CS221: Logic Design

Instructors:

Dr. Ahmed Shalaby http://bu.edu.eg/staff/ahmedshalaby14#

Dr. Fatma Sakr

Digital Fundamentals

CHAPTER 4

Boolean Algebra and Logic Simplification

Boolean Operations and Expressions

In Boolean algebra, a **variable** is a symbol used to represent an action, a condition, or data. A single variable can only have a value of 1 or 0.

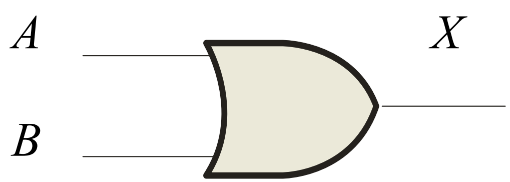
The **complement** represents the inverse of a variable and is indicated with an overbar. Thus, the complement of A is $\overline{A}$.

A **literal** is a variable or its complement.

Boolean Operations and Expressions

Boolean Addition

Addition is equivalent to the OR operation. The sum term is 1 if one or more if the literals are 1. The sum term is zero only if each literal is 0.



Inputs		Output
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

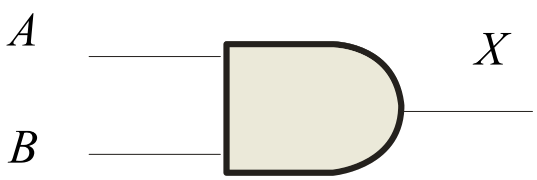
Example Determine the values of A, B, and C that make the sum term of the expression $\overline{A} + B + \overline{C} = 0$?

Solution Each literal must = 0; therefore $A = 1$, $B = 0$ and $C = 1$.

Boolean Operations and Expressions

Boolean Multiplication

In Boolean algebra, multiplication is equivalent to the AND operation. The product of literals forms a product term. The product term will be 1 only if all of the literals are 1.



Inputs		Output
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

Example What are the values of the A, B and C if the product term of $A \cdot \overline{B} \cdot \overline{C} = 1$?

Solution Each literal must = 1; therefore $A = 1$, $B = 0$ and $C = 0$.

Laws and Rules of Boolean Algebra

Laws Boolean Algebra

- Commutative Laws
- Associative Laws
- Distributive Law

Laws Boolean Algebra

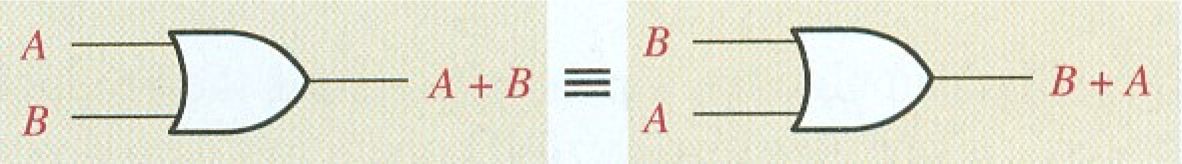
Commutative Laws

The **commutative laws** are applied to addition and multiplication.

- **For addition, the commutative law states**

  In terms of the result, the order in which variables are ORed makes no difference.

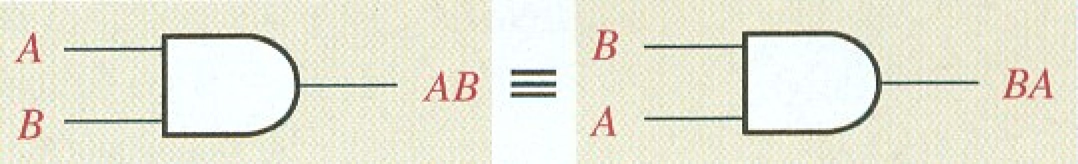
$$A + B = B + A$$



- **For multiplication, the commutative law states**

  In terms of the result, the order in which variables are ANDed makes no difference.

$$AB = BA$$



Laws Boolean Algebra

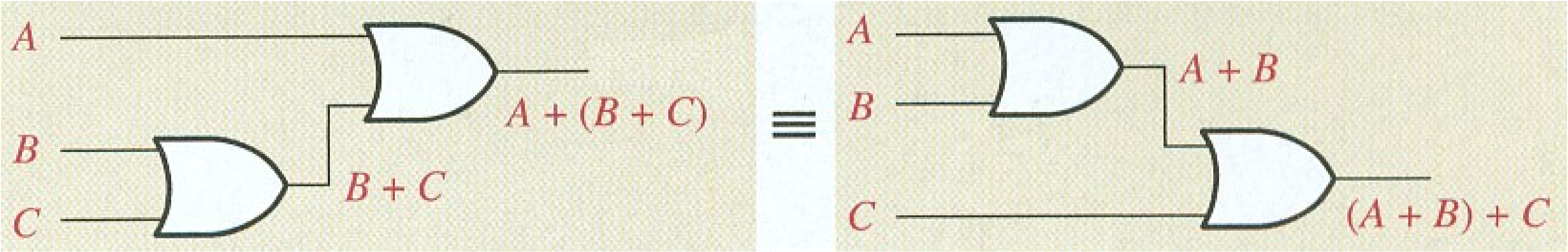
Associative Laws

The **associative laws** are also applied to addition and multiplication.

- **For addition, the associative law states**

  When ORing more than two variables, the result is the same regardless of the grouping of the variables.

$$\boxed{A + (B + C) = (A + B) + C}$$



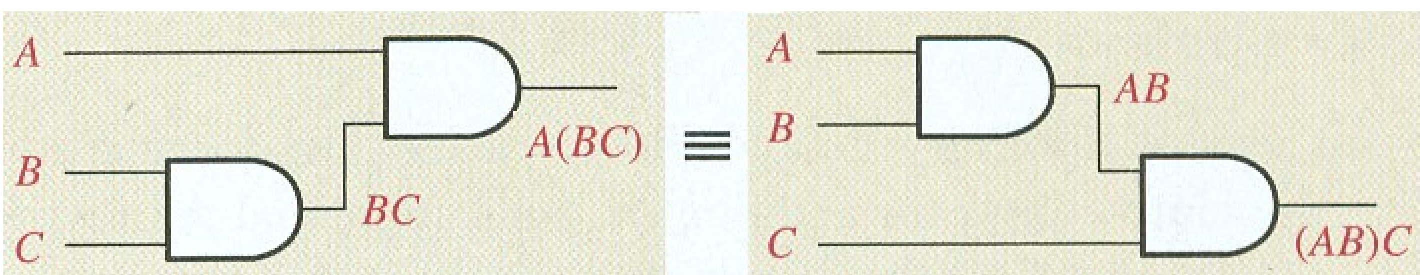
Laws Boolean Algebra

Associative Laws

- **For multiplication, the associative law states**

  When ANDing more than two variables, the result is the same regardless of the grouping of the variables.

