Lecture - 6 & 7
Formulas with distinct truth tables

Recall the previous lectures; primary statements, truth tables, compound statements, connectives, negation, conjunction, disjunction, statement formulas, conditional and biconditional statements, converse, contrapositive and inverse, well-formed formula, tautology, substitution instances, equivalence of formulas, replacement process, duality laws and tautological implications.

Using connectives and the rule of constructing well-formed formula we have constructed several statement formulas. We shall now see that how many of these these formulas have distinct truth tables.

Let us consider all possible truth tables that can be obtained when the formulas involve only one variable P. These possible truth tables are shown in the table below;

P	1	2	3	4
T	T	F	T	F
F	F	T	T	F

Any formula involving only one variable will have one of these four truth tables. The most simplest formulas having the truth tables $1, 2, 3$ and 4 are $P, \neg P, P \vee \neg P$ and $P \wedge \neg P$, respectively. Any other formula which involves P only will be equivalent to one of these four formulas.

Let us now consider the formulas obtained by using the two variables and any connectives, then we shall get several formulas. The number of distinct truth tables for formulas involving two variables is equals to $2^{2^2} = 16$. Since there are there are 2^2 rows in the truth tables and each row could have any of the two entries T or F, therefore, we have 2^{2^2} possible tables (given in the table below);

P	Q	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
T	T	T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F
T	F	T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F
F	T	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F
F	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F

Thus, any formula involving two variables will have one of these 16 truth tables. Also, formulas which have one of these truth tables are equivalent.
Similarly, a statement formula containing n variables must have as its truth table of the 2^{2^n} possible truth tables and each of them will have 2^n rows. Thus we see that there are several formulas which may look different but are equivalent.

One method to check whether two formulas A and B are equivalent is to construct their truth table and compare them. However, this method is tedious and difficult to perform even on a computer because the number of entries increase very rapidly as n increases. Therefore, another approach could be to transform A and B to some standard forms A' and B' and compare A' and B' to check whether $A \iff B$. These standard forms are called canonical or normal forms.

Functionally complete sets of connectives

We defined the connectives $\wedge, \vee, \neg, \rightarrow$ and $\leftrightarrow$. We shall see some more examples of connectives. We prove that not all the connectives defined so far are necessary. In fact, we can find certain proper subsets of these connectives which are sufficient to express any formula in an equivalent form. Any set of connectives in which every formula can be expressed in terms of an equivalent formula involving the connectives from this set is called a *functionally complete* set of connectives. We assume that a functionally complete set does not contain any redundant connectives, i.e., a connective which can be expresses in terms of the other connectives.

Consider the following equivalence:

$$P \leftrightarrow Q \iff (P \rightarrow Q) \wedge (Q \rightarrow P).$$

This means that in a formula we can replace the part involving biconditional by an equivalent formula which does not contain biconditional. Thus we can replace all the biconditional in a formula.

Example (1): Write an equivalent formula for $P \wedge (Q \leftrightarrow R) \vee (R \leftrightarrow P)$ which does not involve the biconditional.

Use the above equivalence to obtain

$$P \wedge (Q \leftrightarrow R) \vee (R \leftrightarrow P) \iff P \wedge ((Q \rightarrow R) \wedge (R \rightarrow Q)) \vee ((R \rightarrow P) \wedge (P \rightarrow R)).$$

Next, consider the equivalence $P \rightarrow Q \iff \neg P \vee Q$. Thus, we can replace the conditional also.

Example (2): Write an equivalent formula for $P \wedge (Q \leftrightarrow R)$ which contains neither conditional nor the biconditional.

$$P \wedge (Q \leftrightarrow R) \iff P \wedge ((Q \rightarrow R) \wedge (R \rightarrow Q)) \iff P \wedge ((\neg Q \vee R) \wedge (\neg R \vee Q)).$$

Let us now recall the De Morgan's law,

$$P \wedge Q \iff \neg(\neg P \vee \neg Q),\ P \vee Q \iff \neg(\neg P \wedge \neg Q).$$

Thus, in view of the first (resp. second) equivalence we can eliminate conjunction (resp. disjunction) in a formula.

If we apply all the previous equivalences, we see that in a formula, we can replace all the biconditionals, then the conditionals, and finally all the conjunctions or all the disjunctions

to obtain an equivalent formula which contains either the negation and disjunction only or the negation and the conjunction only. This means that the following sets of connectives $\{\neg, \vee\}$ and $\{\neg, \wedge\}$ are functionally complete.
Thus from the set of five connectives $\wedge, \vee, \neg, \rightarrow$ and $\leftrightarrow$we obtained at least two sets of functionally complete connectives.
We now ask the following question;
Does there exists a single connective which is functionally complete?
Answer to the above question is NO, if we consider the above five connectives only. However, there are some connectives (which we shall define) are functionally complete.

Note that if a formula is replaced by an equivalent formula in which the number of different connectives is less than the given formula, the reulting formula may become more complex.

So far we have seen that not all connectives defined are necessary. For any formula, there exists an equivalent formula which involves only those connectives belonging to one of the functionally complete set of connectives. In spite of this fact, we did define other connectives because, by using them, some of the formulas become simpler. There are some other connectives which serves the similar purpose. We shall now discuss those connectives.

