CHAPTER 3

BOOLEAN ALGEBRA

And basic logic GATES

Pof.Dr. Mehmet Akbaba

BOOLEAN ALGEBRA

Introduction

❑Basic math needed for study of the design of digital systems is Boolean algebra.

❑All switching devices which we will use are two-state (high or low)devices.

❑We will study the special case of Boolean algebra in which all variables assume one of the two values.

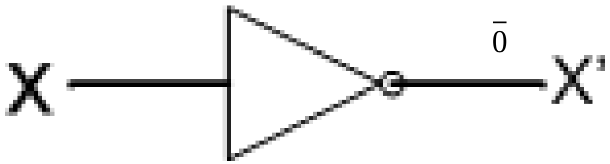
- This two valued Boolean algebra is referred to as switching algebra
- Two states are ‘0’ or ‘1’
- A variable X will assume a value of either ‘0’ or ‘1’ or alternatively T (true) or F (false)
- Although ‘0’ and ‘1’ used in switching algebra looks like binary numbers , they are not.

Basic Operations

Basic operations of Boolean algebra are AND, OR, and NOT or complement (or inverse). Complement of 0 is 1 and complement of 1 is 0. Symbolically:

0'=1 (and 1'=0 X'=0 if X=1 and X'=1 if X=0)

$$(\bar{0} = 1 \text{ and } \bar{1} = 0)$$

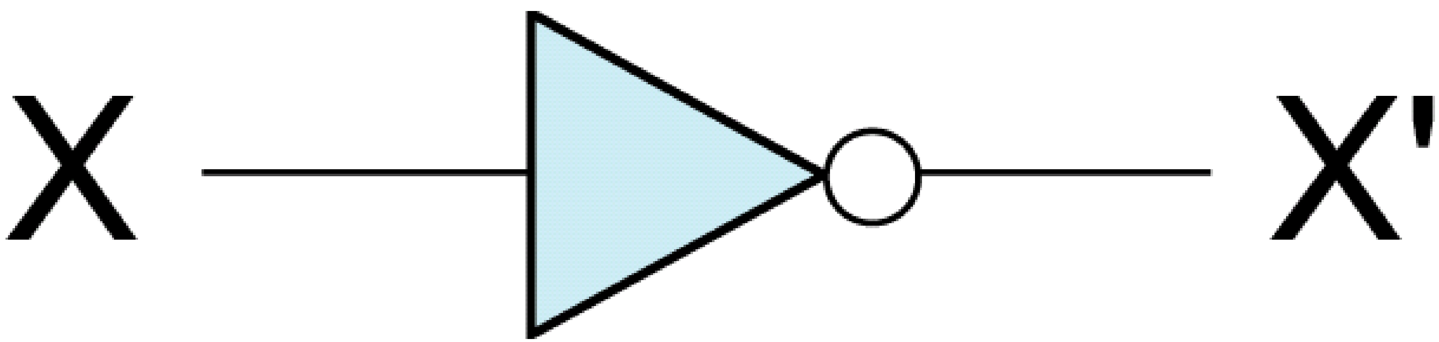


- **Complement is also called inversion. Electronic device that perform inversion is called inverter and the above symbol is used for it.**

- **Complementation is also sometimes is called as NOT.**

X' = 1 if X = 0

X' = 0 if X = 1



AND operation can be defined as:

- : AND

0.0=0,

0.1=0 ,

1.0=0

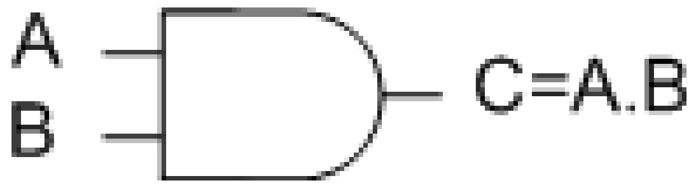
and 1.1=1

(Althoug it works like multiplication but it is not a binary multiplication)

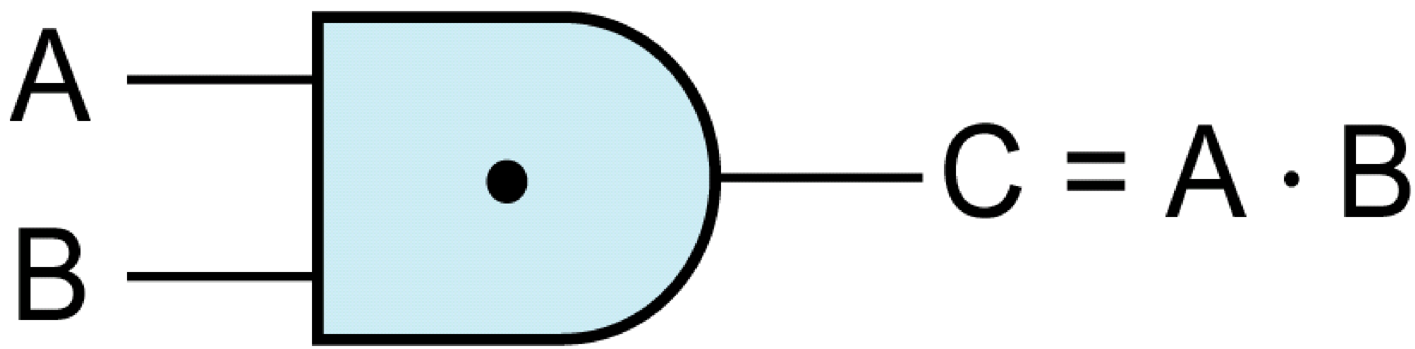
Truth table of AND:

A	B	C=A.B
0	0	0
0	1	0
1	0	0
1	1	1

A logic gate which performs the AND operation is represented by :



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



- **OR operation can be defined as:**

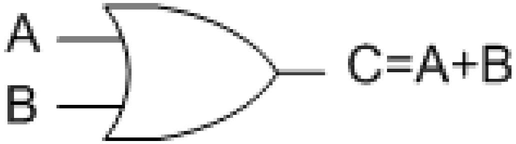
0+0=0, 0+1=1 , 1+0=1 and 1+1=1

(This is not a binary ADDITION)

