Boolean Algebra

outline

- Boolean Algebra
  - Basic Boolean Equations
  - Multiple Level Logic Representation
  - Basic Identities
  - Algebraic Manipulation
  - Complements and Duals

Overview

- Gate Circuits and Boolean Equations
  - Binary Logic and Gates
  - Boolean Algebra
  - Standard Forms
- Additional Gates and Circuits
  - Other Gate Types
  - Exclusive-OR Operator and Gates
  - High-Impedance Outputs

Binary Logic and Gates

- Digital circuits are hardware components (based on transistors) that manipulate binary information
- We model the transistor-based electronic circuits as logic gates.

- Boolean algebra is a mathematical system for the manipulation of variables that can have one of two values.
  - In formal logic, these values are “true” and “false.”
  - In digital systems, these values are “on” and “off,” 1 and 0, or “high” and “low.”
- Boolean expressions are created by performing operations on Boolean variables.
  - Common Boolean operators include AND, OR, and NOT.

Binary Logic

- Binary variables take on one of two values.
- Logical operators operate on binary values and binary variables.
- Basic logical operators are the logic functions AND, OR and NOT.
- Logic gates implement logic functions.
- Boolean Algebra: a useful mathematical system for specifying and transforming logic functions.

Boolean values

two discrete values 1 and 0, from which binary numbers

For simplicity, we often still write digits instead:

- 1 (high) is true
- 0 (low) is false

- We will use these values 1 and 0 as the elements of our Boolean System.

Binary Variables

- Recall that the two binary values have different names:
  - True/False
  - On/Off
  - Yes/No
  - 1/0
- We use 1 and 0 to denote the two values.

APPLICATION OF BOOLEAN ALGEBRA

- It is used to perform the logical operations in digital computer.
- In digital computer True represent by ‘1’ (high volt) and False represent by ‘0’ (low volt)
- Logical operations are performed by logical operators. The fundamental logical operators are:
  1. AND (conjunction)
  2. OR (disjunction)
  3. NOT (negation/complement)

Logical Operations

- The three basic logical operations are:
  - AND
  - OR
  - NOT
- AND is denoted by a dot (·).
- OR is denoted by a plus (+).
- NOT is denoted by an overbar (¯), a single quote mark (') after, or (~) before the variable.

Basic Identities of Boolean Algebra

- 1. $X + 0 = X$

OR +

A	0	RESULT
0	0	0
1	0	1

- 2. $X \cdot 1 = X$

AND •

A	1	RESULT
0	1	0
1	1	1

- 3. $X + 1 = 1$
- 4. $X \cdot 0 = 0$
- 5. $X + X = X$
- 6. $X \cdot X = X$

Basic Identities

- 7. $X + X' = 1$

X	X'	RES
0	1	1
1	0	1

- 8. $X \cdot X' = 0$

X	X'	RES
0	1	0
1	0	0

- 9. $(X')' = X$

Basic Properties

- Commutative
  - 10. $X + Y = Y + X$
- Associative
  - 12. $X+(Y+Z)=(X+Y)+Z$
- Distributive
  - 14. $X(Y+Z) = XY+XZ$
  - AND distributes over OR

- Commutative
  - 11. $X \cdot Y = Y \cdot X$
- Associative
  - 13. $X(YZ) = (XY)Z$
- Distributive
  - 15. $X+YZ=(X+Y)(X+Z)$
  - OR distributes over AND

Basic Properties

- DeMorgan’s Theorem
- Very important in simplifying equations
  - 16. (X + Y)’ = X’ • Y’
  - 17. (XY)’ = X’ + Y’

X	Y	$X+Y$	$\overline{X+Y}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

X	Y	$\overline{X}$	$\overline{Y}$	$\overline{X}\bullet\overline{Y}$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

DeMorgan's Theorem

- $(x + y)' = x'y'$,$(xy)' = x' + y'$
- By means of truth table

x	y	x'	y'	$x+y$	$(x+y)'$	$x'y'$	xy	$x'+y'$	$(xy)'$
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

Proofing the theorems using axioms

- Idempotency: $x + x = x$

Proof:
$$\begin{aligned} x + x &= (x + x) \bullet 1 && \text{by identity} \\ &= (x + x) \bullet (x + x') && \text{by complement} \\ &= x + x \bullet x' && \text{by distributivity} \\ &= x + 0 && \text{by complement} \\ &= x && \text{by identity} \end{aligned}$$

- Idempotency: $x \bullet x = x$

Proof:
$$\begin{aligned} x \bullet x &= (x \bullet x) + 0 && \text{by identity} \\ &= (x \bullet x) + (x \bullet x') && \text{by complement} \\ &= x \bullet (x + x') && \text{by distributivity} \\ &= x \bullet 1 && \text{by complement} \\ &= x && \text{by identity} \end{aligned}$$

Implementation

- Boolean Algebra applied in computers electronic circuits. These circuits perform Boolean operations and these are called logic circuits or logic gates.

