Logic and Computer Design Fundamentals

Chapter 2 – Combinational Logic Circuits

Part 1 – Gate Circuits and Boolean Equations

Charles Kime & Thomas Kaminski

(Hyperlinks are active in View Show mode)

Updated by Dr. Waleed Dweik

Combinational Logic Circuits

- Digital (logic) circuits are hardware components that manipulate binary information.
- Integrated circuits: transistors and interconnections.
  - Basic circuits is referred to as ***logic gates***
  - The outputs of gates are applied to the inputs of other gates to form a digital circuit
- Combinational? Later…

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

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 (∧)
- OR is denoted by a plus (+) or (∨)
- NOT is denoted by an over-bar (‾), a single quote mark (') after, or (~) before the variable

Notation Examples

- Examples:
  - $Z = X \cdot \mathrm{Y} = \mathrm{XY} = X \wedge Y$: is read "Z is equal to X AND Y"
    - Z = 1 if and only if X = 1 and Y = 1; otherwise, Z = 0
  - $Z = X + Y = X \vee Y$: is read "Z is equal to X OR Y"
    - Z = 1 if (only X = 1) or if (only Y = 1) or if (X =1 and Y = 1)
  - Z= $\bar{X} = X' = \sim X$: is read "Z is equal to NOT X"
    - Z = 1 if X = 0; otherwise, Z = 0

- Notice the difference between arithmetic addition and logical OR:
  - 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 . 0 = 0
0 . 1 = 0
1 . 0 = 0
1 . 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		
Inputs		Output
X	Y	Z = X . Y
0	0	0
0	1	0
1	0	0
1	1	1

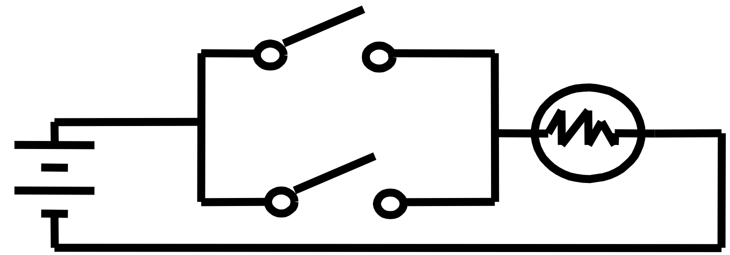
OR		
Inputs		Output
X	Y	Z = X + Y
0	0	0
0	1	1
1	0	1
1	1	1

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

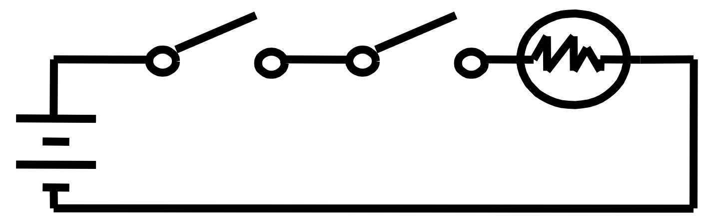
Logic Function Implementation

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

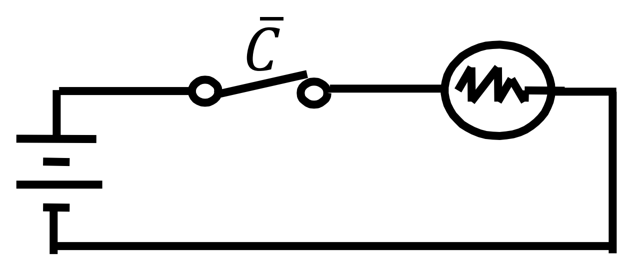
Switches in parallel => OR



Switches in series => AND

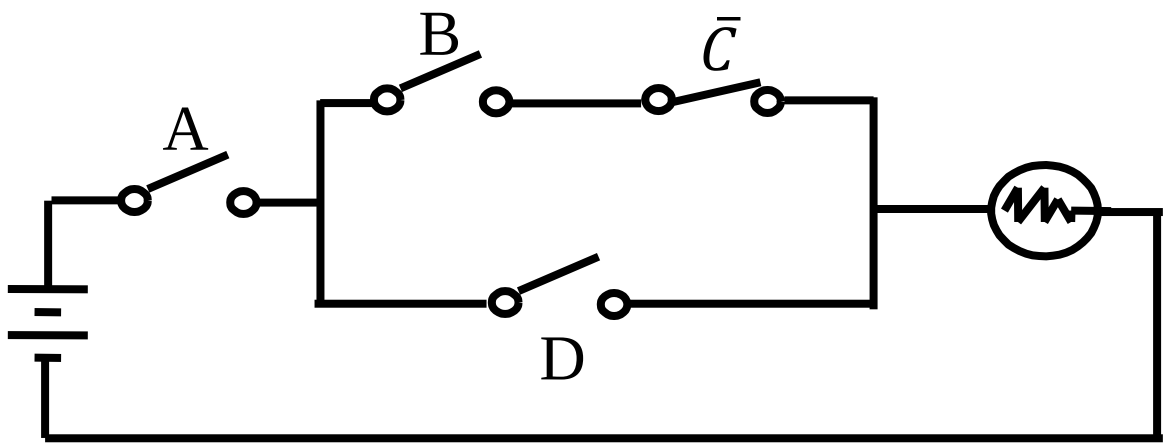


Normally-closed switch => NOT



Logic Function Implementation (Continued)

- Example: Logic Using Switches



- Light is
  ON (L = 1) for $L\ (A, B, C, D) = A\,.(B\bar{C} + D) = \mathrm{AB}\bar{C} + AD$
  and ***OFF*** (L = 0), otherwise.

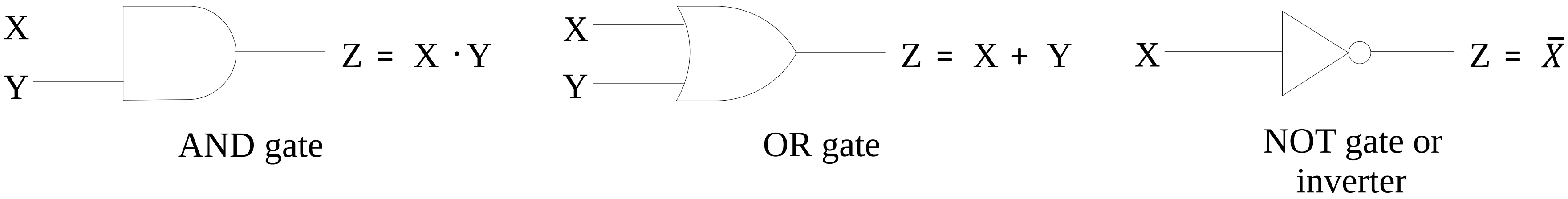
- Useful model for relay circuits and for CMOS gate circuits, the foundation of current digital logic technology

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
- Optional: Chapter 6 – Part 1: The Design Space

