Boolean Algebra

Rab Nawaz Khan Jadoon

Lecturer

COMSATS Lahore

Department of Computer Science

DCS

COMSATS Institute of
Information Technology

Boolean algebra

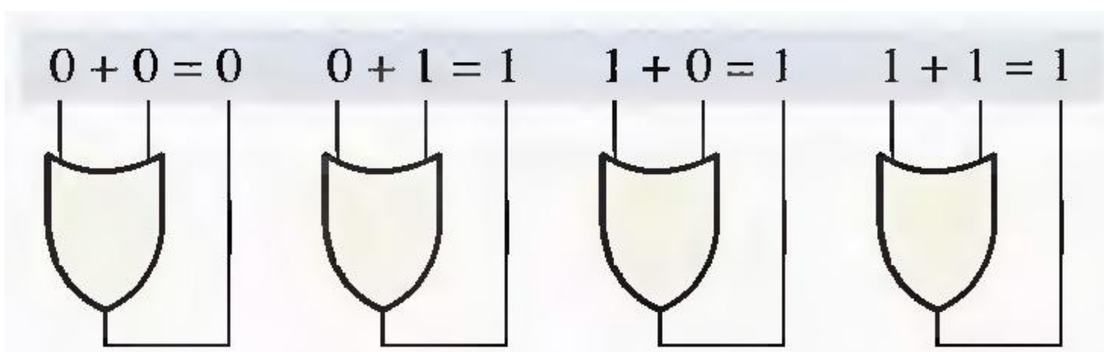
- Boolean algebra is the mathematics of digital systems.
- A basic knowledge of Boolean algebra is indispensable to the study and analysis of logic.
- In the last lecture, Boolean operations and expressions in terms of their relationship to NOT, AND, OR, NAND, and NOR gates were introduced.

- Variable, complement, and literal are terms used in Boolean algebra.
- A variable is a symbol (usually an italic uppercase letter) used to represent a logical quantity.
- Any single variable can have a 1 or a 0 value.
- The complement is the inverse of a variable and is indicated by a bar over the variable.
- A literal is a variable or the complement of a variable.

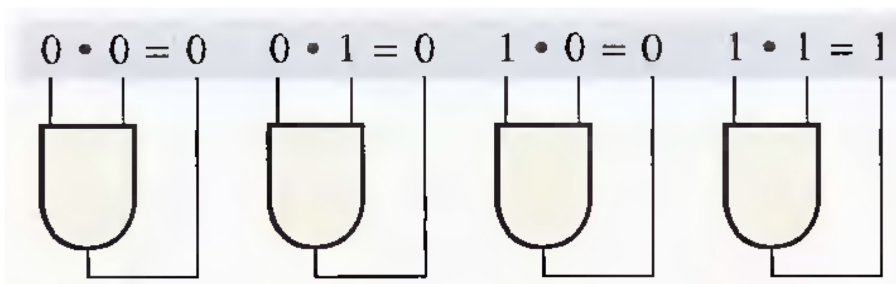
Boolean Algebra

Boolean Addition

- Boolean addition is equivalent to the OR operation.
- In Boolean algebra, a sum term is a sum of literals.
- In logic circuits, a sum term is produced by an OR operation with no AND operations involved.
- Some examples of sum terms are
  - **A + B, A + B, A + B + C, and A + B + C + D.**



- **Boolean Multiplication**
  - Boolean multiplication is equivalent to the AND operation.
  - In Boolean algebra, a product term is the product of literals.
  - A product term is equal to I only if each of the literals in the term is 1.
  - Some examples of product terms are AB, AB, ABC, and ABCD.



Laws and rules of Boolean Algebra

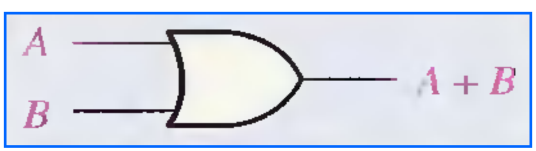
- The most important rules of BA are,
  - Apply the commutative laws of addition and multiplication
  - Apply the associative laws of addition and multiplication.
  - Apply the distributive law.
  - Apply twelve basic rules of Boolean algebra

Commutative law

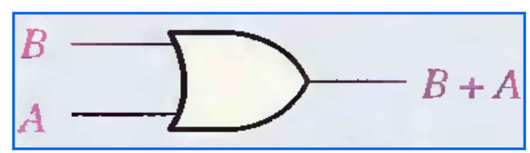
- The commutative law of addition for two variables is written as,

  $$A + B = B + A$$

  - This law states that the order in which the variables are ORed makes no difference.
  - in Boolean algebra as applied to logic circuits, addition and the OR operation are the same.
  - Figure below shows the commutative law as applied to the OR gate and shows that it doesn't matter to which input each variable is applied.



