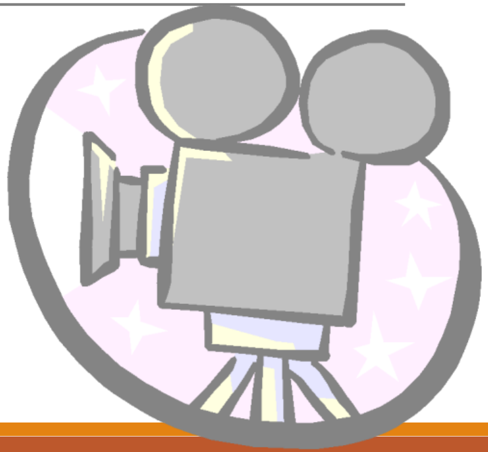
Digital Logic and Circuit
Paper Code: CS-102



Outline

- Basic theorem of Boolean Algebra
- Boolean Function
- Universal gates

Basic theorem & properties of Boolean Algebra

Duality: The Huntington postulates have been listed in pairs and designated by part (a) and part (b). One part may be obtained from the other if the binary operators and the identity elements are interchanged.

This important property of Boolean algebra is called the **duality principle**.

It states that every algebraic expression deducible from the postulates of Boolean algebra remains valid if the operators and identity elements are interchanged.

Example: B={0, 1} e={0, 1}

$$x+0=0+x=x$$

$$x.1=1.x=x$$

Basic theorem & properties of Boolean Algebra

Postulates and Theorems of Boolean Algebra

Postulate 2	(a) $x + 0 = x$	(b) $x \cdot 1 = x$
Postulate 5	(a) $x + x' = 1$	(b) $x \cdot x' = 0$
Theorem 1	(a) $x + x = x$	(b) $x \cdot x = x$
Theorem 2	(a) $x + 1 = 1$	(b) $x \cdot 0 = 0$
Theorem 3, involution	$(x')' = x$	
Postulate 3, commutative	(a) $x + y = y + x$	(b) $xy = yx$
Theorem 4, associative	(a) $x + (y + z) = (x + y) + z$	(b) $x(yz) = (xy)z$
Postulate 4, distributive	(a) $x(y + z) = xy + xz$	(b) $x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a) $(x + y)' = x'y'$	(b) $(xy)' = x' + y'$
Theorem 6, absorption	(a) $x + xy = x$	(b) $x(x + y) = x$

THEOREM 1(a): $x + x = x$.

$$\begin{aligned} x + x &= (x + x) \cdot 1 && \text{by postulate: } 2(\text{b}) \\ &= (x + x)(x + x') && 5(\text{a}) \\ &= x + xx' && 4(\text{b}) \\ &= x + 0 && 5(\text{b}) \\ &= x && 2(\text{a}) \end{aligned}$$

THEOREM 1(b): $x \cdot x = x$.

$$\begin{aligned} x \cdot x &= xx + 0 && \text{by postulate: } 2(\text{a}) \\ &= xx + xx' && 5(\text{b}) \\ &= x(x + x') && 4(\text{a}) \\ &= x \cdot 1 && 5(\text{a}) \\ &= x && 2(\text{b}) \end{aligned}$$

THEOREM 2(a): $x + 1 = 1$.

$$
\begin{aligned}
x + 1 &= 1 \cdot (x + 1) && \text{by postulate:} \quad 2(b)\\
&= (x + x')(x + 1) && 5(a)\\
&= x + x' \cdot 1 && 4(b)\\
&= x + x' && 2(b)\\
&= 1 && 5(a)
\end{aligned}
$$

THEOREM 2(b): $x \cdot 0 = 0$ by duality.

THEOREM 6(a): $x + xy = x$.

$$
\begin{aligned}
x + xy &= x \cdot 1 + xy && \text{by postulate:} \quad 2(b) \\
&= x(1 + y) && 4(a) \\
&= x(y + 1) && 3(a) \\
&= x \cdot 1 && 2(a) \\
&= x && 2(b)
\end{aligned}
$$

THEOREM 6(b): $x(x + y) = x$ by duality.