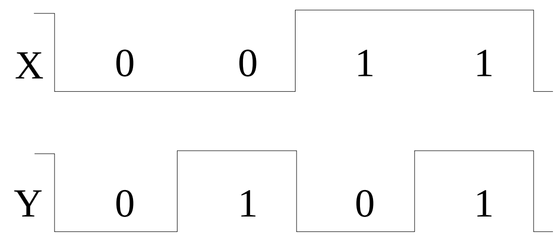
Logic Gate Symbols and Behavior

- **Logic gates have special symbols:**



(a) Graphic symbols

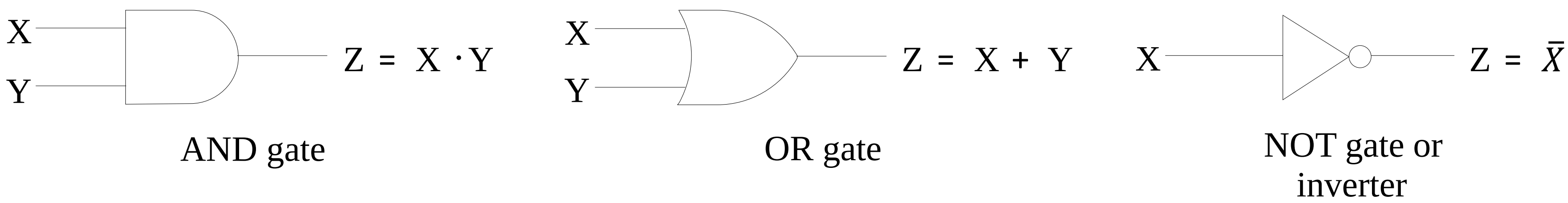
- **And waveform behavior in time as follows:**



(b) Timing diagram

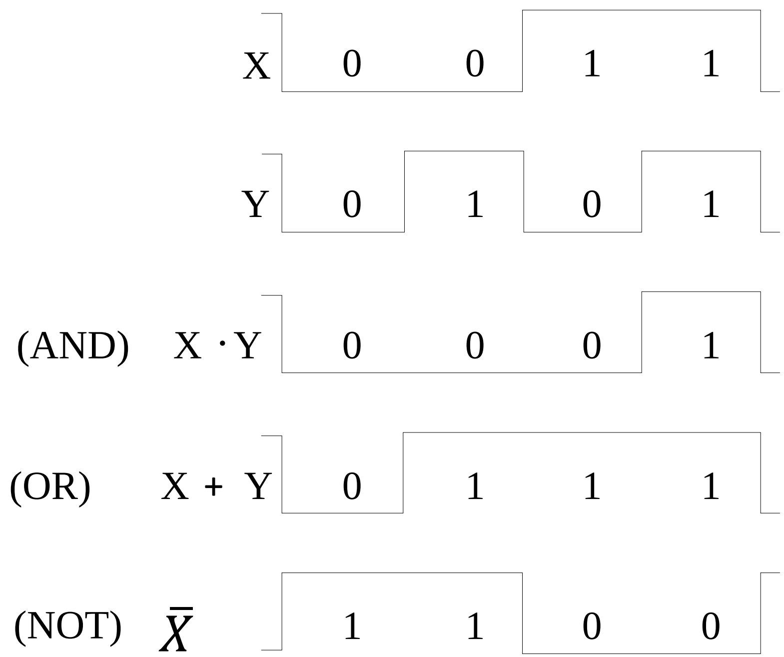
Logic Gate Symbols and Behavior

- **Logic gates have special symbols:**



(a) Graphic symbols

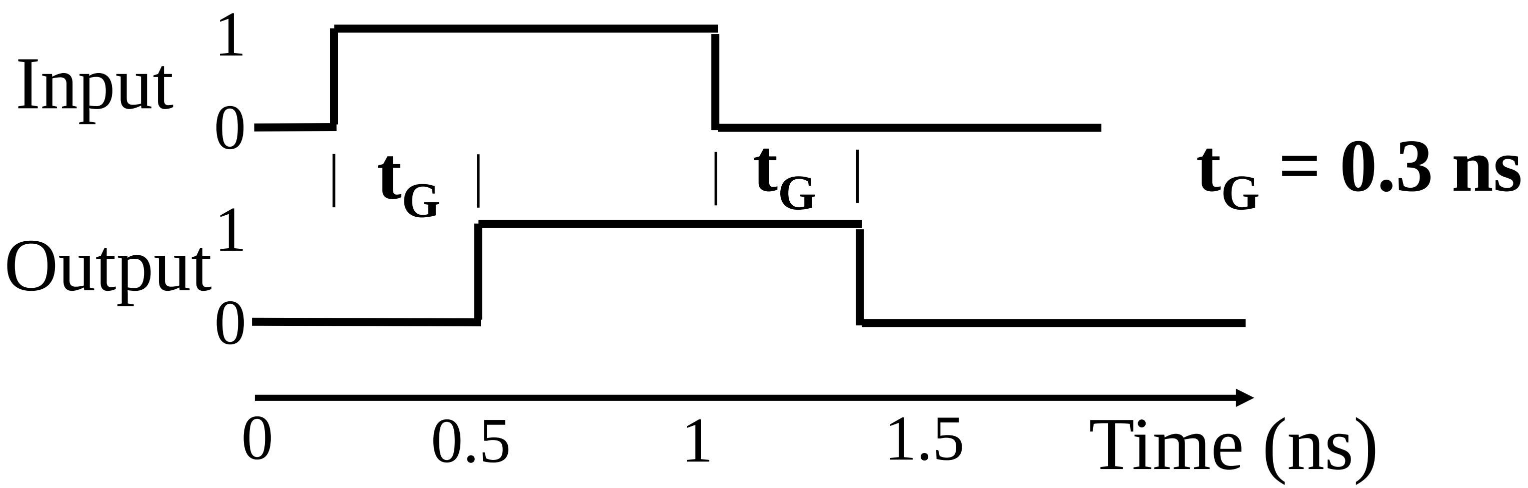
- **And waveform behavior in time as follows:**



(b) Timing diagram

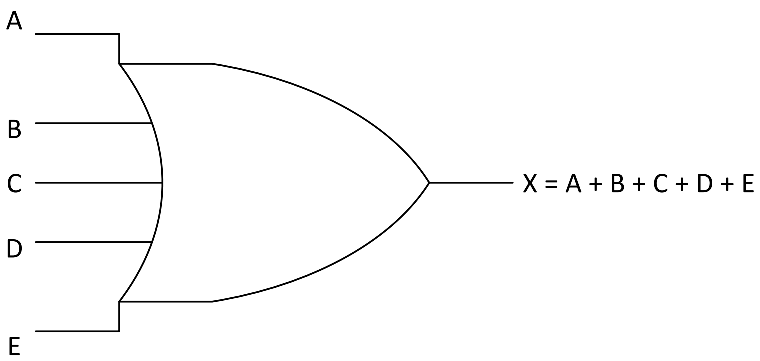
Gate Delay

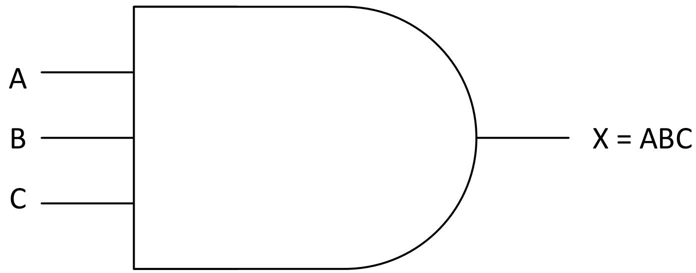
- In actual physical gates, if one or more input changes causes the output to change, the output change does not occur instantaneously
- The delay between an input change(s) and the resulting output change is the ***gate delay*** denoted by t_G:



Logic Gates: Inputs and Outputs

- NOT (inverter)
  - Always one input and one output
- AND and OR gates
  - Always one output
  - Two or more inputs