Equivalent

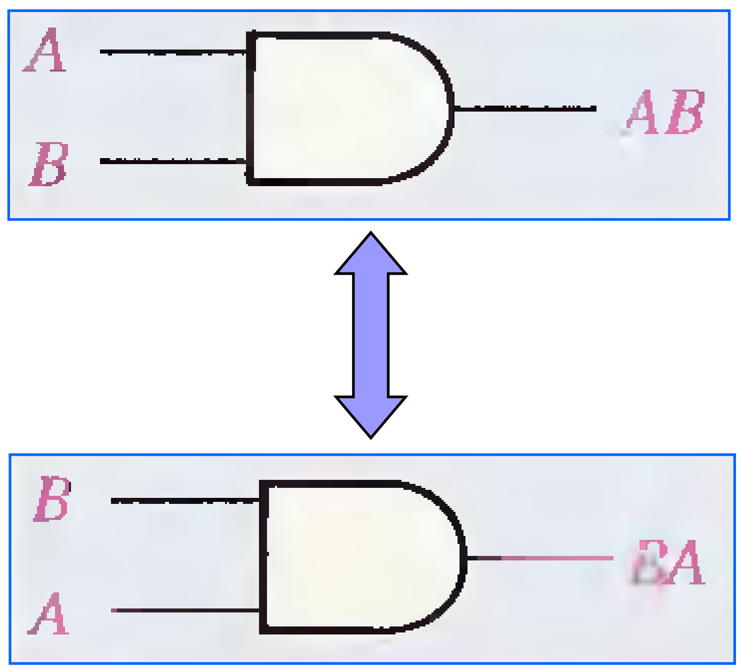


Commutative law

The commutative law of multiplication for two variables is,

$$AB = BA$$

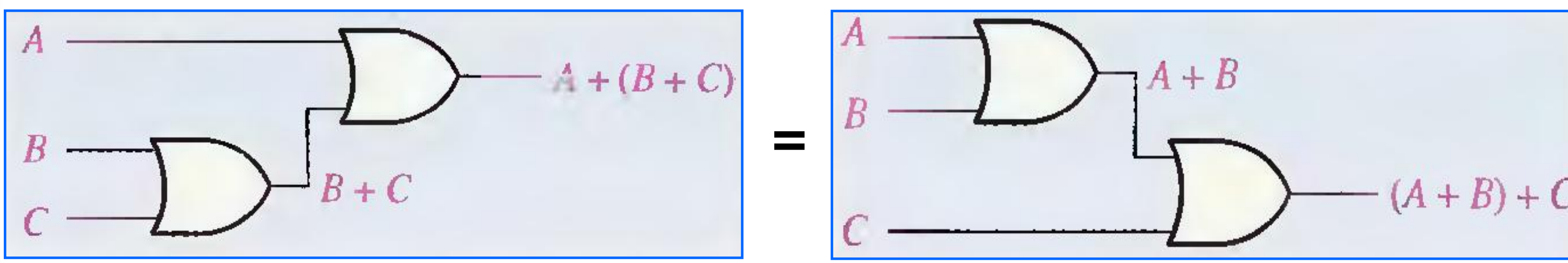
This law states that the order in which the variables are ANDed makes no difference.



- The associative law
  - The associative law of addition is written as follows for three variables:

$$A + (B + C) = (A + B) + C$$

  - This law states that when ORing more than two variables, the result is the same regardless of the grouping of the variables.

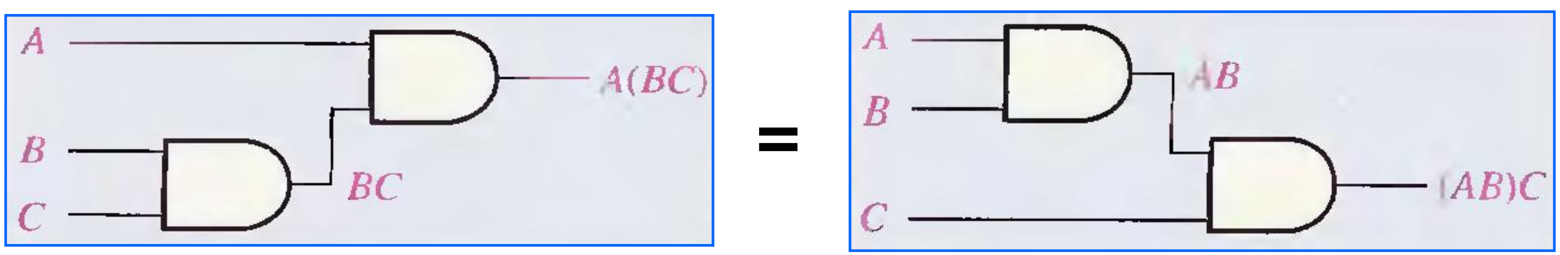


The associative law

- The associative law of multiplication is written as follows for three variables:

  A(BC) = (AB)C

- This law states that it makes no difference in what order the variables are grouped when ANDing more than two variables.

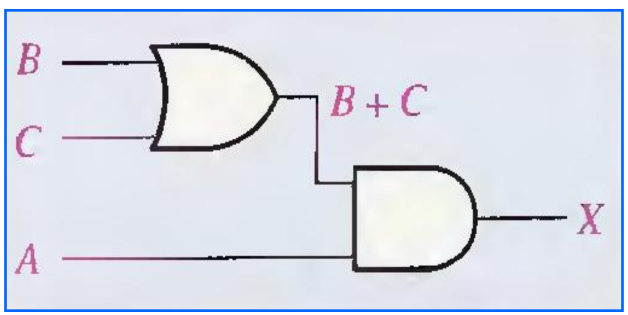


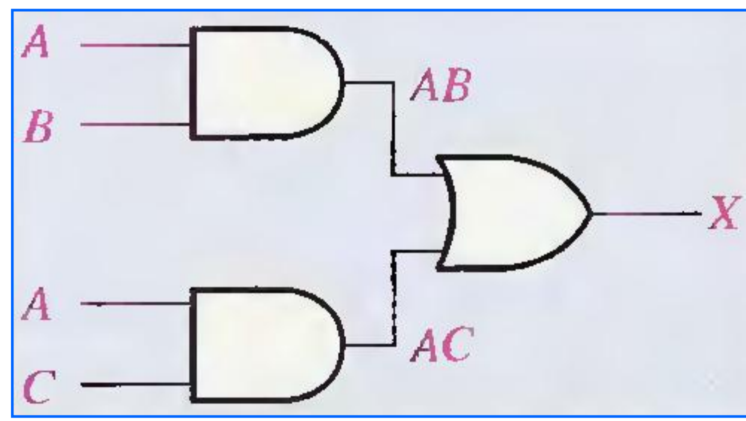
Distributive law

The distributive law is written for three variables as follows:

$$A(B + C) = AB + AC$$

This law states that ORing two or more variables and then ANDing the result with a single variable is equivalent to ANDing the single variable with each of the two or more variables and then ORing the products.





Rules for Boolean Algebra

- Table on next slide enlist 12 basic rules that are useful in manipulating and simplifying Boolean expressions.
- Rules 1 through 9 will be viewed in terms of their application to logic gates.
- Rules 10 through 12 will be derived in terms of the simpler rules and the laws previously discussed.

Rules for Boolean Algebra

1. $A + 0 = A$	7. $A \cdot A = A$
2. $A + 1 = 1$	8. $A \cdot \overline{A} = 0$
3. $A \cdot 0 = 0$	9. $\overline{\overline{A}} = A$
4. $A \cdot 1 = A$	10. $A + AB = A$
5. $A + A = A$	11. $A + \overline{A}B = A + B$
6. $A + \overline{A} = 1$	12. $(A + B)(A + C) = A + BC$

A, B, or *C* can represent a single variable or a combination of variables.