- Computers are implementations of Boolean logic.
- Boolean functions are completely described by truth tables.
- Logic gates are small circuits that implement Boolean operators.
- The **basic gates** are AND, OR, and NOT.
    - The XOR gate is very useful in parity checkers and adders.
- The “**universal gates**” are NOR, and NAND.
- The **Special Gates** XOR andXNOR

Logic Gate

Logic Gate

- A gate is an digital circuit which operates on one or more signals and produce single output.
- Gates are digital circuits because the input and output signals are denoted by either 1(high voltage) or 0(low voltage).
- A Boolean operator can be completely described using a truth table.

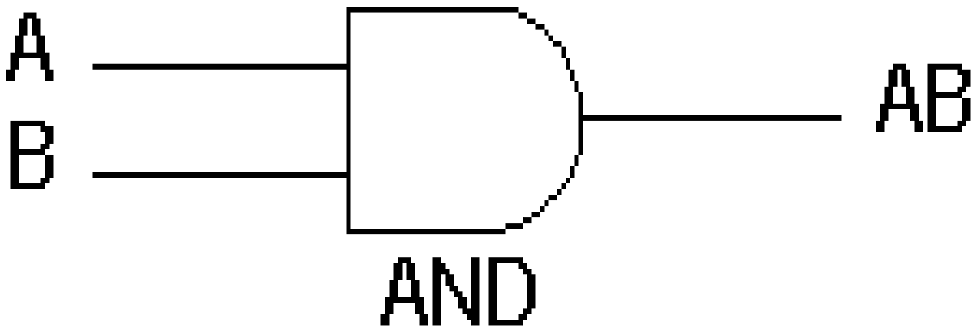
There are three basic gates :

1. **AND gate**
2. **OR gate**
3. **NOT gate**

Basic gates

AND gate

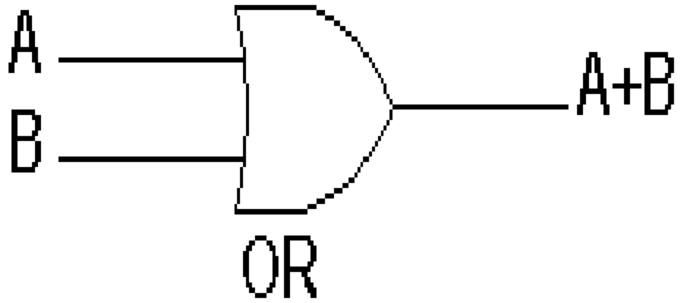
- The AND gate is an electronic circuit that gives a high output (1) only if all its inputs are high. It performs logical multiplication and denoted by (.) dot.
- AND gate takes two or more input signals and produce only one output signal. The AND operator is also known as a Boolean product.



Input A	Input B	Output A.B
0	0	0
0	1	0
1	0	0
1	1	1

OR gate

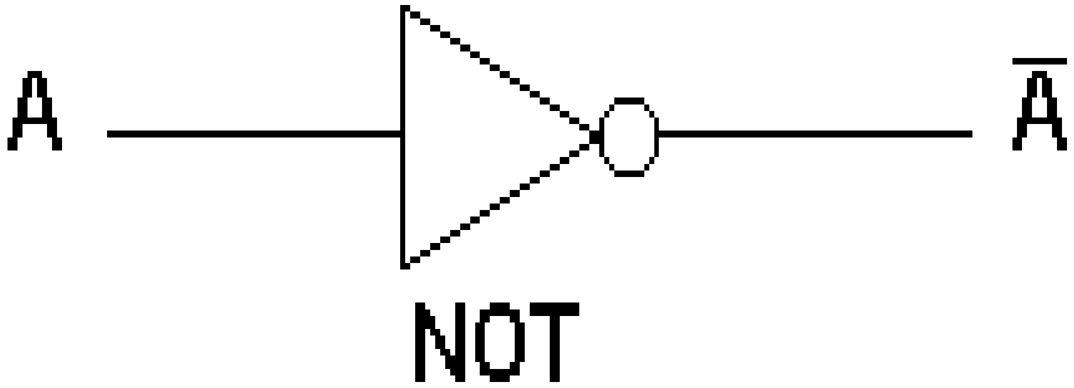
- The OR gate is an electronic circuit that gives a high output (1) if one or more of its inputs are high.
- OR gate also takes two or more input signals and produce only one output signal.
- It performs logical addition and denoted by (+) plus.
- The OR operator is the Boolean sum.



