UNIT - IV

DIGITAL LOGIC

Digital Logic:

Boolean Algebra, Basic and Universal Logic gates, Half adder, Full adder, Simplification of logic expressions using K-maps, Multiplexer, De-multiplexer, Encoder and Decoder.

BOOLEAN ALGEBRA

George Boole in 1854 invented a new kind of algebra known as Boolean algebra. It is sometimes called switching algebra. Boolean algebra is the mathematical frame work on which logic design based. It is used in synthesis & analysis of binary logical function.

Laws of Boolean Algebra:

1. **0'=1**
2. **1'=0**
3. **If A=0, A'=1**
4. **IF A=1, A'=0**
5. **A''=A**
6. **A.0=0**
7. **A.1=A**
8. **A.A=A**
9. **A.A'=0**
10. **A+0=A**
11. **A+1=1**

12. **A+A=A**
13. **A+A'=1**
14. **A+AB=A(1+B)=A**
15. **A+A'B=A+AB+A'B=A+B(A+A')=A+B**

LOGIC GATES

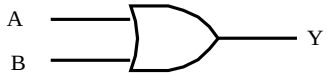
- It is an electronic circuit which makes logic decisions. A logic gate is a digital circuit with one or more input signal and only one output signal. All input or output signals either low voltage or high voltage. A digital circuit is referred to as logic gate for simple reason i.e. it can be analyzed on the basis of Boolean algebra.
- To make logical decisions, three gates are used. They are OR, AND and NOT gate. These logic gates are building blocks which are available in the form of IC.
- The input and output of the binary variables for each gate can be represented in a tabular column or truth table.

1. OR Gate: The OR gate performs logical additions commonly known as OR function. The OR gate has two or more inputs and only one output. The operation of OR gate is such that a HIGH(1) on the output is produced when any of the input is HIGH. The output is LOW(0) only when all the inputs are LOW.

- If A & B are the input variables of an OR gate and c is its output, then A+B. similarly for more than two

variables, the OR function can be expressed as Y=A+B+C.

- Logical Symbol:



- Truth table

Input		Output
A	B	Y= A+B
0	0	0
0	1	1
1	0	1
1	1	1

Realization of OR gate using diodes

- Two input OR gate using "diode-resistor" logic is shown in figure below. Where X, Y are the inputs and F is the output.

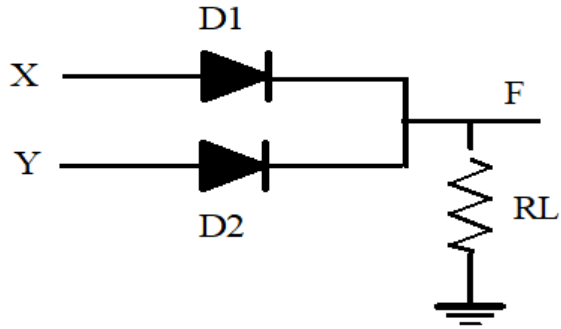
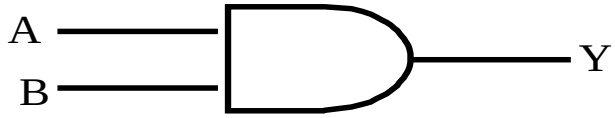


Fig 5.1: OR gate using diodes

2. **AND Gate** : The AND gate performs logical multiplication, commonly known as AND function. The AND gate has two or more inputs and a single output. The output of an AND gate is HIGH only when all the inputs are HIGH. Even if any one of the input is LOW, the output will be LOW. If a & b are input variables of an AND gate and c is its output, then Y=A.B

Logic Symbol



Truth table

Input		Output
A	B	Y=A.B
0	0	0
0	1	0
1	0	0
1	1	1

Realization of AND gate using diodes

- Two input AND gate using "diode-resistor" logic is shown in figure below. Where X, Y are inputs and F is the output.

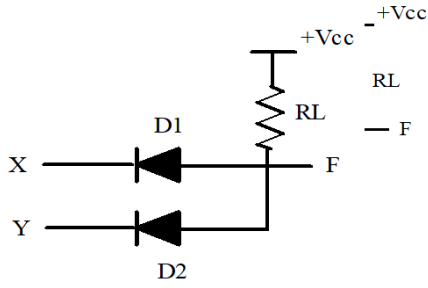


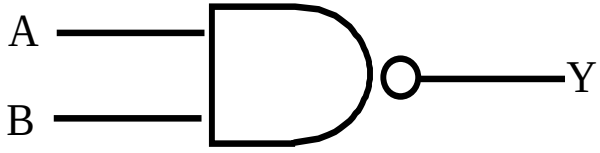
Fig 5.2AND gate using diodes

3. Not Gate (Inverter): The NOT gate performs the basic logical function called inversion or complementation. The purpose of this gate is to convert one logic level into the opposite logic level. It has one input and one output. When a HIGH level is applied to an inverter, a LOW level appears at the output and vice-versa.

Truth Table:

Input	output
A	$Y=\overline{A}$
0	1
1	0

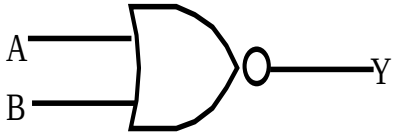
4.NAND Gate: The output of a NAND gate is LOW only when all inputs are HIGH and output of the NAND is HIGH if one or more inputs are LOW.



Truth Table:

Input		Output
A	B	$Y = \overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

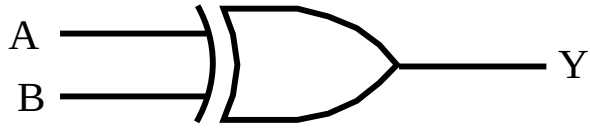
5. NOR Gate: The output of the NOR gate is HIGH only when all the inputs are LOW.



Truth Table:

Input		Output
A	B	$Y = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

6.XOR Gate or Exclusive OR gate: In this gate output is HIGH only when any one of the input is HIGH. The circuit is also called as inequality comparator, because it produces output when two inputs are different. When both the inputs are high, then the output is low.

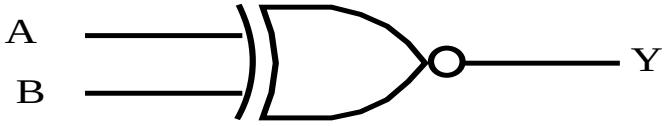


Truth Table:

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

$Y = A \oplus B = A\overline{B} + \overline{A}B$

7. XNOR Gate or Exclusive NOR Gate: An XNOR gate is a gate with two or more inputs and one output. XNOR operation is complimentary of XOR operation. i.e. The output of XNOR gate is High, when all the inputs are identical; otherwise it is low.



Truth Table:

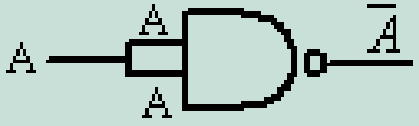
Input		Output
A	B	$Y = \overline{A}\ \overline{B} + AB$
0	0	1
0	1	0
1	0	0
1	1	1

Universal Logic Gate

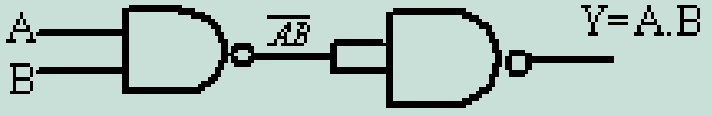
NAND and NOR gates are called Universal gates or Universal building blocks, because both can be used to implement any gate like AND,OR an NOT gates or any combination of these basic gates.