Truth table of OR is:

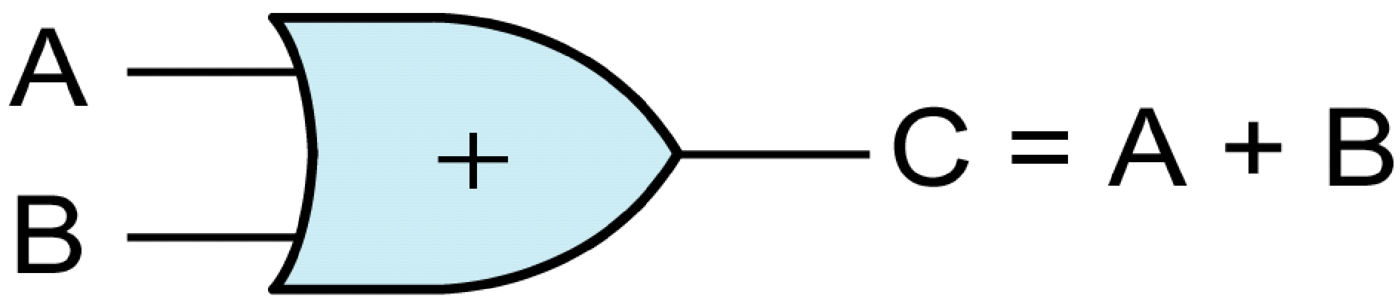
A	B	C=A+B
0	0	0
0	1	1
1	0	1
1	1	1

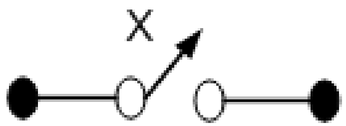
A logic gate which performs the AND operation is represented by :



The OR operation is also referred to as logical (or Boolean) addition.

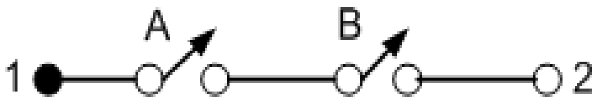
A	B	C = A + B
0	0	0
0	1	1
1	0	1
1	1	1





X=0 → Switch is open

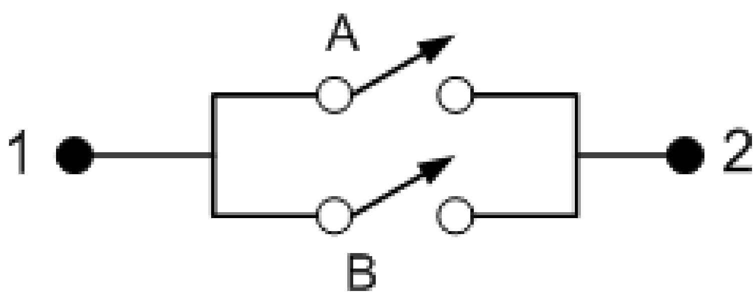
X=1 → Switch is closed



T=0 → open circuit between terminals 1 and 2

T=1 → closed circuit between terminals 1 and 2

T=A.B (AND)



T=A+B (OR)

In this case we have a closed circuit between terminals 1 and 2 iff A is closed or B is closed.

BOOLEAN EXPRESSIONS AND TRUTH TABLES

Boolean expressions are formed by application of basic operations to one or more variables or constants. The simplest expression consist of a single constant or variables, such as X, Y'. More complicated expressions are formed by combining two or more other expressions using AND or OR, or by complementing another expression. Examples of expressions are:

AB'+C (3.1)

[A(C+D)]'+BE (3.2)

Each expression corresponds directly to a circuit of logic gates. Figure 3.1 gives the circuit for expressions (3.1) and (3.2).