Input A	Input B	Output A+B
0	0	0
0	1	1
1	0	1
1	1	1

NOT gate

- The NOT gate is an electronic circuit that gives a high output (1) if its input is low .
- NOT gate takes only one input signal and produce only one output signal.
- The output of NOT gate is complement of its input.
- It is also called inverter.
- It performs logical negation and denoted by (-) bar. It operates on single variable.
- It is sometimes indicated by a prime mark (‘) or an “elbow” (¬).



Input A	Output $\overline{A}$
0	1
1	0

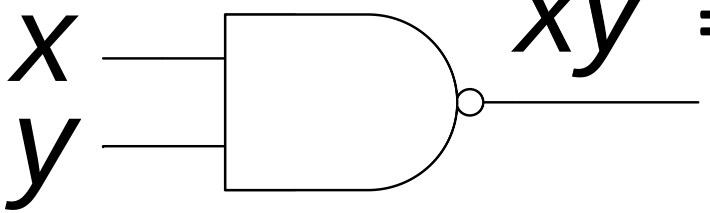
Universal gate

NAND Gate

Known as a “universal” gate because ANY digital circuit can be implemented with NAND gates alone.

NAND Gate

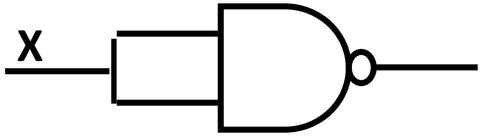
NAND



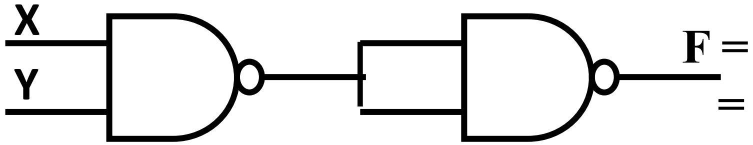
$\overline{xy}$ =Z

X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

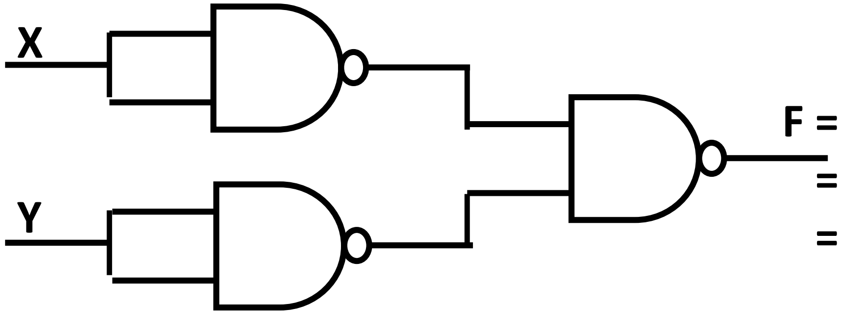
NAND Gate



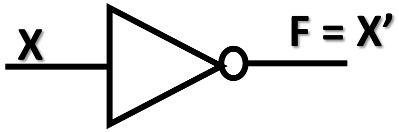
$F = (X \bullet X)'$
$= X' + X'$
$= X'$

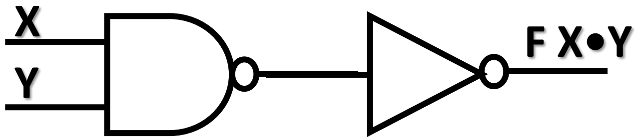


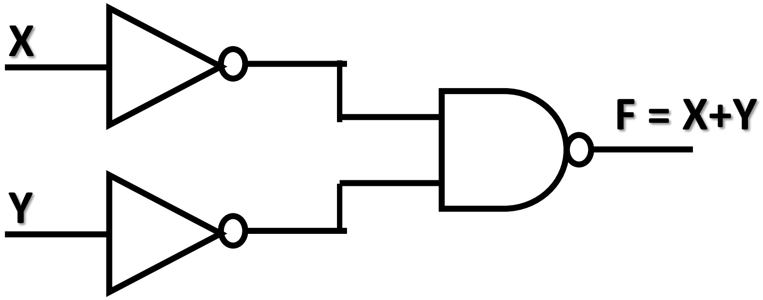
$F = ((X \bullet Y)')'$
$= (X' + Y')'$
$= X'' \bullet Y''$
$= X \bullet Y$



