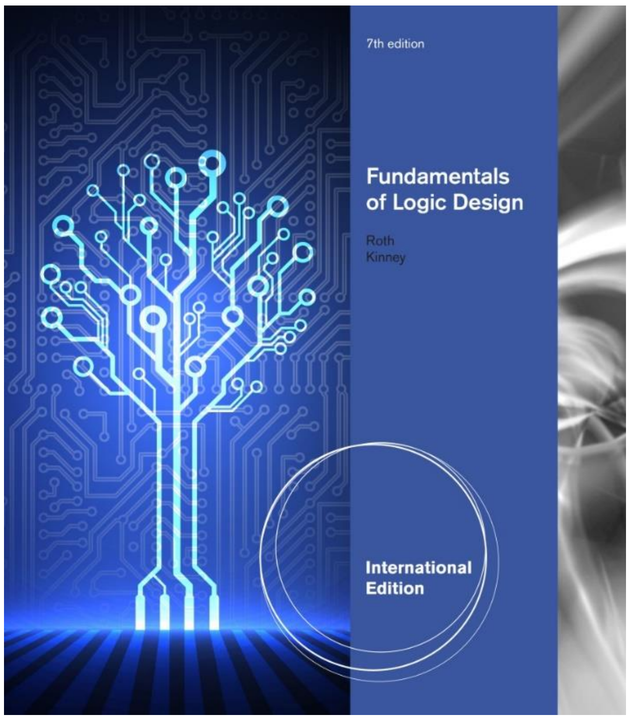



# Boolean Algebra I

성균관대학교
SUNG KYUN KWAN UNIVERSITY

# Contents

# Objectives

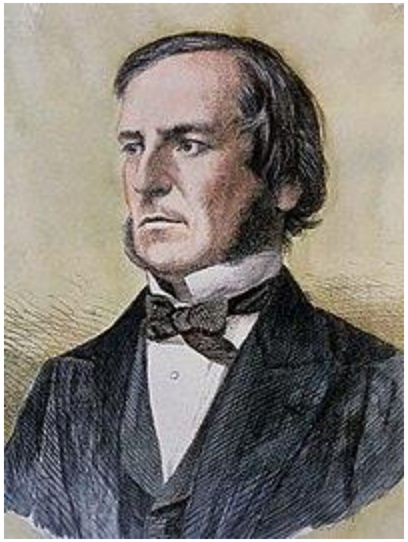
- Understand the basic operations and laws of Boolean algebra.
- Relate these operations and laws to circuits composed of AND gates, OR gates, INVERTERS and switches.
- Prove any of these laws in switching algebra using a truth table.
- Apply these laws to the manipulation of algebraic expressions including: obtaining a sum of products or product of sums, simplifying an expression and/or finding the complement of an expression

# Introduction

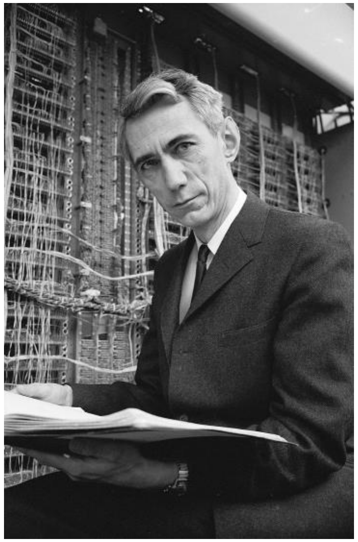
- All switching devices we will use are two-state devices, so we will emphasize the case in which all variables assume only one of two values.
- Boolean variable *X* or *Y* will be used to represent input or output of switching circuit.
- Symbols "0" and "1" represent the two values any variable can take on. These represent states in a logic circuit, and do not have numeric value.
- **Logic gate**: 0 usually represents range of low voltages and 1 represents range of high voltages
- **Switch circuit**: 0 represents open switch and 1 represents closed switch
- 0 and 1 can be used to represent the two states in any binary valued system.