$$A(BC) = (AB)C$$

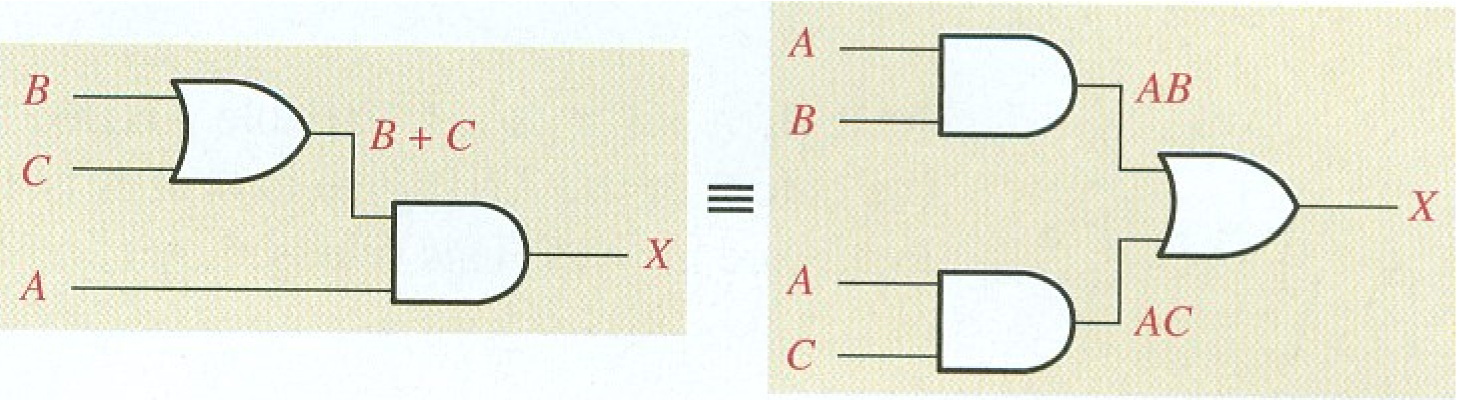


Laws Boolean Algebra

Distributive Law

The **distributive law** is the factoring law. A common variable can be factored from an expression just as in ordinary algebra. That is

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



$X = A(B + C)$ $\quad$ $X = AB + AC$

Rules of Boolean Algebra

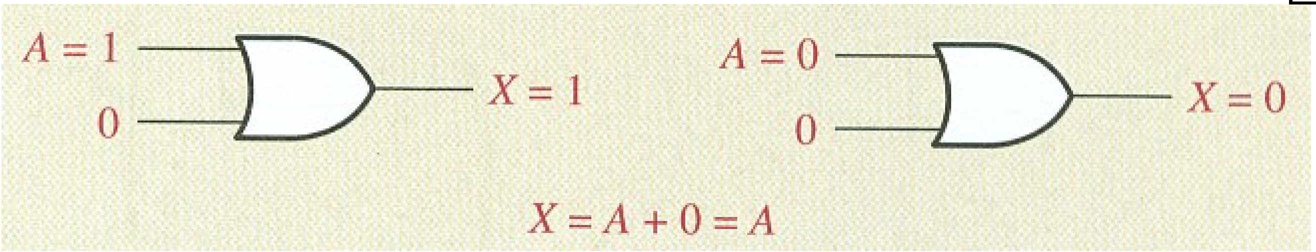
1. $A + 0 = A$
2. $A + 1 = 1$
3. $A \cdot 0 = 0$
4. $A \cdot 1 = A$
5. $A + A = A$
6. $A + \overline{A} = 1$
7. $A \cdot A = A$
8. $A \cdot \overline{A} = 0$
9. $\overline{\overline{A}} = A$
10. $A + AB = A$
11. $A + \overline{A}B = A + B$
12. $(A + B)(A + C) = A + BC$

Rules of Boolean Algebra

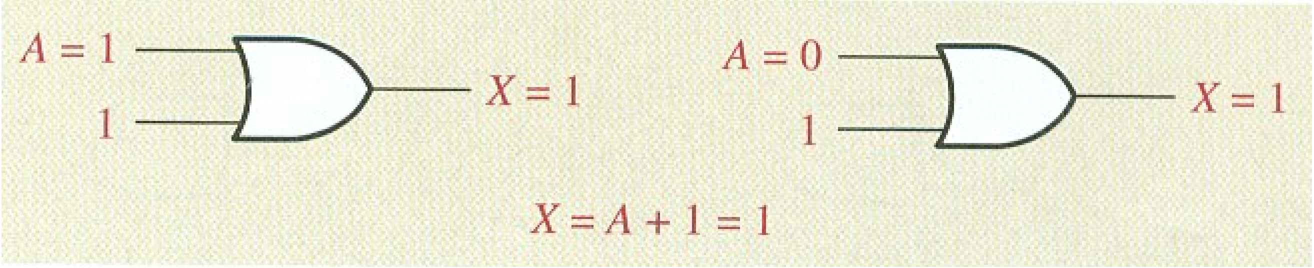
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