NAND gate as Universal gate

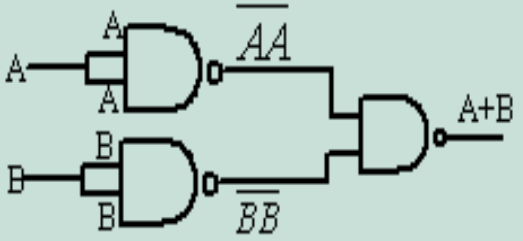
NOT operation:



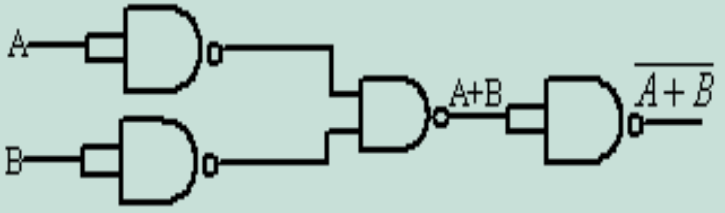
AND operation:



OR operation:

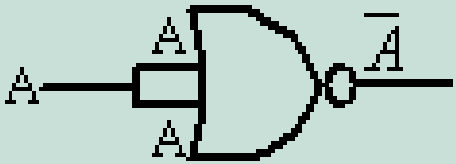


NOR operation:

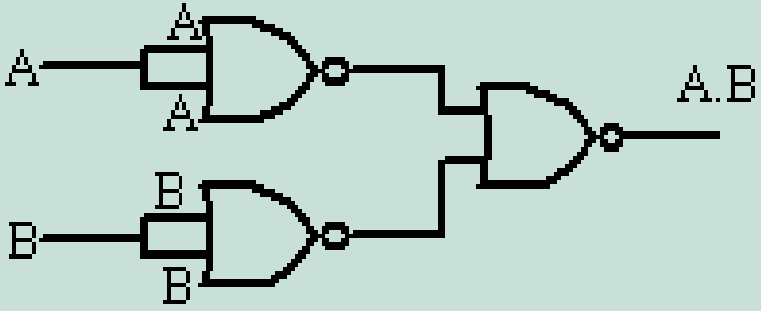


NOR gate as Universal gate:

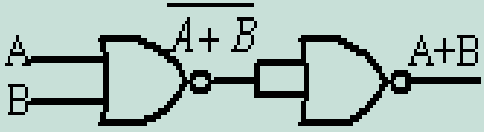
NOT operation:



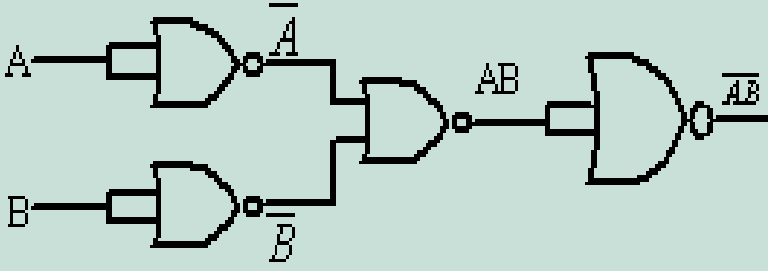
AND operation:



OR operation:

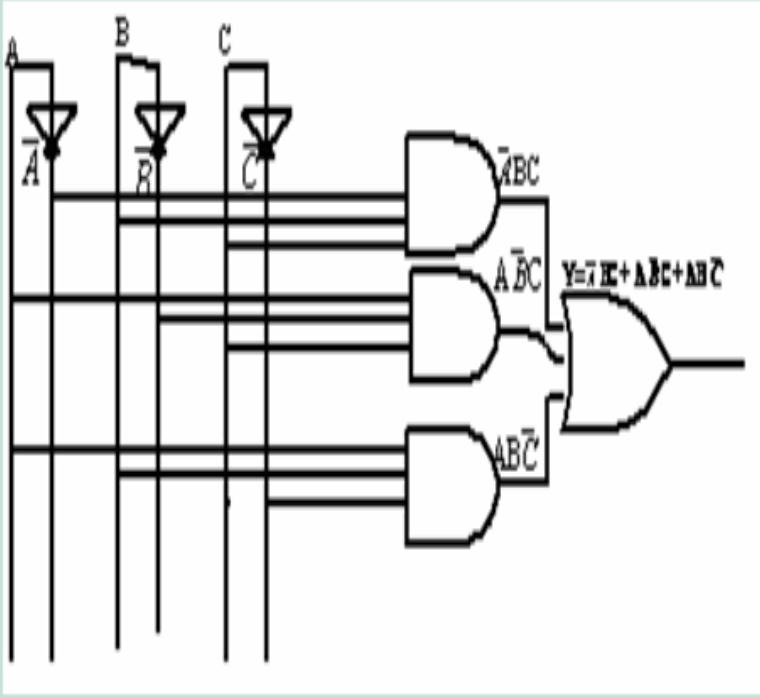


NAND operation:



Draw the logic circuit for the Boolean expression.

$Y=\overline{A}BC+A\overline{B}C+ABC.$



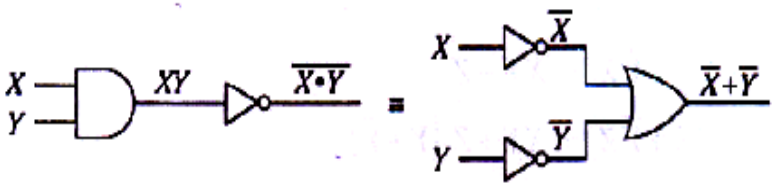
De Morgan's Theorems:

It is one of the important properties of Boolean algebra. It is extensively useful in simplifying complex Boolean expression.

Theorem 1: It states that "the compliments of product of two variables equal to sum of the compliments of individual variable".

$$\overline{X \bullet Y} = \overline{X} + \overline{Y}$$

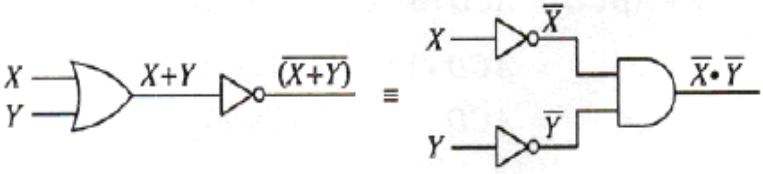
X	Y	$X \bullet Y$	$\overline{X \bullet Y}$	X	Y	$\overline{X}$	$\overline{Y}$	$\overline{X}+\overline{Y}$
0	0	0	1	0	0	1	1	1
0	1	0	1	0	1	1	0	1
1	0	0	1	1	0	0	1	1
1	1	1	0	1	1	0	0	0



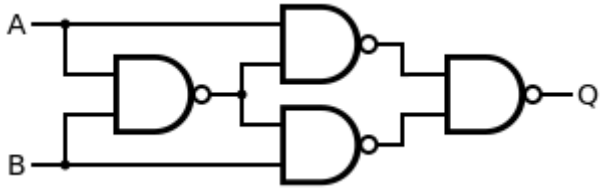
Theorem 2: It states that compliment of sum of two variables is equal to product of compliment of two individual variables.

