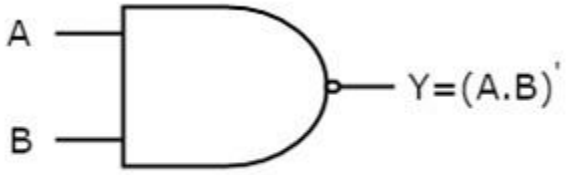
The following image shows the symbol of NAND gate, which is having two inputs A, B and one output, Y.



NAND gate operation is same as that of AND gate followed by an inverter. That's why the NAND gate symbol is represented like that.

NOR gate

NOR gate is a digital circuit that has two or more inputs and produces an output, which is the inversion of logical OR of all those inputs.

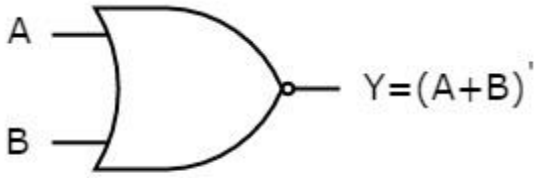
The following table shows the truth table of 2-input NOR gate

A	B	Y = A+BA+B'
0	0	1
0	1	0
1	0	0
1	1	0

Here A, B are the inputs and Y is the output. If both inputs are '0', then the output, Y is '1'. If at least one of the input is '1', then

the output, Y is '0'. This is just opposite to that of two input OR gate operation.

The following figure shows the symbol of NOR gate, which is having two inputs A, B and one output, Y.



NOR gate operation is same as that of OR gate followed by an inverter. That's why the NOR gate symbol is represented like that.

Special Gates

Ex-OR & Ex-NOR gates are called as special gates. Because, these two gates are special cases of OR & NOR gates.

Ex-OR gate

The full form of Ex-OR gate is Exclusive-OR gate. Its function is same as that of OR gate except for some cases, when the inputs having even number of ones.

The following table shows the truth table of 2-input Ex-OR gate.

A	B	$Y = A \oplus B$
0	0	0
0	1	1

Degenerative Form

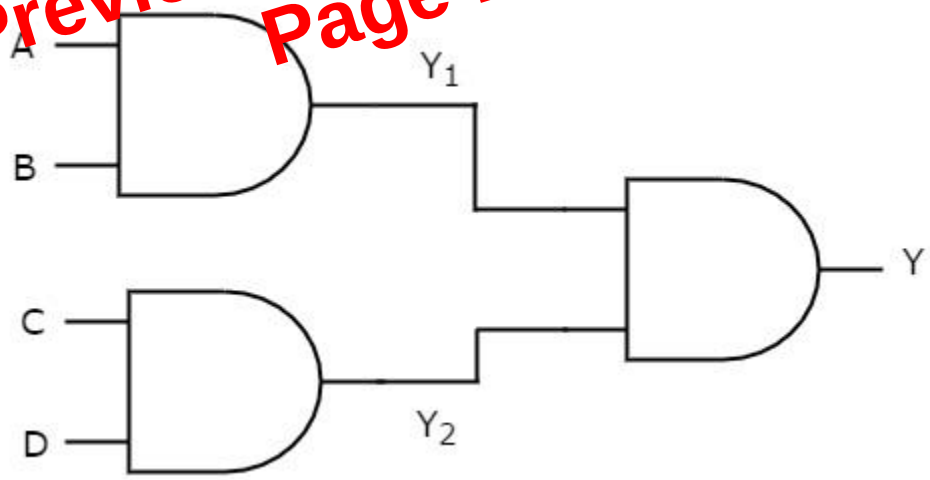
If the output of two level logic realization can be obtained by using single Logic gate, then it is called as degenerative form. Obviously, the number of inputs of single Logic gate increases. Due to this, the fan-in of Logic gate increases. This is an advantage of degenerative form.

Only 6 combinations of two level logic realizations out of 16 combinations come under degenerative form. Those are AND-AND, AND-NAND, OR-OR, OR-NOR, NAND-NOR, NORNAND.

In this section, let us discuss some realizations. Assume, A, B, C & D are the inputs and Y is the output in each logic realization.

AND-AND Logic

In this logic realization, AND gates are present in both levels. Below figure shows an example for AND-AND logic realization.



We will get the outputs of first level logic gates as Y1=ABY1=AB and Y2=CDY2=CD

These outputs, Y1Y1 and Y2Y2 are applied as inputs of AND gate that is present in second level. So, the output of this AND gate is