OR Truth Table

1. $A + 0 = A$



2. $A + 1 = 1$

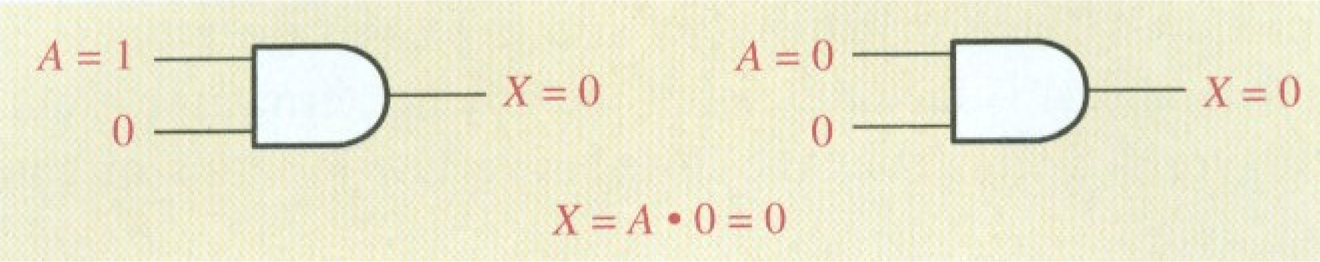


Rules of Boolean Algebra

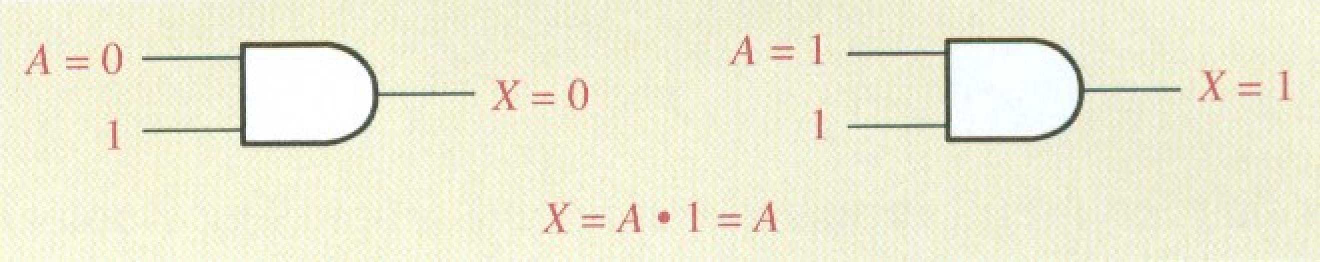
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

AND Truth Table

3. $A \cdot 0 = 0$



4. $A \cdot 1 = A$

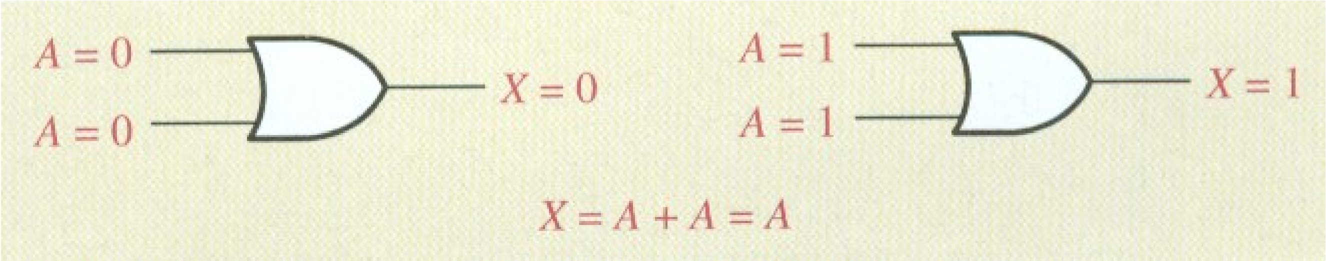


Rules of Boolean Algebra

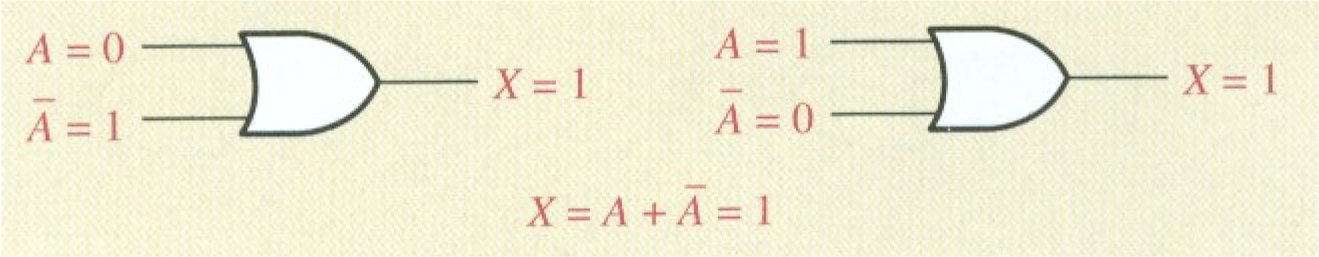
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

OR Truth Table

5. $A + A = A$



6. $A + \overline{A} = 1$

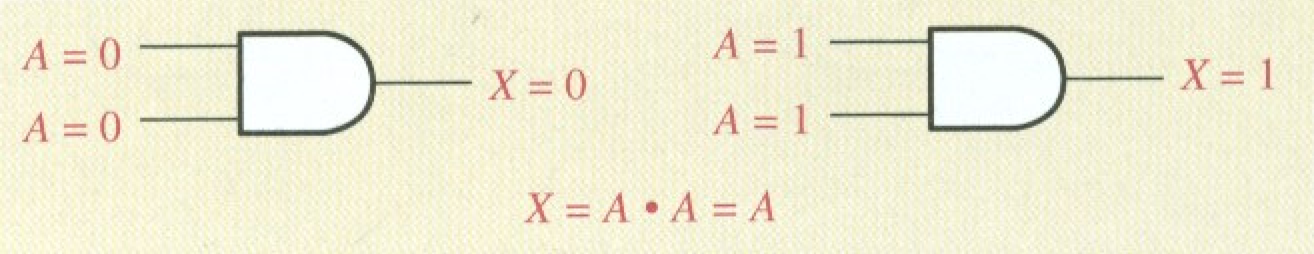


Rules of Boolean Algebra

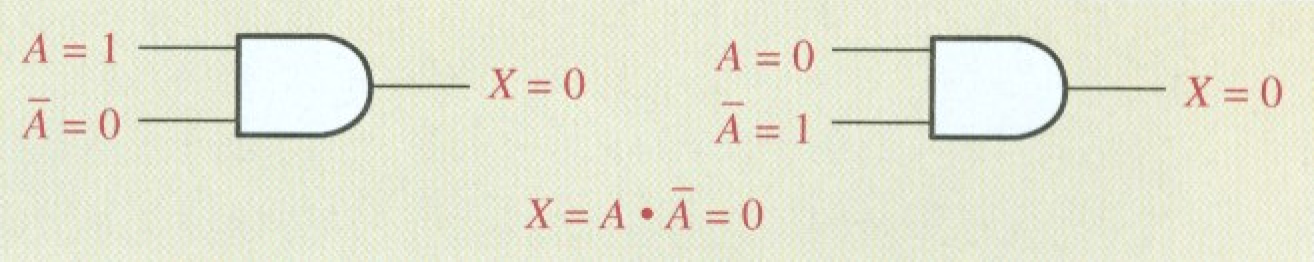
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