Rules for Boolean Algebra

- **Rule 10**:

$$A + AB = A$$

- This rule can be proved by applying the distributive law, rule 2, and rule 4 as follows:

Proof

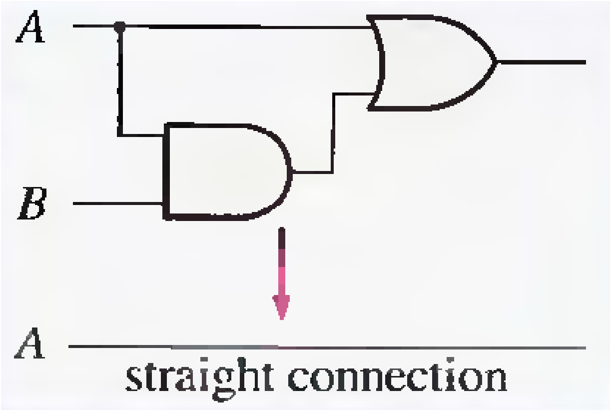
$A + AB$	$= A(1 + B)$	Factoring (distributive law)
	$= A \cdot 1$	Rule 2: $(1 + B) = 1$
	$= A$	Rule 4: $A \cdot 1 = A$

Rules for Boolean Algebra

The proof is shown in Table below, which shows the truth table and the resulting logic circuit simplification.

A	B	AB	$A + AB$
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

equal



Rule 11.

A + A’B = A + B

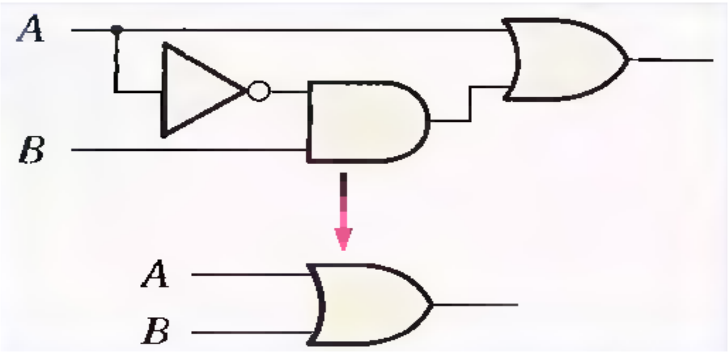
This rule can be proved as follows:

$A + \overline{A}B$	$= (A + AB) + \overline{A}B$	Rule 10: $A = A + AB$
	$= (AA + AB) + \overline{A}B$	Rule 7: $A = AA$
	$= AA + AB + A\overline{A} + \overline{A}B$	Rule 8: adding $A\overline{A} = 0$
	$= (A + \overline{A})(A + B)$	Factoring
	$= 1 \cdot (A + B)$	Rule 6: $A + \overline{A} = 1$
	$= A + B$	Rule 4: drop the 1

Example

- **The proof is shown in Table 4--3, which shows the truth table and the resulting logic circuit simplification.**

A	B	$\overline{A}B$	$A + \overline{A}B$	$A + B$
0	0	0	0	0
0	1	1	1	1
1	0	0	1	1
1	1	0	1	1
			equal	



Rules for Boolean Algebra

Rule 12.

$$(A + B)(A + C) = A + BC$$

Proof

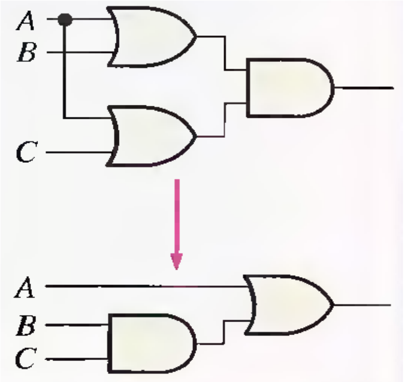
$(A + B)(A + C)$	$= AA + AC + AB + BC$	Distributive law
	$= A + AC + AB + BC$	Rule 7: $AA = A$
	$= A(1 + C) + AB + BC$	Factoring (distributive law)
	$= A \cdot 1 + AB + BC$	Rule 2: $1 + C = 1$
	$= A(1 + B) + BC$	Factoring (distributive law)
	$= A \cdot 1 + BC$	Rule 2: $1 + B = 1$
	$= A + BC$	Rule 4: $A \cdot 1 = A$

Truth Table Proof of Rule 12

A	B	C	$A+B$	$A+C$	$(A+B)(A+C)$	BC	$A+BC$
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	1	1	1	1
1	0	0	1	1	1	0	1
1	0	1	1	1	1	0	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

equal

Circuit Diagram



Examples

- Simplify the boolean functions to minimum number of literals,
  - $X+X'Y = (X+X')(X+Y) = X+Y$
  - $X(X'+Y) = XX' + XY = 0 + XY = XY$
  - $X'Y'Z + X'YZ + XY'$

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

    $= X'Z + XY'$

DEMORGAN'S THEOREMS

- One of DeMorgan's theorems is stated as follows:
  - The complement of a product of variables is equal to the sum of the complements of the variables.
- Stated another way,
  - The complement of two or more ANDed variables is equivalent to the OR of the complements of the individual variables.
- The formula for expressing this theorem for two variables is,

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

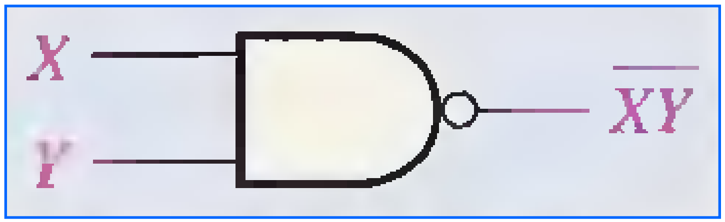
DeMorgan’s Theorem

- DeMorgan's second theorem is stated as follows:
  - The complement of a sum of variables is equal to the product of the complements of the variables.
- Stated another way,
  - The complement of two or more ORed variables is equivalent to the AND of the complements of the individual variables,
- The formula for expressing this theorem for two variables is,