# Boole/Shannon/Shockley

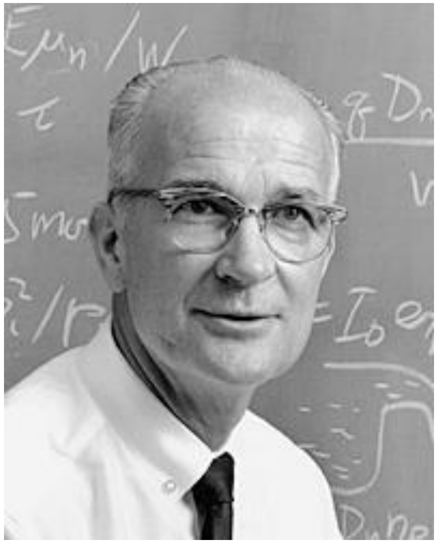
- Basic mathematics needed for logic design of digital systems is Boolean algebra.
- George Boole developed Boolean algebra in 1847 and used it to solve problems in mathematical logic.
- Two-valued Boolean algebra is often called *switching algebra*.
- Claude Shannon first applied Boolean algebra to the design of switching circuits in 1939.
- In 1947, William Shockley invented the first transistor.



George Boole (1815~1864)



Claude Shannon (1916~2001)

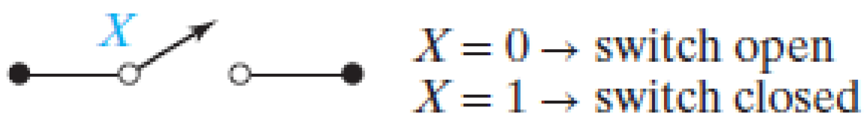


William Bradford Shockley Jr. (1910~1989)

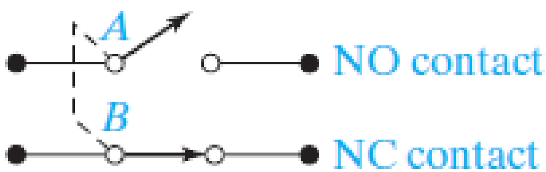
# Basic Operations

**Basic Operations:**

- The basic operations of Boolean (switching) algebra are called AND, OR, and Complement (or Inverse).
- To apply switching algebra to a switch circuit, each switch contact is labeled with a variable.




- NC (normally closed) and NO (normally open) contacts are always in opposite states.



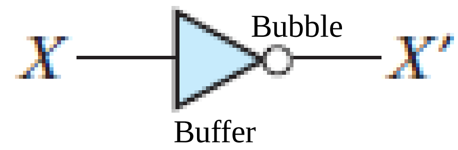

- If variable $X$ is assigned to NO contact, then $X'$ will be assigned for NC.

성균관대학교 SUNG KYUN KWAN UNIVERSITY

# Basic Operations

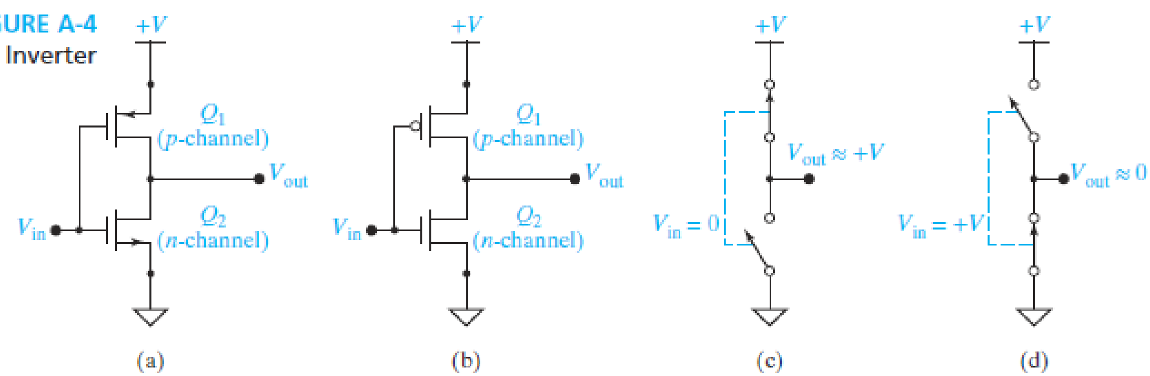
**Complementation / Inversion:**

- Prime (’) denotes complementation, i.e., 0’ = 1 and 1’ = 0
- For a switching variable, $X$:
  - ✓ $X'$ = 1 if $X$ = 0 and $X'$ = 0 if $X$ = 1’