AND Truth Table

7. $A \cdot A = A$

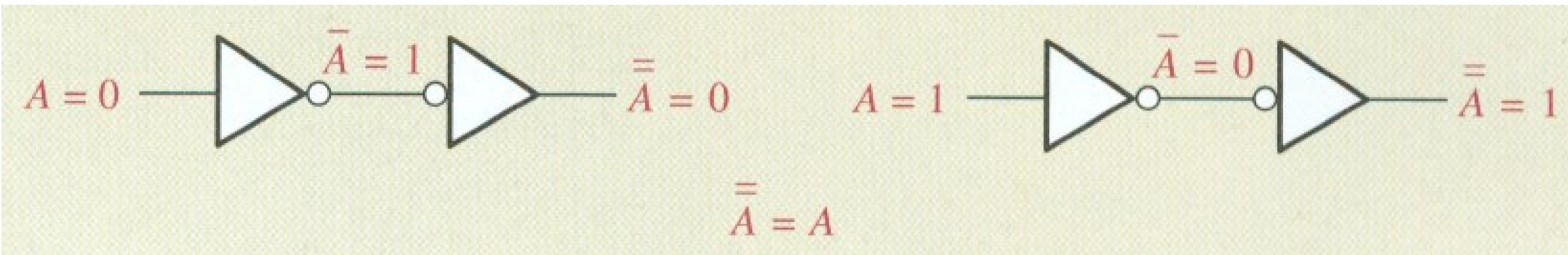


8. $A \cdot \overline{A} = 0$



Rules of Boolean Algebra

9. $\overline{\overline{A}} = A$

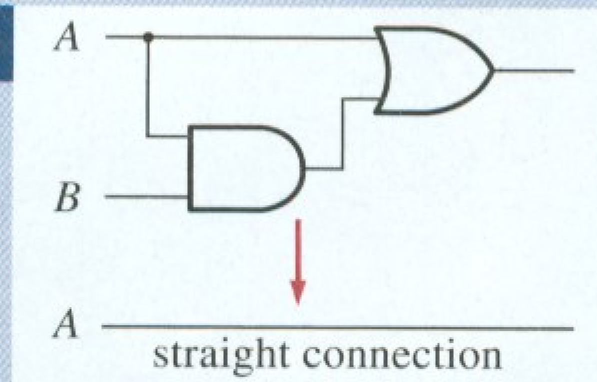


Rules of Boolean Algebra

10. $A + AB = A$

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

equal



A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

AND Truth Table

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

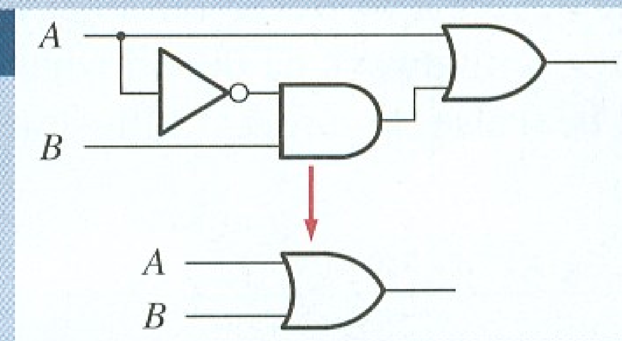
OR Truth Table

Rules of Boolean Algebra

11. $A + \overline{A}B = A + B$

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



A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

AND Truth Table

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

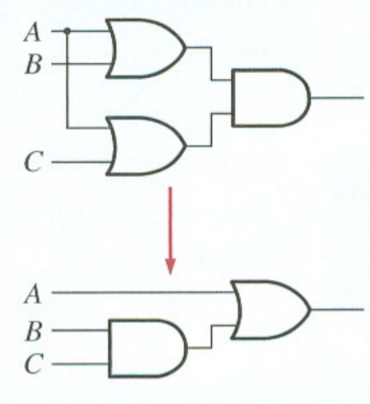
OR Truth Table

Rules of Boolean Algebra

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

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



A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

AND Truth Table

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

OR Truth Table

DeMorgan's Theorem

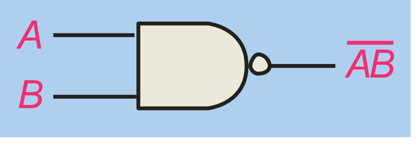
DeMorgan's Theorem

DeMorgan's 1st Theorem

The complement of a product of variables is equal to the sum of the complemented variables.

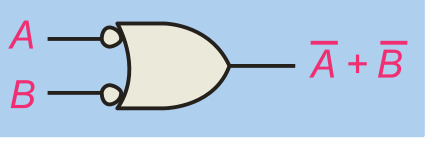
$$\overline{AB} = \overline{A} + \overline{B}$$

Applying DeMorgan's first theorem to gates:



NAND

≡



Negative-OR

Inputs		Output	
A	B	$\overline{AB}$	$\overline{A} + \overline{B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

DeMorgan's Theorem

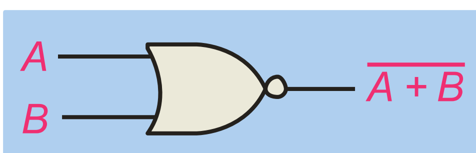
DeMorgan's Theorem

DeMorgan's 2nd Theorem

The complement of a sum of variables is equal to the product of the complemented variables.

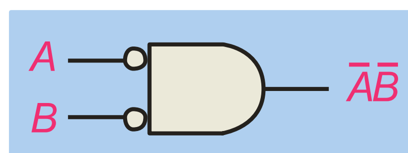
$$\overline{A + B} = \overline{A} \cdot \overline{B}$$

Applying DeMorgan's second theorem to gates:



NOR

≡



Negative-AND

Inputs		Output	
A	B	$\overline{A+B}$	$\overline{A}\overline{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

DeMorgan's Theorem

DeMorgan's Theorem

- Theorem 1 $\overline{XY} = \overline{X} + \overline{Y}$
- Theorem 2 $\overline{X + Y} = \overline{X}\,\overline{Y}$

Example Apply DeMorgan's theorem to remove the overbar covering both terms from the expression $X = \overline{\overline{C} + D}$.

Solution To apply DeMorgan's theorem to the expression, you can break the overbar covering both terms and change the sign between the terms. This results in $X = \overline{\overline{C}} \cdot \overline{D}$. Deleting the double bar gives $X = C \cdot \overline{D}$.