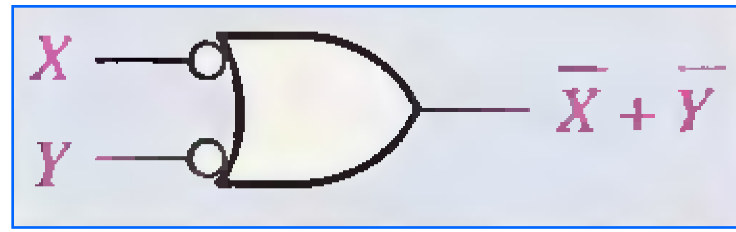
$$(X+Y)' = X'Y'$$

Demorgan’s Theorem

Gate equivalencies and truth table of both the demorgan’s Theorem is as follow,



NAND

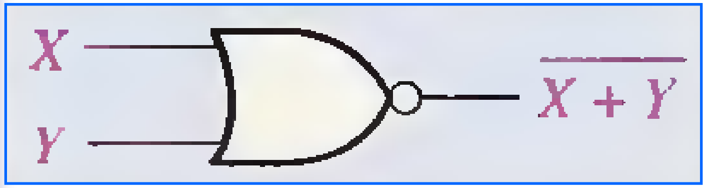


Negative-OR

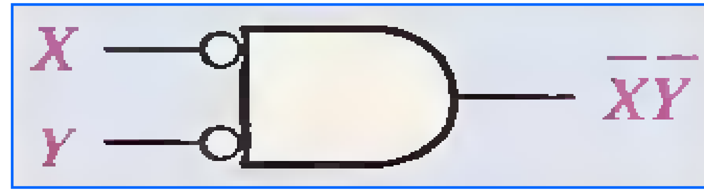
Inputs		Output	
X	Y	$\overline{XY}$	$\overline{X} + \overline{Y}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

Truth Table Proof

2nd theorem Proof



NOR



Negative-AND

Inputs		Output	
X	Y	$\overline{X + Y}$	$\overline{X}\,\overline{Y}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

Truth Table Proof

Example

- Proof the Following

  $$[(A + BC')' + D(E + F')']'$$

Proof

- **Step 1.** Identify the terms to which you ca n apply D eMorgan's theorems, and think of each term as a single term.

  Let $(A + BC')' = X$ and $D(E + F')' = Y$.

- **Step 2.** Since $(X + Y)' = X' Y'$

  $[(A + BC')' + D(E + F')']' = [((A + BC')')' (D(E + F')')']$

Example

- **Step 3.** Use rule 9 ($A'' = A$) to cancel the double bars over the left term (this is not part of DeMorgan's theorem).

  $$[((A + BC')')' (D(E + F')')'] = (A + BC') (D(E + F')')'$$

- **Step 4.** Applying DeMorgan's theorem to the second term,

  $$(A + BC') (D(E + F')')' = (A + BC')(D' + ((E + F')')')$$

- **Step 5.** Use rule 9 ($A'' = A$) to cancel the double bars over the E + F part of the term.

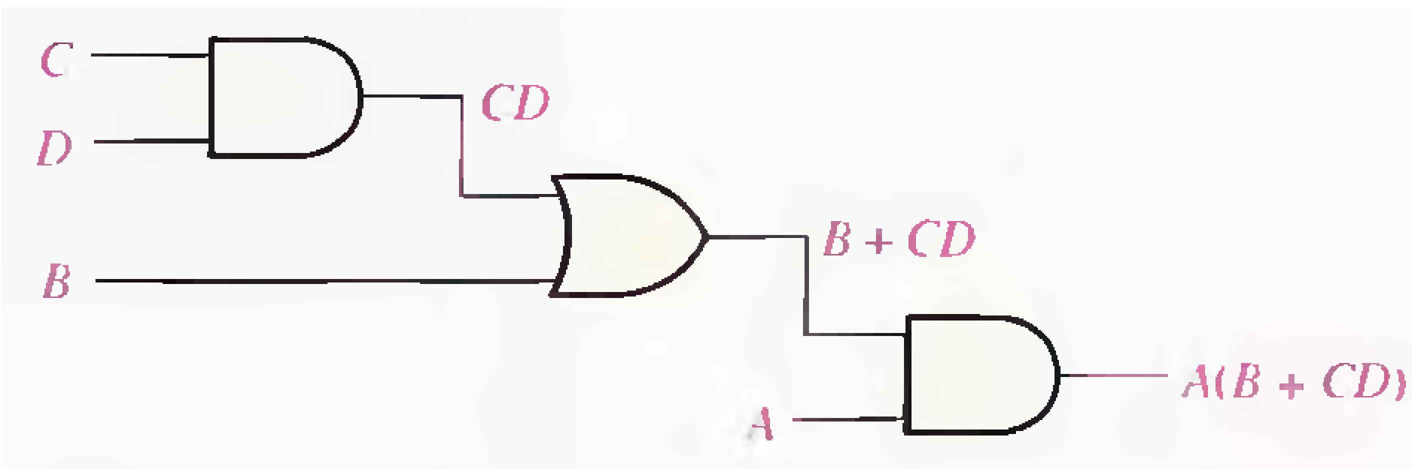
  $$(A + BC')(D' + ((E + F')')') = (A + BC')(D' + E + F')$$

BOOLEAN ANALYSIS OF LOGIC CIRCUITS

- Boolean algebra provides a concise way to express the operation of a logic circuit formed by a combination of logic gates so that the output can be determined for various combinations of input values.
  - Boolean Expression for a Logic Circuit
  - Constructing a Truth Table for a Logic Circuit

Boolean Expression for a Logic Circuit

- To derive the Boolean expression for a given logic circuit, begin at the left-most inputs and work toward the final output, writing the expression for each gate.
- For the example circuit in Figure below, the Boolean expression is determined as follows:



Boolean Expression for a Logic Circuit

- The expression for the left-most AND gate with inputs C and D is CD.
- The output of the left-most AND gate is one of the inputs to the OR gate and B is the other input. Therefore, the expression for the OR gate is B + CD.
- The output of the OR gate is one of the inputs to the right-most AND gate and A is the other input. Therefore, the expression for this AND gate is A(B + CD), which is the final output expression for the entire circuit.
  - Create the truth table for A(B+CD)

Simplification using Boolean algebra

- Many times in the application of Boolean algebra, you have to reduce a particular expression to its simplest form or change its form to a more convenient one to implement the expression most efficiently.
- The approach taken in this section is to use the basic laws, rules, and theorems of Boolean algebra to manipulate and simplify an expression.
- This method depends on a thorough knowledge of Boolean algebra and considerable practice in its application,

Example

Using Boolean algebra techniques, simplify this expression:

$$AB + A(B + C) + B(B + C)$$

Solution: The following is not necessarily the only approach.