Let P and Q be any two formulas. Then the formula $P \,\overline{\vee}\, Q$, is true whenever either P or Q, but not both, is true. The connective $\overline{\vee}$ is called *exclusive* OR or *exclusive disjunction*. The truth table for $\overline{\vee}$ is;

P	Q	$P \,\overline{\vee}\, Q$
T	T	F
T	F	T
F	T	T
F	F	F

We now list some equivalences which is easy to prove and hence left as an exercise;

(1) $P \,\overline{\vee}\, Q \iff Q \,\overline{\vee}\, P$ (symmetric)
(2) $(P \,\overline{\vee}\, Q) \,\overline{\vee}\, R \iff P \,\overline{\vee}\, (Q \,\overline{\vee}\, R)$ (associative)
(3) $P \wedge (Q \,\overline{\vee}\, R) \iff (P \wedge Q) \,\overline{\vee}\, (P \wedge R)$ (distributive)
(4) $(P \,\overline{\vee}\, Q) \iff (P \wedge \neg Q) \vee (\neg P \wedge Q)$
(5) $(P \,\overline{\vee}\, Q) \iff \neg(P \leftrightarrow Q)$

Exercise: If $P \,\overline{\vee}\, Q \iff R$, then $P \,\overline{\vee}\, R \iff Q$ and $Q \,\overline{\vee}\, R \iff P$, and $P \,\overline{\vee}\, Q \,\overline{\vee}\, R$ is contradiction.

Thus, for a given formula involving $\overline{\vee}$ can be replaced by an equivalent formula which contains only the connectives $\wedge, \vee$, and $\neg$ by using the equivalence (4).

We now define other connectives which are important in design of computers, called *NAND* and *NOR*. The connective *NAND* is denoted by the symbol $\uparrow$. For any two formula P and Q;

$$P \uparrow Q \iff \neg(P \wedge Q).$$

The connective *NOR* is denoted by the symbol $\downarrow$, and for formulas P and Q;

$$P \downarrow Q \iff \neg(P \vee Q).$$

The connectives $\uparrow$ and $\downarrow$ have been defined in terms of the connectives $\vee, \wedge$ and $\neg$. Therefore, for a formula involving the connectives $\uparrow$ or $\downarrow$, we can obtain an equivalent formula containing the connectives $\vee, \wedge$ and $\neg$, only.

Also, note that $\uparrow$ and $\downarrow$ are duals to each other. Therefore, in order to find the dual of a formula involving $\uparrow$ and $\downarrow$, we interchange $\uparrow$ and $\downarrow$ in addition to other interchanges which we already have discussed.

We now prove that each of the connectives $\uparrow$ and $\downarrow$ is functionally complete. To see this, it suffices to show that the set of connectives $\{\wedge, \neg\}$ and $\{\vee, \neg\}$ can be expressed either in terms of $\uparrow$ alone or in terms of $\downarrow$ alone. Which can be seen through the following equivalences;

(1) $P \uparrow P \iff \neg(P \wedge P) \iff \neg P$
(2) $(P \uparrow Q) \uparrow (P \uparrow Q) \iff \neg(P \uparrow Q) \iff P \wedge Q$
(3) $(P \uparrow P) \uparrow (Q \uparrow Q) \iff \neg P \uparrow \neg Q \iff \neg(\neg P \wedge \neg Q) \iff P \vee Q$

Similarly we express $\neg, \vee$ and $\wedge$ in terms of $\downarrow$ alone as follows;

(1) $P \downarrow P \iff \neg(P \vee P) \iff \neg P$
(2) $(P \downarrow Q) \downarrow (P \downarrow Q) \iff \neg(P \downarrow Q) \iff P \vee Q$
(3) $(P \downarrow P) \downarrow (Q \downarrow Q) \iff \neg P \downarrow \neg Q \iff P \wedge Q$

Thus, we have proved that a single operator *NAND* and *NOR* is functionally complete. Each of the sets $\{\uparrow\}$ and $\{\downarrow\}$ is called *minimal functionally complete set* or *minimal set.*

We now list some basic properties of the connectives *NAND* and *NOR*.

(1) $P \uparrow Q \iff Q \uparrow P,\ P \downarrow Q \iff Q \downarrow P$ (commutative)
(2) $P \uparrow (Q \uparrow R) \iff P \uparrow \neg(Q \wedge R) \iff \neg(P \wedge \neg(Q \wedge R)) \iff \neg P \vee (Q \wedge R)$

As

$$(P \uparrow Q) \uparrow R \iff (P \wedge Q) \vee \neg R,$$

we see that $\uparrow$ is not associative. Similarly,

$$P \downarrow (Q \downarrow R) \iff \neg P(Q \vee R),\ (P \downarrow Q) \downarrow R \iff (P \vee Q)\neg R.$$

Two-state devices and statement logic

Since we admit only those statements which have a truth value either true or false, the logic is called two-valued logic.

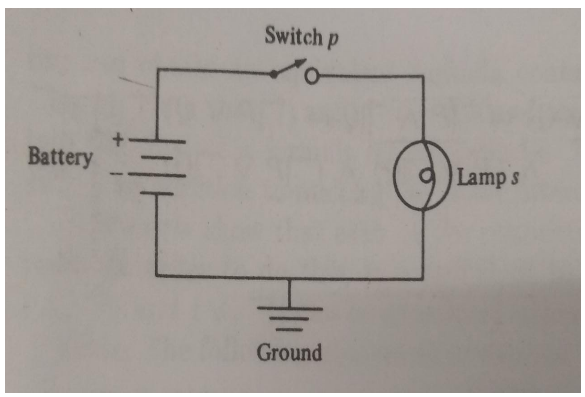
A similar situation exists in several electrical and mechanical devices which are assumed to be in one of two possible configurations and because of this reason, these devices are called two-state devices.

Let us look at some examples of such commonly known devices and then we shall discuss their connection to two-valued logic.

- An electric switch is used for turning "on" and "off" and electric light is two-state device.
- A vaccum tube or a transistor is also a two-state device in which the current is either passing or not passing.
- A mechanical clutch is either engaged or disengaged.
- A small doughnut-shaped metal disc with a wire coil wrapped around it (called a magnetic core in computers) may be magnetized in one direction if the current is passed through the coil in one way and may be magnetized in the opposite direction if the current is reversed.

A general description of such devices can be given by replacing the word "switch" by the word "gate" to mean a device which permits or stops the flow of not only electric current but any quantity that can go through the device, such as water, information, person, etc.