$$\overline{X + Y} = \overline{X} \bullet \overline{Y}$$

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



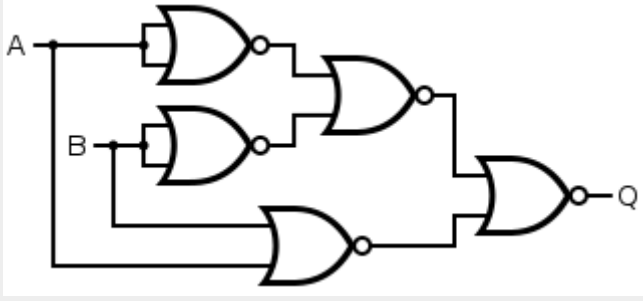
1. **RealizeEXOR Gate using only minimum NAND Gates**



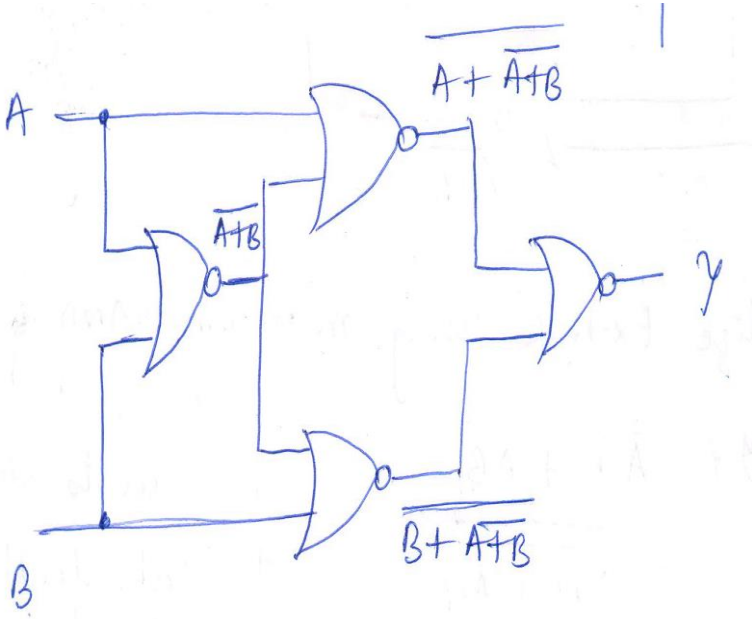
2. RealizeEXOR Gate using only NOR Gates

XNOR + NOR = XOR i.e. Add NOR gate to the ouptut of XNOR gate. Here is the boolean algebra for XOR gate:

Y = (AB + B'A')'
= (AB)'.(B'.A')'
= (A'+B').((B')'+(A')')
= (A'+B').(B+A)
= A'B+A'A+BB'+B'A
= A'B+AB' (Boolean expression for XOR Gate)



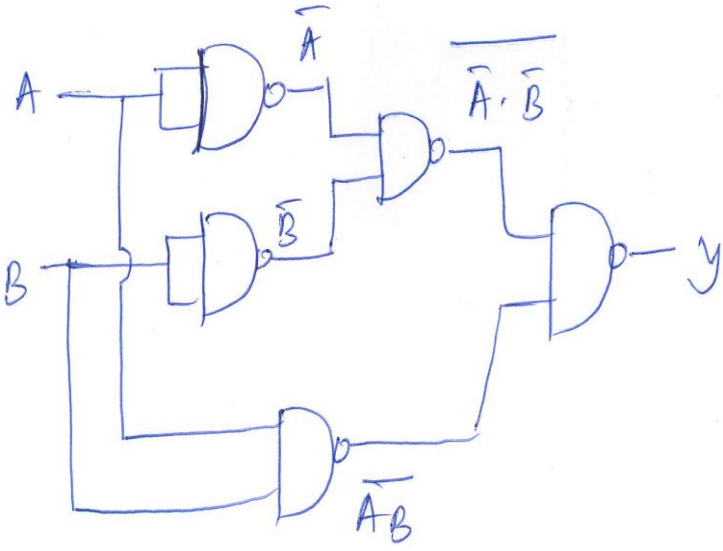
3. RealizeEX-NOR Gate using only minimum NOR Gates



4. RealizeEX-NOR Gate using only NAND Gates

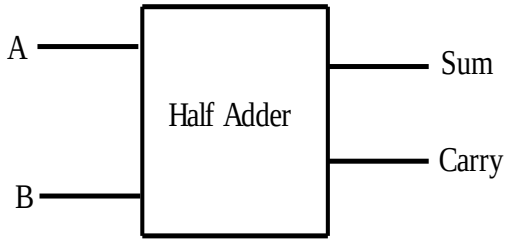
XOR + NAND Inverter(NOT) = XNOR i.e. Add NOT Gate to the output of XOR gate as shown in the image above. Here is the boolean algebra for XNOR gate:

$$
\begin{aligned}
Y &= (AB' + BA')' \\
&= (AB')'.(B.A')' \\
&= (A'+(B')').(B'+(A')') \\
&= (A'+B).(B'+A) \\
&= A'B'+A'A+BB'+BA \\
&= AB+A'B' \text{ (Boolean expression for XNOR Gate)}
\end{aligned}
$$


A
B
$\bar{A}$
$\bar{B}$
$\overline{\bar{A} \cdot \bar{B}}$
$\bar{A}B$
y

Half Adder:

A combinational circuit which performs the arithmetic addition of two binary digits is called Half Adder. In the half adder circuit, there are two inputs, one is addend and augends and two outputs are Sum and Carry.



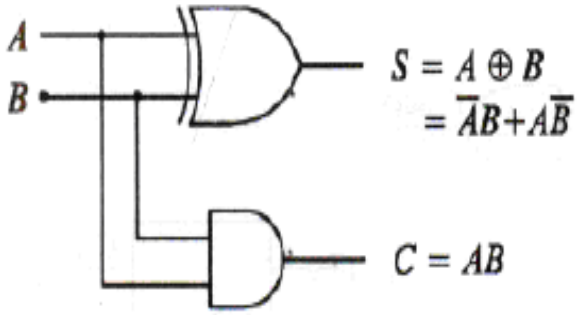
Truth Table for Half Adder

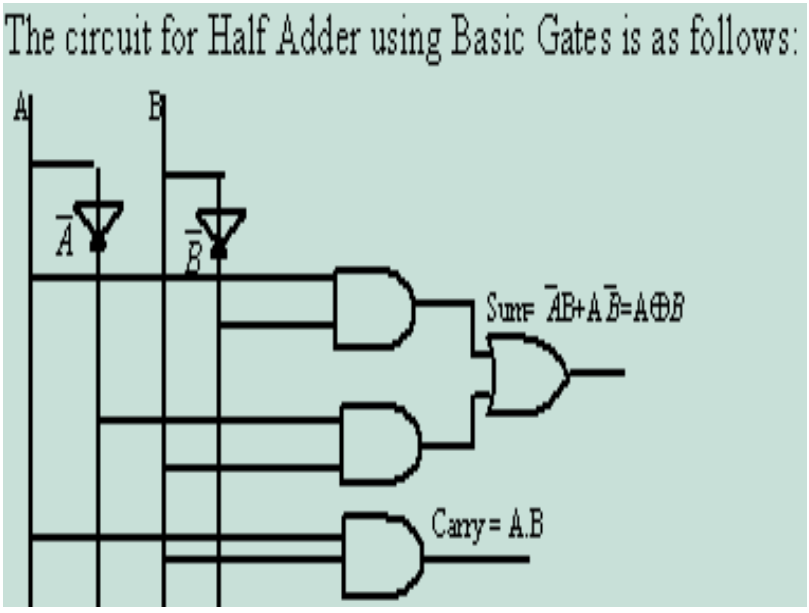
Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Logic Expression :

$$\text{Sum} = \overline{A}B + A\overline{B} = A \oplus B$$
$$\text{Carry} = A.B$$

Circuit Symbol of Half adder:

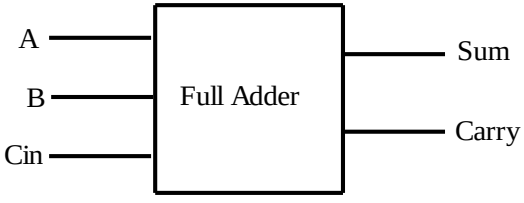




Full Adder: The full adder is a combinational circuit that performs the arithmetic sum of three input bits.