- The theorems of Boolean algebra can be shown to hold true by means of truth tables.
- In truth tables, both sides of the relation are checked to yield identical results for all possible combinations of variables involved.

- The following truth table verifies the first absorption theorem.
  - x+xy=x

x	y	xy	$x + xy$
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

The truth table for the first DeMorgan's theorem (x + y)' = x'y' is shown below

x	y	$x + y$	$(x + y)'$	x'	y'	$x'y'$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

Operator Precedence

The operator precedence for evaluating Boolean expressions is

(1) parentheses

(2) NOT

(3) AND

(4) OR.

BOOLEAN FUNCTIONS

A binary variable can take the value of 0 or 1. A Boolean function is an expression formed with binary variables, the two binary operators OR and AND, and unary operator NOT, parentheses, and an equal sign.

For a given value of the variables, the function can be either 0 or 1. Consider, for example, the Boolean function

$$F_1 = xyz'$$

The function F_1 is equal to 1 if $x = 1$ and $y = 1$ and $z' = 1$; otherwise $F_1 = 0$.

$F_1 = xyz'$, $F_2 = x + y'z$. $F_3 = x'y'z + x'yz + xy'$, and $F_4 = xy' + x'z$

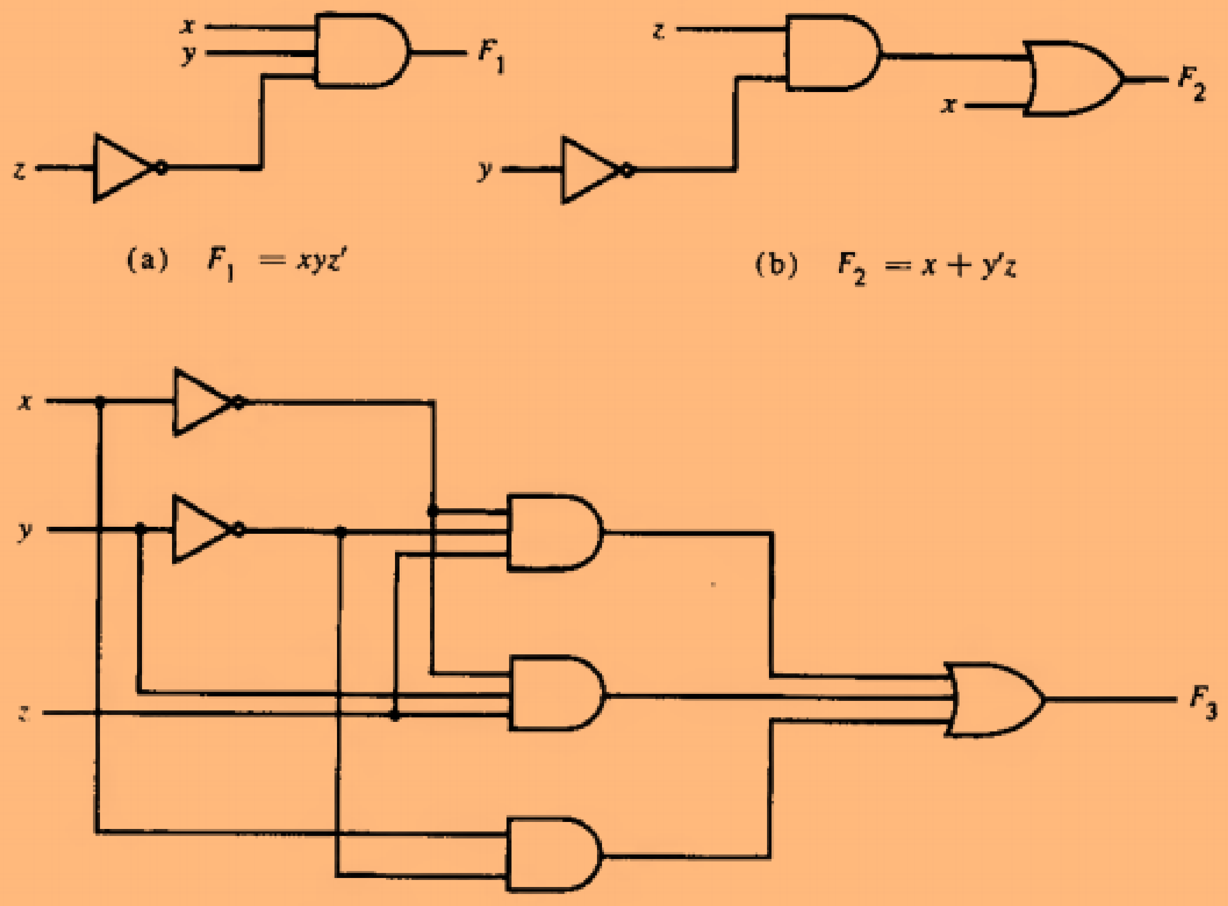
Truth Tables for $F_1 = xyz'$, $F_2 = x + y'z$, $F_3 = x'y'z + x'yz + xy'$, and $F_4 = xy' + x'z$