Boolean Analysis of Logic Circuits

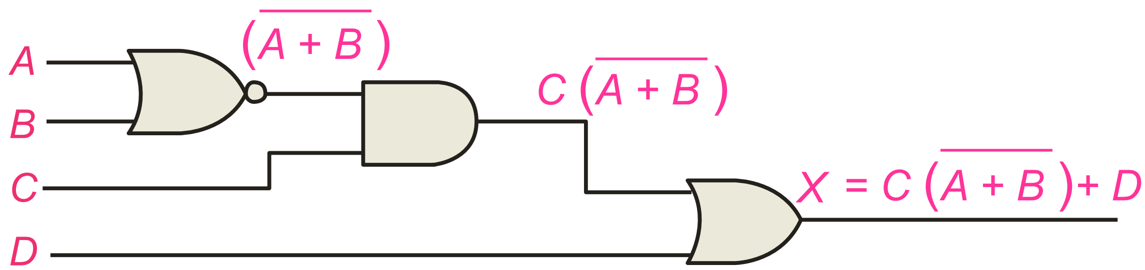
Boolean Analysis of Logic Circuits

Boolean Analysis of Logic Circuits

Combinational logic circuits can be analyzed by writing the expression for each gate and combining the expressions according to the rules for Boolean algebra.

Example Apply Boolean algebra to derive the expression for X.

Solution Write the expression for each gate:



Applying DeMorgan's theorem and the distribution law:

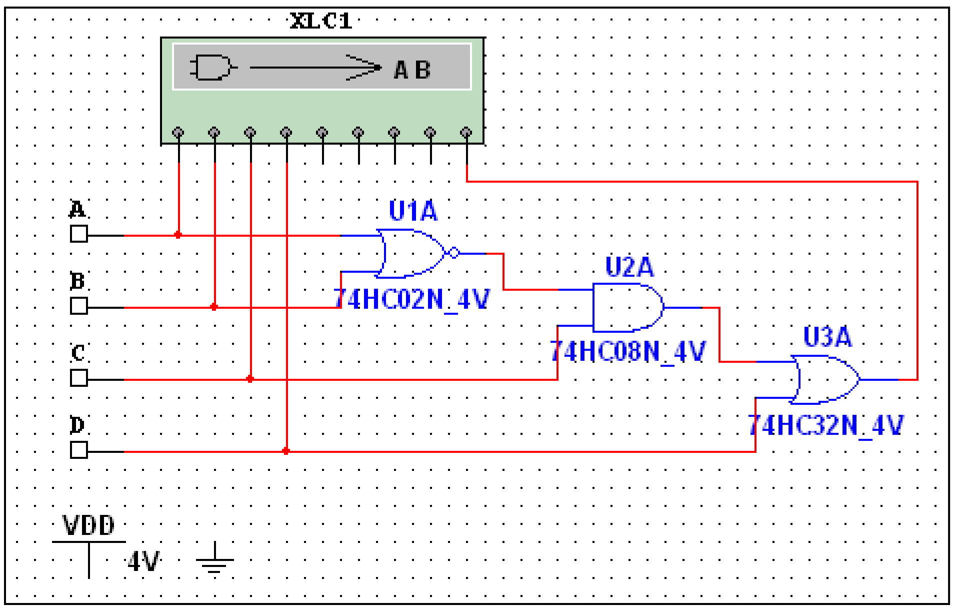
$$X = C\,\overline{(A + B)} + D = \overline{A}\,\overline{B}\,C + D$$

Boolean Analysis of Logic Circuits

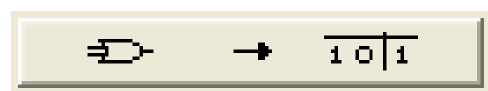
Boolean Analysis of Logic Circuits

Example Use Multisim to generate the truth table for the circuit in the previous example.

Solution Set up the circuit using the Logic Converter as shown. (Note that the logic converter has no "real-world" counterpart.)

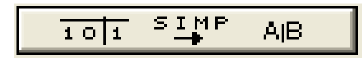


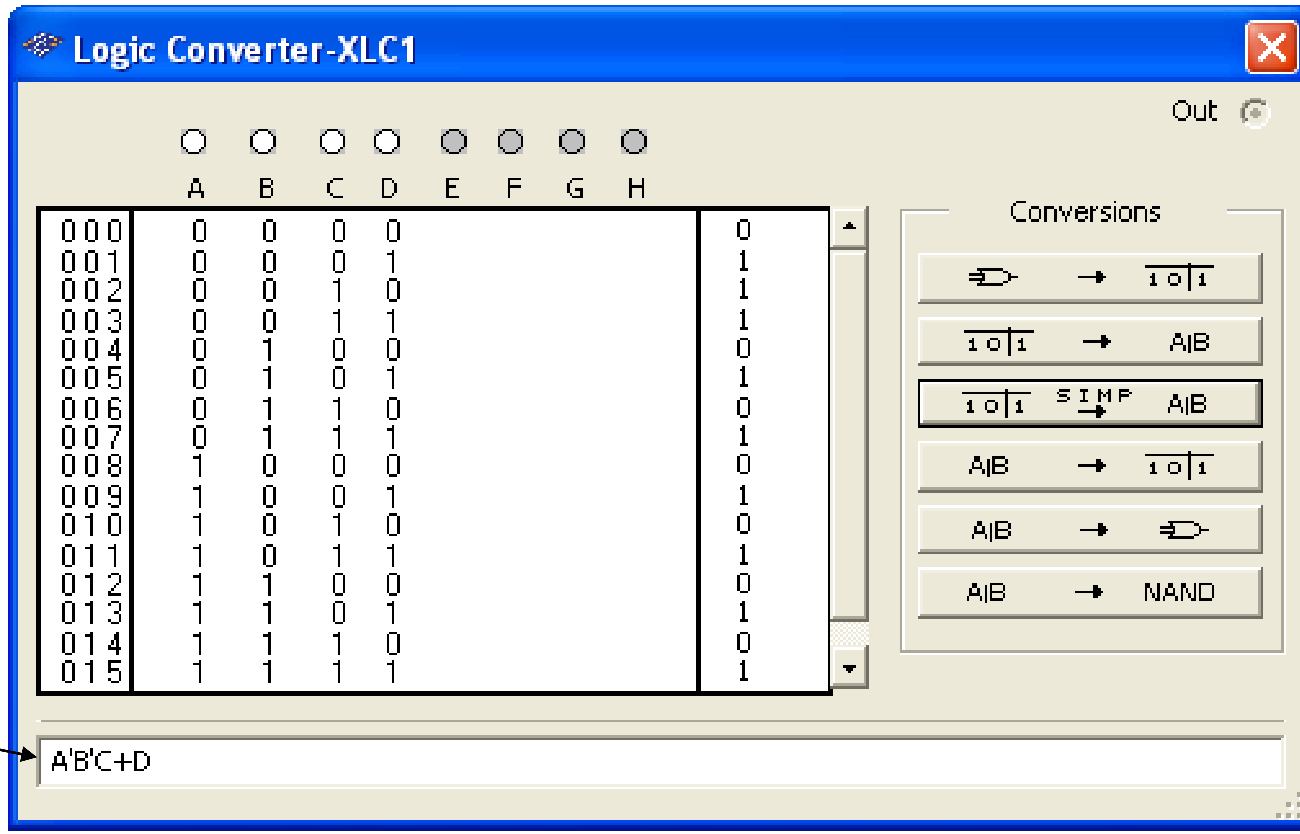
Double-click the Logic Converter top open it. Then click on the conversion bar on the right side to see the truth table for the circuit (see next slide).