Consider the example of an electric lamp controlled by a mechanical switch as shown in the figure;



state of switch p	state of lamp s
closed	open
open	off

when the switch p is open, there is no current flowing in the circuit and the lamp s is "off." When p is closed, the lamp s is "on." The state of the switch and the lamp is shown in the table above.

Now consider the following statements;

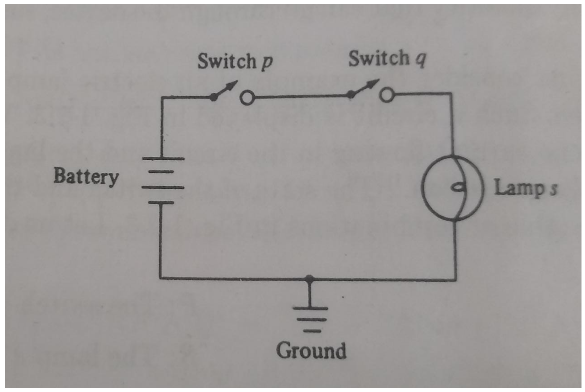
$$P : \text{The switch p is closed}, \ \ S : \text{The lamp s is on}$$

Then the above table can be rewritten as;

$p(P)$	$s(S)$
T 1	T 1
F 0	F 0

Consider an extension of the preceding circuit in which we have two switches p and q in series

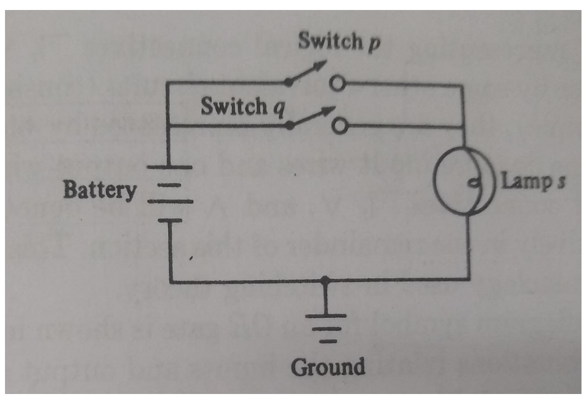
$$P: \text{The switch p is closed}, \; Q: \text{The switch q is closed}, \; S: \text{The lamp s is on}$$



P	Q	S
1	1	1
1	0	0
0	1	0
0	0	0

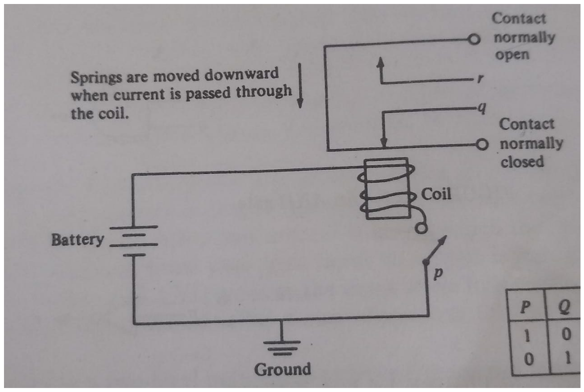
It is immediate from the table that $P \wedge Q \iff S$.

Parallel:



P	Q	S
1	1	1
1	0	1
0	1	1
0	0	0

It is immediate from the table that $P \vee Q \iff S$.
Thus we see that switches connected in series and in parallel corresponds to the connectives $\wedge$ and $\vee$, respectively.
We next see an example of a switch controlled by a relay.



P	Q	R
1	0	1
0	1	0

When the switch p is open (P is false, P : The switch p is closed) no current flows and the contact q which is normally closed remains closed and the contact r remains open. When p is closed, the current will flow from the battery through the coil which will cause the movement of a relay armature, which in turn causes the springs to move downward and the normally closed contact q to open while the normally open contact r closed. If p is open, then the contact q closes and r opens, because the spring moves upward to its original position.

Consider the statements:

$$P : \text{The switch p is closed},\ Q : \text{The switch q is closed},\ R : \text{The switch r is closed}$$

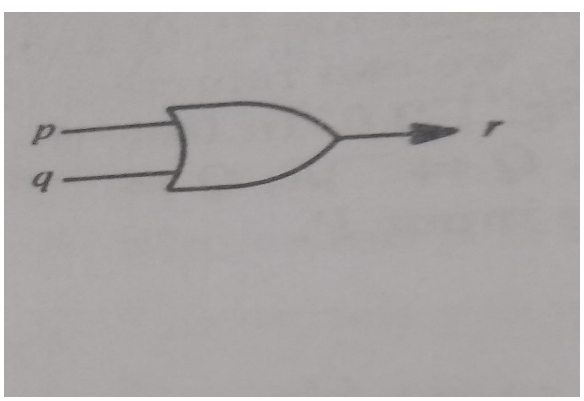
The switches q and r are always in the opposite states, i.e.,

$$Q \iff \neg R,\ Q \iff \neg P, R \iff P.$$

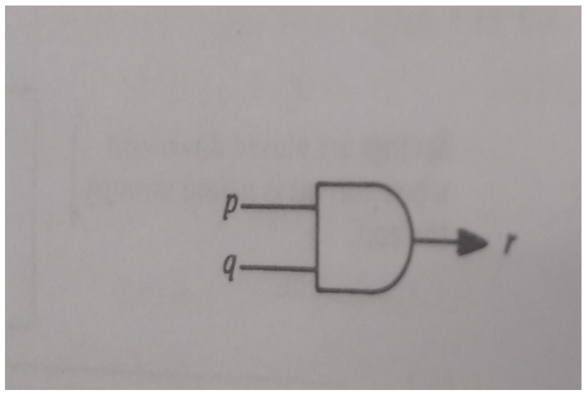
Note that, the output Q is the negation of the input P.

Instead of representing the logical connectives $\neg$, $\vee$, and $\wedge$ by the circuits, they are represented by block diagrams or gates. Each gate has one or more input wire and one output wire.