- Complementation is also called inversion. An inverter is represented as shown below, where circle (bubble) at the output denotes inversion:
- Figure A-4(b) shows CMOS (Complementary Metal Oxide Semiconductor) inverter. The switch analog in Figure A-4(c) illustrates the operation of the inverter when the inverter input is 0. $Q_1$ is on and $Q_2$ is off as indicated by the closed and open switches. When the input is $+V$ (logic 1), $Q_1$ is off and $Q_2$ is on, as indicated by the open and closed switches in Figure A-4(d). The following table summarizes the operation:

FIGURE A-4
CMOS Inverter






| $V_{in}$ | $V_{out}$ | $Q_1$ | $Q_2$ |
|---|---|---|---|
| 0 | $+V$ | ON | OFF |
| $+V$ | 0 | OFF | ON |

# Basic Operations

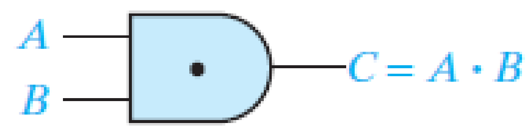
**Series Switching Circuits / AND operation:**
A) Truth table B) Logic gate diagram
C) Switch circuit diagram

- The operation defined by the table is called AND.
- It is written algebraically as $C = A \cdot B$.
- We will usually write $AB$ instead of $A \cdot B$.
- The AND operation is also referred to as logical (or Boolean) multiplication.

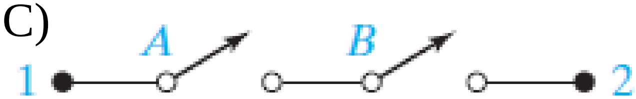
A)

| $A$ | $B$ | $C = A \cdot B$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

B)




C)




$C = 0 \rightarrow$ open circuit between terminals 1 and 2
$C = 1 \rightarrow$ closed circuit between terminals 1 and 2

- When switch contacts $A$ and $B$ are connected in series, there is an open circuit between the terminals if either $A$ or $B$ or both are open (0), and there is a closed circuit between the terminals only if both $A$ and $B$ are closed (1).

# Basic Operations

**Parallel Switching Circuits / OR operation:**
A) Truth table B) Logic gate diagram
C) Switch circuit diagram

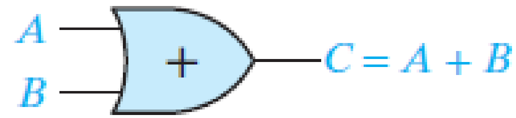
- The operation defined by the table is called OR.
- It is written algebraically as $C=A+B$.
- The OR operation is also referred to as logical (or Boolean) addition.

If switches $A$ and $B$ are connected in parallel, there is a closed circuit if either $A$ or $B$, or both, are closed and an open circuit only if $A$ and $B$ are both open.

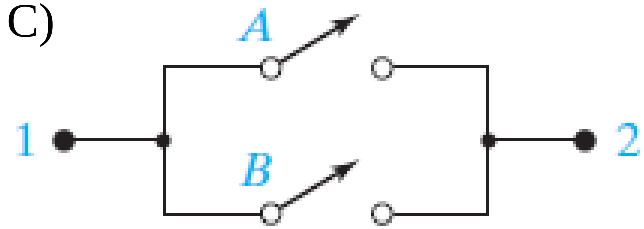
A)

| $A$ | $B$ | $C = A + B$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

B)




C)