Boolean Algebra

- An algebra dealing with binary variables and logic operations
  - Variables are designated by letters of the alphabet
  - Basic logic operations: AND, OR, and NOT

- ***A Boolean expression*** is an algebraic expression formed by using **binary variables**, **constants 0 and 1**, the **logic operation symbols**, and **parentheses**
  - E.g.: X . 1, A + B + C, (A + B)(C + D)

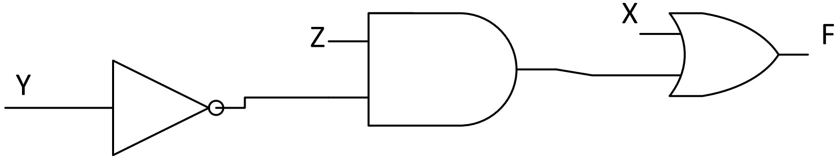
- ***A Boolean function*** consists of a binary variable identifying the function followed by equals sign and a Boolean expression
  - E.g.: $F = A + B + C, L(D, X, A) = DX + \bar{A}$

Logic Diagrams and Expressions

1. Equation: $F = X + \bar{Y}Z$
2. Logic Diagram:
3. Truth Table:

Logic Diagrams and Expressions

1. Equation: $F = X + \bar{Y}Z$
2. Logic Diagram:



3. Truth Table:

X	Y	Z	F
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

- 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.

Logic Diagrams and Expressions

1. Equation: $F = X + \bar{Y}Z$
2. Logic Diagram:
3. Truth Table:

X	Y	Z	F
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

- 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.

Example

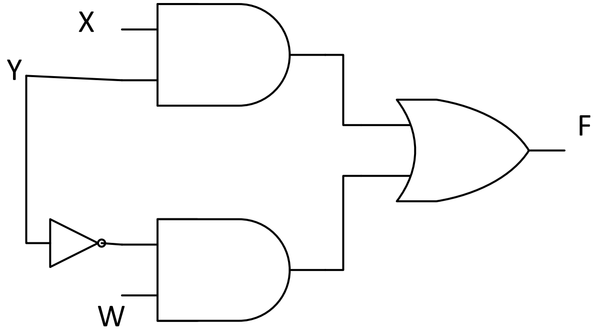
- Draw the logic diagram and the truth table of the following Boolean function: $F(W, X, Y) = XY + W\bar{Y}$
- Logic Diagram:
- Truth Table:

W	X	Y	F
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

- This example represents a ***Single Output Function***

Example

- Draw the logic diagram and the truth table of the following Boolean function: $F(W, X, Y) = XY + W\bar{Y}$
- Logic Diagram:



- Truth Table:

W	X	Y	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

- This example represents a ***Single Output Function***

Example

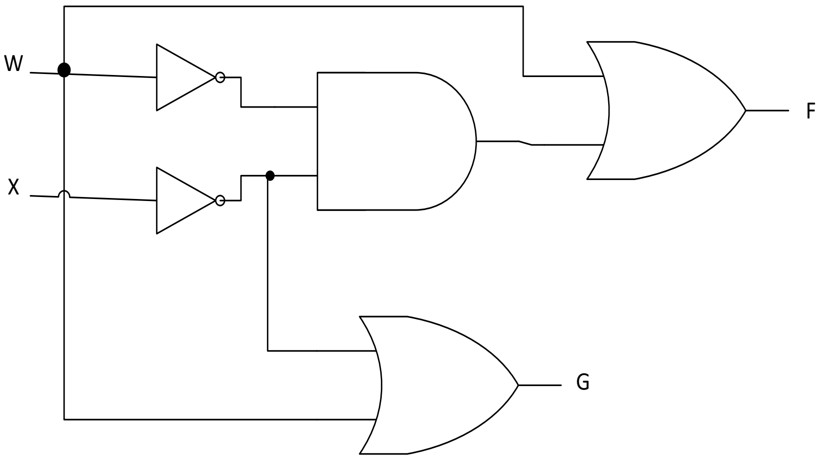
- Draw the logic diagram and the truth table of the following Boolean functions: $F(W, X) = \bar{W}\bar{X} + W, G(W, X) = W + \bar{X}$
- Logic Diagram:
- Truth Table:

W	X	F	G
0	0		
0	1		
1	0		
1	1		

Example

- Draw the logic diagram and the truth table of the following Boolean functions: $F(W,X) = \bar{W}\bar{X} + W, G(W,X) = W + \bar{X}$

- Logic Diagram:



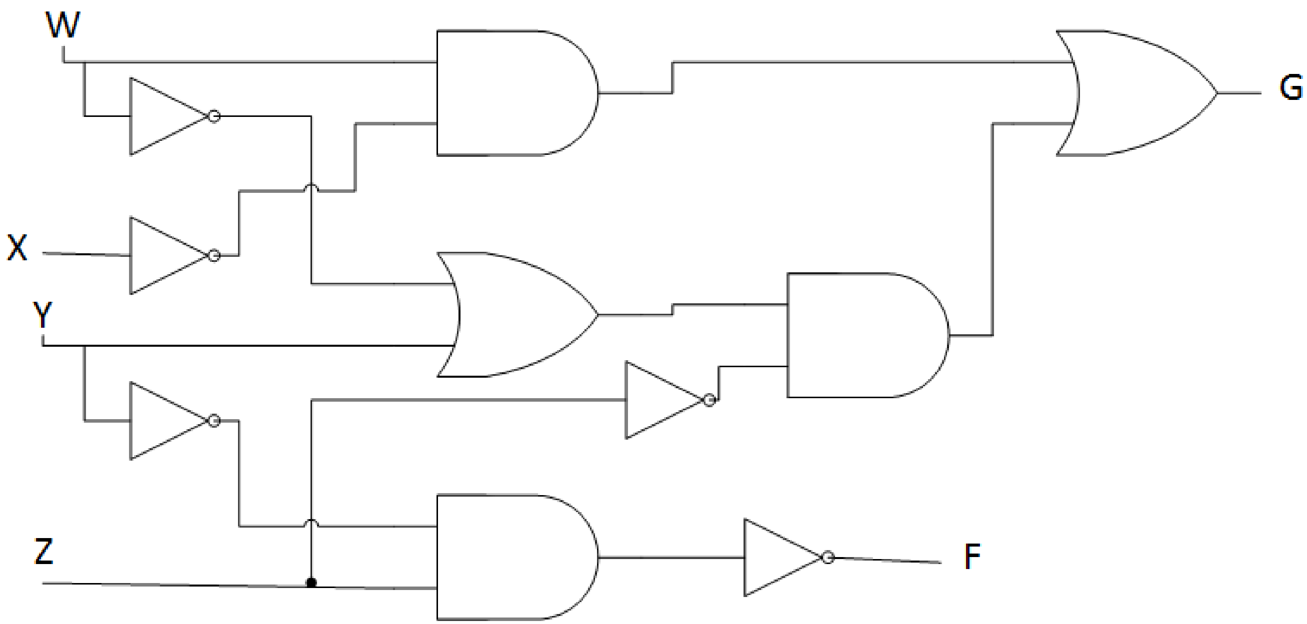
- Truth Table:

W	X	F	G
0	0	1	1
0	1	0	0
1	0	1	1
1	1	1	1

- This example represents a ***Multiple Output Function***

Example:

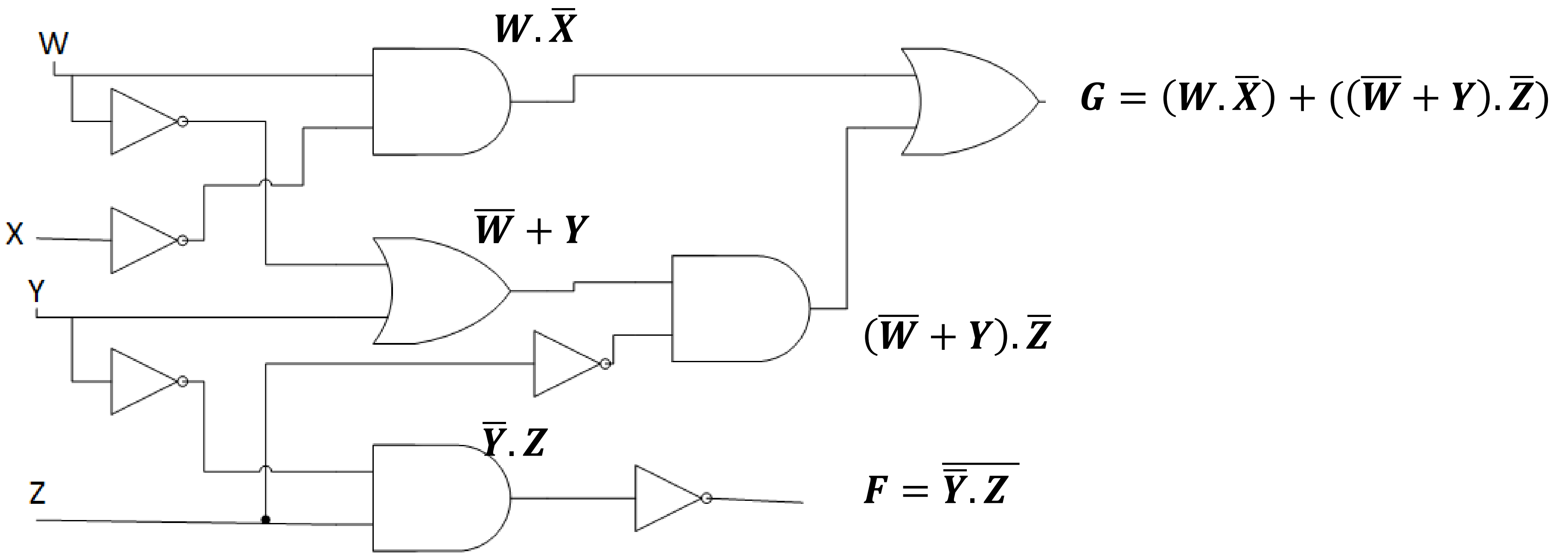
- Given the following logic diagram, write the corresponding Boolean equation:



- Logic circuits of this type are called combinational logic circuits since the variables are combined by logical operations

Example:

- Given the following logic diagram, write the corresponding Boolean equation:



- Logic circuits of this type are called combinational logic circuits since the variables are combined by logical operations

Basic Identities of Boolean Algebra

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

Some Properties of Identities & the Algebra

- If the meaning is unambiguous, we leave out the symbol "·"

- The identities above are organized into pairs
  - 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.

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
- Examples:
  - $F = (A + \bar{C}) \,.\, B + 0$
    - $Dual\ F =$
  - $G = \mathrm{XY} + (\overline{W + Z})$
    - $Dual\ G =$
  - $H = AB + AC + BC$
    - $Dual\ H =$
- Are any of these functions self-dual?
  - Yes, H is self-dual

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
- Examples:
  - $F = (A + \bar{C}) \,.\, B + 0$
    - $Dual\ F = (A \,.\, \bar{C}) + \mathrm{B} \,.\, 1 = \mathrm{A} \,.\, \bar{C} + B$ (Not Accurate)
    - $Dual\ F = \big((A \,.\, \bar{C}) + \mathrm{B}\big).\, 1 = \mathrm{A} \,.\, \bar{C} + B$ (Accurate)
  - $G = \mathrm{XY} + (\overline{W + Z})$
    - $Dual\ G = (X + Y).\overline{WZ} = (X + Y)\,.\,(\bar{W} + \bar{Z})$
  - $H = AB + AC + BC$
    - $Dual\ H = (A + B)(A + C)(B + C) = (A + BC)(B + C)$

    $= AB + AC + BC$
- Are any of these functions self-dual?
  - Yes, H is self-dual

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

- Examples:
  - $F = A(B + C)(C + \overline{D})$
  - $F = {\sim}AB = \bar{A}B$
  - $F = AB + C$
  - $F = A(B + C)$

Useful Boolean Theorems

<table>
<tr><th>Theorem</th><th>Dual</th><th>Name</th></tr>
<tr><td>$x.y + \bar{x}.y = y$</td><td>$(x + y)(\bar{x} + y) = y$</td><td>Minimization</td></tr>
<tr><td>$x + x.y = x$</td><td>$x.(x + y) = x$</td><td>Absorption</td></tr>
<tr><td>$x + \bar{x}.y = x + y$</td><td>$x.(\bar{x} + y) = x.y$</td><td>Simplification</td></tr>
<tr><td colspan="2">$x.y + \bar{x}.z + y.z = x.y + \bar{x}.z$</td><td rowspan="2">Consensus</td></tr>
<tr><td colspan="2">$(x + y)(\bar{x} + z)(y + z) = (x + y)(\bar{x} + z)$</td></tr>
</table>

Example 1: Boolean Algebraic Proof

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

Proof Steps	Justification (identity or theorem)
$A + A \cdot B$	
$= A \cdot 1 + A \cdot B$	$X = X \cdot 1$
$= A \cdot (1 + B)$	*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

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

Proof Steps	Justification (identity or theorem)
$\boldsymbol{AB + \bar{A}C + BC}$	
$\boldsymbol{= AB + \bar{A}C + 1.BC}$	$\boldsymbol{1.X = X}$
$\boldsymbol{= AB + \bar{A}C + (A + \bar{A}).BC}$	$\boldsymbol{X + \bar{X} = 1}$
$\boldsymbol{= AB + \bar{A}C + ABC + \bar{A}BC}$	***Distributive Law***
$\boldsymbol{= AB + ABC + \bar{A}C + \bar{A}BC}$	***Commutative Law***
$\boldsymbol{= AB.1 + AB.C + \bar{A}C.1 + \bar{A}C.B}$	$\boldsymbol{X.1 = X}$ ***and Commutative Law***
$\boldsymbol{= AB(1 + C) + \bar{A}C(1 + B)}$	***Distributive Law***
$\boldsymbol{= AB.1 + \bar{A}C.1}$	$\boldsymbol{1 + X = 1}$
$\boldsymbol{= AB + \bar{A}C}$	$\boldsymbol{X.1 = X}$

Proof of Simplification

- $A + \bar{A}.B = A + B$ (Simplification Theorem)

Proof Steps	Justification (identity or theorem)
$A + \bar{A}.B$	
$=(A+\bar{A})(A+B)$	*Distributive law*
$=1.(A+B)$	*Factor B out (Distributive Laws)*
$= (A + B)$	$X + \bar{X} = 1$

- $A.(\bar{A} + B) = AB$ (Simplification Theorem)

Proof Steps	Justification (identity or theorem)
$A.(\bar{A} + B)$	
$= (A.\bar{A}) + (A.B)$	*Distributive Law*
$= 0 + AB$	$X.\bar{X} = 0$
$= AB$	$X + 0 = X$