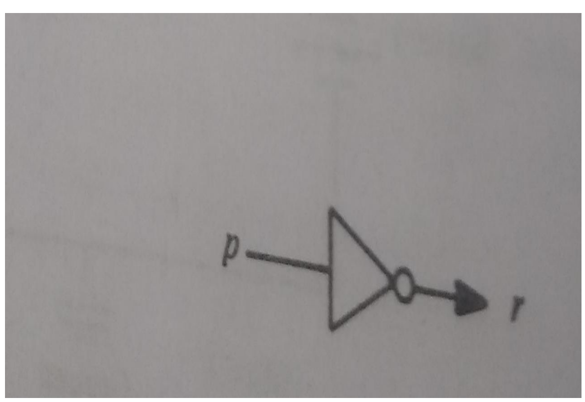
Also, we shall use $+, -$ and $\cdot$ for the logical connectives $\vee, \neg$ and $\wedge$, respectively.



Input	Output
$p\ q$	$r(p+q)$
1 1	1
1 0	1
0 1	1
0 0	0



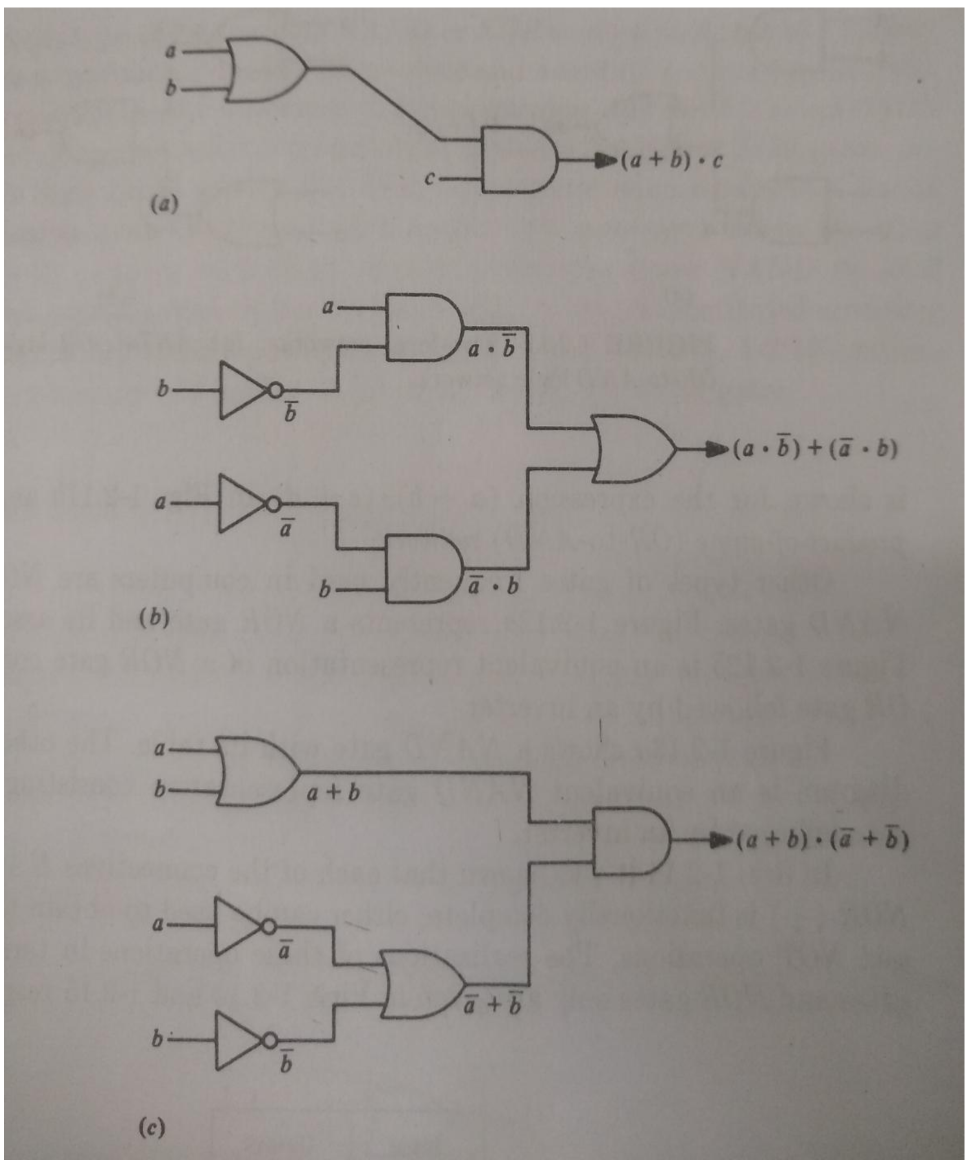
Input	Output
$p\ q$	$r(p \cdot q)$
1 1	1
1 0	0
0 1	0
0 0	0



Input	Output
p	$r(\overline{p})$
1	0
0	1

The block diagrams not only replace the switches and relay but can also be used to represent "gates" in a much more general sense. We may use p to denote voltage potential of an input which is "high" or "low" to allow a transistor to be in a conducting or non-conducting state.