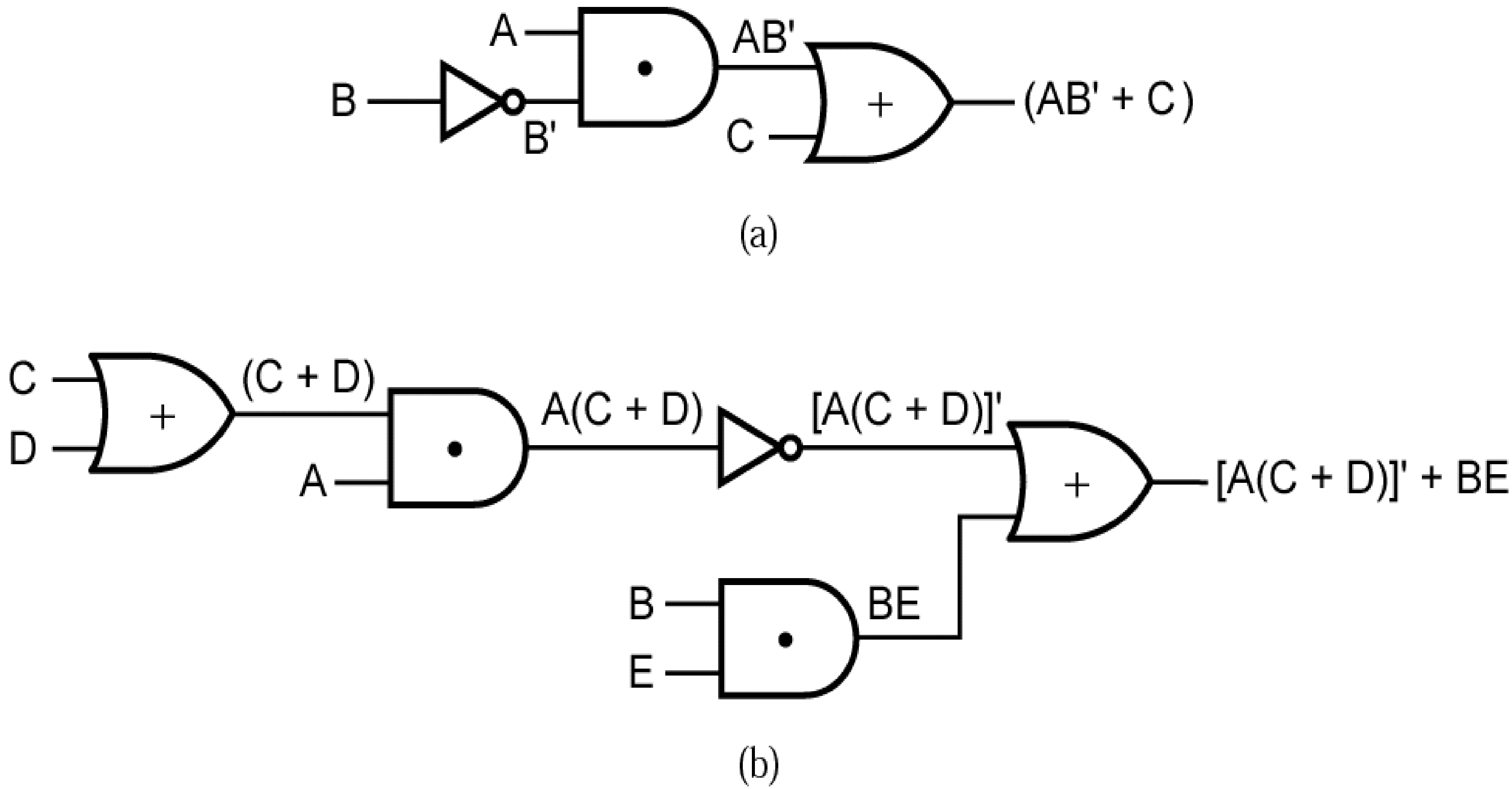


Figure 3.1. Circuits for Expressions (3-1) and (3-2)

An expression is evaluated by substituting a value of 0 or 1 for each variable. If A=B=C=1 and D=E=0, the value of expression (3.2) is:

[A(C+D)]'+BE=[1(1+0)]'+1x0=1x(1)'+1x0=1x0+0=0
(x: used multiplication sign)
Each appearance of a variable or its complement in an expression will be referred to as a literal. Thus following expression, which has three variables, has 10 literals:
ab'c+a'b+a'bc'+b'c'
A truth table (also called a table of combinations) shows specific the values of a Boolean expression for every possible combinations of values of the variables in the expression.
The circuit and the truth table for the expression F=A'+B is given in Figures (3.2a) and (3.2b).

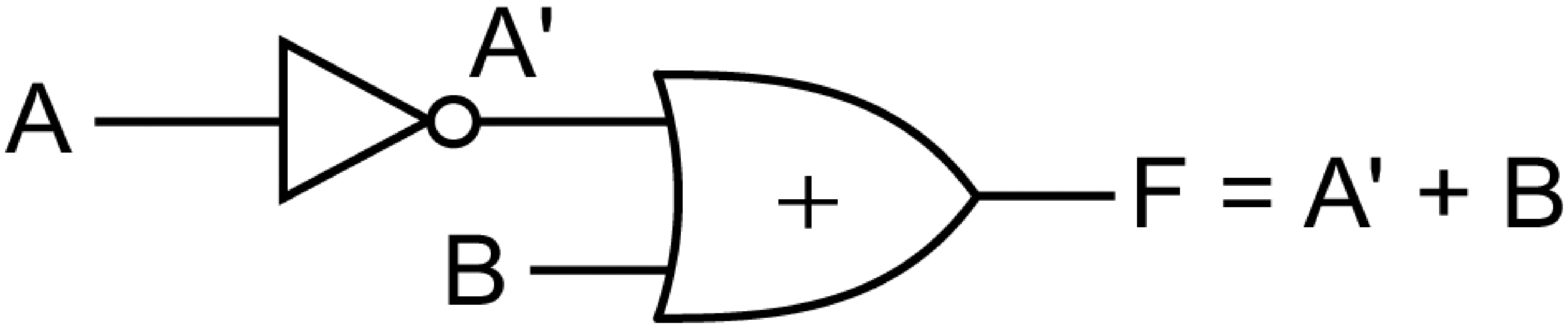


Fig. (3.2a)

(a)

2-Input Circuit

Fig. (3.2b)

A	B	A′	F = A′ + B
0	0	1	1
0	1	1	1
1	0	0	0
1	1	0	1

(b)

Table 2.1 is illustrating truth table for expressions AB’+C and (A+C)(B’+C).

Since there are 3 variables (A, B, C), there will be 2^3=8 combinations of input variables in the truth table.

In truth tables in general 2^n combinations for n input variables.

In fact AB’+C=(A+C)(B’+C). Table 3.1 will also give the proof of this identity.

Table 3-1: Truth Table for 3 variables

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

BASIC THEOREMS

Operation with 0 and 1:

$X+0=X$ (3.4.a) $X.1=X$ (3.4.b)

$X+1=1$ (3.5.a) $X.0=0$ (3.5.b)

Idempotent laws

$X+X=X$ (3.6.a) $X.X=X$ (3.6.b)

Involution law:

$(X')'=X$ (3.7)

Laws of complementarity:

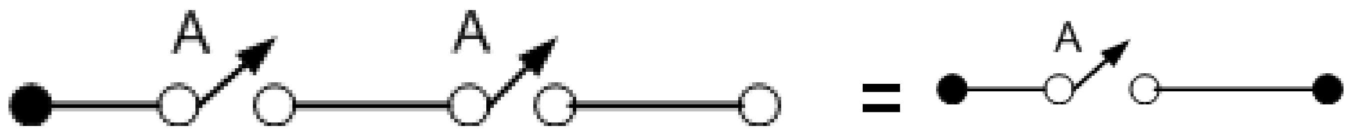
$X+X'=1$ (3.8.a) $X.X'=0$ (3.8.b)

Any expression can be substituted for the variable X in theses theorems. Thus, by Theorem (3.5.a):