Proof of Minimization

- $A.B + \bar{A}.B = B$ (Minimization Theorem)

Proof Steps	Justification (identity or theorem)
$A.B + \bar{A}.B$	
$= B(A + \bar{A})$	*Distributive Law*
$= B.1$	$X + \bar{X} = 1$
$= B$	$X.1 = X$

- $(A + B)(\bar{A} + B) = B$ (Minimization Theorem)

Proof Steps	Justification (identity or theorem)
$(A + B)(\bar{A} + B)$	
$= B + (A.\bar{A})$	*Distributive Law*
$= B + 0$	$X.\bar{X} = 0$
$= B$	$X + 0 = X$

Proof of DeMorgan's Laws (1)

- $\overline{X+Y} = \bar{X}.\bar{Y}$ (DeMorgan's Law)
  - We will show that, $\bar{X}.\bar{Y}$, satisfies the definition of the complement of $(X + Y)$, defined as $\overline{X+Y}$ by DeMorgan's Law.
  - To show this, we need to show that $A + A' = 1$ and $A.A' = 0$ with $A = X + Y$ and $A' = X'.Y'$. This proves that $X'.Y' = \overline{X+Y}$.

- Part 1: Show $X + Y + X'.Y' = 1$

Proof of DeMorgan's Laws (1)

- $\overline{X+Y} = \bar{X}.\bar{Y}$ (DeMorgan's Law)
  - We will show that, $\bar{X}.\bar{Y}$, satisfies the definition of the complement of $(X + Y)$, defined as $\overline{X+Y}$ by DeMorgan's Law.
  - To show this, we need to show that $A + A' = 1$ and $A.A' = 0$ with $A = X + Y$ and $A' = X'.Y'$. This proves that $X'.Y' = \overline{X+Y}$.

- Part 1: Show $X + Y + X'.Y' = 1$

Proof Steps	Justification (identity or theorem)
$(X+Y) + X'.Y'$	
$= (X+Y+X')(X+Y+Y')$	*Distributive Law*
$= (1+Y)(X+1)$	$X + \bar{X} = 1$
$= 1.1$	$X + 1 = 1$
$= 1$	$X.1 = X$

Proof of DeMorgan's Laws (2)

- Part 2: Show $(X+Y).X'.Y' = 0$

- Based on the above two parts, $X'.Y' = \overline{X+Y}$
- The second DeMorgans' law is proved by duality
- Note that DeMorgan's law, given as an identity is not an axiom in the sense that it can be proved using the other identities.

Example 3: Boolean Algebraic Proofs

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

Example 3: Boolean Algebraic Proofs

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

Proof Steps	Justification (identity or theorem)
$\overline{(X+Y)}Z + X\bar{Y}$	
$= X'Y'Z + X.Y'$	*DeMorgan's law*
$= Y'(X'Z + X)$	*Distributive law*
$= Y'(X + X'Z)$	*Commutative law*
$= Y'(X + Z)$	*Simplification Theorem*

Boolean Function Evaluation

- $F_1 = xy\bar{z}$
- $F_2 = x + \bar{y}z$
- $F_3 = \bar{x}\bar{y}\bar{z} + \bar{x}yz + x\bar{y}$
- $F_4 = x\bar{y} + \bar{x}z$

x	y	z	F_1	F_2	F_3	F_4
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

Expression Simplification

- An application of Boolean algebra
- Simplify to contain the smallest number of literals (complemented and uncomplemented variables)
- Example: Simplify the following Boolean expression
  - $AB + A'CD + A'BD + A'CD' + ABCD$

Simplification Steps	Justification (identity or theorem)
$AB + A'CD + A'BD + A'CD' + ABCD$	
$= AB + ABCD + A'CD + A'CD' + A'BD$	*Commutative law*
$= AB(1 + CD) + A'C(D + D') + A'BD$	*Distributive law*
$= AB.1 + A'C.1 + A'BD$	$1 + X = 1$ *and* $X + X' = 1$
$= AB + A'C + A'BD$	$X.1 = X$
$= AB + A'BD + A'C$	*Commutative law*
$= B(A + A'D) + A'C$	*Distributive law*
$= B(A + D) + A'C$ → *5 Literals*	*Simplification Theorem*

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 $F = x'yz' + xy'z'$

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

- Example: Complement $G = (a' + bc)d' + e$

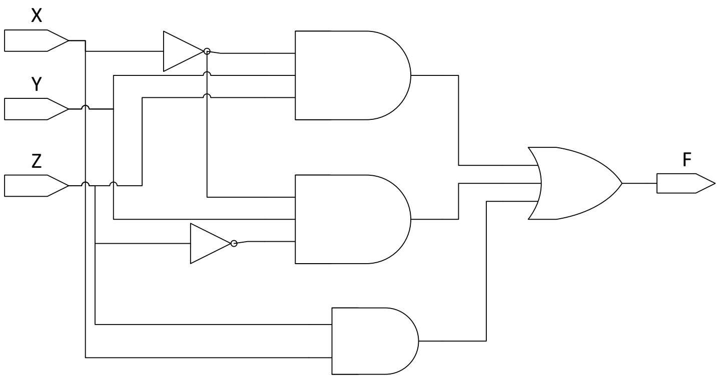
$$G' = (a(b' + c') + d).e'$$

Example

- Simplify the following:
  - $F = X'YZ + X'YZ' + XZ$

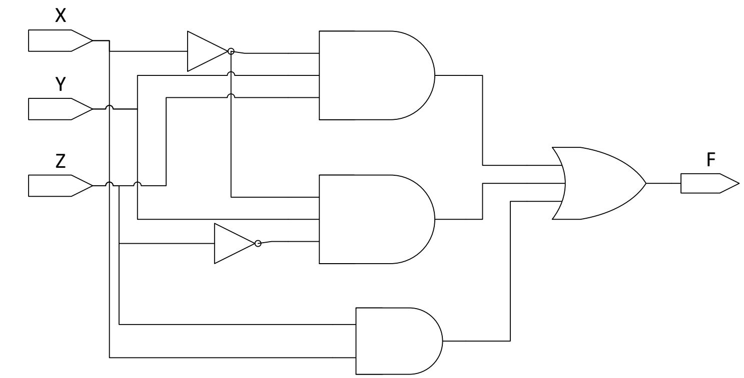
Example

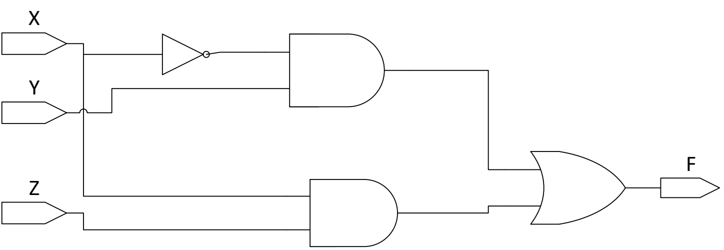
- Simplify the following:
  - $F = X'YZ + X'YZ' + XZ$



Example

- Simplify the following:
  - $F = X'YZ + X'YZ' + XZ$





Simplification Steps	(identity or theorem)
$X'YZ + X'YZ' + XZ$	
$= X'Y(Z + Z') + XZ$	*Distributive law*
$= X'Y.1 + XZ$	$X + X' = 1$
$= X'Y + XZ$	$X.1 = X$

x	y	z	$X'YZ + X'YZ' + XZ$	$X'Y + XZ$
0	0	0	0	0
0	0	1	0	0
0	1	0	1	1
0	1	1	1	1
1	0	0	0	0
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1
			3 terms and 8 literals	2 terms and 4 literals

Example

- Show that $F = x'y' + xy' + x'y + xy = 1$
  - Solution1: Truth Table

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

  - Solution2: Boolean Algebra

Proof Steps	(identity or theorem)
$x'y' + xy' + x'y + xy$	
$= y'(x' + x) + y(x' + x)$	*Distributive law*
$= y'.1 + y.1$	$X + X' = 1$
$= y' + y$	$X.1 = X$
$= 1$	$X + X' = 1$

Examples

- Show that $ABC + A'C' + AC' = AB + C'$ using Boolean algebra.