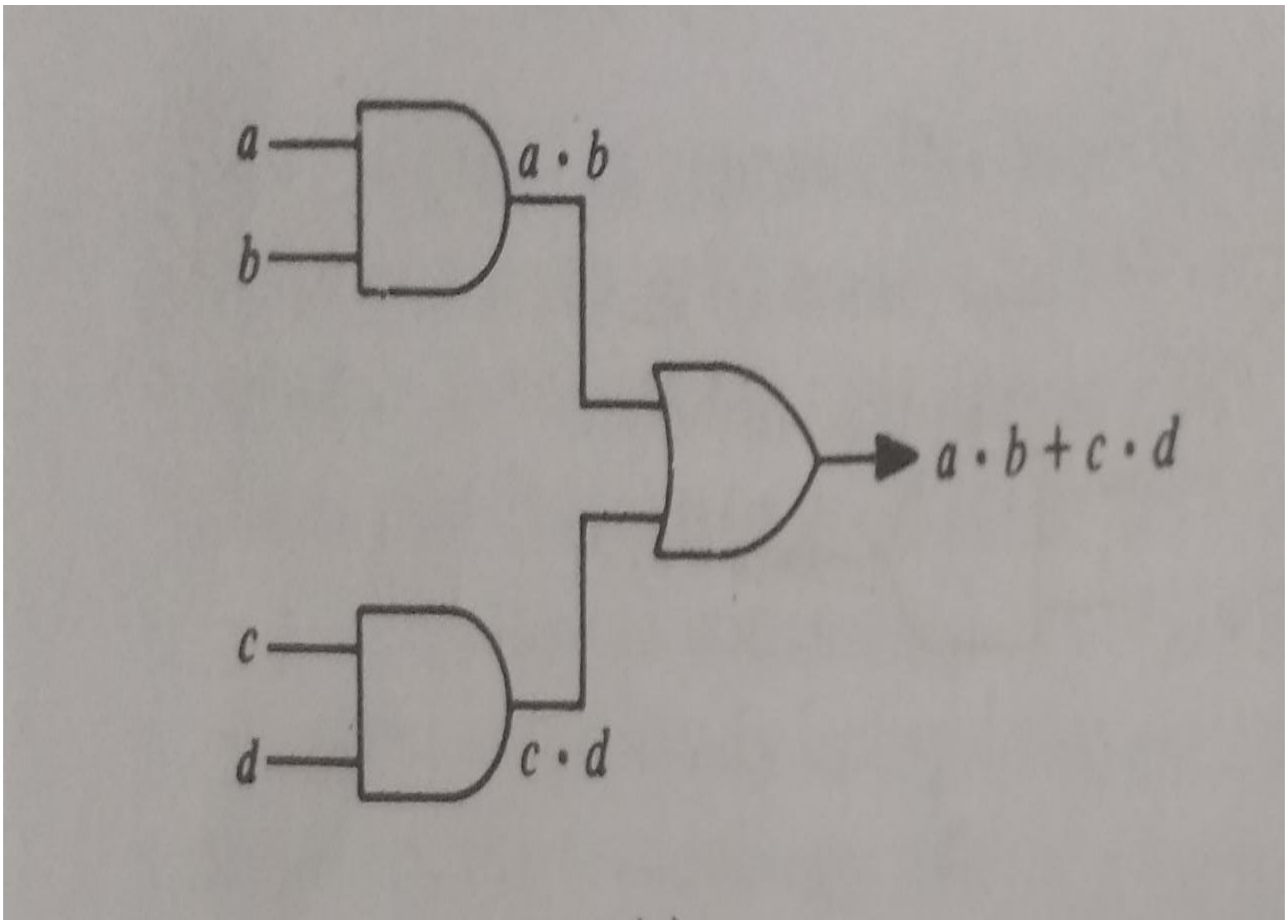
The above gates can be interconnected to realize various logical expressions. These systems of gates are known as logic or combinatorial networks.



To reduce the number of parentheses we write

$$a \cdot b + c \cdot d \text{ for } (a \cdot b) + (c \cdot d).$$

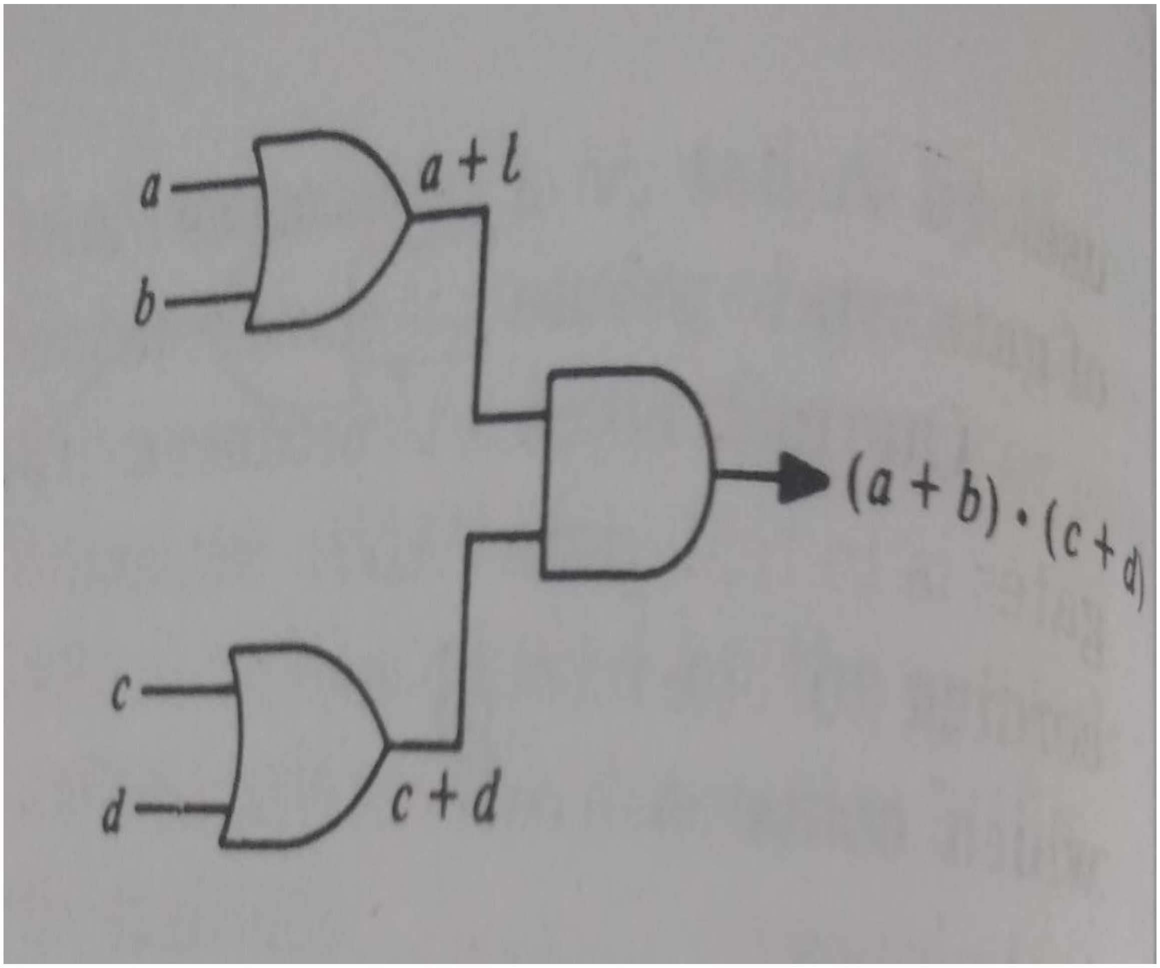
Now, consider the logical expression $a \cdot b + c \cdot d$, which involves disjunction of two conjunctions. A logic network to realize this expression can be two - level network, as shown in the figure below;