$F = (X' \bullet Y')'$
$= X'' + Y''$
$= X + Y$

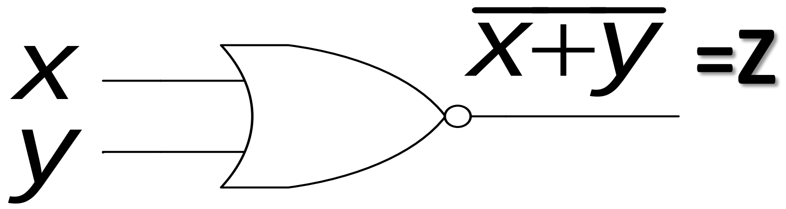






NOR Gate

NOR

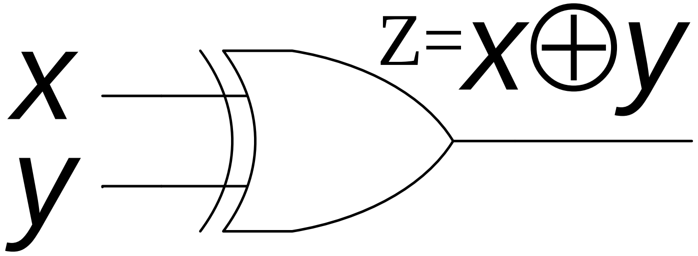


X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	0

Special Gates

Exclusive-OR Gate

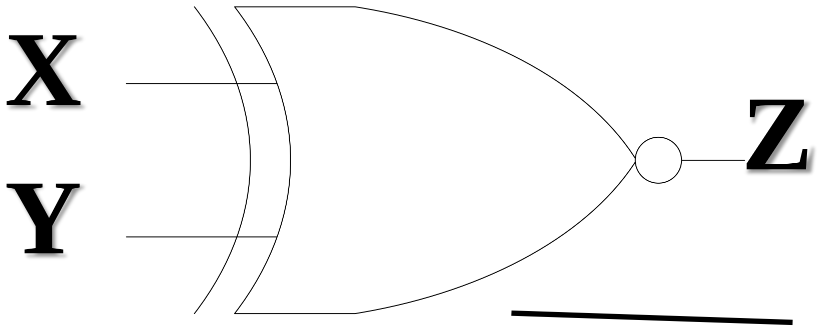
XOR



X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

Exclusive-NOR Gate

XNOR



$$A \odot B = \overline{x \oplus y}$$

X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	1

Truth Table

- Truth table is a table that contains all possible values of logical variables/statements in a Boolean expression.

No. of possible combination = 2^n,

where n = number of variables used in a Boolean expression.

Truth Tables

- ***Truth table*** **-** a tabular listing of the values of a function **for all possible combinations** of values on its arguments
- Example: Truth tables for the basic logic operations:

AND		
X	**Y**	**$Z = X \cdot Y$**
0	0	0
0	1	0
1	0	0
1	1	1

OR		
X	**Y**	**$Z = X + Y$**
0	0	0
0	1	1
1	0	1
1	1	1

NOT	
X	**$Z = \overline{X}$**
0	1
1	0

Truth Table

The truth table for XY + Z is as follows:

Dec	X	Y	Z	XY	XY+Z
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	0
3	0	1	1	0	1
4	1	0	0	0	0
5	1	0	1	0	1
6	1	1	0	1	1
7	1	1	1	1	1

- DeMorgan's law can be extended to any number of variables.
- Replace each variable by its complement and change all ANDs to ORs and all ORs to ANDs.
- The complement of:

$$F(X,Y,Z) = (XY) + (\overline{X}Z) + (Y\overline{Z})$$

is:

$$\begin{aligned}\overline{F}(X,Y,Z) &= \overline{(XY) + (\overline{X}Z) + (Y\overline{Z})} \\ &= \overline{(XY)}\,\overline{(\overline{X}Z)}\,\overline{(Y\overline{Z})} \\ &= (\overline{X}+\overline{Y})(X+\overline{Z})(\overline{Y}+Z)\end{aligned}$$

Tautology & Fallacy

Tautology & Fallacy

- If the output of Boolean expression is always True or 1 is called Tautology.

- If the output of Boolean expression is always False or 0 is called Fallacy.

- A Boolean function has:
  - At least one Boolean variable,
  - At least one Boolean operator, and
  - At least one input from the set {0,1}.
- It produces an output that is also a member of the set {0,1}.

- The truth table for the Boolean function:

$$F(x,y,z) = x\overline{z}+y$$

is shown at the right.

- To make evaluation of the Boolean function easier, the truth table contains extra (shaded) columns to hold evaluations of subparts of the function.

$$F(x,y,z) = x\overline{z}+y$$

x	y	z	$\overline{z}$	$x\overline{z}$	$x\overline{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	0	1

Exercise

1. Evaluate the following Boolean expression using Truth Table.

(a) $X'Y'+X'Y$ (b) $X'YZ'+XY'$ (c) $XY'(Z+YZ')+Z'$

2. Verify that $P+(PQ)'$ is a Tautology.

3. Verify that $(X+Y)'=X'Y'$

Boolean Functions

- Computers take inputs and produce outputs, just like functions in math!
- Mathematical functions can be expressed in two ways:
- We can represent logical functions in two analogous ways too:
  - A finite, but non-unique Boolean expression.
  - A truth table, which will turn out to be unique *and* finite.