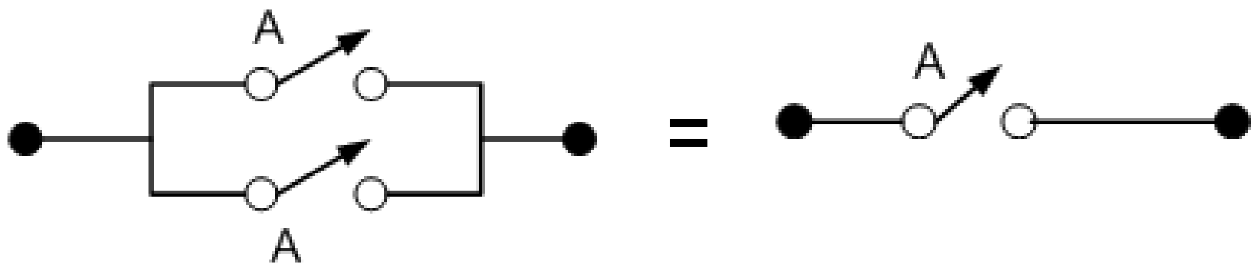
$(AB'+D)E+1=1$ (where (AB'+D)E is cosidered as X)

And by Theorem (3.8.b)

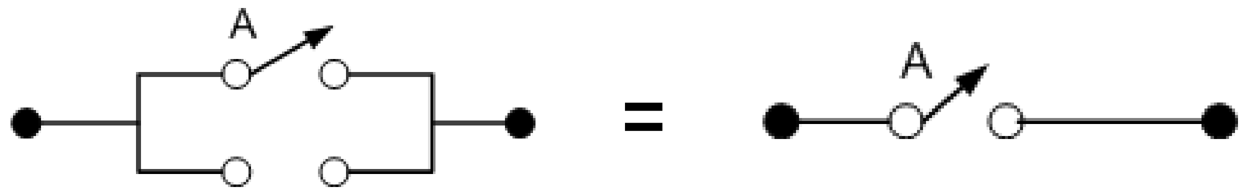
$(AB'+D)(AB'+D)'=0$



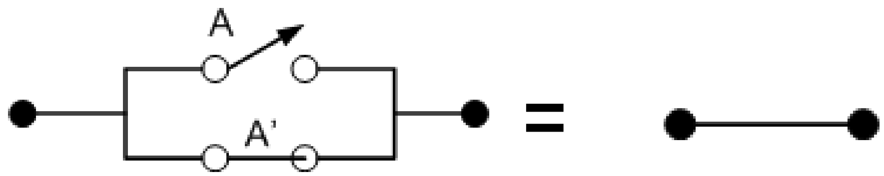
This illustrates the theorem A.A=A



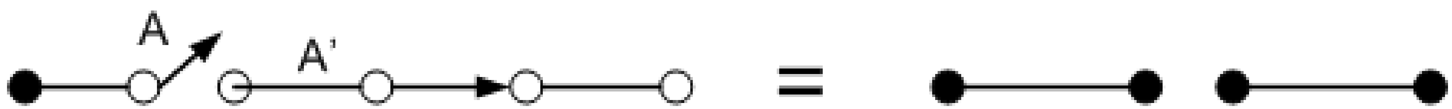
This illustrates the theorem A+A=A



This illustrates the theorem A+0=A



This illustrates the theorem A+A'=1



This illustrates the theorem A.A'=0

Commutative, Associative and Distributive Laws

Many of the laws of ordinary algebra, such as commutative and associative laws, also apply to Boolean algebra. The commutative laws for the AND and OR operations are:

XY=YX (3.9.a) X+Y=Y+X (3.9.b)

Which means the order of the writing variables at first or at last, will not affect the results of the AND and OR operations.

The associative laws also apply to AND and OR:

(XY)Z=X(YZ)=XYZ (3.10.a)

(X+Y)+Z=X+(Y+Z)=X+Y+Z (3.10.b)

When forming AND (or OR) of three variables, the results is independent of which pair of variables are associated first, so the parentheses can be omitted as indicated in (3.10.a) and (3.10.b).
Proof of the associative law for AND can be made using the truth table as shown in Table 3.2.

Table 3.2: Proof of Associative Law for AND

X	Y	Z	XY	YZ	$(XY)Z$	$X(YZ)$
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	1	0	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	1	0	0	0
1	1	1	1	1	1	1

Fig. 3.3 illustrates the associative laws using AND and OR gates. In figure 3.3a two 2-input AND gated are replaced with 3-input AND gate.

Similarly, in Fig. 3.3b two 2-input OR gates are replaced with a single 3-input OR gate.