- It consists of three inputs and two outputs. Two of the inputs are variables, denoted by A and B, represent the two significant bit to be added. The third input C_{in} represents carry form the previous lower significant position.

Truth Table for Full Adder

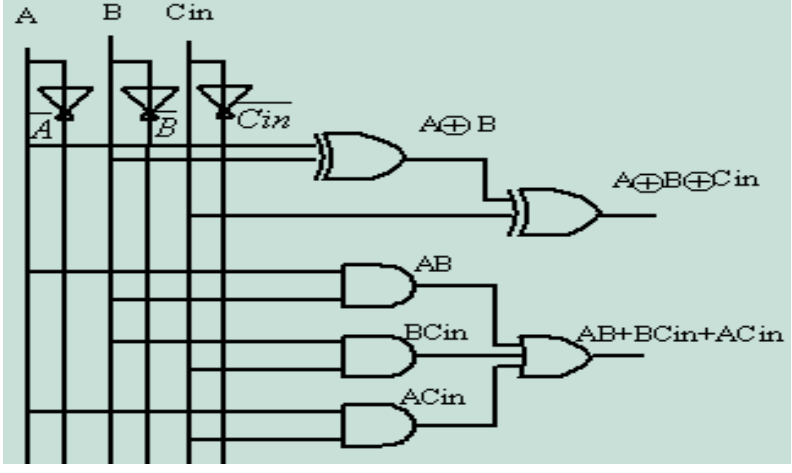


Inputs			Outputs	
A	B	C_{in}	Carry	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$\text{Sum} = \bar{A}\,\bar{B}\,\text{Cin} + \bar{A}\,\text{B}\,\overline{Cin} + \text{A}\,\bar{B}\,\overline{Cin} + \text{ABCin}$$
$$= \bar{A}\,[\bar{B}\,\text{Cin} + \text{B}\,\overline{Cin}] + \text{A}[\bar{B}\,\overline{Cin} + \text{BCin}]$$
$$= \bar{A}\,[\text{B} \oplus \text{Cin}] + \text{A}[\overline{B \oplus Cin}]$$
$$= \text{A} \oplus \text{B} \oplus \text{Cin}$$

$$\text{Carry} = \bar{A}\,\text{BCin} + \text{A}\,\bar{B}\,\text{Cin} + \text{AB}\,\overline{Cin} + \text{ABCin}$$
$$= \bar{A}\,\text{BCin} + \text{A}\,\bar{B}\,\text{Cin} + \text{AB}(\overline{Cin} + \text{Cin})$$
$$= \bar{A}\,\text{BCin} + \text{A}\,\bar{B}\,\text{Cin} + \text{AB}$$
$$= \bar{A}\,\text{BCin} + \text{A}(\bar{B}\,\text{Cin} + \text{B})$$
$$= \bar{A}\,\text{BCin} + \text{AB} + \text{ACin}$$
$$= \text{B}(\bar{A}\,\text{Cin} + \text{A}) + \text{ACin}$$
$$= \text{B}(\text{A} + \text{Cin}) + \text{ACin}$$
$$= \text{AB} + \text{BCin} + \text{ACin}$$

Implementation of Full Adder:



Full adder Circuit using two Half adders:

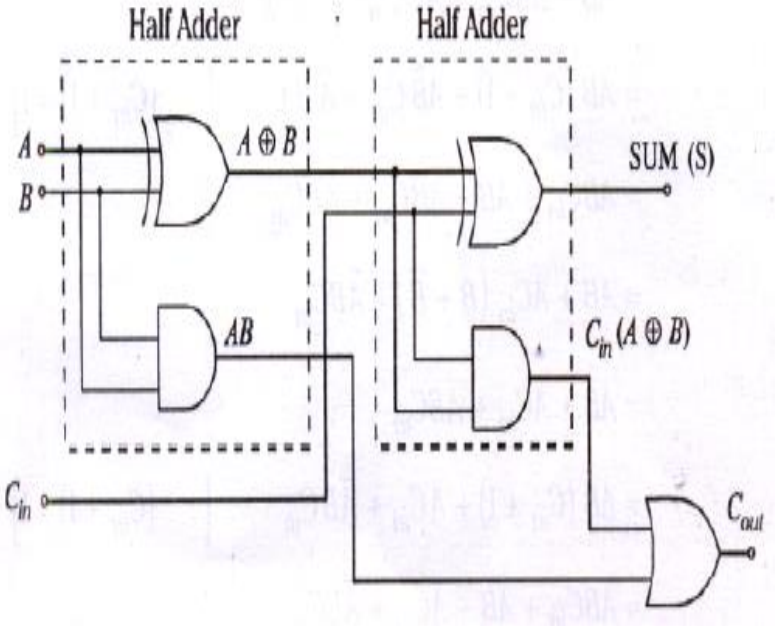


Fig 5.4:Full adder Circuit using two Half adders

SIMPLIFICATION USING BOOLEANALGEBRA

A simplified Boolean expression uses the fewest gates possibleto implement a givenexpression.

Example

Using Boolean algebra techniques,

simplify

thisexpression:

AB + A(B + C) + B(B +C)

Solution

Step 1: Apply the distributive law to the second and third terms inthe expression, asfollows:

AB + AB + AC + BB +BC

Step 2: Apply rule 7 (BB = B) to the fourthterm.

AB + AB + AC + B +BC

Step 3: Apply rule 5 (AB + AB = AB) to the first twoterms.

AB + AC + B +BC

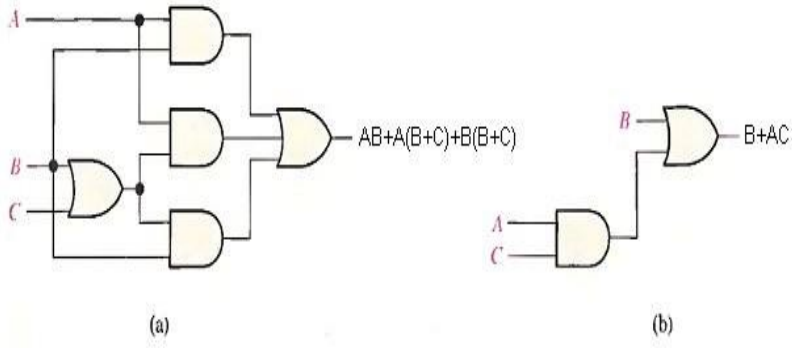
Step 4: Apply rule 10 (B + BC = B) to the last twoterms.

AB + AC +B

Step 5: Apply rule 10 (AB + B = B) to the first and thirdterms.

B+AC

At this point the expression is simplified as much aspossible.