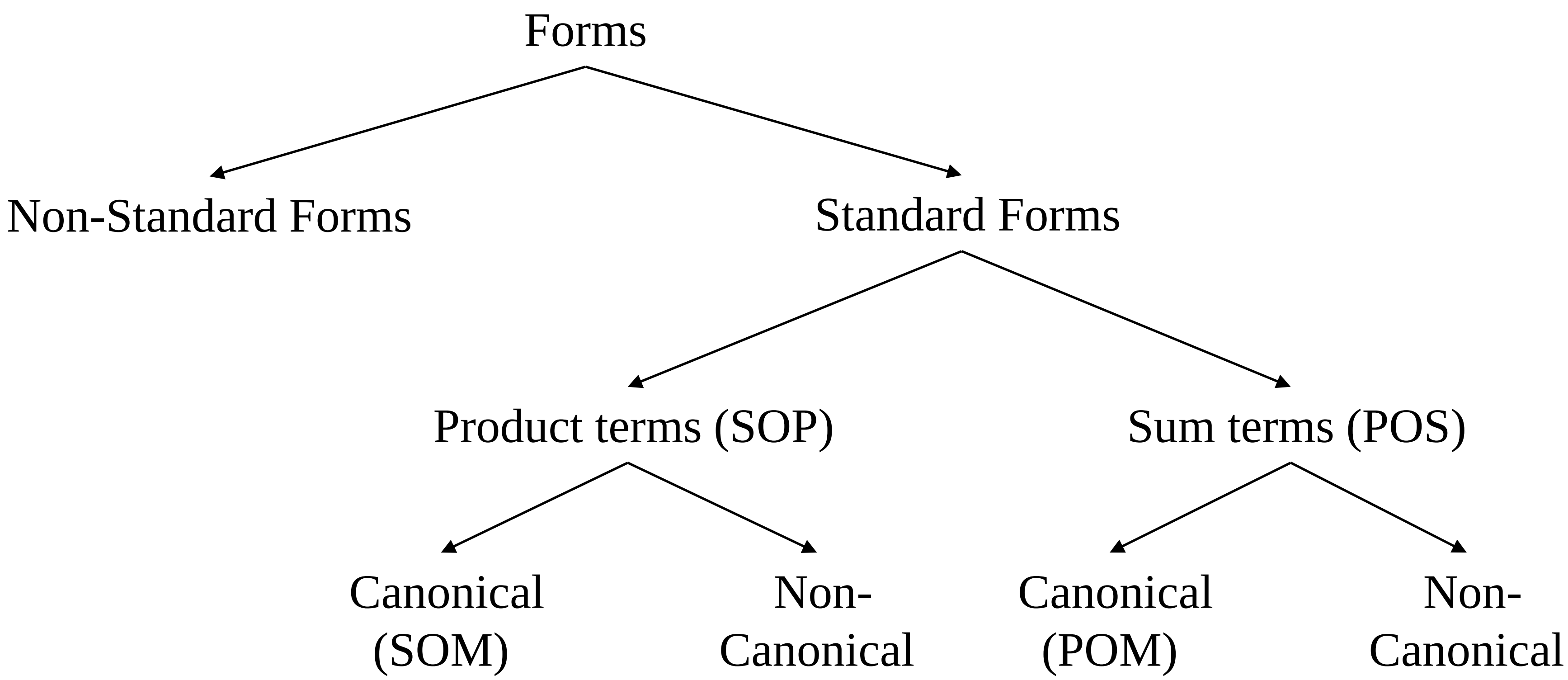
Proof Steps	(identity or theorem)
$ABC + A'C' + AC'$	
$= ABC + C'(A' + A)$	*Distributive law*
$= ABC + C'.1$	$X + X' = 1$
$= ABC + C'$	$X.1 = X$
$= (AB + C')(C + C')$	*Distributive law*
$= (AB + C').1$	$X + X' = 1$
$= AB + C'$	$X.1 = X$

- Find the dual and the complement of $f = wx + y'z.0 + w'z$
  - $Dual(f) = (w + x)(y' + z + 1)(w' + z)$
  - $f' = (w' + x')(y + z' + 1)(w + z')$

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

Boolean Representation Forms



Canonical Forms

- It is useful to specify Boolean functions in a form that:
  - Allows comparison for equality
  - Has a correspondence to the truth tables
  - Facilitates simplification
- 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., $\bar{x}$), there are 2^n minterms for n variables
- Example: Two variables (X and Y) produce $2^2 = 4$ combinations:

  XY (both normal)

  $X\bar{Y}$ (X normal, Y complemented)

  $\bar{X}Y$ (X complemented, Y normal)

  $\bar{X}\bar{Y}$ (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., $\bar{x}$), there are 2^n maxterms for n variables
- Example: Two variables (X and Y) produce $2^2 = 4$ combinations:

$X + Y$ (both normal)

$X + \bar{Y}$ (X normal, Y complemented)

$\bar{X} + Y$ (X complemented, Y normal)

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

Maxterms and Minterms

- Examples: Three variable (X, Y, Z) minterms and maxterms

Index	X,Y,Z	Minterm (m)	Maxterm (M)
0	000	$\bar{X}\bar{Y}\bar{Z}$	$X + Y + Z$
1	001	$\bar{X}\bar{Y}Z$	$X + Y + \bar{Z}$
2	010	$\bar{X}Y\bar{Z}$	$X + \bar{Y} + Z$
3	011	$\bar{X}YZ$	$X + \bar{Y} + \bar{Z}$
4	100	$X\bar{Y}\bar{Z}$	$\bar{X} + Y + Z$
5	101	$X\bar{Y}Z$	$\bar{X} + Y + \bar{Z}$
6	110	$XY\bar{Z}$	$\bar{X} + \bar{Y} + Z$
7	111	XYZ	$\bar{X} + \bar{Y} + \bar{Z}$

- **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 ***same order (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:** $a\bar{b}c, abc, \bar{a}\bar{b}c$
  - **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:**
  - **"0" means the variable is "Complemented"**
  - **"1" means the variable is "Not Complemented"**
- **For Maxterms:**
  - **"0" means the variable is "Not Complemented"**
  - **"1" means the variable is "Complemented"**

Index Example: Three Variables

Index (Decimal)	Index (Binary) n = 3 Variables	Minterm (m)	Maxterm (M)
0	000	$m_0 = \bar{X}\bar{Y}\bar{Z}$	$M_0 = X + Y + Z$
1	001	$m_1 = \bar{X}\bar{Y}Z$	$M_1 = X + Y + \bar{Z}$
2	010	$m_2 = \bar{X}Y\bar{Z}$	$M_2 = X + \bar{Y} + Z$
3	011	$m_3 = \bar{X}YZ$	$M_3 = X + \bar{Y} + \bar{Z}$
4	100	$m_4 = X\bar{Y}\bar{Z}$	$M_4 = \bar{X} + Y + Z$
5	101	$m_5 = X\bar{Y}Z$	$M_5 = \bar{X} + Y + \bar{Z}$
6	110	$m_6 = XY\bar{Z}$	$M_6 = \bar{X} + \bar{Y} + Z$
7	111	$m_7 = XYZ$	$M_7 = \bar{X} + \bar{Y} + \bar{Z}$