Step 1: Apply the distributive law to the second and third terms in the expression, as follows:

$$AB + AB + AC + BB + BC$$

Step 2: Apply rule 7 (BB = B) to the fourth term.

$$AB + AB + AC + B + BC$$

Step 3: Apply rule 5 (AB + AB = AB) to the first two terms.

$$AB + AC + B + BC$$

Example cont.

Step 4: Apply rule 10 ($B + BC = B$) to the last two terms.

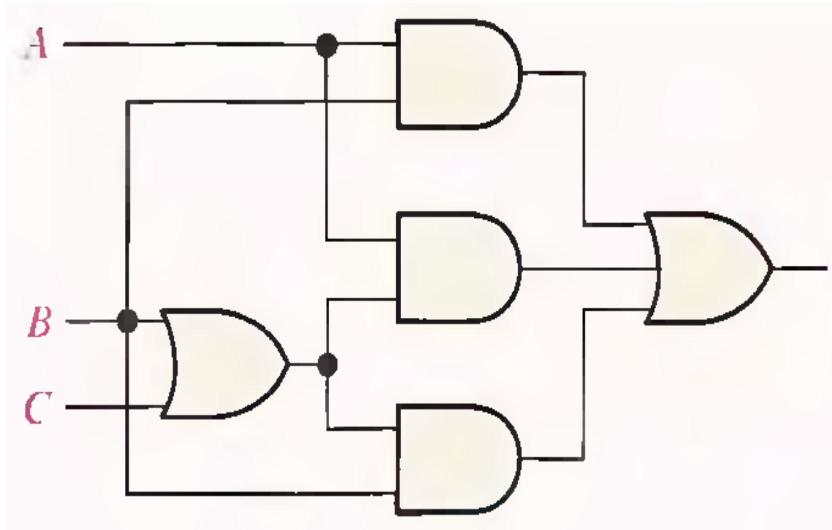
$$AB + AC + B$$

Step 5: Apply rule 10 ($AB + B = B$) to the first and third terms.

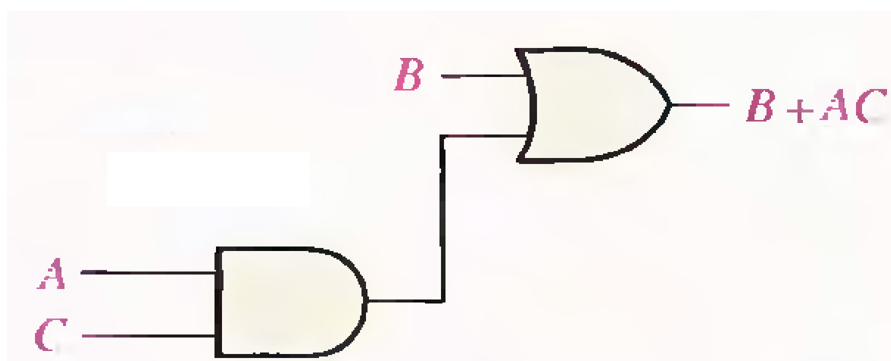
$$B+AC$$

- **At this point the expression is simplified as much as possible.**

Example Cont.



=



Example

- Simplify the following Boolean expression to minimum number of literals,

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

Solution

Step 1: Apply the distributive law to the terms within the brackets.

$$(AB'C + AB'BD + A'B')C$$

Step 2: Apply rule 8 (BB = 0) to the second term within the parentheses.

$$(AB'C + A.O.D + A'B')C$$

Step 3: Apply rule 3 (A.0.D = 0) to the second term within the parentheses.

$$(AB'C + O + A'B')C$$

Example cont.

Step 4: Apply rule 1 (drop the 0) within the parentheses.

(AB′C + A′ B′)C

Step 5: Apply the distributive law.

AB′CC + A′B′C

Step 6: Apply rule 7 (CC = C) to the first term.

AB′C + A′B′C

Step 7: Factor out B′C.

B′C(A + A′)

Step 8: Apply rule 6 (A + A′ = 1).

B′C . 1 = B′C

Example

- **Solve the expression** to minimum number of literals

$$A'BC + AB'C' + A'B'C' + AB'C + ABC$$

- **Solution:**

$A'BC + AB'C' + A'B'C' + AB'C + ABC$

$BC(A' + A) + AB'C' + A'B'C' + AB'C$

$BC.\ 1 + AB'(C' + C) + A'B'C'$

$BC + AB'.\ 1 + A'B'C'$

$BC + AB' + A'B'C'$

$BC + B'(A + A'C')$ **($A+A'C'=A+C'$ Rule11)**

$BC + B'(A + C')$

$BC + AB' + B'C'$ (Ans)

Self Questions

- Simplify the Followings if possible,
  - $(AB + AC)' + A'B'C$
  - $(AB)' + (AC)' + (ABC)'$
  - $A + AB + AB'C$
  - $(A' + B)C + ABC$
  - $AB'C(BD + CDE) + AC'$

Standard forms of Boolean expressions

- All Boolean expressions, regardless of their form, can be converted into either of two standard forms
  - **Sum-of-products (SOP) form**
    - **Also called Minterms**
  - **Product of Sum (POS) form**
    - **Also called Maxterms**
- Standardization makes the evaluation, simplification, and implementation of Boolean expressions much more systematic and easier.