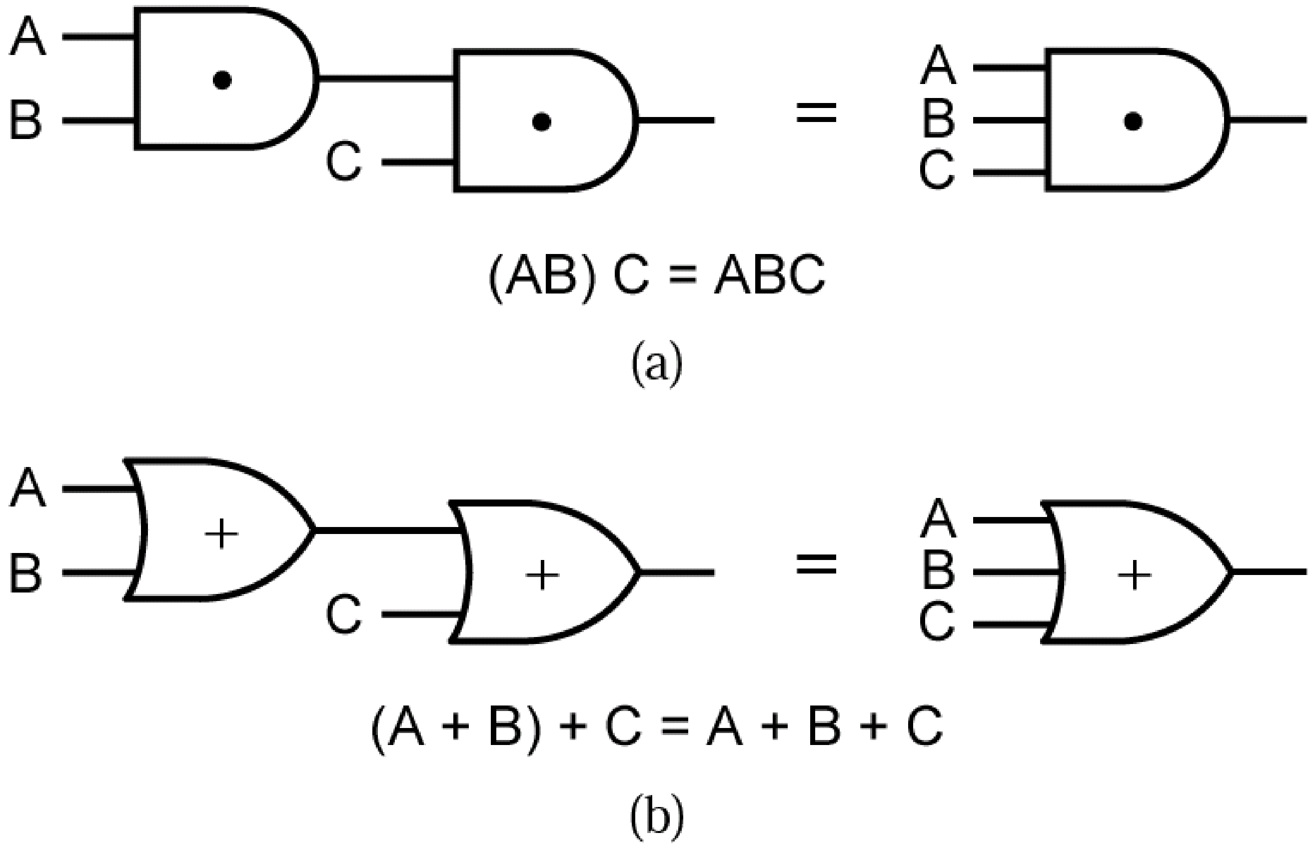


Figure 3-3: **Associative Law for AND and OR**

When 1 or more variables are AND ed together, the value of the result will be 1 iff all of the variables have the value 1. If any of the variables have the value 0, the result of the AND operation will be 0. For example,

XYZ=1 iff X=Y=Z=1

When two or more variables are ORed together, the value of the result will be 1 if any of the variables have the value 1. The result of the OR operation will be zero iff all the variables have the value 0. For example,

X+Y+Z=0 iff X=Y=Z=0

Using the truth table, it is easy to show that distributive law is valid:

$X(Y+Z)=XY+XZ$ (3.11.a)

In addition to ordinary distributive law, a second distributive law (exteremly important law) is valid for Boolean algebra but not for ordinary algebra:

$X+YZ=(X+Y)(X+Z)$ (3.11.b) (very important)

Prof of the second distributive law follows:

$(X+Y)(X+Z)=X(X+Z)+Y(X+Z)=XX+XZ+YX+YZ$
$=X+XZ+XY+YZ=X.1+XZ+XY+YZ$
$=X(1+Z+Y)+YZ=X.1+YZ=X+YZ$

The ordinary distributive law states that the AND operation distributes over OR, while the second distributive law states that OR distributes over AND. This second law is useful in manipulating Boolean expressions. In particular, an expression like A+BC, which can not be factored in ordinary algebra, is easily factored using the second distributive law:

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

SIMPLIFICATION THEOREMS

$XY + XY' = X$ (3.12.a) $(X + Y)(X + Y') = X$ (3.12.b)

$X + XY = X$ (3.13.a) $X(X + Y) = X$ (3.13.b)

$(X + Y')Y = XY$ (3.14.a) $XY' + Y = X + Y$ (3.14.b)

In each case, one expression can be replaced by a simpler one. ***Because each expression corresponds to a logic circuit composed of logic gates, simplifying an expression leads to simplifying the corresponding logic circuit.***

Each of the preceding theorems can be proved by using truth table, or they can be proved algebraically starting with the basic theorems.

Proof of (3.13.a):

X+XY=X.1+XY=X(1+Y)=X.1=X
(because 1+Y=1, (Theorem 3.5.a))

Proof of (3.13.b):

X(X+Y)=XX+XY=X+XY=X(1+Y)=X

Proof of (3.14.b):

Y+XY'=(Y+X)(Y+Y')=(Y+X).1=Y+X
(because Y+Y'=1 (Theorem 3.8.a))

Illustration of Theorem (3.14.b) using switches:

Fig. ‘a’

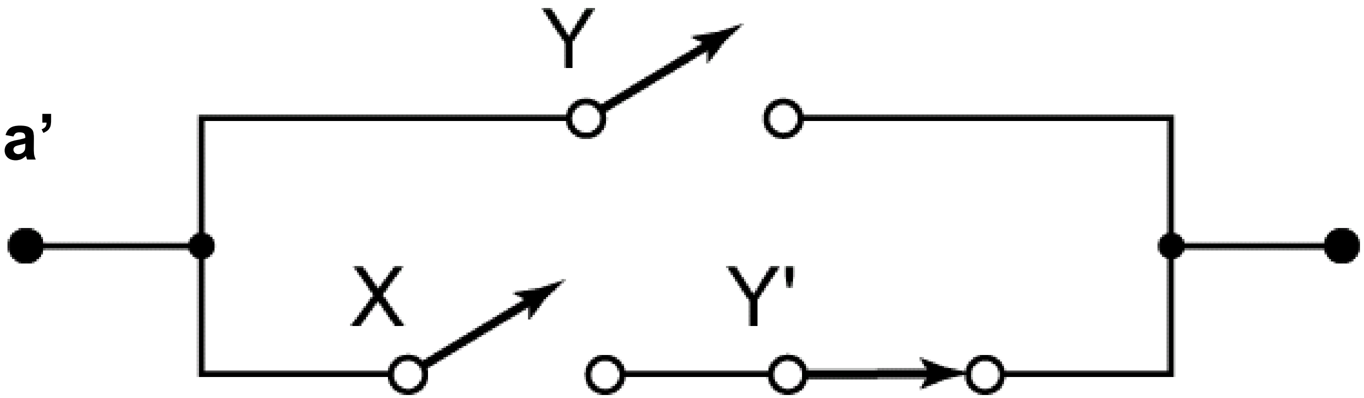
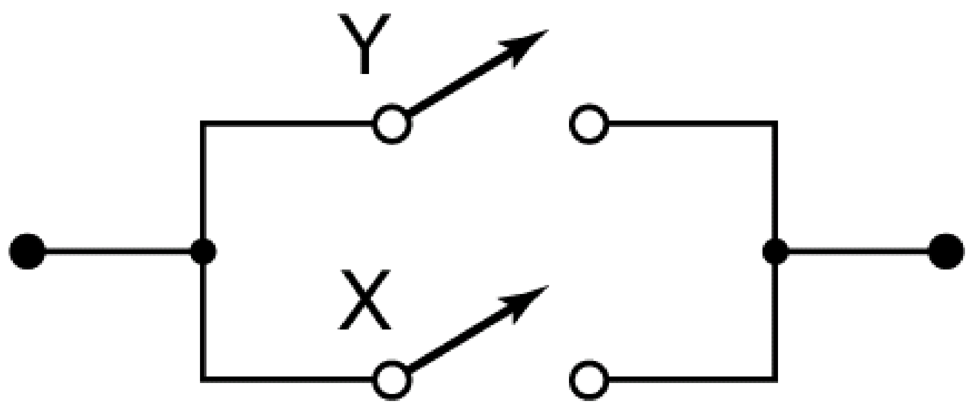