x	y	z	F_1	F_2	F_3	F_4
0	0	0	0	0	0	0
0	0	1	0	1	1	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

Algebraic Manipulation

- A literal is a primed or unprimed variable. When a Boolean function is implemented with logic gates, each literal in the function designates an input to a gate, and each term is implemented with a gate.
- The minimization of the number of literals and the number of terms results in a circuit with less equipment.
- It is not always possible to minimize both simultaneously; usually, further criteria must be available.

Implementation of Boolean functions with gates

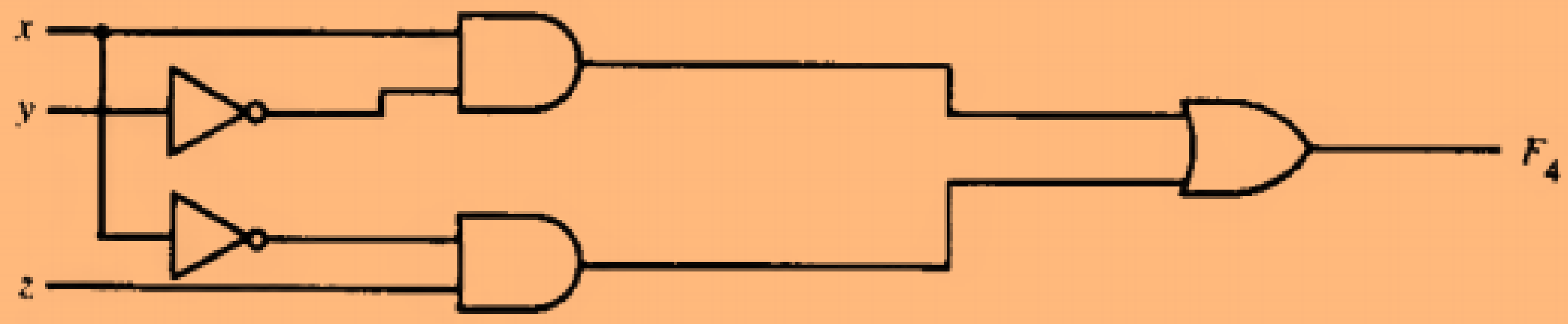


(a) $F_1 = xyz'$

(b) $F_2 = x + y'z$

(c) $F_3 = x'y'z + x'yz + xy'$

Implementation of Boolean functions with gates



(d) $F_4 = xy' + x'z$

Complement of a Function

The complement of a function F is F ' and is obtained from an interchange of 0's for l's and l's for 0's in the value of F.

The complement of a function may be derived algebraically through DeMorgan's theorem.