Sum of product (SOP) Form

- A product term was defined as a term consisting of the product (Boolean multiplication) of literals (variables or their complements).
- When two or more product terms are summed by Boolean addition. the resulting expression is a sum-of-products (SOP).
  - Some examples are
    - AB + ABC
    - ABC + CDE + B‘CD‘
    - A‘B + A‘BC‘ + AC

SOP (Minterms)

- Also, an SOP expression can contain a single-variable like,

  A + A′BC′ + BC′D′

- **Domain of a Boolean Expression**
  - The domain of a general Boolean expression is the set of variables contained in the expression in either complemented or uncomplemented form.
  - For example, the domain of the expression A′B + AB′C is the set of variables A, B, C.

SOP Form

- Truth Table for SOP form

x	y	z	Terms	
0	0	0	$x'y'z'$	m_0
0	0	1	$x'y'z$	m_1
0	1	0	$x'y\,z'$	m_2
0	1	1	$x'y\,z$	m_3
1	0	0	$x\,y'z'$	m_4
1	0	1	$x\,y'z$	m_5
1	1	0	$x\,y\,z'$	m_6
1	1	1	$x\,y\,z$	m_7

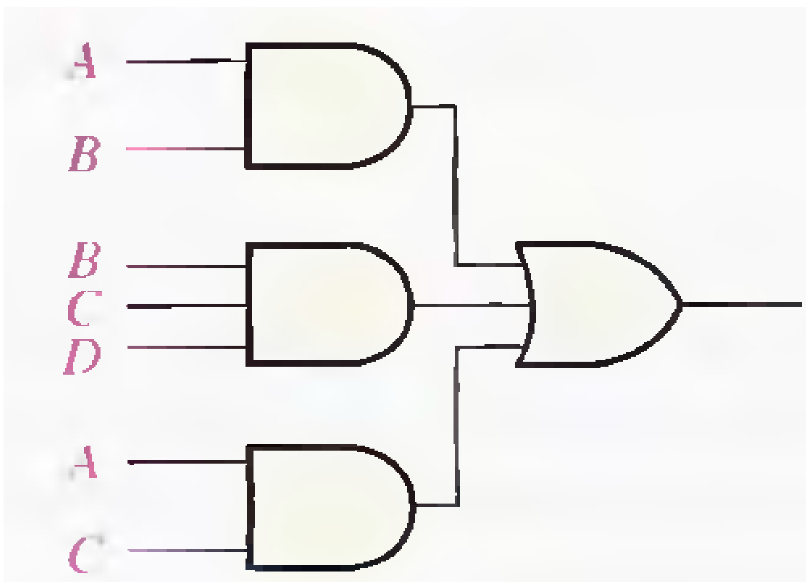
- **AND/OR Implementation of an SOP Expression**
  - Implementing an SOP expression simply requires ORing the outputs of two or more AND gates.
  - A product term is produced by an AND operation, and the sum (addition) of two or more product terms is produced by an OR operation.
  - Therefore, an SOP expression can be implemented by AND-OR logic in which the outputs of a number of AND gates connect to the inputs of an OR gate.

SOP form

- Figure below for the expression

X = AB + BCD + AC

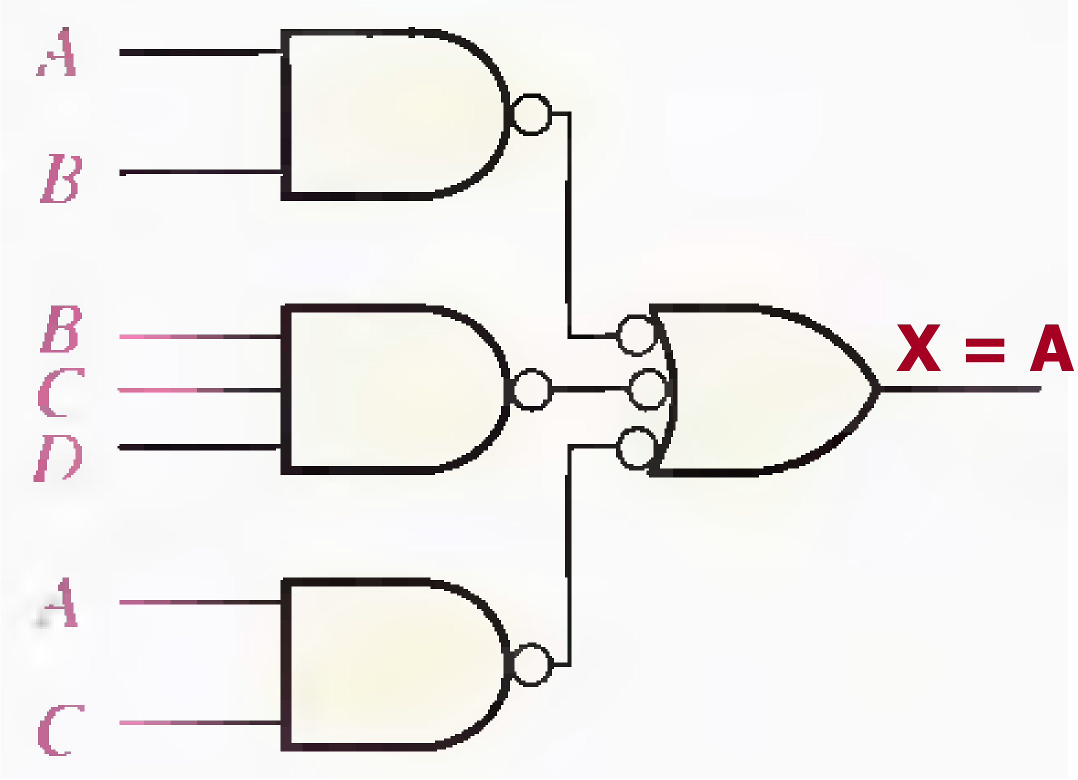
- The output X of the OR gate equals the SOP expression.



X=AB + BCD + AC.

- **NAND/NAND Implementation of an SOP Expression**
  - NAND gates can be used to implement an SOP expression.
  - Using only NAND gates, an AND/OR function can be accomplished.
  - The first level of NAND gates feed into a NAND gate that acts as a negative-OR gate.
  - The NAND and negative-OR inversions cancel and the result is effectively an AND/OR circuit.