Boolean Algebra Practice Problems

1) $a + 0 =$ _____

2) $\bar{a} \cdot 0 =$ _____

3) $a + \bar{a} =$ _____

4) $a + a =$ _____

5) $a + ab =$ _____

6) $a + ab =$ _____

7) $a(\bar{a} + b) =$ _____

8) $ab + a\bar{b} =$ _____

9) $(\bar{a} + \bar{b})(a + b) =$ _____

10) $a(a + b + c + \ldots) =$ _____

For (11),(12),(13), $f(a,b,c) = a + b + c$

14) $y + y\bar{y} =$ _____

15) $xy + x\bar{y} =$ _____

16) $\bar{x} + yx =$ _____

17) _____ $(w + x + y + z)\ y =$ _____

18) $(x + y)(x + y) =$ _____

19) $w + [w + (wx)] =$ _____

20) $x[x + (xy)] =$ _____

21) $\overline{(x + x)} =$ _____

22) $\overline{(x + x)} =$ _____

23) $w + (\overline{wxyz}) =$ _____

24) $\bar{w} \cdot (\overline{wxyz}) =$ _____

11) $f(a, b, ab)=$_____ 25) $xz+\bar{x}y+zy=$_____

12) $f(a,b,a\cdot\bar{b})=$_____ 26) $(x+z)(\bar{x}+y)(z+y)=$_____

13) $f[a,b,\overline{(ab)}]=$_____ 27) $\bar{x}+\bar{y}+xyz=$_____

Solutions to the Boolean Algebra PracticeProblems

1) $a+0=a$

2) $a\cdot 0=0$

3) $a+\bar{a}=1$

1) $a+a=a$

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

6) $a+a\bar{b}=(a+a)(\bar{a}+b)=a+b$

7) $a(\bar{a}+b)=a\bar{a}+ab=ab$

8) $ab+\bar{a}b=b(a+\bar{a})=b$

9) $(\bar{a}+\bar{b})(a+b)=a\bar{a}+a\bar{b}+ba+\bar{b}b=a+a\bar{b}+\bar{a}b=a(1+\bar{b}+b)=a$

10) $a(a+b+c+\ldots)=aa+ab+ac+\ldots=a+ab+ac+\ldots=a$

11) $f(a, b, ab)=a+b+ab=a+b$

12) $f(a,b,a\cdot\bar{b})=a+b+a\bar{b}=\bar{a}+b+a=1$

13) $f[a,b,\overline{(ab)}]=a+b+\overline{(ab)}=a+b+a+\bar{b}=1$

14) $y+\bar{y}y=y$

15) $x\bar{y}+xy=x(\bar{y}+y)=x$

16) $x+\bar{x}x=x(1+y)=x$

17) $(w+\bar{x}+y+z)y=y$

18) $(x+\bar{y})(x+y)=x$

19) $w+[w+(wx)]=w$

20) $x[x+(xy)]=x$

21) $\overline{(\bar{x}+x)}=x$

22) $\overline{(x+\bar{x})}=0$

23) $w+(w\bar{x}yz)=w(1+\bar{x}yz)=w$

24) $\overline{w\cdot(wxyz)}=\bar{w}(\bar{w}+\bar{x}+\bar{y}+\bar{z})=\bar{w}$

25) $xz+\bar{x}y+zy=xz+\bar{x}y$

26) $(x+z)(\bar{x}+y)(z+y)=(x+z)(\bar{x}+y)=xy+\bar{x}z$

27) $\bar{x}+\bar{y}+xyz=\bar{x}+\bar{y}+z$

Simplify the Boolean expression

1. $\overline{X}\,\overline{Y}\,Z + \overline{X}\,YZ$
   $= \overline{X}\,Z[\overline{Y} + Y]$
   $= \overline{X}\,Z[1] = \overline{X}\,Z$

2. $f = X(\overline{X} + Y)$
   $= X\,\overline{X} + XY = 0 + XY = XY$

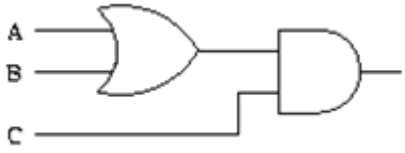
3. $f = B(A+C)+C$
   $= BA+BC+C$
   $= BA+C(1+B)$
   $= BA+C$

4. $XY+XYZ+XY\,\overline{Z}+\overline{X}\,YZ$
   $= XY(Z+\overline{Z})+XYZ+XY\,\overline{Z}+\overline{X}\,YZ$
   $XYZ+XY\,\overline{Z}+XY\,\overline{Z}+\overline{X}\,YZ+XYZ$
   $XYZ(1+1)+XY\,\overline{Z}\,(1+1)+\overline{X}\,YZ$
   $XYZ+XY\,\overline{Z}+\overline{X}\,YZ$
   $XY(Z+\overline{Z})+\overline{X}\,YZ$
   $XY+\overline{X}\,YZ$
   $Y(X+\overline{X}\,Z)$
   $Y(X+Z)$

5. $XYZ+\overline{X}\,Y+XY\,\overline{Z}$
   $= Y(\overline{X} + X\,\overline{Z})+XY\,\overline{Z}$
   $= Y(\overline{X} + \overline{Z})+XY\,\overline{Z}$
   $= Y\,\overline{X} + Y\,\overline{Z} + XY\,\overline{Z}$
   $= Y\,\overline{X} + Y(Z+X\,\overline{Z})$
   $= Y\,\overline{X} + Y(Z+X)$
   $= Y(Z+\overline{X}+X)$
   $= Y(Z+1)$
   $= Y$

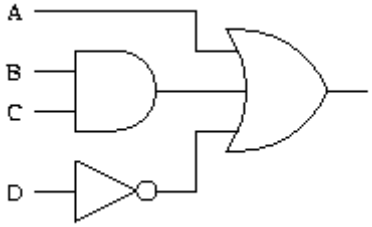
•Draw a logic circuit for (A + B)C.

Soln:



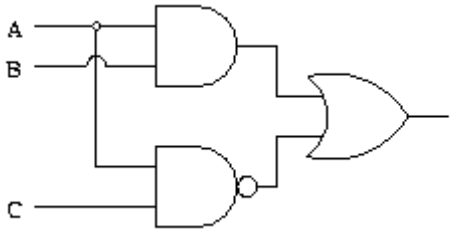
•Draw a logic circuit for A + BC + D.

Soln:



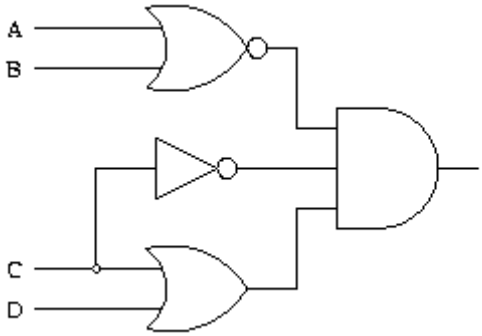
•Draw a logic circuit for AB + AC.

Soln:



•Draw a logic circuit for (A + B)(C + D)C.

Soln:



MULTIPLEXER (MUX)

•A MUX is a digital switchthat has multiple inputs(sources) and a singleoutput (destination).