An expression is

finite but not unique

$$f(x,y) = 2x + y$$

$$= x + x + y$$

$$= 2(x + y/2)$$

$$= \ldots$$

A function table is

unique but infinite

x	y	f(x,y)
0	0	0
…	…	…
2	2	6
…	…	…
23	41	87
…	…	…

Boolean expressions

- We can use these basic operations to form more complex expressions:

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

- Some terminology and notation:
  - f is the name of the function.
  - (x,y,z) are the input variables, each representing 1 or 0. Listing the inputs is optional, but sometimes helpful.
  - A literal is any occurrence of an input variable or its complement. The function above has four literals: x, y', z, and x'.

Precedence is important, but not too difficult.

NOT has the highest precedence, followed by AND, and then OR.

Fully parenthesized,

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

- As with common arithmetic, Boolean operations have rules of precedence.
- The NOT operator has highest priority, followed by AND and then OR.
- This is how we chose the (shaded) function subparts in our table.

$F(x,y,z) = x\bar{z}+y$

x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	0	1

Truth tables

- A truth table shows all possible inputs and outputs of a function.
- Remember that each input variable represents either 1 or 0.
  - Because there are only a finite number of values (1 and 0), truth tables themselves are finite.
  - A function with n variables has 2^n possible combinations of inputs.
- Inputs are listed in binary order—in this example, from 000 to 111.

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

$f(0,0,0)$	$= (0 + 1)0 + 1$	$= 1$
$f(0,0,1)$	$= (0 + 1)1 + 1$	$= 1$
$f(0,1,0)$	$= (0 + 0)0 + 1$	$= 1$
$f(0,1,1)$	$= (0 + 0)1 + 1$	$= 1$
$f(1,0,0)$	$= (1 + 1)0 + 0$	$= 0$
$f(1,0,1)$	$= (1 + 1)1 + 0$	$= 1$
$f(1,1,0)$	$= (1 + 0)0 + 0$	$= 0$
$f(1,1,1)$	$= (1 + 0)1 + 0$	$= 1$

x	y	z	f(x,y,z)
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Complement of a function

- The complement of a function always outputs 0 where the original function outputted 1, and 1 where the original produced 0.
- In a truth table, we can just exchange 0s and 1s in the output column(s)

$$f(x,y,z) = x\,(y'z' + yz)$$

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

x	y	z	f'(x,y,z)
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Complementing a function algebraically

- You can use DeMorgan's law to keep "pushing" the complements inwards

$$f(x,y,z) = x\,(y'\,z' + y\,z)$$

$$\begin{aligned}
f'(x,y,z) &= (\,x(y'z' + yz)\,)' && [\text{ complement both sides }]\\
&= x' + (y'z' + yz)' && [\text{ because } (xy)' = x' + y'\ ]\\
&= x' + (y'z')'\,(yz)' && [\text{ because } (x + y)' = x'\,y'\ ]\\
&= x' + (y + z)(y' + z') && [\text{ because } (xy)' = x' + y',\text{ twice}]
\end{aligned}$$

- You can also take the dual of the function, and then complement each literal
  - If $f(x,y,z) = x(y'z' + yz)$…
  - …the dual of f is $x + (y' + z')(y + z)$…
  - …then complementing each literal gives $x' + (y + z)(y' + z')$…
  - …so $f'(x,y,z) = x' + (y + z)(y' + z')$

- There are two canonical forms for Boolean expressions: sum-of-products and product-of-sums.
  - Recall the Boolean product is the AND operation and the Boolean sum is the OR operation.
- In the sum-of-products form, ANDed variables are ORed together.
  - For example: $F(x,y,z) = xy + xz + yz$
- In the product-of-sums form, ORed variables are ANDed together:
  - For example:

$$F(x,y,z) = (x+y)(x+z)(y+z)$$

- It is easy to convert a function to sum-of-products form using its truth table.
- We are interested in the values of the variables that make the function true (=1).
- Using the truth table, we list the values of the variables that result in a true function value.
- Each group of variables is then ORed together.

$$F(x,y,z) = x\bar{z}+y$$

x	y	z	$x\bar{z}+y$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

- The sum-of-products form for our function is:

$$F(x,y,z) = \bar{x}y\bar{z}+\bar{x}yz+x\bar{y}\bar{z}+xy\bar{z}+xyz$$

We note that this function is not in simplest terms. Our aim is only to rewrite our function in canonical sum-of-products form.

$$F(x,y,z) = x\bar{z}+y$$

x	y	z	$x\bar{z}+y$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

- Boolean functions are implemented in digital computer circuits called gates.
- A gate is an electronic device that produces a result based on two or more input values.
    - In reality, gates consist of one to six transistors, but digital designers think of them as a single unit.
    - Integrated circuits contain collections of gates suited to a particular purpose.