Fig. ‘b’

=

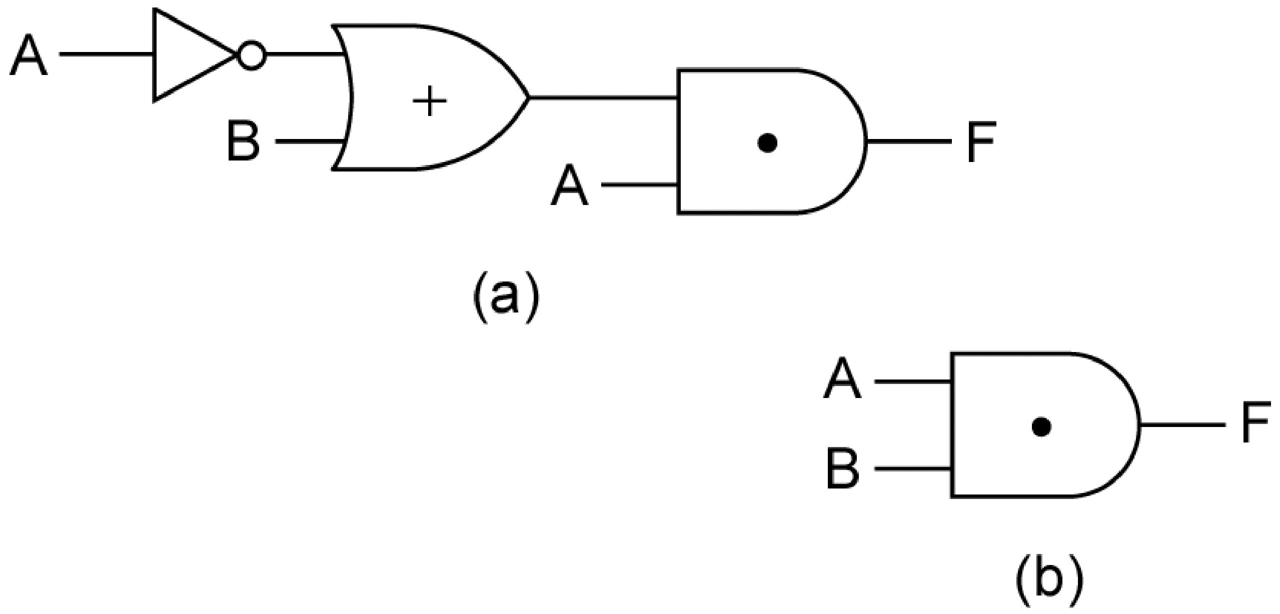


$$(Y+XY' = Y+X)$$

Transmission of Fig. ‘a’ above, is T=Y+XY’ because there is a closed circuit between the terminals if Y is closed or X is closed and Y’ is closed. Therefore Fig. ‘b’ above equivalent of Fig.’a’ because if Y is closed (Y=1) both circuits have a transmission of 1; if Y is open (Y=0 and Y’=1) both circuits have transmission of X.

Example 1: Simplify circuit in Fig. 3.4.a using one of the theorems.

Fig. 3.4



The output of Fig. 3.4a is F=A(A'+B). Using theorem 3.14 (X(X'+Y)=XY) this reduces to F=AB. Therefore logic circuit in Fig. 3.4a simplifies to Fig. 3.4b.

Example 2: **Simplify Z=A'BC+A'**

Let BC=Y and A'=X. Then above expression is F=X+XY
Applying theorem (3.13.a) the last expression reduces to
F=X which mean F=A'

Example 3:
Simplify Z=[A+B'C+D+EF][A+B'C+(D+EF)']

Let X=A+B'C and Y=D+EF Then above expression is
Z=[X+Y][X+Y'] (=X+X(Y+Y')+YY'=X+YY'=X)
Applying theorem (3.8.b) the last expression reduces to
F=X which mean F=A+B'C

Example 4: Simplify Z= (AB+C)(B'D+C'E')+(AB+C)'

Let X=AB+C and Y=B'D+C'E', then
Z=XY+X' =X'+XY   (=(X'+X)(X'+Y),   X+X'=1, Z=Y+X')

Applying theorem 3.14.b we obtain Z=Y+X' or
Z=B'D+C'E'+(AB+C)'

Multiplying Out and Factoring

The two distributive laws are used to multiply out an expression to obtain a sum- of-products (SOP) (çarpımların toplamı) form. An expression is said to be in *sum-of-products* form when all products are the products of single variables. This form is the end result when an expression is fully multiplied out. It is usually easy to recognize a sum-of-products expression because it consists of a sum of product terms:

$$AB' + CD'E + AC'E' \qquad (3\text{-}15)$$

However, in degenerate cases, one or more of the product terms may consist of a single variable. For example

$$ABC' + DEFG + H \qquad (3.16)$$

and

$$A + B' + C + D'E \qquad (3.17)$$

are still considered to be in sum-of-products form. The expression

$$(A + B)CD + EF$$

is not in sum-of-products form because the *A + B* term enters into a product but is not a single variable.