A *two - level network* is a logic network in which the longest path through which information must pass from input to output is two gates.

In the above figure, we see that this is a two - level network consisting of AND gates at the input stage and OR gate at the output stage. It is sometimes called *sum of products* (AND- to OR) *network.*

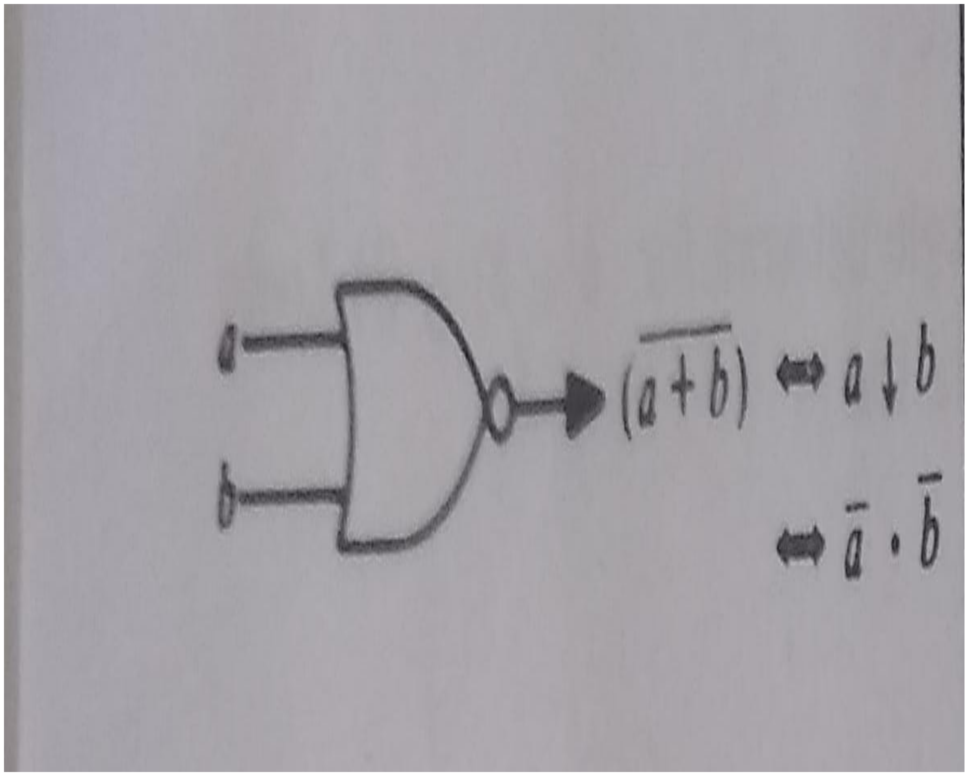
Another possibility for a two - level network is to keep OR gates at the input stage and AND gate at output stage (see the figure below);



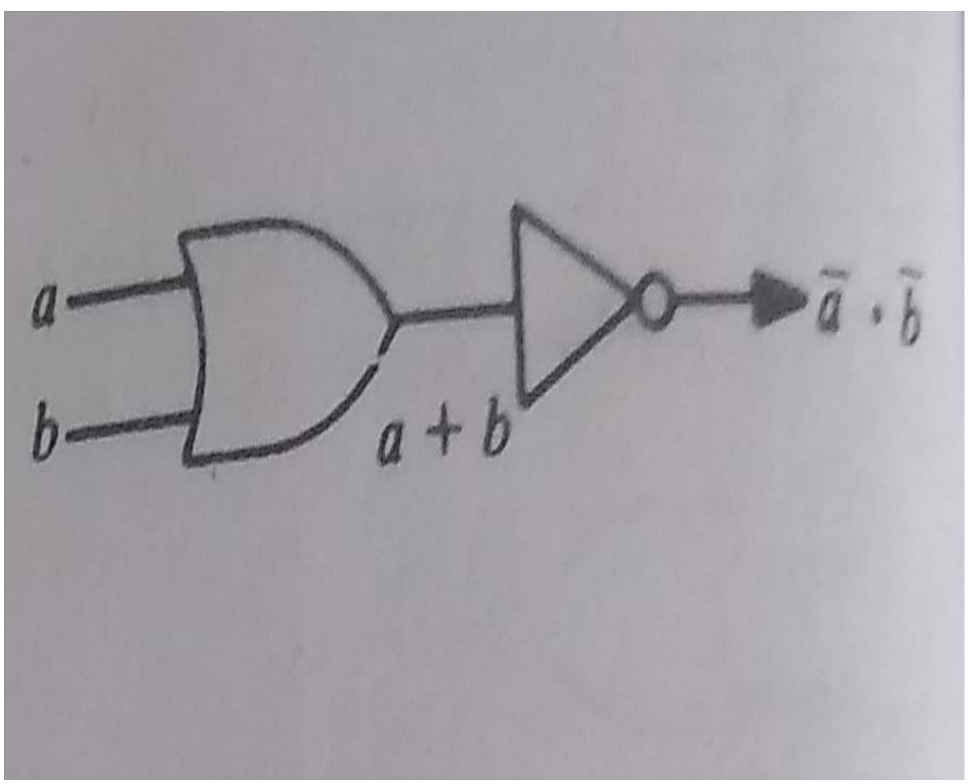
It is called *product of sums* (*OR*- to *AND*) *network.*