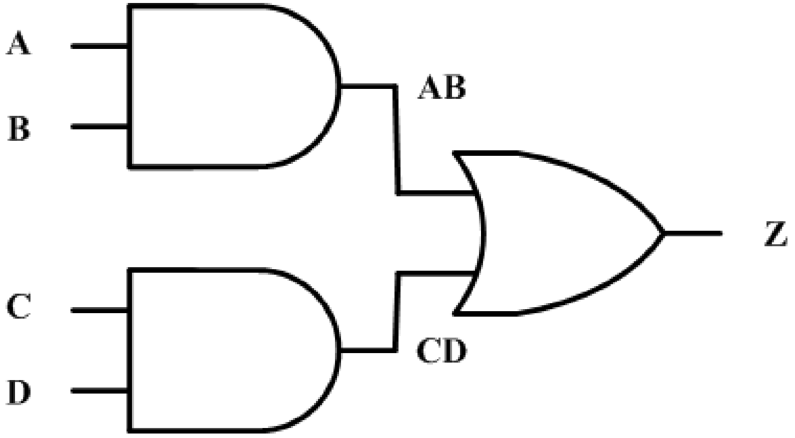
Basic Boolean Equations

- For the basic gates/functions
- AND
  - $Z = A\,B$
  - $X = C\,D\,E$ 3 input gate
  - $Y = F\,G\,H\,K$ 4 input gate
- OR
  - $Z = A + B$
  - $Y = F + G + H + K$ 4 input gate
- NOT
  - $Z = \overline{A}$
  - $Y = (\overline{F\,G\,H\,K})$ actually 2 level logic

- Consider the following logic equation
  - Z (A,B,C,D) = A B + C D
  - The Z (A,B,C,D) means that the output is a function of the four variables within the ( ).
  - The AB and CD are terms of the expression.
  - This form of representing the function is an algebraic expression.
  - For this function to be True, either both A AND B are True OR both C AND D are True.

Truth table expression

- the truth tables for the basic functions, we can also construct truth tables for any function.



A	B	C	D	Z	AB	CD
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	0	1	1	1	0	1
0	1	0	0	0	0	0
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	1	1	1	0	1
1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0
1	0	1	1	1	0	1
1	1	0	0	1	1	0
1	1	0	1	1	1	0
1	1	1	0	1	1	0
1	1	1	1	1	1	1

Examples of Boolean Equations

- Some examples
  - $F = AB + CD + BD'$
  - $Y = CD + A'B'$
  - quations can be very complex
  - Usually desire a minimal expression

Simplify

- Digital computers contain circuits that implement Boolean functions.
- The simpler that we can make a Boolean function, the smaller the circuit that will result.
  - Simpler circuits are cheaper to build, consume less power, and run faster than complex circuits.
- With this in mind, we always want to reduce our Boolean functions to their simplest form.
- There are a number of Boolean identities that help us to do this.

- Most Boolean identities have an AND (product) form as well as an OR (sum) form. We give our identities using both forms. Our first group is rather intuitive:

Identity Name	AND Form	OR Form
Identity Law	$1x = x$	$0 + x = x$
Null Law	$0x = 0$	$1 + x = 1$
Idempotent Law	$xx = x$	$x + x = x$
Inverse Law	$x\overline{x} = 0$	$x + \overline{x} = 1$

- Second group of Boolean identities should be familiar to you from your study of algebra:

Identity Name	AND Form	OR Form
Commutative Law	$xy = yx$	$x+y = y+x$
Associative Law	$(xy)z = x(yz)$	$(x+y)+z = x + (y+z)$
Distributive Law	$x+yz = (x+y)(x+z)$	$x(y+z) = xy+xz$

• Last group of Boolean identities are perhaps the most useful.