SOP form



X = AB + BCD + AC

- **Conversion of a General Expression to SOP Form**
  - Any logic expression can be changed into SOP form by applying Boolean algebra techniques.
  - For example. the expression A(B + CD) can be converted to SOP form by applying the distributive law.

$$A(B + CD) = AB + ACD$$

Examples

- Convert each of the following Boolean expressions to SOP form,
  - $AB + B(CD + EF)$

    $= AB + BCD + BEF$

  - $(A + B)(B + C + D)$

    $= AB + AC + AD + BB + BC + BD$

  - $[(A + B)' + C]'$

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

    $= AC' + BC'$

Problem: **Convert $A'BC' + (A + B')(B + C' + AB')$ to SOP form.**

Standard SOP Form

- SOP expressions in which some of the product terms do not contain some of the variables in the domain of the expression.
- For example
  - A′ BC′ + AB′ D + A′ B C′ D
  - Domain made up of the variables A, B, C. and D.
- complete set of variables in the domain is not represented in the first two terms of the expression.
  - D or D′ is missing from the first term
  - C and C′ is missing from the second term.

SOP form

- A standard SOP expression is one in which all the variables in the domain appear in each product term in the expression.
- For example.
  - AB′CD + A′ B′ CD′ + ABC′ D′

STANDARD SOP

- Converting Product Terms to Standard SOP
  - **Step 1.** Multiply each nonstandard product term by a term made up of the sum of a missing variable and its complement. This results in two product terms. As you know, you can multiply anything by I without changing its value.
  - **Step 2.** Repeat Step I until all resulting product terms contain all variables in the domain in either complemented or uncomplemented form. In converting a product term to standard form, the number of product terms is doubled for each missing variable

Example

- Convert the following Boolean expression into standard SOP form:

  $AB'C + A'B' + ABC'D$

Solution

$$AB'C = AB'C(D + D') = AB'CD + AB'CD'$$

  - In this case, two standard product terms are the result.
  - The second term, $A'B'$, is missing variables C or C' and D or D', so first multiply the second term by $C + C'$ as follows.

Example

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

- The two resulting terms are missing variable D or D′, so multiply both terms by D + D′ as follows:

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

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

In this case, four standard product terms are the result.

The third term, ABCD, is already in standard form. The complete standard SOP form of the original expression is as follows:

Example cont.

Accumulated terms are,

$$AB'C + A'B' + ABC'D = AB'CD + AB'CD' + A'B'CD + A'B'CD' + A'B'C'D + A'B'C'D' + ABC'D$$

Related Problem

Convert the expression

$WX'Y + X'YZ' + WXY'$

to standard SOP form.

- Binary Representation of a Standard Product Term

$$AB'CD' = 1.\ 0'.\ 1\ .\ 0' = 1.1.1.1=1$$

In this case, the product term has a binary value of 1010 (decimal ten).

Note

An SOP expression is equal to 1 only if one or more of the product terms in the expression is equal to 1.

SOP form

The Product-of-Sums (POS) Form

- When two or more sum terms are multiplied, the resulting expression is a **product-of-sums (POS).**
  - $(A' + B)(A + B' + C)$
  - $(A + B' + C)(C + D' + E')(B + C' + D)$
  - $(A' + B)(A' + B' + C')(A + C')$
    - **A POS expression can contain a single-variable term, as in the following i.e. A′,**

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

POS Form

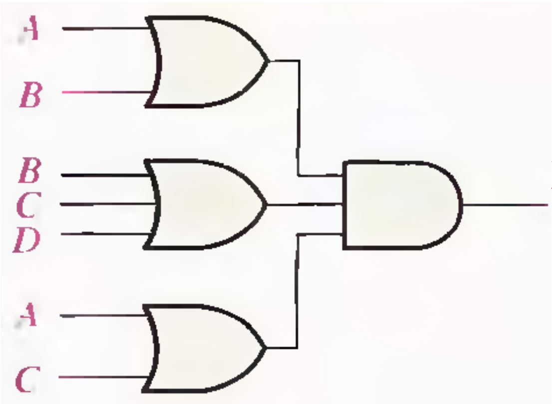
- Truth Table of POS form

Row#	A	B	C	Maxterms
0	0	0	0	$A+B+C = M_0$
1	0	0	1	$A+B+C' = M_1$
2	0	1	0	$A+B'+C = M_2$
3	0	1	1	$A+B'+C' = M_3$
4	1	0	0	$A'+B+C = M_4$
5	1	0	1	$A'+B+C' = M_5$
6	1	1	0	$A'+B'+C = M_6$
7	1	1	1	$A'+B'+C' = M_7$

POS Form

- Implementation of a POS Expression
  - Implementing a POS expression simply requires ANDing the outputs of two or more OR gates.
  - A sum term is produced by an OR operation. And the product of two or more sum terms is Produced by an AND operation.

POS Expression (A + B)(B + C + D)(A + C).



X=(A + B)(B + C + D)(A + C).

The Standard POS Form

- POS expressions in which some of the sum terms do not contain all of the variables in the domain of the expression. For example. the expression.

  $$(A + B' + C)\,(A + B + D')\,(A + B' + C' + D)$$

- It has a domain made up of the variables A. B. C, and D.
- A standard POS expression is one in which all the variables in the domain appear in each sum term in the expression. For example,

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

  is a standard POS expression.

- Converting a Sum Term to Standard POS
  - Each sum term in a POS expression that does not contain all the variables in the domain can be expanded to standard form to include all variables in the domain and their complements.
  - a non- standard POS expression is converted into standard form using Boolean algebra rule 8,

$$(A \cdot A' = 0)$$

  - A variable multiplied by its complement equals 0.

POS Form

- Rules for Conversion

Step 1: Add to each nonstandard product term a term made up of the product of the missing variable and its complement. This results in two sum terms. As you know, you can add 0 to anything without changing its value.

Step 2: Apply rule 12, $A + BC = (A + B)(A + C)$

Step 3: Repeat Step 1 until all resulting sum terms contain all variables in the domain in either complemented or uncomplemented form.

Example

- **Convert the following Boolean expression into standard POS form:**

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

Solution

- **The domain of this POS expression is A, B, C, D.**
- **The first term, A + B′ + C, is missing variable D or D′, so add DD′ and apply rule 12 as follows:**

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

- **The second term, B + C + D, is missing variable A or A, so add AA′ and apply rule 12 as follows:**

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

- **The third term, A + B′ + C′ + D, is already in standard form.**

POS Form Example

- The standard POS form of the original expression is as follows:

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

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

Related Problem

Related Problem Convert the expression $(A + B')(B + C)$ to standard POS form.