Index Example: Four Variables

i (Decimal)	i (Binary) n = 4 Variables	m_i	M_i
0	0000	$\bar{a}\bar{b}\bar{c}\bar{d}$	$a+b+c+d$
1	0001	$\bar{a}\bar{b}\bar{c}d$	$a+b+c+\bar{d}$
3	0011	$\bar{a}\bar{b}cd$	$a+b+\bar{c}+\bar{d}$
5	0101	$\bar{a}b\bar{c}d$	$a+\bar{b}+c+\bar{d}$
7	0111	$\bar{a}bcd$	$a+\bar{b}+\bar{c}+\bar{d}$
10	1010	$a\bar{b}c\bar{d}$	$\bar{a}+b+\bar{c}+d$
13	1101	$ab\bar{c}d$	$\bar{a}+\bar{b}+c+\bar{d}$
15	1111	$abcd$	$\bar{a}+\bar{b}+\bar{c}+\bar{d}$

Minterm and Maxterm Relationship

- Review: DeMorgan's Theorem
  - $\overline{x.y} = \bar{x} + \bar{y}$ and $\overline{x + y} = \bar{x}.\bar{y}$
- Two-variable example:
  - $M_2 = \bar{x} + y$ and $m_2 = x.\bar{y}$
  - Using DeMorgan's Theorem → $\overline{\bar{x} + y} = \bar{\bar{x}}.\bar{y} = x.\bar{y}$
  - Using DeMorgan's Theorem → $\overline{x.\bar{y}} = \bar{x} + \bar{\bar{y}} = \bar{x}.y$
  - 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:

$$M_i = \overline{m_i} \text{ and } m_i = \overline{M_i}$$

- Thus, M_i is the complement of m_i and vice-versa

Function Tables for Both

- Minterms of 2 variables:

xy	m_0	m_1	m_2	m_3
00	1	0	0	0
01	0	1	0	0
10	0	0	1	0
11	0	0	0	1

- Maxterms of 2 variables:

xy	M_0	M_1	M_2	M_3
00	0	1	1	1
01	1	0	1	1
10	1	1	0	1
11	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.
  - "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 for stating any Boolean function:
  - ***Sum of Minterms (SOM)***
  - ***Product of Maxterms (POM)***

Minterm Function Example

- Example: Find $F_1 = m_1 + m_4 + m_7$
- $F_1 \; = x'y'z + xy'z' + xyz$

xyz	Index	$m_1 + m_4 + m_7 = F_1$
000	0	$0 + 0 + 0 = 0$
001	1	$1 + 0 + 0 = 1$
010	2	$0 + 0 + 0 = 0$
011	3	$0 + 0 + 0 = 0$
100	4	$0 + 1 + 0 = 1$
101	5	$0 + 0 + 0 = 0$
110	6	$0 + 0 + 0 = 0$
111	7	$0 + 0 + 1 = 1$

Minterm Function Example

- $\boldsymbol{F(A,B,C,D,E) = m_2 + m_9 + m_{17} + m_{23}}$
- $F(A,B,C,D,E) = A'B'C'DE' + A'BC'D'E + AB'C'D'E + AB'CDE$

Maxterm Function Example

- **Example: Implement F1 in maxterms:**
- $F_1 = M_0 \,.\, M_2 \,.\, M_3 \,.\, M_5 \,.\, M_6$
- $F_1 = (x + y + z) \,.\, (x + y' + z) \,.\, (x + y' + z') \,.\, (x' + y + z') \,.\, (x' + y' + z)$

xyz	Index	$M_0 \,.\, M_2 \,.\, M_3 \,.\, M_5 \,.\, M_6 = F_1$
000	0	$0 \,.\, 1 \,.\, 1 \,.\, 1 \,.\, 1 = 0$
001	1	$1 \,.\, 1 \,.\, 1 \,.\, 1 \,.\, 1 = 1$
010	2	$1 \,.\, 0 \,.\, 1 \,.\, 1 \,.\, 1 = 0$
011	3	$1 \,.\, 1 \,.\, 0 \,.\, 1 \,.\, 1 = 0$
100	4	$1 \,.\, 1 \,.\, 1 \,.\, 1 \,.\, 1 = 1$
101	5	$1 \,.\, 1 \,.\, 1 \,.\, 0 \,.\, 1 = 0$
110	6	$1 \,.\, 1 \,.\, 1 \,.\, 1 \,.\, 0 = 0$
111	7	$1 \,.\, 1 \,.\, 1 \,.\, 1 \,.\, 1 = 1$

Maxterm Function Example

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

- $F(A,B,C,D) = (A + B + C' + D') . (A' + B + C + D) . (A' + B + C' + D') . (A' + B' + C' + D)$

Canonical Sum of Minterms

- Any Boolean function can be expressed as a Sum of Minterms (SOM):
  - 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 $(v + \bar{v})$
- Example: Implement $f = x + \bar{x}\bar{y}$ as a SOM?
  1. Expand terms → $f = x(y + \bar{y}) + \bar{x}\bar{y}$
  2. Distributive law → $f = xy + x\bar{y} + \bar{x}\bar{y}$
  3. Express as SOM → $f = m_3 + m_2 + m_0 = m_0 + m_2 + m_3$

Another SOM Example

- Example: $F = A + \bar{B}C$
- There are three variables: A, B, and C which we take to be the standard order
- Expanding the terms with missing variables:
  - $F = A(B + \bar{B})(C + \bar{C}) + (A + \bar{A})\bar{B}C$
- Distributive law:
  - $F = ABC + A\bar{B}C + AB\bar{C} + A\bar{B}\bar{C} + A\bar{B}C + \bar{A}\bar{B}C$
- Collect terms (removing all but one of duplicate terms):
  - $F = ABC + AB\bar{C} + A\bar{B}C + A\bar{B}\bar{C} + \bar{A}\bar{B}C$
- Express as SOM:
  - $F = m_7 + m_6 + m_5 + m_4 + m_1$
  - $F = m_1 + m_4 + m_5 + m_6 + m_7$

Shorthand SOM Form

- **From the previous example, we started with:**
  - $F = A + \bar{B}C$
- **We ended up with:**
  - $F = m_1 + m_4 + m_5 + m_6 + m_7$
- **This can be denoted in the *formal shorthand*:**
  - $F(A, B, C) = \sum_m(1,4,5,6,7)$
- **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 $(v \,.\, \bar{v})$ and then applying the distributive law again
- Example: Convert $f(x, y, z) = x + \bar{x}\bar{y}$ to POM?
  - Distributive law → $f = (x + \bar{x}) \,.\, (x + \bar{y}) = x + \bar{y}$
  - ORing with missing variable (z) → $f = x + \bar{y} + z \,.\, \bar{z}$
  - Distributive law → $f = (x + \bar{y} + z) \,.\, (x + \bar{y} + \bar{z})$
  - Express as POS → $f = M_2 \,.\, M_3$