$(A + B + C)'$	$= (A + X)'$	let $B + C = X$
	$= A'X'$	by theorem 5(a) (DeMorgan)
	$= A' \cdot (B + C)'$	substitute $B + C = X$
	$= A' \cdot (B'C')$	by theorem 5(a) (DeMorgan)
	$= A'B'C'$	by theorem 4(b) (associative)

Find the complement of the functions $F_1 = x'yz' + x'y'z$, By applying DeMorgan's theorem as many times as necessary

$$F'_1 = (x'yz' + x'y'z)'$$

$$= (x'yz')'(x'y'z)'$$

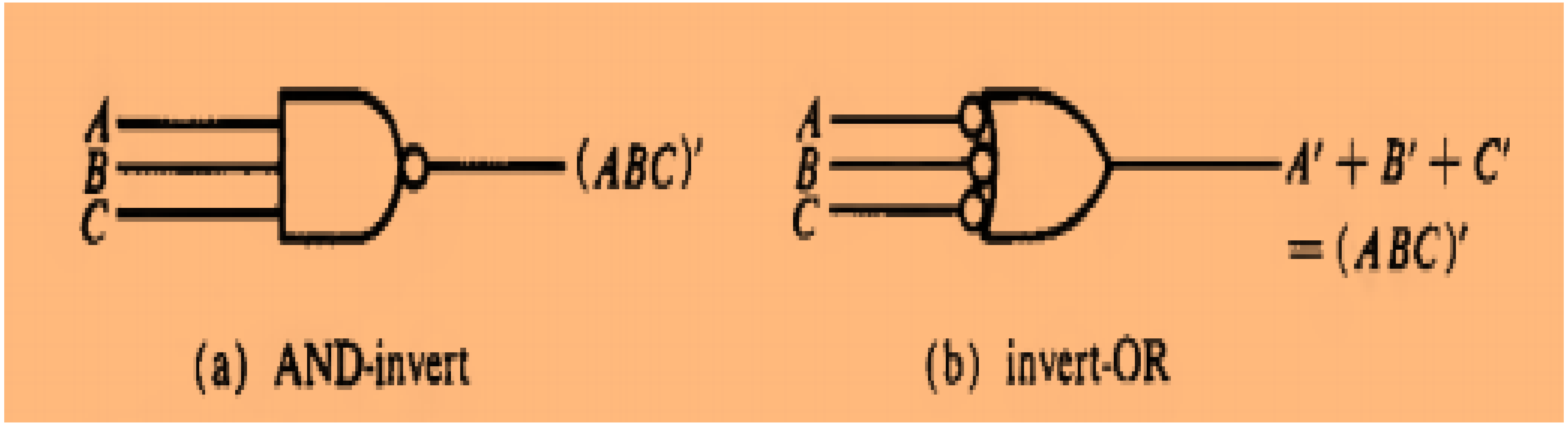
$$= (x + y' + z)(x + y + z')$$

Universal gates: NAND and NOR

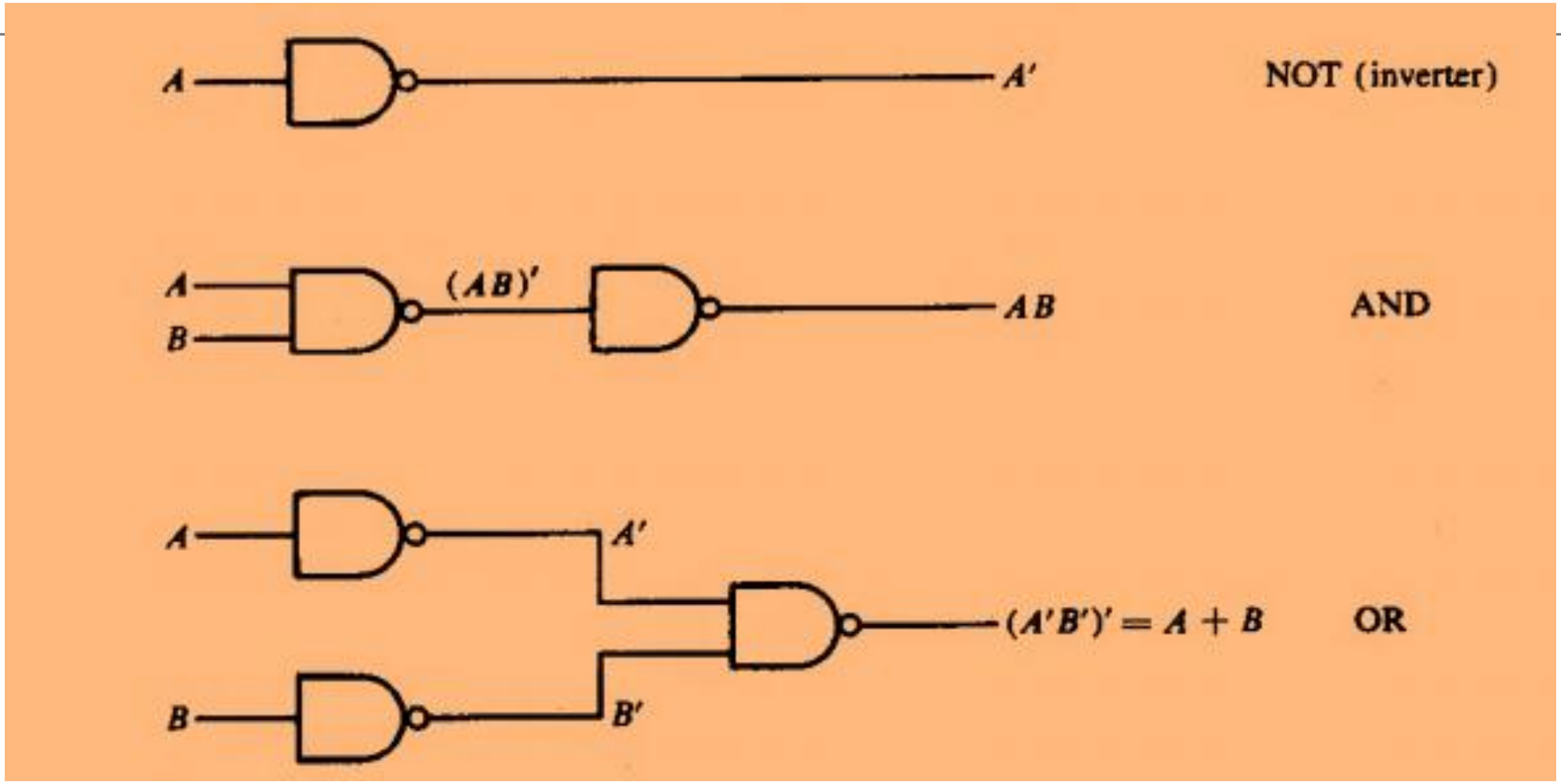
The NAND and NOR gates are said to be a universal gate because any digital system can be implemented with it.

To show that any Boolean function can be implemented with NAND gates, we need to only show that the logical operations AND, OR, and NOT can be implemented with NAND gate.

Graphical symbol for NAND gate



AND, OR and Not implementation using NAND



Check for NOR gate

CANONICAL AND STANDARD FORMS

Minterms: The binary numbers from 0 to 2" - 1 are listed under the n variables.

Each minterm is obtained from an AND term of the n variables, with each variable being primed if the corresponding bit of the binary number is a 0 and unprimed if a 1.

Maxterms: In a similar fashion, n variables forming an OR term, with each variable being primed or unprimed, provide 2" possible combinations, called maxterms, or standard sums.

Minterms and Maxterms for Three Binary Variables

			Minterms		Maxterms	
x	y	z	Term	Designation	Term	Designation
0	0	0	$x'y'z'$	m_0	$x + y + z$	M_0
0	0	1	$x'y'z$	m_1	$x + y + z'$	M_1
0	1	0	$x'yz'$	m_2	$x + y' + z$	M_2
0	1	1	$x'yz$	m_3	$x + y' + z'$	M_3
1	0	0	$xy'z'$	m_4	$x' + y + z$	M_4
1	0	1	$xy'z$	m_5	$x' + y + z'$	M_5
1	1	0	xyz'	m_6	$x' + y' + z$	M_6
1	1	1	xyz	m_7	$x' + y' + z'$	M_7

Example

A Boolean function may be expressed algebraically from a given truth table by forming a minterm for each combination of the variables that produces a 1 in the function, and then taking the OR of all those terms.