•The select linesdetermine which input is connected to the output

•MUXTypes

2-to-1(1selectline)

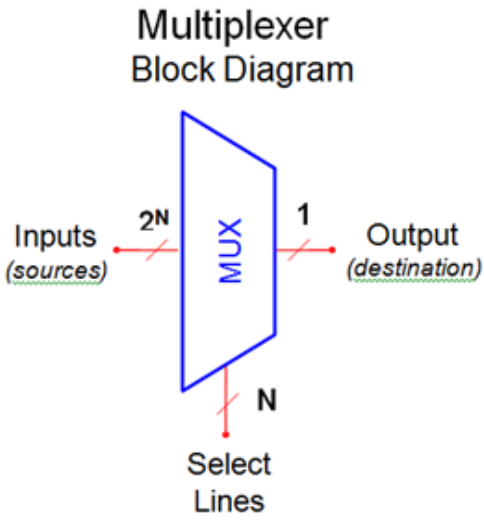
4-to-1(2selectlines)

8-to-1(3selectlines)

16-to-1(4selectlines)

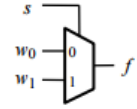
•Normally there are 2n input lines and n select lines

•The basic multiplexer has several data input lines and a single output line and hence the name ‘Many to One’.



- 2:1 Mux

•2:1 Mux contains 2 input lines, 1 select line and one output line



(a) Graphical symbol

s	f
0	w_0
1	w_1

(b) Truth table

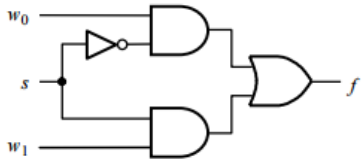
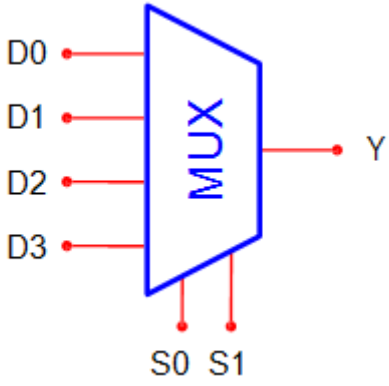


Fig 5.8: 2:1 MUX

- 4:1 MUX

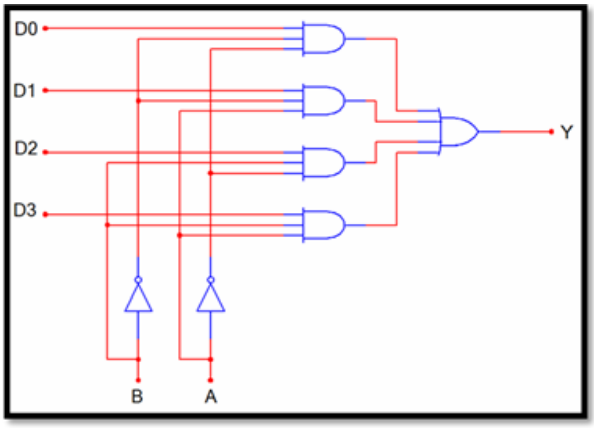
•4:1 MUX contains 4 input lines, 2 select lines and one input line



Logic Symbol

Truth Table

S0	S1	Y
0	0	D0
0	1	D1
1	0	D2
1	1	D3



Logic Expression:

Y= S0' S1'D0 +S0' S1 D1 +S0 S1' D2+ S0 S1 D3

No of AND Gates : 04

No of NOT Gates : 02

No of OR Gates : 01

Total Gates : 07

DEMULTIPEXER (De-MUX)

•A DEMUX is a digital switch with a single input(source) and a multiple outputs (destinations).

•The select lines determine which output the input is connected to. The selection of specific output is controlled by the values of 'n' select lines.

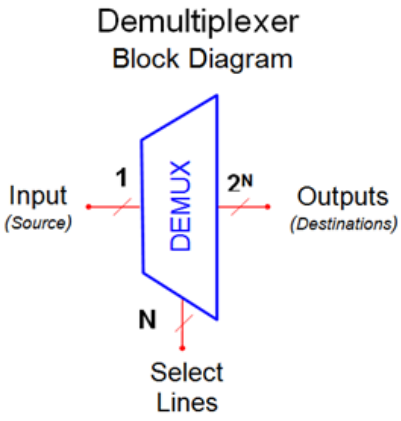
•DEMUX Types

1-to-2(1selectline)

1-to-4(2selectlines)

1-to-8(3selectlines)

1-to-16(4selectlines)



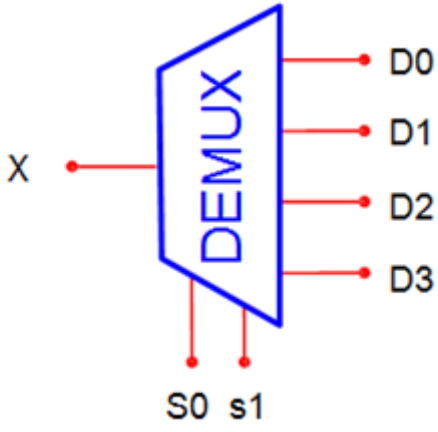
1:4 DEMUX:

Input = 01

Select lines ‘n’= 2

Outputs = 2n = 4

Logic symbol:



Truth Table:

S0	S1	D0	D1	D2	D3
0	0	X	0	0	0
0	1	0	X	0	0
1	0	0	0	X	0
1	1	0	0	0	X

Logic Expression:

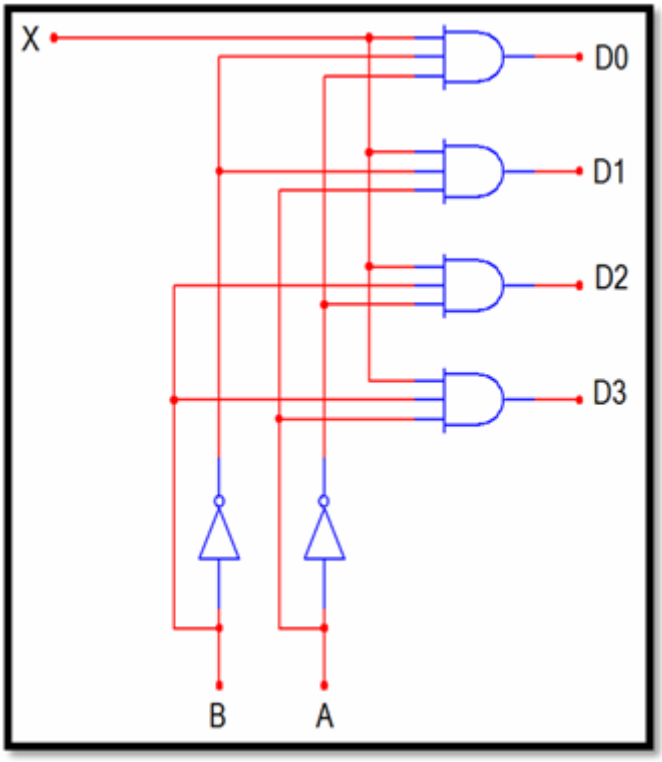
D0=S0’ S1’ X

D1= S0’ S1 X

D2= S0 S1’ X

D3= S0 S1 X

1: 4 DEMUX Circuit:



•No of Gates for 1:4 DEMUX

AND Gates= 04

NOT Gates = 02

ENCODERS:

- Device which converts one binary code to another
- Multiple inputs , multiple outputs
- Converts complex to simplest form
- More inputs, less outputs

DECODER

• Device which transforms the coded bits to generate the original data again

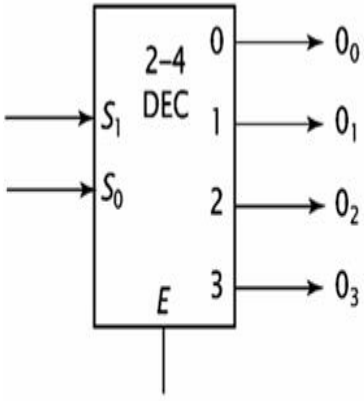
• Multiple inputs multiple output logic circuit