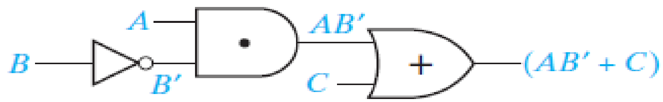
# Boolean Expressions and Truth Tables

**Examples of Boolean Expressions and Corresponding Diagrams:**

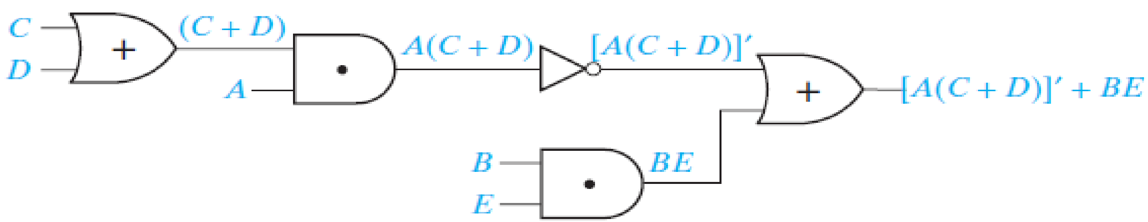
- **Expressions**

$$AB' + C \tag{2-1}$$
$$[A(C + D)]' + BE \tag{2-2}$$

- Order of operations- Parentheses, Inversion, AND, OR
- **Logic Diagrams**




(a)




(b)

- The following expression

$$ab'c + a'b + a'bc' + b'c'$$

has 3 variables ($a$, $b$, and c) and 10 literals.

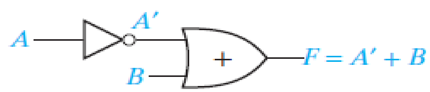
# Boolean Expressions and Truth Tables

## Truth Tables:

A truth table specifies the values of a Boolean expression for every possible combination of values of the variables in the expression.

FIGURE 2-2
Two-Input Circuit and Truth Table






(a)

| $A$ | $B$ | $A'$ | $F = A' + B$ |
|---|---|---|---|
| 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 |

(b)

## Equal Boolean Expressions:

Two Boolean expressions are said to be **equal** if they have the same value for every possible combination of the variables.

TABLE 2-1



| $A$ $B$ $C$ | $B'$ | $AB'$ | $AB' + C$ | $A + C$ | $B' + C$ | $(A + C)(B' + C)$ |
|---|---|---|---|---|---|---|
| 0 0 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 0 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 1 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 1 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 0 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 0 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 1 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 1 1 | 0 | 0 | 1 | 1 | 1 | 1 |

$$AB' + C = (A + C)(B' + C) \tag{2-3}$$

An *n*-variable expression will have $2^n$ rows in its truth table.

# Basic Theorems

**Single Variable Basic Theorems:**

- The following basic laws and theorems of Boolean algebra involve only a single variable:

Operations with 0 and 1:

| | | | |
|---|---|---|---|
| $X + 0 = X$ | (2-4) | $X \cdot 1 = X$ | (2-4D) |
| $X + 1 = 1$ | (2-5) | $X \cdot 0 = 0$ | (2-5D) |

Idempotent laws:

| | | | |
|---|---|---|---|
| $X + X = X$ | (2-6) | $X \cdot X = X$ | (2-6D) |

Involution law:

| | |
|---|---|
| $(X')' = X$ | (2-7) |

Laws of complementarity:

| | | | |
|---|---|---|---|
| $X + X' = 1$ | (2-8) | $X \cdot X' = 0$ | (2-8D) |

- Any expression can be substituted for the variable $X$ in these theorems.

$(AB+D)E + 1 = 1$ from (2-5)

$(AB'+D)(AB'+D)' = 0$ from (2-5D)

**Commutative and Associative Laws:**

**Commutative:** Order in which variables are written does not affect result of applying AND and OR operations.

$$XY = YX \qquad (2\text{-}9) \qquad \text{and} \qquad X+Y = Y+X \qquad (2\text{-}9D)$$

**Associative:** Result of AND and OR operations is independent of which variables we associate together first.

$$(XY)Z = X(YZ) = XYZ \qquad (2\text{-}10)$$

$$(X+Y)+Z = X+(Y+Z) = X+Y+Z \qquad (2\text{-}10D)$$

**Distributive Law:**

**Distributive:** The distributive law of Boolean algebra is as follows:

$$X\,(Y + Z) = XY + XZ$$

Furthermore, a second distributive law is valid for Boolean algebra but not ordinary algebra and very useful in manipulating Boolean expressions.

$$\boxed{X + YZ = (X + Y)(X + Z)}$$

Proof of a second distributed law:

$$\begin{aligned}(X + Y)(X + Z) &= X(X+Z) + Y(X+Z) = XX + XZ + YX + YZ \\ &= X + XZ + XY + YZ = X\cdot 1 + XZ + XY + YZ \\ &= X(1+Z+Y) + YZ = X\cdot 1 + YZ \\ &= X + YZ\end{aligned}$$

This second law is very useful in manipulating Boolean expressions.