When multiplying out an expression, apply the second distributive law first when possible. For example, to multiply out *(A + BC)(A + D + E)* let

$$X=A, \quad Y=BC, \quad Z=D+E$$

Then

$$(X + Y)(X + Z) = X + YZ = A + BC(D + E) = A + BCD + BCE$$

Of course, the same result could be obtained by hard way, by multiplying out the original expression completely and then eliminating redundant terms:

$$(A + BC)(A + D + E) = A + AD + AE + ABC + BCD + BCE$$

$$= A(l + D + E + BC) + BCD + BCE$$

$$= A + BCD + BCE$$

You will save yourself a lot of time if you learn to apply the second distributive law instead of doing the problem the hard way.

Both distributive laws can be used to factor an expression to obtain a product- of-sums form. An expression is in *product-of-sums* (POS) (toplamların çerpımı) 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

$$(A + B')(C + D' + E)(A + C' + E') \qquad (3\text{-}18$$

However, in degenerate cases, one or more of the sum terms may consist of a single variable. For example,

Example 1: ***A + B'CD ifadesini faktörlerine ayırınız (POS: Toplamların çarpımı şeklinde yazınız)***

Çözüm: Verilen Lojik ifade X + *YZ* şeklindedir. Burada X = *A, Y* = *B'*, ve Z = *CD,* Dolayısiyle:

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

A + CD ifadesi ikinci dağılım kuralı (second distributive low) uygulanarak faktörlerine ayrılabilir:

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

Example 2: Factor AB' + C'D.

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

(note note that second distributive law ia applied as follows

X + YZ = (X + Y)(X + Z))

Applying second distributive law again to each term we obtain:

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

Example 3:

Factor *C'D* + C' *E'* + G' *H*

= (C' + G'H)(D + E' + G'H)

= (C' + G')(C + H)(D + E' + G')(D + E' + H)

As in Example 1, the ordinary distributive law should be applied before the second law when factoring an expression.

A sum-of-products expression can always be realized directly by one or more AND gates feeding a single OR gate at the circuit output. Figure 3-5 shows the circuits for Equations (3-15) and (3-17). Inverters required to generate the complemented variabies have been omitted

The circuits shown in Figures 3-5 and 3-6 are often referred to as two-level circuits because they have a maximum of two gates in series between an input and the circuit output.