Binary Representation of POS

- Binary Representation of a Standard Sum Term
  - A standard sum term is equal to 0 for only one combination of variable values. For example, the sum term $A + B' + C + D'$ is 0 when $A = 0$, $B = 1$, $C = 0$, and $D = 1$.
  - $A+B'+C+D'=0+1'+0+1'=0+0+0+0=0$
  - In this case, the sum term has a binary value of 0101 (decimal 5).

Converting Standard SOP to Standard POS form

- Therefore, to convert from standard SOP to standard POS, the following steps are taken:

Step 1:Evaluate each product term in the SOP expression. That is, determine the binary numbers that represent the product terms.

Step 2:Determine all of the binary numbers not included in the evaluation in Step 1.

Step 3:Write the equivalent sum term for each binary number from Step 2 and express in POS form.

Using a similar procedure, you can go From POS to SOP.

EXAMPLE

- Convert the following SOP expression to an equivalent POS expression:

  $A' B' C' + A' B C' + A' BC + AB' C + ABC$

Solution

The evaluation is as follows:

000 + 010 + 011 + 101 + 111

Since there are three variables in the domain of this expression.

There are a total of eight (2^3) possible combinations.

The POS must contain the other three which are 001,100, and 1I0.

Example

- Remember, these are the binary values that make the sum term 0. The equivalent POS expression is,

$$(A + B + C')(A' + B + C)\ (A' + B' + C)$$

Converting SOP Expressions to Truth Table Format

- SOP expression is equal to 1 only if at least one of the product terms is equal to 1.
- A truth table is simply a list of the possible combinations of input variable values and the corresponding output values (1 or 0).
- For an expression with a domain of two variables, there are four different combinations of those variables ($2^2 = 4$).
- For three variable we have $2^3=8$ and so on for the higher value of variables same mechanism to be applied.

Example

- The first step in constructing a truth table is to list all possible combinations of binary values of the variables in the expression.
- Finally, place a 1 in the output column (X) for each binary value that makes the standard SOP expression a 1 and place a 0 for all the remaining binary values.
- Develop a truth table for the standard SOP expression $A' B' C + AB' C' + ABC$.

Example

- The binary values that make the product terms in the expressions equal to 1 are

  A’B’C: 001; AB’C’: 100; and ABC: 111.

INPUTS A	B	C	OUTPUT X	PRODUCT TERM
0	0	0	0	
0	0	1	1	$\overline{A}\,\overline{B}C$
0	1	0	0	
0	1	1	0	
1	0	0	1	$A\overline{B}\,\overline{C}$
1	0	1	0	
1	1	0	0	
1	1	1	1	ABC

Converting POS Expressions to Truth Table Format

- Recall that a POS expression is equal to 0 only if at least one of the sum terms is equal to 0.
- Convert the POS expression to standard form if it is not already.
- Finally, place a 0 in the output column (X) for each binary value that makes the expression a 0 and place a 1 for all the remaining binary values.

Example

- Determine the truth table for the following standard POS expression:

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

Solution

There are three variables in the domain so eight possible binary values would be,

- The binary values that make the sum terms in the expression equal to 0 are,
  - $A + B + C$: 000; $A + B' + C$: 010; $A + B' + C'$: O11; $A' + B + C'$: 101; and $A' + B' + C$: 110.

Example

- For each of these binary values, place a 0 in the output column as shown below in the table.

INPUTS A	B	C	OUTPUT X	SUM TERM
0	0	0	0	$(A + B + C)$
0	0	1	1	
0	1	0	0	$(A + \overline{B} + C)$
0	1	1	0	$(A + \overline{B} + \overline{C})$
1	0	0	1	
1	0	1	0	$(\overline{A} + B + \overline{C})$
1	1	0	0	$(\overline{A} + \overline{B} + C)$
1	1	1	1	

Problem: Develop a truth table for the following standard POS expression:

(A + B′ + C) (A + B + C′) (A′ + B′ + C′)

Determining Standard Expressions from a Truth Table

- List only the binary values of the input variables for which the output is 1.
- Convert each binary value to the corresponding product term by replacing each 1 with the corresponding variable and each 0 with the corresponding variable complement.

1010 → AB'CD'

- If you substitute, you can see that the product term is 1,   AB'CD' = 1.0.1.0 = 1.1.1.1 = 1

- To determine the standard POS expression represented by a truth table, list the binary values for which the output is O.
- For example. the binary value 1001 is converted to a sum term as follows:

  1001 ➔ A′ + B + C + D′

- If you substitute, you can see that the sum term is 0,

A′+B+C+D′=1′+0+0+1′=0+0+0+0=0

Example

- From the truth table, determine the standard SOP expression and the equivalent standard POS expression.

INPUTS			OUTPUT
A	*B*	*C*	*X*
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

Example cont…

- **Solution**
  - There are four I s in the output column and the corresponding binary values are 011, 100. 110. and 111.
  - Convert these binary values to product terms as follows:

$$
\begin{aligned}
011 &\longrightarrow \overline{A}BC \\
100 &\longrightarrow A\overline{B}\,\overline{C} \\
110 &\longrightarrow AB\overline{C} \\
111 &\longrightarrow ABC
\end{aligned}
$$

Example cont…

- The resulting standard SOP expression for the output X is,

  X = A'BC + AB' C' + ABC' + ABC

- For the POS expression, the output is 0 for binary values 000. 001, 010, and 101. Convert these binary values to sum terms as follows:

$$\begin{aligned} 000 &\longrightarrow A + B + C \\ 001 &\longrightarrow A + B + \overline{C} \\ 010 &\longrightarrow A + \overline{B} + C \\ 101 &\longrightarrow \overline{A} + B + \overline{C} \end{aligned}$$

Example cont…

- The resulting standard POS expression for the output X is,

$$X=(A+B+C)\,(A+B+C')\,(A+B'+C)\,(A'+B+C')$$

Note

Attempt the relevant exercise question at the end of this chapter.


thanks
for the
tolerance