# Commutative, Associative, Distributive, and DeMorgan's Law

**DeMorgan's Law:**

DeMorgan's Law is stated as follows:

$$(X + Y)' = X'Y'$$
$$(XY)' = X' + Y'$$

Truth table proof of DeMorgan's Laws is shown below:

| $X$ | $Y$ | $X'$ | $Y'$ | $X+Y$ | $(X+Y)'$ | $X'Y'$ | $XY$ | $(XY)'$ | $X'+Y'$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

# Commutative, Associative, Distributive, and DeMorgan's Law

## Laws of Boolean Algebra:

Operations with 0 and 1:

1. $X + 0 = X$ 1D. $X \cdot 1 = X$
2. $X + 1 = 1$ 2D. $X \cdot 0 = 0$

Idempotent laws:

3. $X + X = X$ 3D. $X \cdot X = X$

Involution law:

4. $(X')' = X$

Laws of complementarity:

5. $X + X' = 1$ 5D. $X \cdot X' = 0$

Commutative laws:

6. $X + Y = Y + X$ 6D. $XY = YX$

Associative laws:

7. $(X + Y) + Z = X + (Y + Z) = X + Y + Z$ 7D. $(XY)Z = X(YZ) = XYZ$

Distributive laws:

8. $X(Y + Z) = XY + XZ$ 8D. $X + YZ = (X + Y)(X + Z)$

DeMorgan's laws:

9. $(X + Y)' = X'Y'$ 9D. $(XY)' = X' + Y'$

# Simplification Theorems

**Simplification Theorems:**

- Theorems used to replace an expression with a simpler expression are called **simplification theorems**.

Uniting theorems:

1. $XY + XY' = X$ — 1D. $(X + Y)(X + Y') = X$

Absorption theorems:

2. $X + XY = X$ — 2D. $X(X + Y) = X$

Elimination theorems:

3. $X + X'Y = X + Y$ — 3D. $X(X' + Y) = XY$

Duality:

4. $(X + Y + Z + \cdots)^D = XYZ\ldots$ — 4D. $(XYZ\ldots)^D = X + Y + Z + \cdots$

Theorems for multiplying out and factoring:

5. $(X + Y)(X' + Z) = XZ + X'Y$ — 5D. $XY + X'Z = (X + Z)(X' + Y)$

Consensus theorems:

6. $XY + YZ + X'Z = XY + X'Z$ — 6D. $(X + Y)(Y + Z)(X' + Z) = (X + Y)(X' + Z)$

# Simplification Theorems

**Proof of Simplification Theorems:**

- Using switching algebra, the theorems on the previous slide can be proven using truth tables.
- In general Boolean algebra, these theorems must be proven algebraically starting with basic theorems.

Proof of (2-15): $XY + XY' = X(Y + Y') = X(1) = X$

Proof of (2-16): $X + XY = X \cdot 1 + XY = X(1 + Y) = X \cdot 1 = X$

Proof of (2-17): $X + X'Y = (X + X')(X + Y) = 1(X + Y) = X + Y$

Proof of (2-18): $XY + X'Z + YZ = XY + X'Z + (1)YZ =$

$XY + X'Z + (X + X')YZ = XY + XYZ + X'Z + X'YZ =$

$XY + X'Z$ (using absorption twice)

$(X+Y)(Y+Z)(X'+Z) = (X+Y)(XX'+Y+Z)(X'+Z) =$

$(X+Y+0)(X+Y+Z)(X'+Y+Z)(X'+0+Z) =$

$(X+Y+0 \cdot Z)(X'+0 \cdot Y+Z) = (X+Y)(X'+Z)$

# Multiplying Out and Factoring

**Sum of Product:**

An expression is said to be in *sum-of-products* (SOP) form when all products are the products of single variables. This form is the end result when an expression is fully multiplied out.