Another POM Example

- Convert $f(A,B,C) = AC' + BC + A'B'$ to POM?
- Use $x + yz = (x + y) . (x + z)$, assuming $x = AC' + BC$ and $y = A'$ and $z = B'$
  - $f(A,B,C) = (AC' + BC + A') . (AC' + BC + B')$
- Use Simplification theorem to get:
  - $f(A,B,C) = (BC + A' + C') . (AC' + B' + C)$
- Use Simplification theorem again to get:
  - $f(A,B,C) = (A' + B + C') . (A + B' + C) = M_5 . M_2$
  - $f(A,B,C) = M_2 . M_5 = \prod_M(2,5)$ → ***Shorthand POM form***

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 $F(x, y, z) = \sum_m (1,3,5,7)$, find complement F as SOM and POM?
  - $\bar{F}(x, y, z) = \sum_m (0,2,4,6)$
  - $\bar{F}(x, y, z) = \prod_M (1,3,5,7)$

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: $F(x,y,z) = \sum_m(1,3,5,7)$
  - Form the Complement:

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

  - Then use the other form with the same indices – this forms the complement again, giving the other form of the original function:

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

Important Properties of Minterms

- Maxterms are seldom used directly to express Boolean functions

- Minterms properties:
  - For n Boolean variables, there are 2^n minterms (0 to 2^n -1)
  - Any Boolean function can be represented as a logical sum of minterms (SOM)
  - 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

Standard Forms

- <u>Standard Sum-of-Products (SOP) form:</u> equations are written as an OR of AND terms
- <u>Standard Product-of-Sums (POS) form:</u> equations are written as an AND of OR terms
- **Examples:**
  - SOP: $ABC + \bar{A}\bar{B}C + B$
  - POS: $(A + B) \,.\, (A + \bar{B} + \bar{C}) \,.\, C$
- **These "mixed" forms are <u>neither SOP nor POS</u>**
  - $(AB + C)(A + C)$
  - $AB\bar{C} + AC(A + B)$

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)

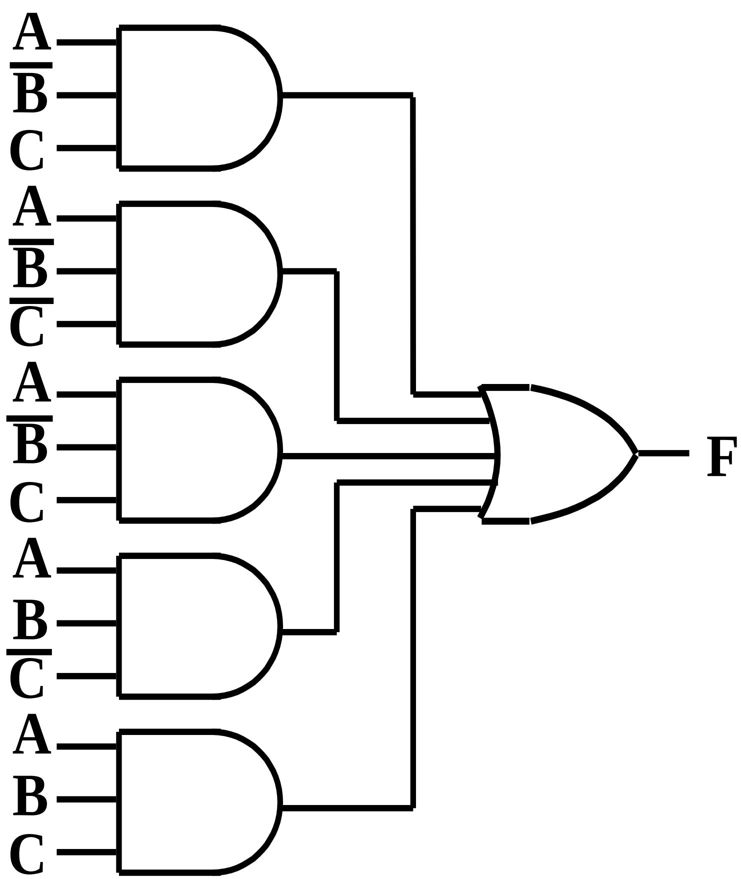
- A Simplification Example: $F(A,B,C) = \sum_m (1,4,5,6,7)$
- Writing the minterm expression:
  - $F(A,B,C) = A'B'C + AB'C' + AB'C + ABC' + ABC$
- Simplifying using boolean Algebra:

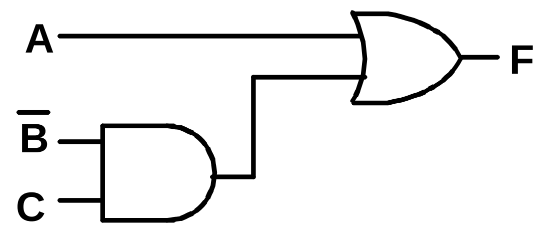
Simplification Steps	(identity or theorem)
$A'B'C + AB'C' + AB'C + ABC' + ABC$	
$= A'B'C + AB'(C' + C) + AB(C' + C)$	*Distributive law*
$= A'B'C + AB' + AB$	$X + X' = 1$
$= A'B'C + A(B' + B)$	*Distributive law*
$= A'B'C + A$	*Simplification Theorem*
$= A + B'C$	

- 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!**





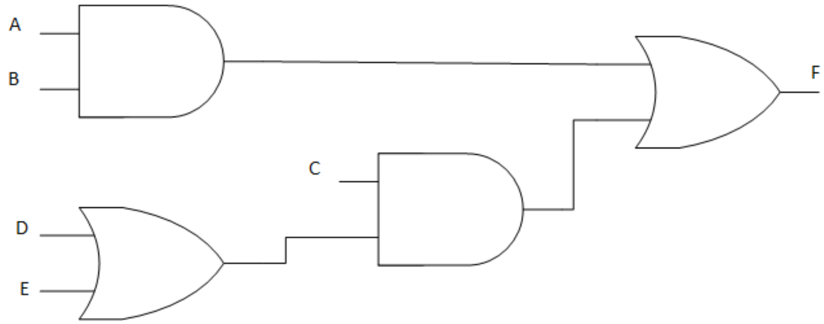
Two-level Implementation

- Draw the logic diagram of the following boolean function:
  - $f = AB + C(D + E)$

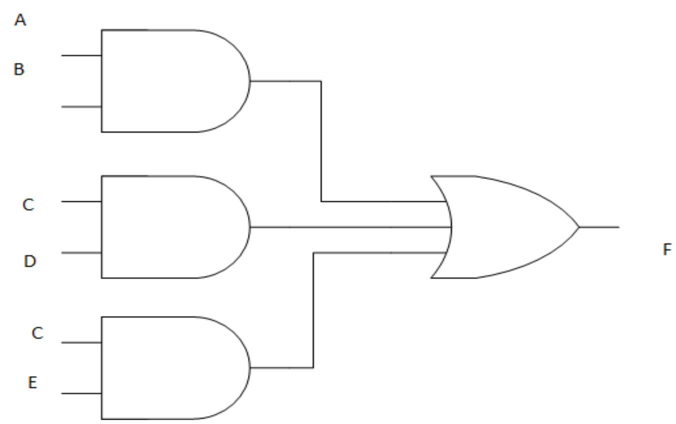
- Represent the function using two-level implementation:
  - $f = AB + CD + CE$ → SOP

Two-level Implementation

- Draw the logic diagram of the following boolean function:
  - $f = AB + C(D + E)$



- Represent the function using two-level implementation:
  - $f = AB + CD + CE$ → SOP



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.