Identity Name	AND Form	OR Form
Absorption Law	$x(x+y) = x$	$x + xy = x$
DeMorgan's Law	$\overline{(xy)} = \overline{x} + \overline{y}$	$\overline{(x+y)} = \overline{x}\,\overline{y}$
Double Complement Law	$\overline{(\overline{x})} = x$	

- We can use Boolean identities to simplify the function:

$$F(X,Y,Z) = (X + Y)(X + \overline{Y})(\overline{X\overline{Z}})$$

as follows:

$(X + Y)(X + \overline{Y})(\overline{X\overline{Z}})$	Idempotent Law (Rewriting)
$(X + Y)(X + \overline{Y})(\overline{X} + Z)$	DeMorgan's Law
$(XX + X\overline{Y} + XY + Y\overline{Y})(\overline{X} + Z)$	Distributive Law
$((X + Y\overline{Y}) + X(Y + \overline{Y}))(\overline{X} + Z)$	Commutative & Distributive Laws
$((X + 0) + X(1))(\overline{X} + Z)$	Inverse Law
$X(\overline{X} + Z)$	Idempotent Law
$X\overline{X} + XZ$	Distributive Law
$0 + XZ$	Inverse Law
XZ	Idempotent Law

Simplify

- These properties (Laws and Theorems) can be used to simplify equations to their simplest form.
  - Simplify F=X’YZ+X’YZ’+XZ

$$F = \overline{X}YZ + \overline{X}Y\overline{Z} + XZ$$

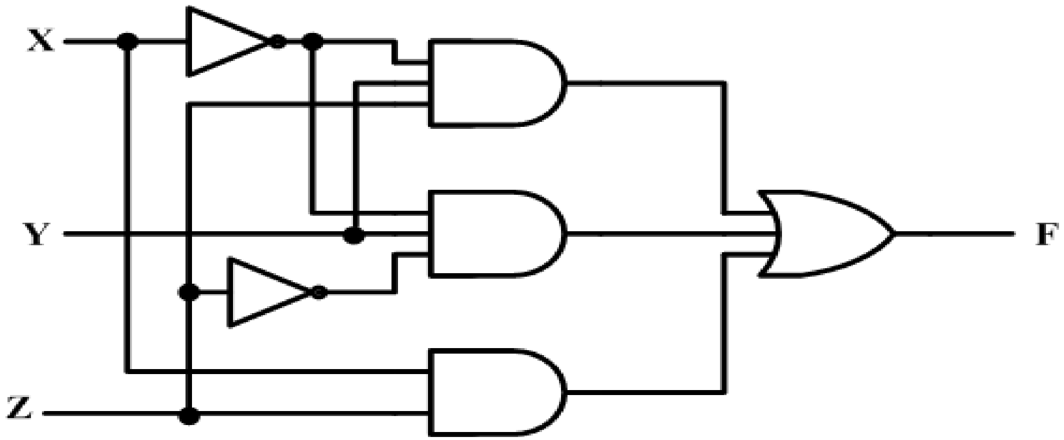
$$= \overline{X}Y(Z + \overline{Z}) + XZ \qquad \text{by identity 14}$$

$$= \overline{X}Y \bullet 1 + XZ \qquad \text{by identity 7}$$

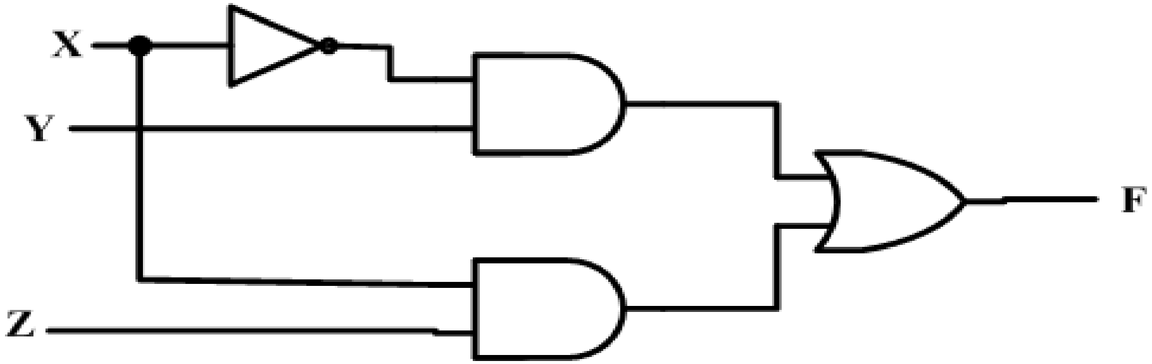
$$= \overline{X}Y + XZ \qquad \text{by identity 2}$$

Affect on implementation

- F = X’YZ + X’YZ’ + XZ



- Reduces to F = X’Y + XZ



Other examples

- 1. $X + XY = X \cdot 1 + XY = X(1+Y) = X \cdot 1 = X$

- 2. $XY + XY' = X(Y + Y') = X \cdot 1 = X$

- 3. $X + X'Y = (X+X')(X+Y) = 1 \cdot (X+Y) = X+Y$

- 4. $X\cdot (X+Y)=X\cdot X+X\cdot Y= X+XY=X(1+Y)=X\cdot 1=X$
- 5. $(X+Y)\cdot(X+Y')=XX+XY'+XY+YY'=$

