$, $(a + b + c)$**
  - **Terms: $(b + a + c)$, $a\,\bar{c}\,b$, and $(c + b + a)$ are NOT in standard order.**
  - **Minterms: $a\,\bar{b}\,c$, $a\,b\,c$, $\bar{a}\,\bar{b}\,c$**
  - **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".**

Index Example in Three Variables

- **Example: (for 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 minterm 0 ($\overline{X}, \overline{Y}, \overline{Z}$) and no variables are complemented for Maxterm 0 (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 ?**
  - **Maxterm 6 ?**

Index Examples – Four Variables

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

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

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

Maxterms of 2 variables

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 <u>min</u>term has one and only one 1 present in the 2^n terms (a <u>minimum</u> of 1s). All other entries are 0.**
  - **Each <u>max</u>term has one and only one 0 present in the 2^n terms All other entries are 1 (a <u>max</u>imum 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 <u>canonical forms</u>:**
  - **<u>Sum of Minterms (SOM)</u>**
  - **<u>Product of Maxterms (POM)</u>**

for stating any Boolean function.

Minterm Function Example

- **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

Minterm Function Example

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

Maxterm Function Example

- **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	$0 \cdot 1 \cdot 1 \cdot 1 \cdot 1$	= 0
0 0 1	1	$1 \cdot 1 \cdot 1 \cdot 1 \cdot 1$	= 1
0 1 0	2	$1 \cdot 0 \cdot 1 \cdot 1 \cdot 1$	= 0
0 1 1	3	$1 \cdot 1 \cdot 0 \cdot 1 \cdot 1$	= 0
1 0 0	4	$1 \cdot 1 \cdot 1 \cdot 1 \cdot 1$	= 1
1 0 1	5	$1 \cdot 1 \cdot 1 \cdot 0 \cdot 1$	= 0
1 1 0	6	$1 \cdot 1 \cdot 1 \cdot 1 \cdot 0$	= 0
1 1 1	7	$1 \cdot 1 \cdot 1 \cdot 1 \cdot 1$	= 1

Maxterm Function Example

- **F(A, B,C,D) = $M_3 \cdot M_8 \cdot M_{11} \cdot M_{14}$**
- **F(A, B,C,D) =**

Canonical Sum of Minterms

- Any Boolean function can be expressed as a <u>Sum of Minterms</u>.
  - For the function table, the <u>minterms</u> used are the terms corresponding to the 1's
  - For expressions, <u>expand</u> 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:**
- **Collect terms (removing all but one of duplicate terms):**
- **Express as SOM:**

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 $v \cdot \overline{v}$ and then applying the distributive law again.**
- **Example: Convert to product of maxterms:**

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

Apply the distributive law:

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

Add missing variable z:

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

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

Another POM Example

- **Convert to Product of Maxterms:**

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

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

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

- **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,**

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

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.
- Example: Given $F(x, y, z) = \Sigma_m(1,3,5,7)$

$$\overline{F}(x, y, z) = \Sigma_m(0,2,4,6)$$

$$\overline{F}(x, y, z) = \Pi_M(1,3,5,7)$$

Conversion Between 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 – this forms the complement again, giving the other form of the original function: $F(x, y, z) = \Pi_M(0,2,4,6)$

Standard Forms

- **Standard Sum-of-Products (SOP) form:** equations are written as an OR of AND terms
- **Standard Product-of-Sums (POS) form:** 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**
  - $(A\,B + C)\,(A + C)$
  - $A\,B\,\overline{C} + A\,C\,(A + B)$

Standard 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.**

Standard Sum-of-Products (SOP)

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

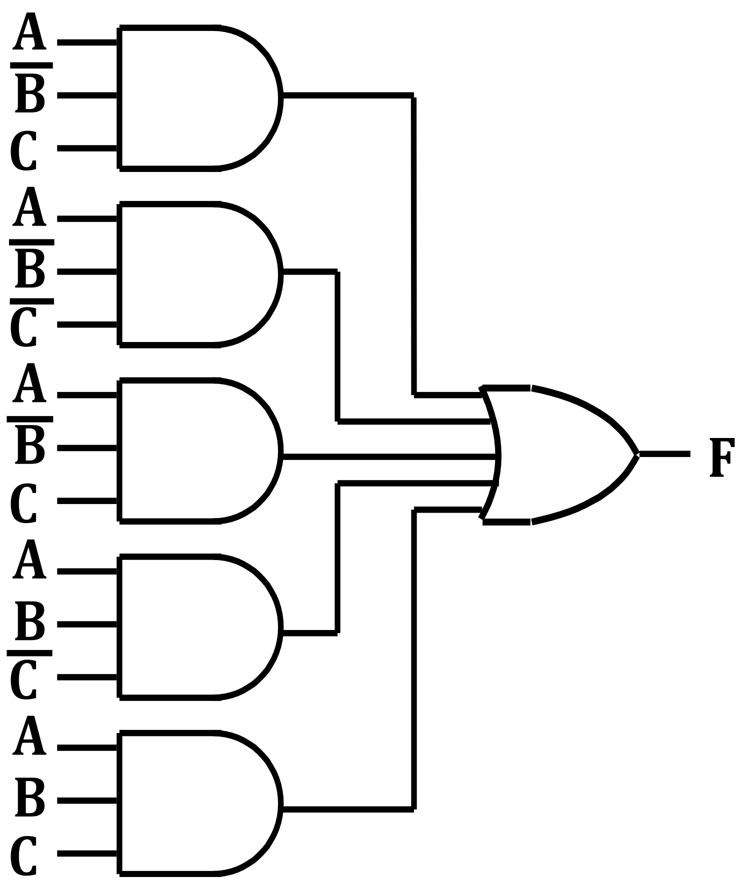
  $F = \overline{A}\,\overline{B}\,C + A\,\overline{B}\,\overline{C} + A\,\overline{B}\,C + AB\overline{C} + ABC$
- **Simplifying:**

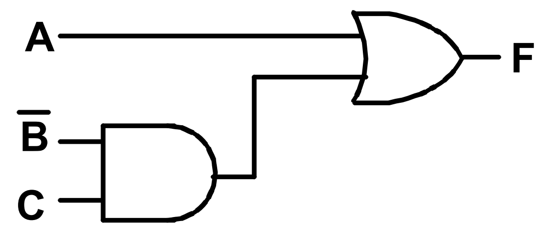
  $F =$

- **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 part will deal with these issues.**

Assignment

2-1a; 2-2a, c; 2-3a, c; 2-6b, d; 2-10a, c; 2-11a, c, d; 2-12b