Other types of gates frequently used in computers are NOR and $NAND$ gates (shown below);

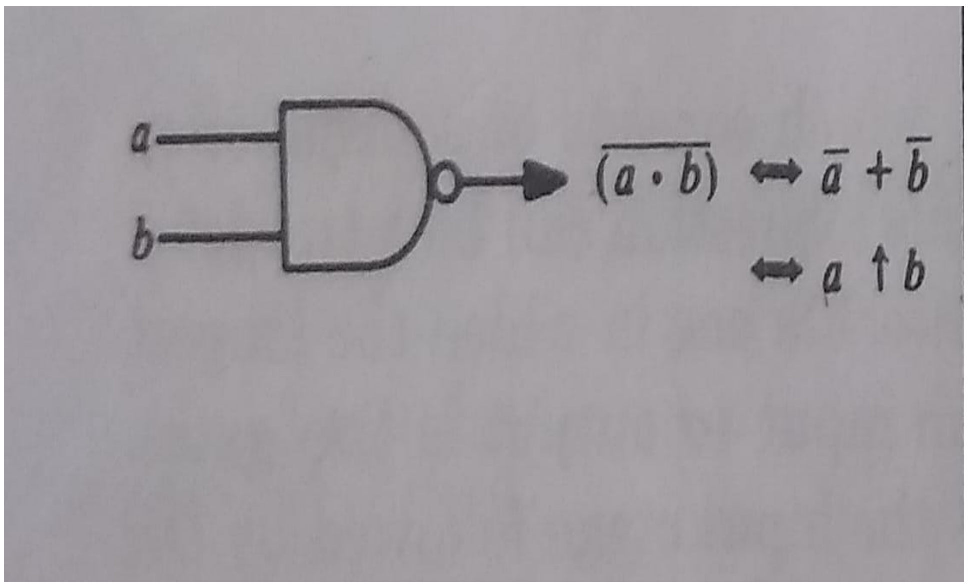
Input	Output
a b	
1 1	0
1 0	0
0 1	0
0 0	1

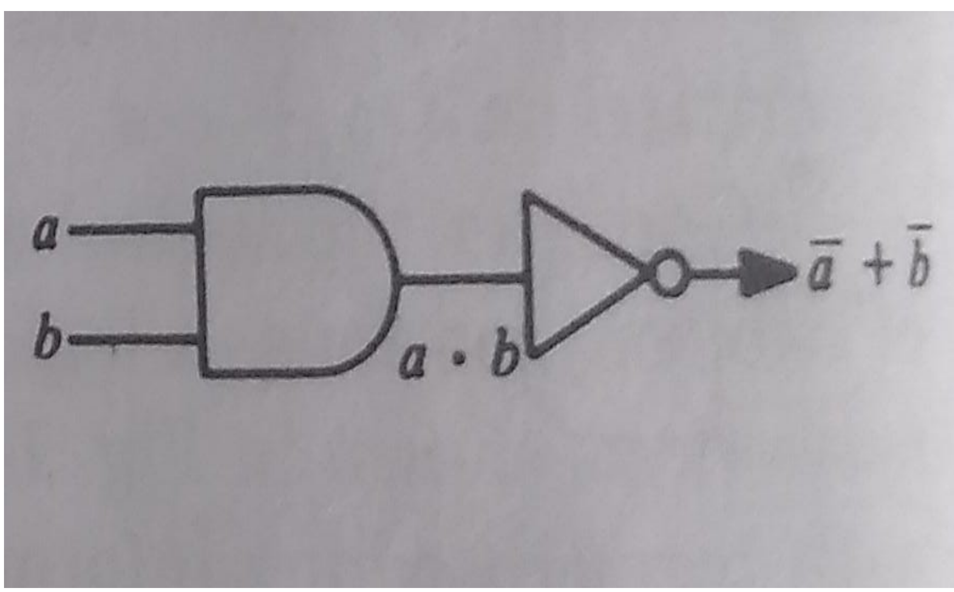

a
b
$\overline{(a+b)} \Leftrightarrow a \downarrow b$
$\Leftrightarrow \bar{a} \cdot \bar{b}$

tables


a
b
a + b
$\bar{a} \cdot \bar{b}$

Input	Output
$a\ b$	
1 1	0
1 0	1
0 1	1
0 0	1





We already have seen that each of the connectives $NAND(\uparrow)$ and $NOR(\downarrow)$ is functionally complete; and either can be used to obtain the AND, OR and NOT operations.