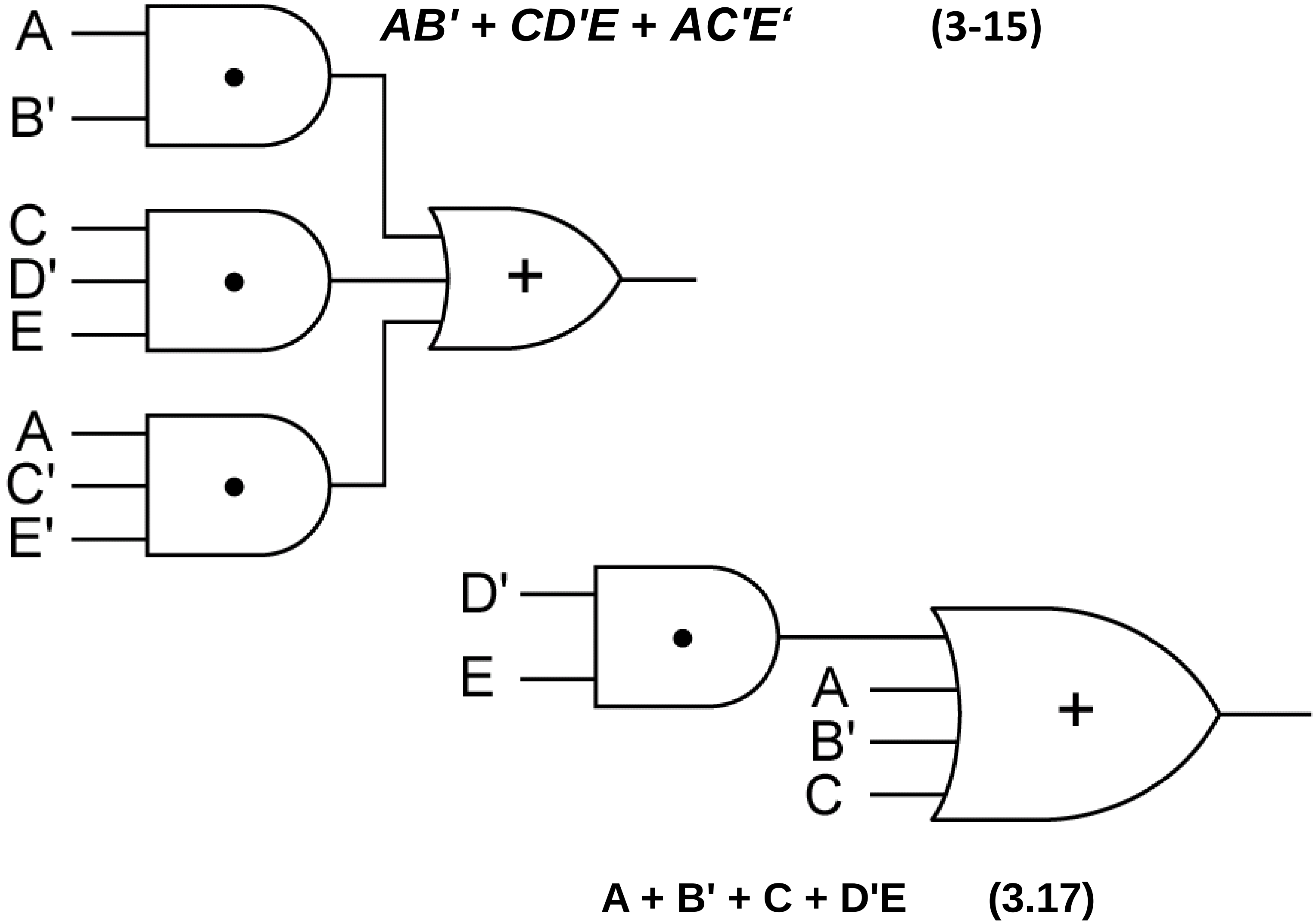


Figure 3-5: **Circuits for Equations (3-15) and (3-17)**

We will verify these laws using truth table

XY	X' *Y'*	*X*+*Y*	(X + *Y)'*	X' *Y'*	*XY*	*(XY)'*	X' + *Y'*
00	**1 1**	**0**	**1**	**1**	**0**	**1**	**1**
01	**1 0**	**1**	**0**	**0**	**0**	**1**	**1**
1 0	**0 1**	**1**	**0**	**0**	**0**	**1**	**1**
1 1	**00**	**1**	**0**	**0**	**1**	**0**	**0**

DeMorgan's laws are easily generalized to *n* variables

$$(X_l + X_2 + X_3 + \ldots + X_n)' = X_l' X_2' X_3 \ldots\ldots X_n' \qquad \textbf{(3.23)}$$

$$(X_l X_2 X_3 \cdot \ldots X_n)' = X_1' + X_2' + X_3' + \ldots + X_n' \qquad \textbf{(3.24)}$$

For example for n=3

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

$$(X_1 + X_2 + X_3)' = (X_1 + X_2)' X_3' = X_1' X_2' X_3'$$

Referring to the OR operation as the logical sum and the AND operation as logical product, DeMorgan's laws can be stated as :

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

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

To form the complement of an expression containing both OR and AND operations, DeMorgan's laws are applied alternately.

Example 1:

To find the complement of *(A' + B)C',* first apply (3-22) and then (2-21).

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

Example 2:

$[(AB' + C)D' + E]' = [(AB' + C)D']E'$	(by (3-21))
$= [(AB' + C)' + D]E'$	(by (3-22))
$= [(AB')'C' + D]E'$	(by (3-21))
$= [(A' + B)C' + D]E'$	(by (3-22)) (3-25)

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

The inverse of F = A'B + AB' is

F' = (A'B + AB')' = (A'B)'(AB')' = (A + B')(A' + B)

= AA' + AB + B'A' + BB' = A'B' + AB

We will verify that this result is correct by constructing a truth table for F and F‘ :

$A\ B$	$A'B$	AB'	$F = A'B + AB'$	$A'B'$	AB	$F' = A'B' + AB$
0 0	0	0	0	1	0	1
0 1	1	0	1	0	0	0
1 0	0	1	1	0	0	0
1 1	0	0	0	0	1	1

In the table, note that for every combination of values of *A* and *B* for which *F* = 0, *F'* = 1; and whenever *F* = 1, *F'* = O.

Given a Boolean expression, the *dual* is formed by replacing AND with OR, OR with AND, 0 with 1, and 1 with 0. Variables and complements are left unchanged.

(2-26)

The dual of AND is OR and the dual of OR is AND:

$$(XYZ\ \ldots)^D = X + Y + Z + \ldots$$

$$(X + Y + Z + \ldots)^D = XY.\ (X + Y + Z + \ldots)^D = XYZ. \qquad (3.26)$$

The dual of an expression may be found by complementing the entire expression and then complementing each individual variable. For example, to find the dual of

AB’+C

(AB' + C)' = (AB)C' = (A' + B)C', so

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

LAWS AND THEOREMS (a)

Operations with 0 and 1:

1.a) $X + 0 = X$ **1.b)** $X \bullet 1 = X$

2.a) $X + 1 = 1$ **2.b)** $X \bullet 0 = 0$

Idempotent laws:

3ab) $X + X = X$ **3.b).** $X \bullet X = \mathrm{X}$

Involution law:

4. $(X')' = X$

Laws of complementarity:

5.a) $X + X' = 1$ **5b).** $X \bullet X' = 0$

LAWS AND THEOREMS (b)

Commutative laws:

6.a) $X + Y = Y + X$ **6.b)** $XY = YX$

Associative laws:

7.a) $(X + Y) + Z = X + (Y + Z) = X + Y + Z$ **7.b)** $(XY)Z = X(YZ) = XYZ$

Distributive laws:

8.a) $X(Y + Z) = XY + XZ$ **8.b).** $X + YZ = (X + Y)(X + Z)$

Simplification theorems:

9.a) $XY + XY' = X$ **9.b)** $(X + Y)(X + Y') = X$

10.a) $X + XY = X$ **10.b)** $X(X + Y) = X$

11.a) $(X + Y')Y = XY$ **11.b)** $XY' + Y = X + Y$

SUMMARY OF LAWS AND THEOREMS (c)

DeMorgan's laws:

12.a) (X + Y + Z +...)' = X'Y'Z'...

12. b) (XYZ...)' = X' + Y' + Z' +...

Duality:

13.a) $(X + Y + Z + \ldots)^D = XYZ\ldots$

13. b) $(XYZ\ldots)^D = X + Y + Z + \ldots$

Theorem for multiplying out and factoring:

14.a) (X + Y)(X' + Z) = XZ + X'Y

14.b) XY + X'Z = (X + Z)(X' + Y)

Consensus theorem:

15.a) XY + YZ + X'Z = XY + X'Z

15.b) (X + Y)(Y + Z)(X' + Z) = (X + Y)(X' + Z)

REFEENCES

- **1. Hüseyin EKİZ, Mantık Devreleri, Değişim Yayınları, 4. Baskı, 2005**
- **2. Prof. M. Akbaba Mantık Devreleri Notları**
- **3 .Thomas L. Floyd, Digital Fundamentals, Prentice-Hall Inc. New Jersey, 2006**
- **4 .M. Morris Mano, Michael D. Ciletti, Digital Design, Prentice-Hall, Inc.,New Jersey, 1997**