For example:

$$AB' + CD'E + AC'E'$$

$$ABC' + DEFG + H$$

**Product of Sum:**

An expression is in *product-of-sums* (POS) form when all sums are the sums of single variables. It is usually easy to recognize a product-of-sums expression since it consists of a product of sum terms.

For example:

$$(A + B')(C + D' + E)(A + C' + E')$$

$$(A + B)(C + D + E)F$$

# Multiplying Out and Factoring

## Examples: Factoring using the distributed law

**Example 1**

Factor $A + B'CD$. This is of the form $X + YZ$ where $X = A, Y = B'$, and $Z = CD$, so

$$A + B'CD = (X + Y)(X + Z) = (A + B')(A + CD)$$

$A + CD$ can be factored again using the second distributive law, so

$$A + B'CD = (A + B')(A + C)(A + D)$$

**Example 2**

Factor $AB' + C'D$.

$AB' + C'D = (AB' + C')(AB' + D)$ ← note how $X + YZ = (X + Y)(X + Z)$ was applied here

$= (A + C')(B' + C')(A + D)(B' + D)$ ← the second distributive law was applied again to each term

성균관대학교
SUNG KYUN KWAN UNIVERSITY

# Complementing Boolean Expressions

**Using DeMorgan's Laws to find Inverse Expression**

- The complement or inverse of any Boolean expression can be found using DeMorgan's Laws.
- DeMorgan's Laws for *n*-variable expressions:

$$(X_1 + X_2 + X_3 + \cdots + X_n)' = X_1{}'X_2{}'X_3{}' \ldots X_n{}' \tag{2-25}$$

$$(X_1X_2X_3 \ldots X_n)' = X_1{}' + X_2{}' + X_3{}' + \cdots + X_n{}' \tag{2-26}$$

For example, for $n = 3$,

$$(X_1 + X_2 + X_3)' = (X_1 + X_2)'X'_3 = X'_1X'_2X'_3$$

- The complement of the product is the sum of the complements.
- The complement of the sum is the product of the complements.

# Complementing Boolean Expressions

## Examples: Complementing Boolean expressions

**Example 1**

To find the complement of $(A' + B)C'$, first apply (2-13) and then (2-12).

$$[(A' + B)C']' = (A' + B)' + (C')' = AB' + C$$

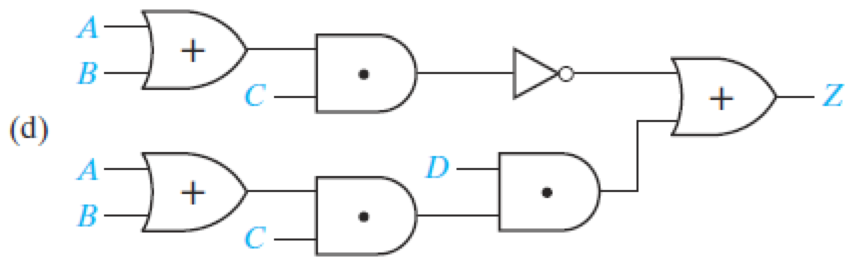
**Example 2**

$$\begin{aligned}
[(AB' + C)D' + E]' &= [(AB' + C)D']'E' && \text{(by (2-12))} \\
&= [(AB' + C)' + D]E' && \text{(by (2-13))} \\
&= [(AB')'C' + D]E' && \text{(by (2-12))} \\
&= [(A' + B)C' + D]E' && \text{(by (2-13))} \quad \text{(2-27)}
\end{aligned}$$

Note that in the final expressions, the complement operation is applied only to single variables.

# Complementing Boolean Expressions

**Example 3:** Find the output and design a simpler circuit that has the same output.




$$Z = [(A+B)C]' + (A+B)CD = [(A+B)C]' + D = A'B' + C' + D$$

(using elimination theorem $X + X'Y = X + Y$)