• It converts coded inputs into coded outputs, where the inputs and outputs are different

• Less inputs with more outputs

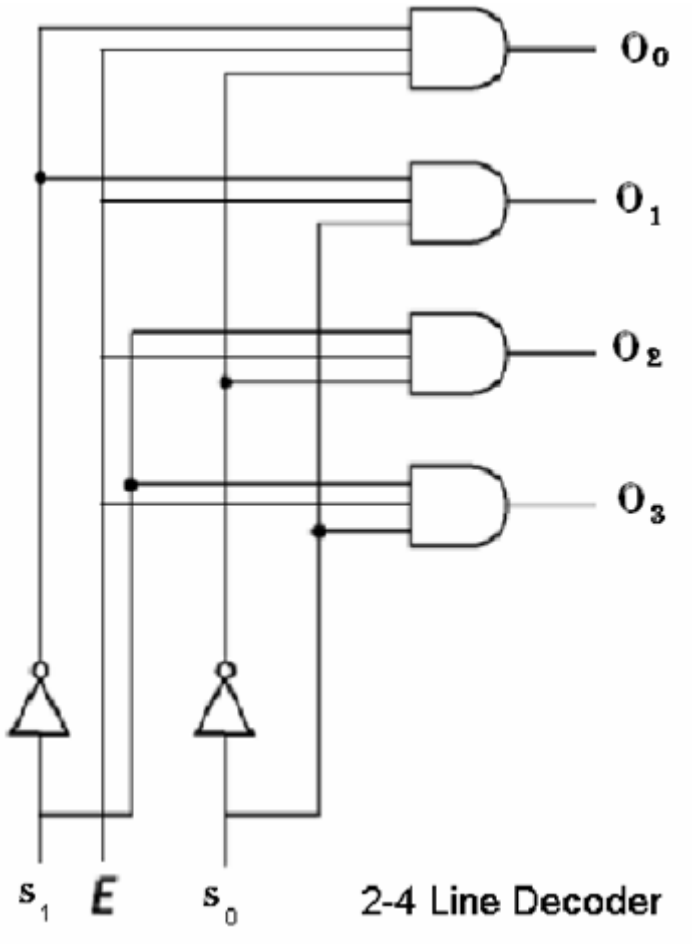
• It takes in 'n' input and gives out 2n outputs

2 to 4 Decoder Logic symbol and Truth table:



(b)

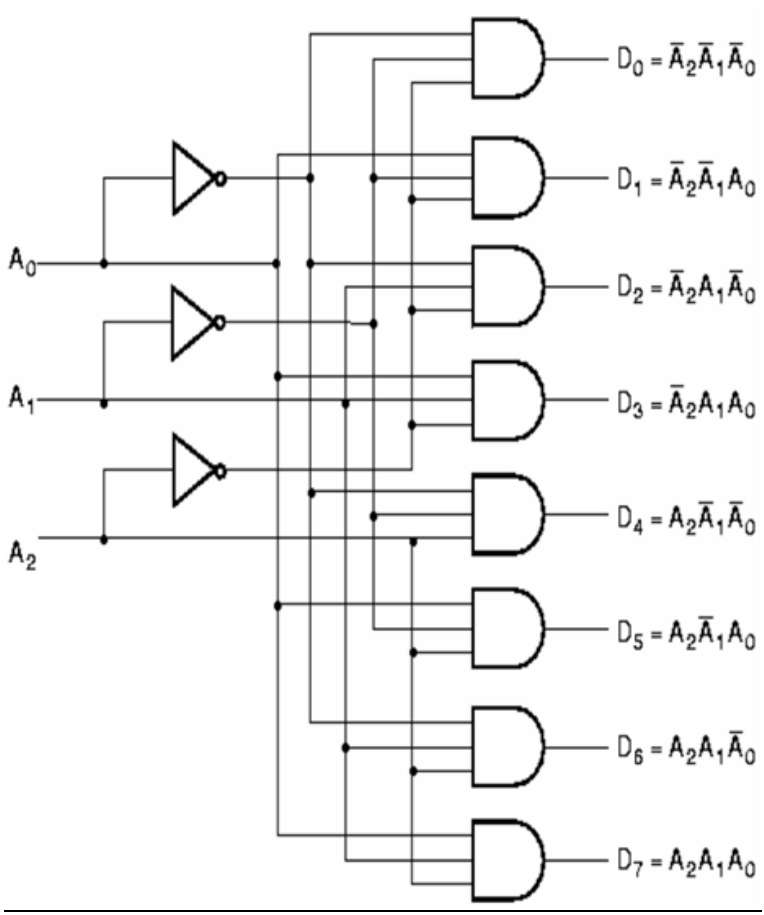
S_1	S_0	E	O_0	O_1	O_2	O_3
X	X	0	0	0	0	0
0	0	1	1	0	0	0
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	1


O0
O1
O2
O3
S1
E
S0
2-4 Line Decoder

3 to 8 decoder: (Binary to Octal Converter)

A2	A1	A0	D0	D1	D2	D3	D4	D5	D6	D7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

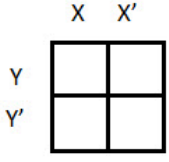
Circuit diagram for 3 to 8 decoder:



---

2 variable K-maps

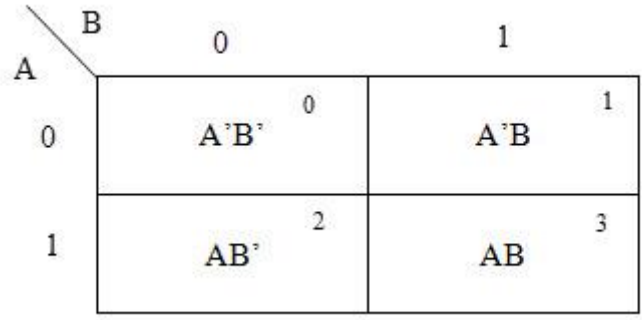
There are 4 cells (22) in the 2-variable k-map. It will look like (see below image)



The possible min terms with 2 variables (A and B) are A.B, A.B', A'.B and A'.B'. The conjunctions of the variables (A, B) and (A', B) are represented in the cells of the top row and (A, B') and (A', B') in cells of the bottom row. The following table shows the positions of all the possible outputs of 2-variable Boolean function on a K-map.

A	B	Possible Outputs	Location on K-map
0	0	A'B'	0
0	1	A'B	1
1	0	AB'	2
1	1	AB	3

A general representation of a 2 variable K-map plot is shown below.



When we are simplifying a Boolean equation using Karnaugh map, we represent the each cell of K-map containing the conjunction term with 1. After that, we group the adjacent cells with possible sizes as 2 or 4. In case of larger k-maps, we can group the variables in larger sizes like 8 or 16.

The groups of variables should be in rectangular shape, that means the groups must be formed by combining adjacent cells either vertically or horizontally. Diagonal shaped or L-shaped groups are

not allowed. The following example demonstrates a K-map simplification of a 2-variable Boolean equation.

Example

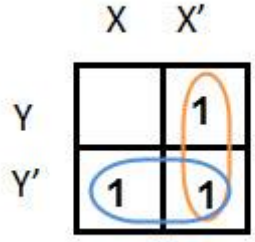
Simplify the given 2-variable Boolean equation by using K-map.