Boolean Analysis of Logic Circuits

Boolean Analysis of Logic Circuits

The simplified logic expression can be viewed by clicking 



Simplified expression

Boolean Analysis of Logic Circuits

SOP and POS forms

Boolean expressions can be written in the **sum-of-products** form (**SOP**) or in the **product-of-sums** form (**POS**). These forms can simplify the implementation of combinational logic, particularly with PLDs. In both forms, an overbar cannot extend over more than one variable.

An expression is in SOP form when two or more product terms are summed as in the following examples:

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

An expression is in POS form when two or more sum terms are multiplied as in the following examples:

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

Boolean Analysis of Logic Circuits

SOP Standard form

In **SOP standard form**, every variable in the domain must appear in each term. This form is useful for constructing truth tables or for implementing logic in PLDs.

You can expand a nonstandard term to standard form by multiplying the term by a term consisting of the sum of the missing variable and its complement.

Example Convert $X = \overline{A}\,\overline{B} + A\,B\,C$ to standard form.

Solution The first term does not include the variable C. Therefore, multiply it by the $(C + \overline{C})$, which = 1:

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

Boolean Analysis of Logic Circuits

POS Standard form

In **POS standard form**, every variable in the domain must appear in each sum term of the expression.

You can expand a nonstandard POS expression to standard form by adding the product of the missing variable and its complement and applying rule 12, which states that $(A + B)(A + C) = A + BC$.

Example Convert $X = (\overline{A} + \overline{B})(A + B + C)$ to standard form.

Solution The first sum term does not include the variable C. Therefore, add $C\overline{C}$ and expand the result by rule 12.

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

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

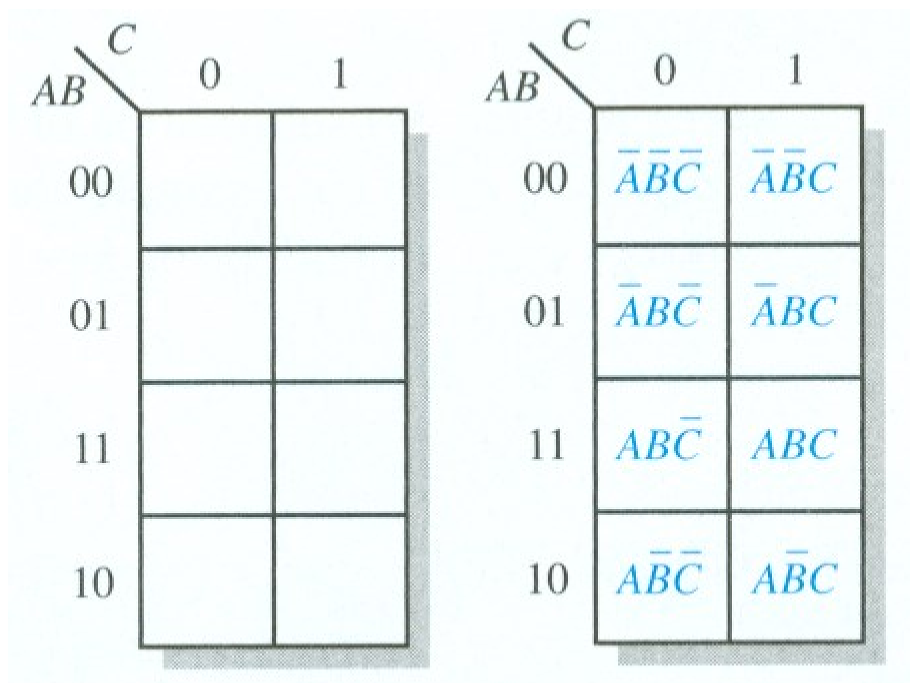
Boolean Analysis of Logic Circuits

Karnaugh maps

The Karnaugh map (K-map) is a tool for simplifying combinational logic with 3 or 4 variables. For 3 variables, 8 cells are required (2^3).

The map shown is for three variables labeled *A, B,* and *C*. Each cell represents one possible product term.

Each cell differs from an adjacent cell by only one variable.



3-Variable Karnaugh Map

Boolean Analysis of Logic Circuits

Karnaugh maps

Cells are usually labeled using 0's and 1's to represent the variable and its complement.

The numbers are entered in gray code, to force adjacent cells to be different by only one variable.

Ones are read as the true variable and zeros are read as the complemented variable.