  $X+XY'+XY+0=X(1+Y'+Y)=X\cdot 1=X$

- 6. $X(X'+Y) = XX'+XY = 0 + XY = XY$

- The Theorem gives us the relationship

  - $XY + X'Z + YZ = XY + X'Z$

  - $XY + X'Z + 1 \cdot YZ = XY + X'Z + (X+X')YZ$

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

$$= xy(1+z) + X'Z(1+y)$$

$$= XY + X'Z$$

Application of Consensus Theorem

- $(A+B)(A'+C) = AA' + AC + A'B + BC$

  $= AC + A'B + BC$

  $= AC + A'B$

Complement of a function

- In real implementation sometimes the complement of a function is needed.
  - Have F=X’YZ’+X’Y’Z

$$F = \overline{X}Y\overline{Z} + \overline{X}\,\overline{Y}Z$$

$$\overline{F} = \overline{\overline{X}Y\overline{Z} + \overline{X}\,\overline{Y}Z}$$

$$= (\overline{\overline{X}Y\overline{Z}})\bullet(\overline{\overline{X}\,\overline{Y}Z})$$

$$= (X+\overline{Y}+Z)\bullet(X+Y+\overline{Z})$$

Duals

- What is meant by the dual of a function?
  - The *dual* of a function is obtained by interchanging OR and AND operations and replacing 1s and 0s with 0s and 1s.
- Shortcut to getting function complement
  - Starting with the equation on the previous slide
  - Generate the dual F=(X'+Y+Z')(X'+Y'+Z)
  - Complement each literal to get:
  - F'=(X+Y'+Z)(X+Y+Z')

Duality principle

- The left and right columns of axioms are duals
  - exchange all ANDs with ORs, and 0s with 1s

1.	$x + y \in B$	$x \bullet y \in B$	Closure
2.	$x + 0 = x$	$x \bullet 1 = x$	Identity
3.	$x + y = y + x$	$xy = yx$	Commutativity
4.	$x(y + z) = xy + xz$	$x + yz = (x + y)(x + z)$	Distributivity
5.	$x + x' = 1$	$x \bullet x' = 0$	Complement
6.	At least 2 elements: $x,y \in B$ such that $x \neq y$		Cardinality

- So are the theorems:

1.	$x + x = x$	$x \bullet x = x$	Idempotency
2.	$x + 1 = 1$	$x \bullet 0 = 0$	
3.	$yx + x = x$	$(y + x) \bullet x = x$	Absorption
4.	$(x')' = x$		Involution
5.	$x + (y + z) = (x + y) + z$	$x(yz) = (xy)z$	Associative
6.	$(x + y)' = x'y'$	$(xy)' = x' + y'$	DeMorgan's

Algebraic manipulation

- We can now start doing some simplifications

$x'y' + xyz + x'y$

$= x'(y' + y) + xyz$	[Distributive; $x'y' + x'y = x'(y' + y)$]
$= x' \bullet 1 + xyz$	[Axiom 5; $y' + y = 1$]
$= x' + xyz$	[Axiom 2; $x' \bullet 1 = x'$]
$= (x' + x)(x' + yz)$	[Distributive]
$= 1 \bullet (x' + yz)$	[Axiom 5; $x' + x = 1$]
$= x' + yz$	[Axiom 2 ; $x' \bullet 1 = x'$]

Function Minimization using Boolean Algebra

- ***Examples***:

(a) $a + ab = a(1+b)=a$

(b) $a(a + b) = a.a + ab=a+ab=a(1+b)=a.$

(c) $a + a'b = (a + a')(a + b)=1(a + b) =a+b$

(d) $a(a' + b) = a. a' +ab=0+ab=ab$

Try

- $F = abc + abc' + a'c$

The other type of question

Show that;

1- $ab + ab' = a$

2- $(a + b)(a + b') = a$

1- $ab + ab' = a(b+b') = a.1=a$

2- $(a + b)(a + b') = a.a + a.b' + a.b + b.b'$

$= a + a.b' + a.b + 0$

$= a + a.(b' + b) + 0$

$= a + a.1 + 0$

$= a + a = a$

More Examples

- Show that;

  (a) $ab + ab'c = ab + ac$

  (b) $(a + b)(a + b' + c) = a + bc$

(a) $ab + ab'c = a(b + b'c)$

$= a((b+b').(b+c))=a(b+c)=ab+ac$

(b) $(a + b)(a + b' + c)$

$= (a.a + a.b' + a.c + ab +b.b' +bc)$

$= \ldots$