F = X Y' + X' Y + X'Y'

First, let's construct the truth table for the given equation,

X	Y	F
0	0	0
0	1	1
1	0	1
1	1	1

We put 1 at the output terms given in equation.



In this K-map, we can create 2 groups by following the rules for grouping, one is by combining (X', Y) and (X', Y') terms and the other is by combining (X, Y') and (X', Y') terms. Here the lower right cell is used in both groups.

After grouping the variables, the next step is determining the minimized expression.

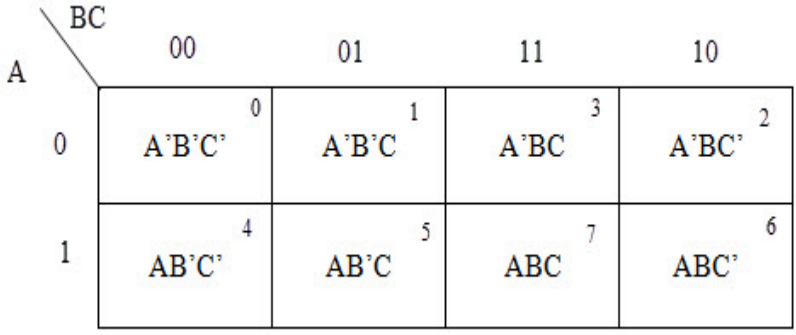
By reducing each group, we obtain a conjunction of the minimized expression such as by taking out the common terms from two groups, i.e. X' and Y'. So the reduced equation will be X' +Y'.

3 variable K-maps

For a 3-variable Boolean function, there is a possibility of 8 output min terms. The general representation of all the min terms using 3-variables is shown below.

A	B	C	Output Function	Location on K-map
0	0	0	A'B'C'	0
0	0	1	A'B'C	1
0	1	0	A'BC'	2
0	1	1	A'BC	3
1	0	0	AB'C'	4
1	0	1	AB'C	5
1	1	0	ABC'	6
1	1	1	ABC	7

A typical plot of a 3-variable K-map is shown below. It can be observed that the positions of columns 10 and 11 are interchanged so that there is only change in one variable across adjacent cells. This modification will allow in minimizing the logic.



Up to 8 cells can be grouped in case of a 3-variable K-map with other possibilities being 1,2 and 4.

Example

Simplify the given 3-variable Boolean equation by using k-map.

F = X' Y Z + X' Y' Z + X Y Z' + X' Y' Z' + X Y Z + X Y' Z'

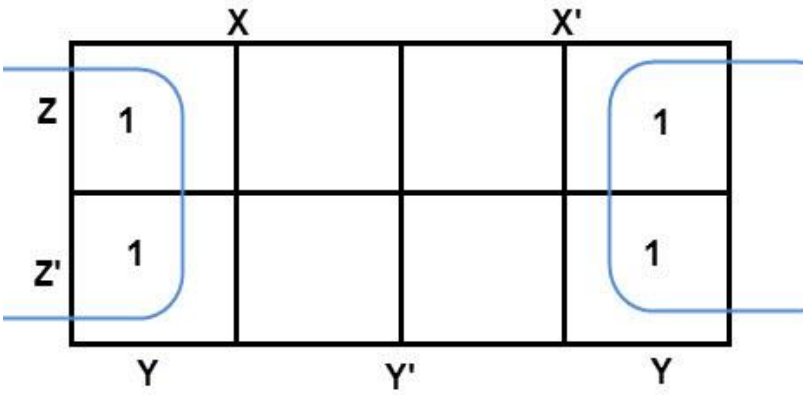
First, let's construct the truth table for the given equation,

x	y	z	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

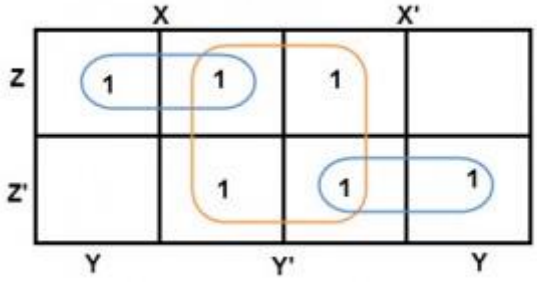
We put 1 at the output terms given in equation.

There are 8 cells (23) in the 3-variable k-map. It will look like (see below image).

The largest group size will be 8 but we can also form the groups of size 4 and size 2, by possibility. In the 3 variable Karnaugh map, we consider the left most column of the k-map as the adjacent column of rightmost column. So the size 4 group is formed as shown below.



And in both the terms, we have 'Y' in common. So the group of size 4 is reduced as the conjunction Y. To consume every cell which has 1 in it, we group the rest of cells to form size 2 group, as shown below.



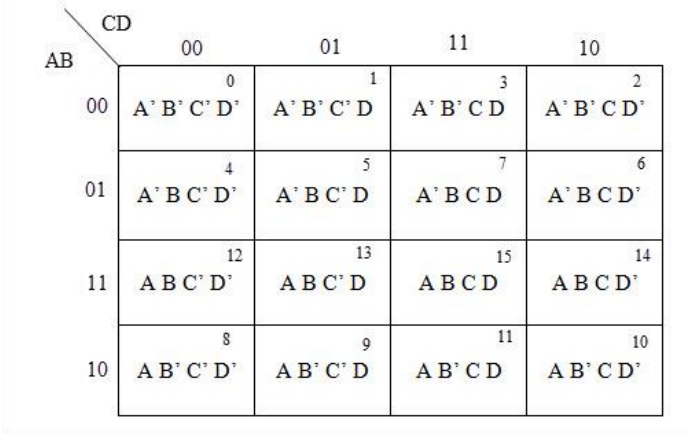
The 2 size group has no common variables, so they are written with their variables and its conjugates. So the reduced equation will be X Z' + Y' + X' Z. In this equation, no further minimization is possible.

4 variable K-maps

There are 16 possible min terms in case of a 4-variable Boolean function. The general representation of minterms using 4 variables is shown below.

A	B	C	D	Output function	K-map location
0	0	0	0	A' B' C' D'	0
0	0	0	1	A' B' C' D	1
0	0	1	0	A' B' C D'	2
0	0	1	1	A' B' C D	3
0	1	0	0	A' B C' D'	4
0	1	0	1	A' B C' D	5
0	1	1	0	A' B C D'	6
0	1	1	1	A' B C D	7
1	0	0	0	A B' C' D'	8
1	0	0	1	A B' C' D	9
1	0	1	0	A B' C D'	10
1	0	1	1	A B' C D	11
1	1	0	0	A B C' D'	12
1	1	0	1	A B C' D	13
1	1	1	0	A B C D'	14
1	1	1	1	A B C D	15

A typical 4-variable K-map plot is shown below. It can be observed that both the columns and rows of 10 and 11 are interchanged.

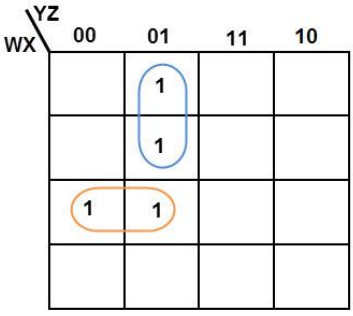


The possible number of cells that can be grouped together are 1, 2, 4, 8 and 16.

Example

Simplify the given 4-variable Boolean equation by using k-map. F (W, X, Y, Z) = (1, 5, 12, 13)

Sol: F (W, X, Y, Z) = (1, 5, 12, 13)



By preparing k-map, we can minimize the given Boolean equation as F = W Y' Z + W 'Y' Z