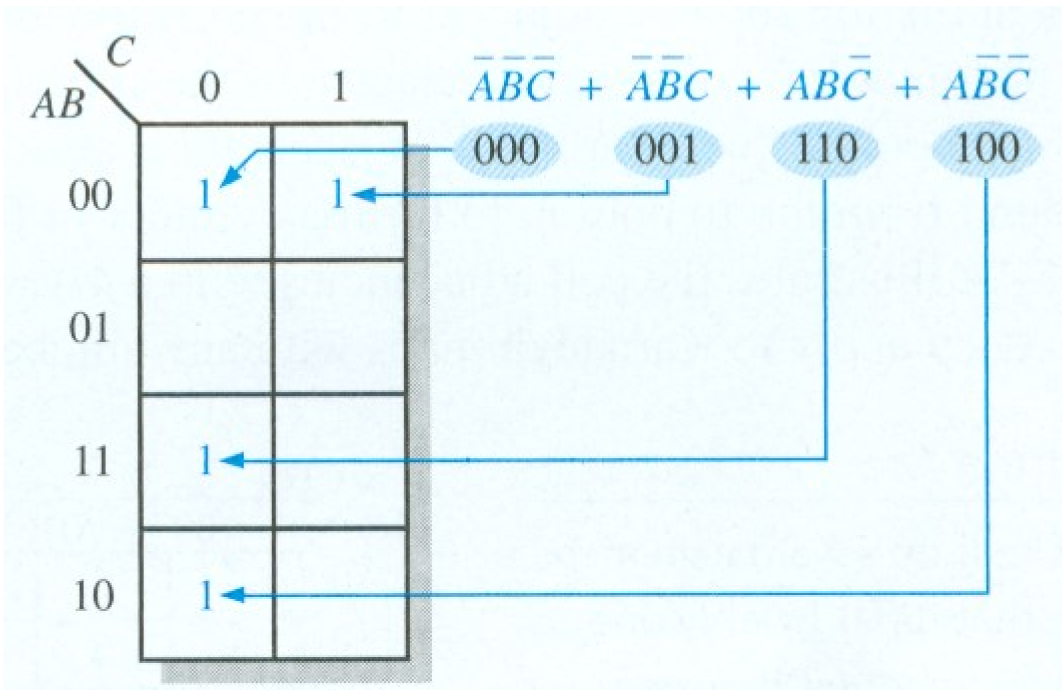
Gray code

$AB \backslash C$	0	1
00	$\bar{A}\bar{B}\bar{C}$	$\bar{A}\bar{B}C$
01	$\bar{A}B\bar{C}$	$\bar{A}BC$
11	$AB\bar{C}$	ABC
10	$A\bar{B}\bar{C}$	$A\bar{B}C$