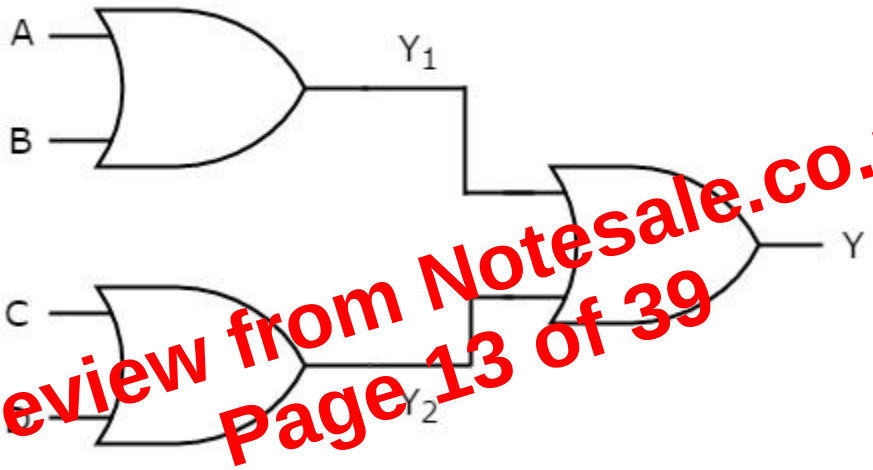
$$Y=((AB)(CD))'Y=((AB)(CD))'$$

$\Rightarrow Y=(ABCD)'\Rightarrow Y=(ABCD)'$

Therefore, the output of this AND-NAND logic realization is (ABCD)'(ABCD)'. This Boolean function can be implemented by using a 4 input NAND gate. Hence, it is degenerative form.

OR-OR Logic

In this logic realization, OR gates are present in both levels. The following figure shows an example for OR-OR logic realization.



We will get the outputs of first level logic gates as Y1=A+BY1=A+B and Y2=C+DY2=C+D.

These outputs, Y1Y1 and Y2Y2 are applied as inputs of OR gate that is present in second level. So, the output of this OR gate is

$$Y=Y1+Y2Y=Y1+Y2$$

Substitute Y1Y1 and Y2Y2 values in the above equation.

$$Y=(A+B)+(C+D)Y=(A+B)+(C+D)$$

$\Rightarrow Y=A+B+C+D\Rightarrow Y=A+B+C+D$

These outputs, Y1 and Y2 are applied as inputs of NOR gate that is present in second level. So, the output of this NOR gate is

$$Y=(Y1+Y2)'Y=(Y1+Y2)'$$

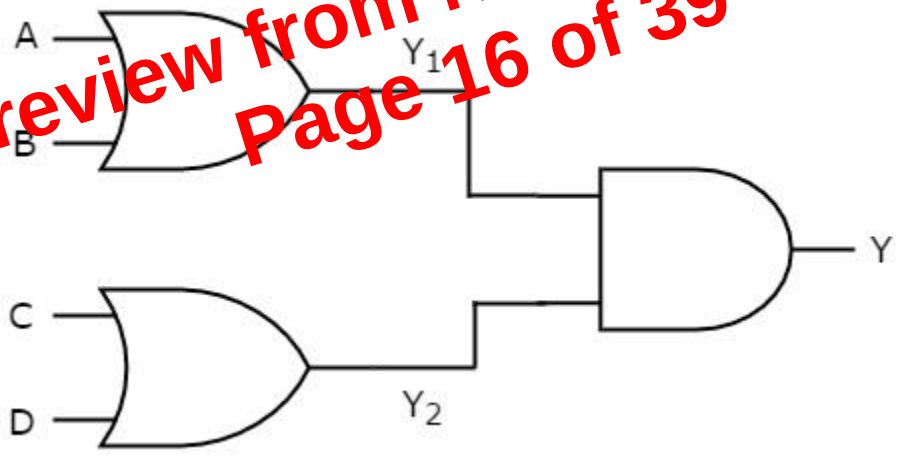
Substitute Y1Y1 and Y2Y2 values in the above equation.

$$Y=(AB+CD)'Y=(AB+CD)'$$

Therefore, the output of this AND-NOR logic realization is (AB+CD)'(AB+CD)'. This Boolean function is in AND-OR-Invert form. Since, we can't implement it by using single logic gate, this AND-NOR logic realization is a non-degenerative form

OR-AND Logic

In this logic realization, OR gates are present in first level & AND gatess are present in second level. The following figure shows an example for OR-AND logic realization.



Previously, we got the outputs of first level logic gates as Y1=A+BY1=A+B and Y2=C+DY2=C+D.

These outputs, Y1Y1 and Y2Y2 are applied as inputs of AND gate that is present in second level. So, the output of this AND gate is

$$Y=Y1Y2Y=Y1Y2$$

Design procedure of Combinational circuits

- Find the required number of input variables and outputs from given specifications.
- Formulate the Truth table. If there are ‘n’ input variables, then there will be 2n possible combinations. For each combination of input, find the output values.
- Find the Boolean expressions for each output. If necessary, simplify those expressions.
- Implement the above Boolean expressions corresponding to each output by using Logic gates.

Code Converters

We have discussed various codes in the chapter named codes. The converters, which convert one code to other code are called as code converters. These code converters basically consist of Logic gates.

Example

Binary code to Gray code converter

Let us implement a converter, which converts a 4-bit binary code WXYZ into its equivalent Gray code ABCD.

The following table shows the Truth table of a 4-bit binary code to Gray code converter.

Binary code WXYZ	WXYZ Gray code ABCD
0000	0000