Functions of Three Variables

x	y	z	Function f_1	Function f_2
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$f_1 = x'y'z + xy'z' + xyz = m_1 + m_4 + m_7$$

Similarly, it may be easily verified that

$$f_2 = x'yz + xy'z + xyz' + xyz = m_3 + m_5 + m_6 + m_7$$

Now consider the complement of a Boolean function.

It may be read from the truth table by forming a minterm for each combination that produces a 0 in the function and then ORing those terms. The complement of f_1 is read as

$$f'_1 = x'y'z' + x'yz' + x'yz + xy'z + xyz'$$

If we take the complement of f'_1, we obtain the function f_1:

$$f_1 = (x + y + z)(x + y' + z)(x + y' + z')(x' + y + z')(x' + y' + z)$$

$$= Mo\ M_2\ M_3\ M_5\ M_6$$

Sum of minterm

It was previously stated that for n binary variables, one can obtain 2^n distinct minterms, and that any Boolean function can be expressed as a sum of minterms.

The minterms whose sum define the Boolean function are those that give the l's of the function in a truth table.

Example: Express the Boolean function F = A + B’C in a sum of minterms.

The function has three variables, A, B, and C. The first term A is missing two variables; therefore:

A = A(B + B’) = AB + AB’
This is still missing one variable: A = AB(C + C') + AB'(C + C’)
= ABC + ABC ' + AB'C + AB'C’

The second term B’C is missing one variable:
B’C = B'C(A + A’) = AB'C + A‘B'C

Combining ail terms, we have
f = A + B 'C = ABC + A BC ' + AB'C + AB'C’ + AB'C + A‘B'C

But AB 'C appears twice, and according to theorem 1 (x + x = 1)

Rearranging the min terms in ascending order, we finally obtain
F = A'B'C + AB ‘C’ + AB'C + ABC’ + ABC
= m1 + m4 + m5 + m6 + m7

It is sometimes convenient to express the Boolean function, when in its sum of min terms, in the following short notation: F(A, B, C) = $\sum$(1, 4, 5, 6, 7)

Example: Express the Boolean function F = A + B'C in a sum of minterms.

An alternate procedure for deriving the minterms of a Boolean function is to obtain the truth table of the function directly from the algebraic expression and then read the minterms from the truth table.

Truth Table for $F = A + B'C$

A	B	C	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

From the truth table,
we can then read the five minterms of the function to be 1, 4, 5, 6, and 7.

Product of Maxterm

Each of the 2^n functions of n binary variables can be also expressed as a product of maxterms.

To express the Boolean function as a product of maxterms, it must first be brought into a form of OR terms.

This may be done by using the distributive law, $x + yz = (x + y)(x + z)$. Then any missing variable x in each OR term is ORed with xx'.

Express the Boolean function F = xy + x 'z in a product of maxterm form. First, convert the function into OR terms using the distributive law:

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

The function has three variables: x, y, and z. Each OR term is missing one variable; therefore:

$$x' + y = x' + y + zz' = (x' + y + z)(x' + y + z')$$
$$x + z = x + z + yy' = (x + y + z)(x + y' + z)$$
$$y + z = y + z + xx' = (x + y + z)(x' + y + z)$$

Combining all the terms and removing those that appear more than once, we finally obtain:

$$F = (x + y + z)(x + y' + z)(x' + y + z)(x' + y + z')$$
$$= M_0M_2M_4M_5$$

A convenient way to express this function is as follows:

$$F(x, y, z) = \Pi(0, 2, 4, 5)$$

The product symbol, Π, denotes the ANDing of maxterms; the numbers are the maxterms of the function.

Suggested Reading

- M. Morris Mano, Digital Logic and Computer Design, PHI.

Thank you