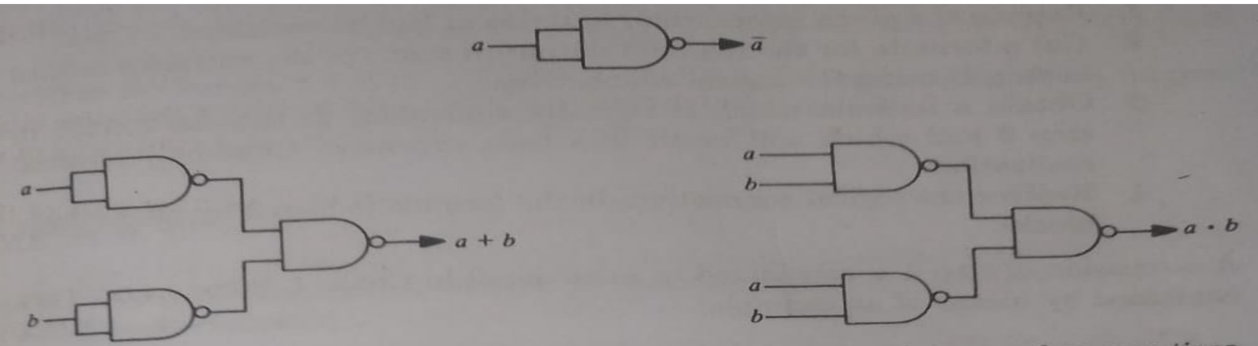


FIGURE 1-2.14. *NAND* gate generation of $+$, $\cdot$, and $^{-}$ operations.

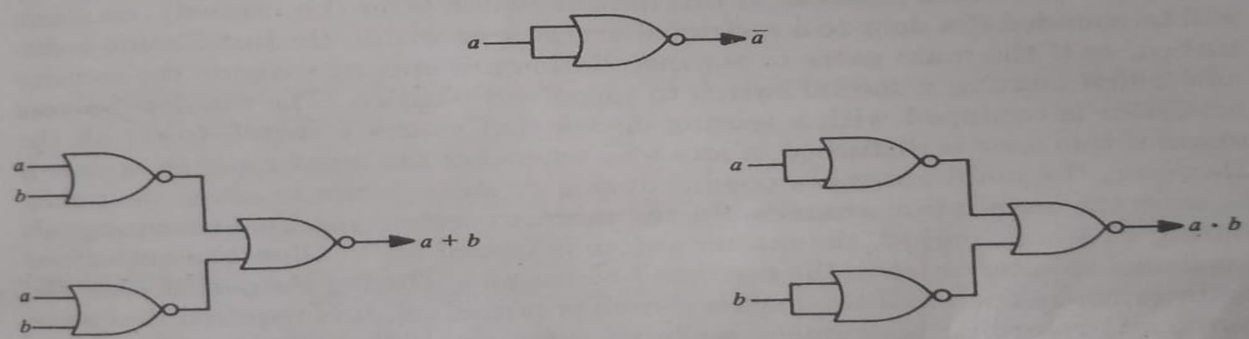


FIGURE 1-2.15. *NOR* gate generation of $+$, $\cdot$, and $^{-}$ operations.

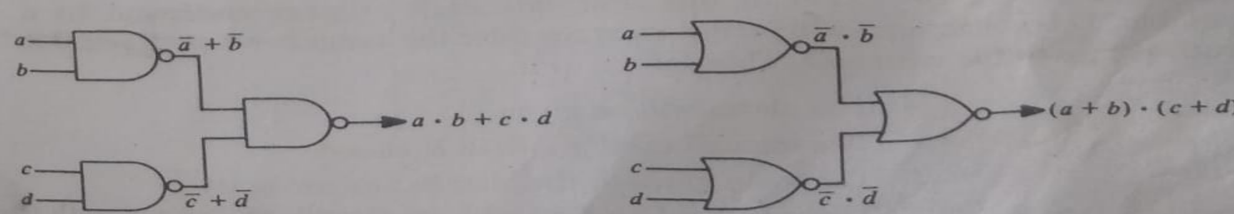


FIGURE 1-2.16. Two-level *NAND* and *NOR* gate networks.

We have discusses certain elements which corresponds to some of the connectives of statement logic. Usually, if a formula involving these connectives is given, we can physically realize a circuit that corresponds to the formula by replacing the connectives by the appropriate gates and the variables by certain physical quantities such as voltage, current, etc.

This method may, however, not yield the best design from the point of view of using a minimum number of gates, or the design it yields may not be a minimal design in some other sense.

Sometimes even the formula may not be available and all we may have is its truth table. Even in this case, we may be required to physically realize the formula, not in any manner, but in some optimal way.

This design process in cases consists of the following steps;

(1) Express the given information in terms of logical variables.
(2) Obtain a formula for the required output in terms of the variables defined in step (1) using the logical connectives.
(3) Obtain a formula which is logically equivalent to the one formed in step (2) and which will result in a least expensive (minimal) physical realization.
(4) Replace the logical connectives in the formula in step (3) by proper logic blocks.

We now see the steps (1) and (2) through example.

Example: A certain government installation has an intruder alarm system which is to be operative only if a manual master switch situated at a security office is in the closed position. If this master switch is on (i.e., closed), an alarm will be sounded if a door to a restricted-access area within the installation is disturbed, or if the main gates to the installation are opened without the security officer first turning a special switch to the closed position. The restricted-access area door is equipped with a sensing device that causes a switch to set off the alarm if this door is disturbed in any way whenever the master switch is closed. However, the main gates are opened during daytime hours to allow the public to enter the installation grounds. Furthermore, at certain specified time intervals during a 24− hour period, the master switch is turned off to allowed the authorized personnel to enter or leave the restricted-access area. During the period when the main gates are open and the master switch is turned off, it is required that some automatic recording instrument make an entry every time the door to the restricted-access area is opened.

Step 1: We associate each primary statement with a variable. This association will allow us to consider all possible truth values to the variables.

$$\begin{aligned} A &: \text{ The alarm will be given.} \\ M &: \text{ The manual master switch is closed.} \\ G &: \text{ The main gates to the installation are open} \\ R &: \text{ The restricted} - \text{area door has been disturbed.} \\ S &: \text{ The special switch is closed.} \\ E &: \text{ The recording equipment isactivated.} \end{aligned}$$

Step 2: The output variables are A and E. The conditions given in the problem require

$$A \iff M \cdot (R + (G \cdot \overline{S})), \; E \iff \overline{M} \cdot G \cdot R.$$