3-Variable Karnaugh Map

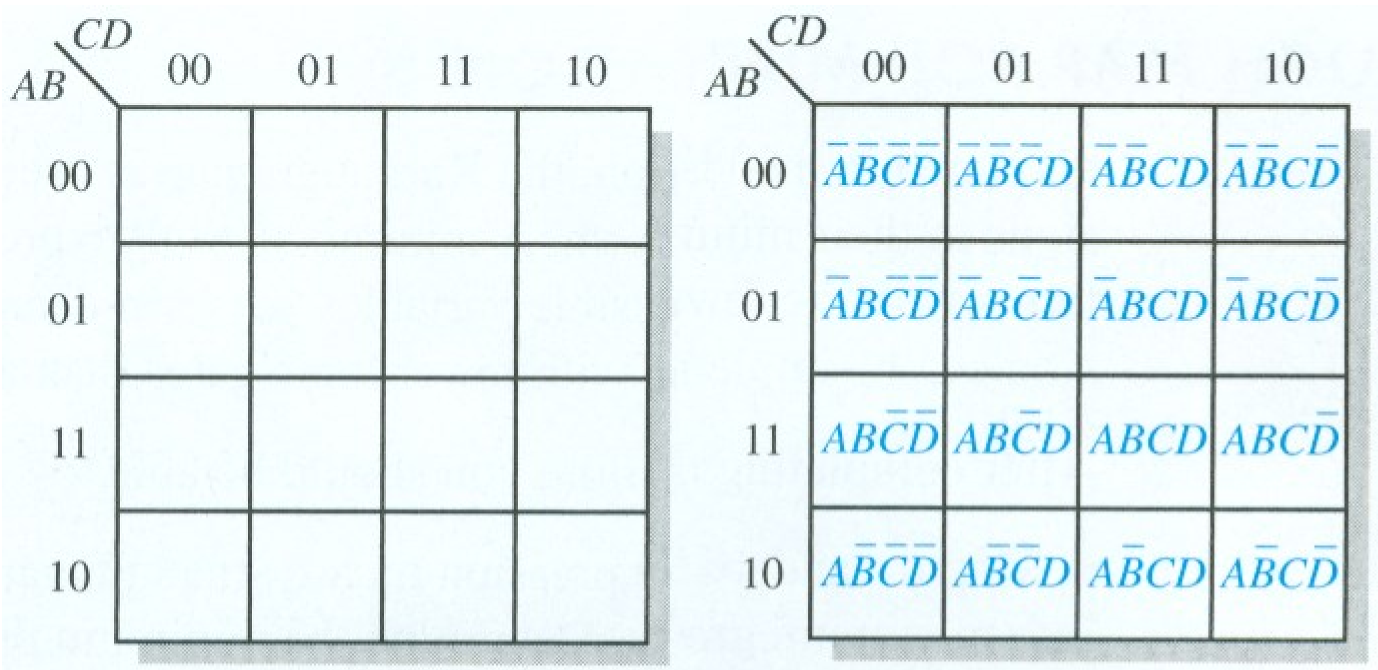
Boolean Analysis of Logic Circuits

Karnaugh maps



Boolean Analysis of Logic Circuits

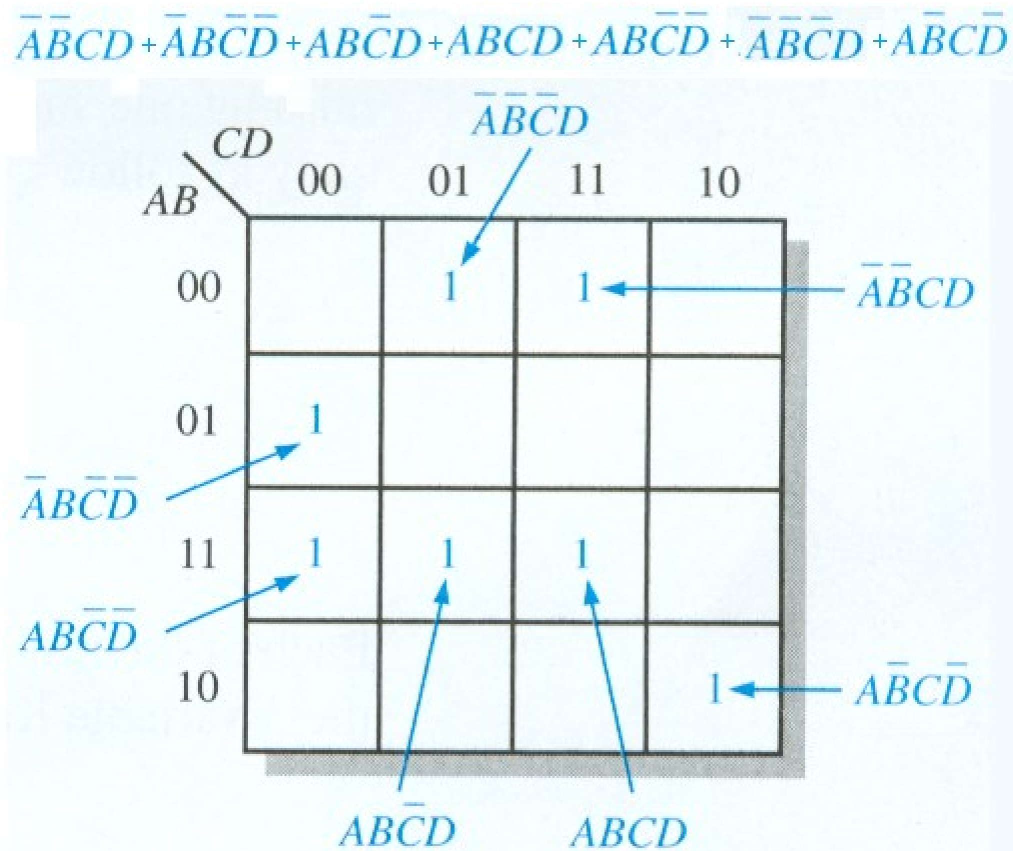
Karnaugh maps



4-Variable Karnaugh Map

Boolean Analysis of Logic Circuits

Karnaugh maps



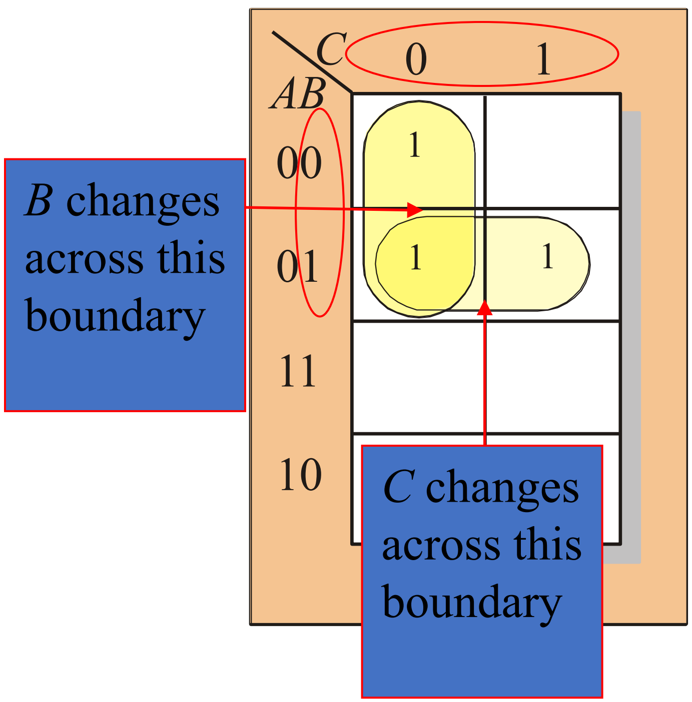
4-Variable Example

Boolean Analysis of Logic Circuits

Karnaugh maps

K-maps can simplify combinational logic by **grouping cells and eliminating variables that change**.

Example Group the 1's on the map and read the minimum logic.



Solution

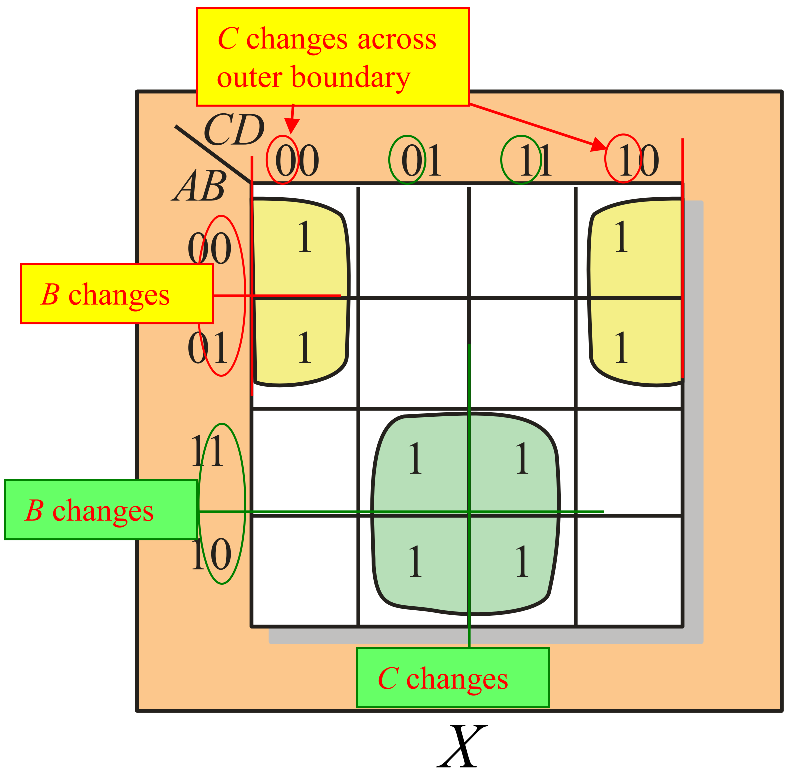
1. Group the 1's into two overlapping groups as indicated.
2. Read each group by eliminating any variable that changes across a boundary.
3. The vertical group is read $\overline{A}\,\overline{C}$.
4. The horizontal group is read $\overline{A}B$.

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

Boolean Analysis of Logic Circuits

Karnaugh maps

Example Group the 1's on the map and read the minimum logic.



Solution

1. Group the 1's into two separate groups as indicated.
2. Read each group by eliminating any variable that changes across a boundary.
3. The upper (yellow) group is read as $\overline{A}\,\overline{D}$.
4. The lower (green) group is read as AD.

$$X = \overline{A}\,\overline{D} + AD$$

QUIZ

The minimum expression that can be read from the Karnaugh map shown is

	$\overline{C}$	C
$\overline{A}\,\overline{B}$		
$\overline{A}B$		
AB	1	1
$A\overline{B}$	1	1