EX.NO: 1 DATE:	STUDY OF LOGIC GATES

AIM:

To Verify the truth table for AND, OR, NOT, NAND, NOR and EX-OR gates.

FORMULA:

* The Boolean equation for AND gate is

  $Y = A \cdot B$

* The Boolean equation for OR gate is

  $Y = A + B$

* The Boolean equation for NOT gate is

  $Y = A' \ (\bar{A})$

* The Boolean equation for NAND gate is

  $Y = \overline{A \cdot B}$

* The Boolean equation for NOR gate is

  $Y = \overline{A + B}$

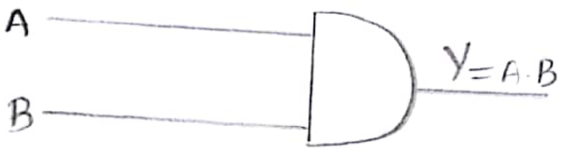
* The Boolean equation for EX-OR gate is $Y = A \oplus B$

Scanned by CamScanner

LOGIC GATES DIAGRAM:

1. AND GATE:

Logic Diagram:

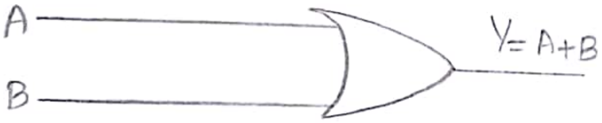


Truth table:

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

2. OR GATE:

Logic Diagram:



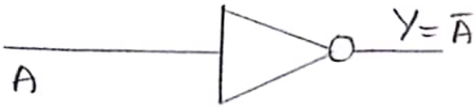
Truth table:

A	B	$Y = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

Scanned by CamScanner

3. NOT GATE:

Logic diagram:

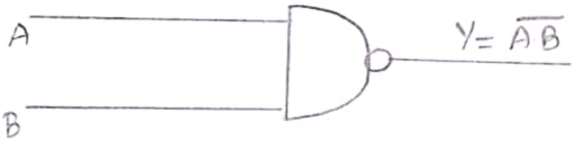


Truth table:

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

4. NAND GATE:

Logic Diagram:



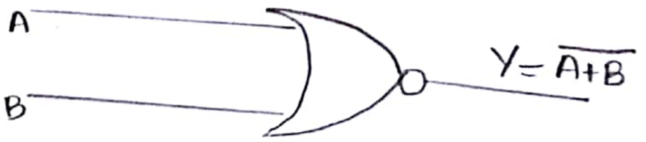
Truth table:

A	B	Y=$\overline{A\cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

Scanned by CamScanner

5. NOR GATE:

Logic Diagram:

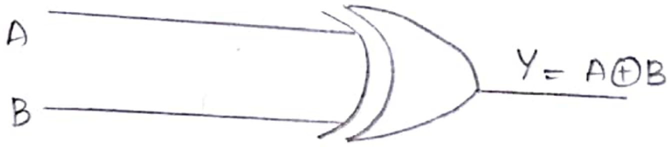


Truth table:

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

6. EX-OR GATE:

Logic Diagram:



Truth table!

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

Scanned by CamScanner

RESULT:

The truth table is verified for AND, OR, NOT, NAND, EX-OR gates successfully.

Scanned by CamScanner

EX.NO:2 DATE:	NAND AS UNIVERSAL GATE

AIM:

To realise NOT, AND, OR, EX-OR gate using NAND gates.

FORMULA:

* The Boolean equation for NOT gate is

  $Y = \overline{A}$

* The Boolean equation for AND gate is

  $Y = A \cdot B$

* The Boolean equation for OR gate is

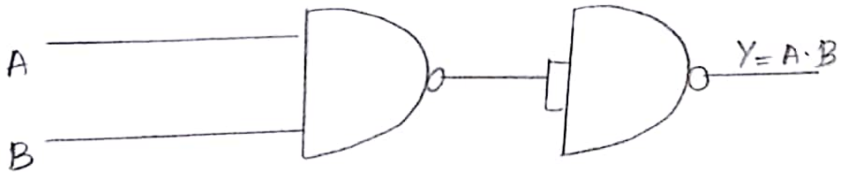
  $Y = A + B$

* The Boolean equation for EX-OR gate is

  $Y = A \oplus B$

Logic gate diagram:

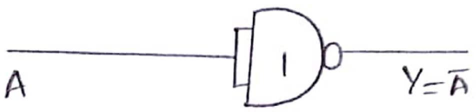
1. AND FUNCTION: USING NAND GATE



Truth table:

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

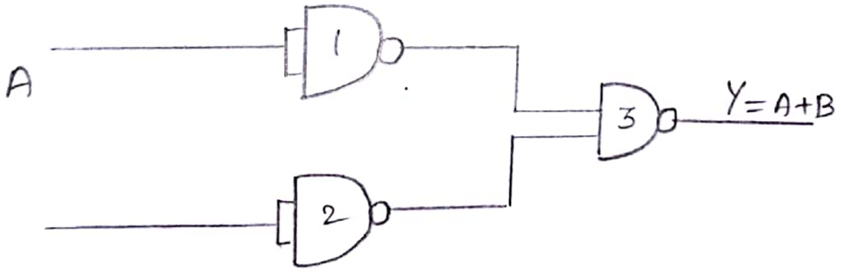
2. NOT FUNCTION: USING NAND GATE



Truth table:

A	$Y=\bar{A}$
0	1
1	0

3. OR FUNCTION: USING NAND GATE

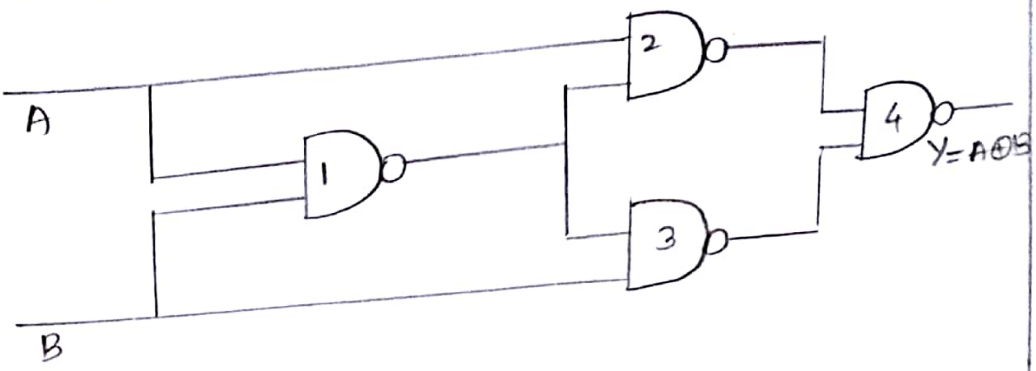


Truth table:

A	B	$Y=A+B$
0	0	0
0	1	1
1	0	1
1	1	1

Scanned by CamScanner

4. EX-OR FUNCTION: USING NAND gate



Truth table:

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

RESULT:

Thus the realisation of NOT, AND, OR, EX-OR gates using NAND gate is verified.

EX.NO : DATE :	NOR AS UNIVERSAL GATE

AIM:

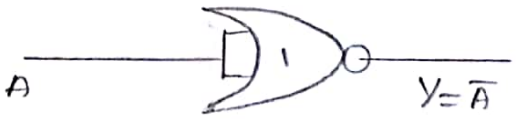
To realise NOT, OR, AND and EX-OR using NOR gates.

FORMULA:

- * The Boolean equation for NOT gate is $Y = \overline{A}$
- * The Boolean equation for AND gate is $Y = A \cdot B$
- * The Boolean equation for OR gate is $Y = A + B$
- * The Boolean equation for EX-OR gate is $Y = A \oplus B$

LOGIC GATE DIAGRAM:

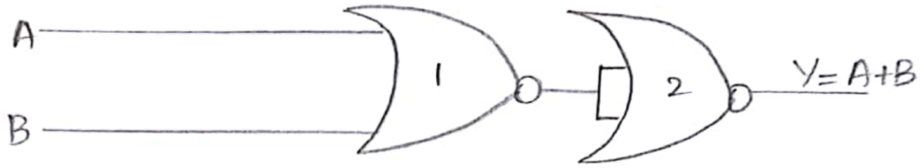
1. NOT FUNCTION: USING NOR GATE



Truth table:

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

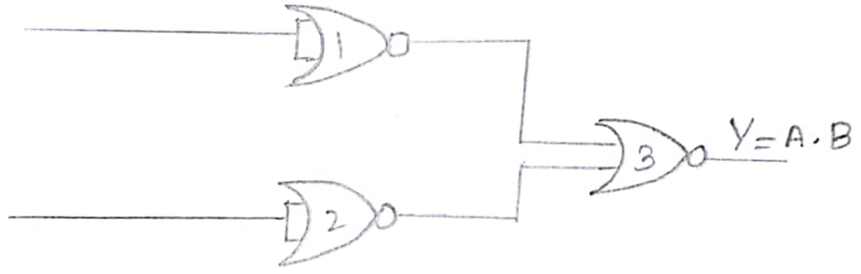
2. OR FUNCTION: USING NOR GATE:



Truth table:

A	B	Y=A+B
0	0	0
0	1	1
1	0	1
1	1	1

3. AND FUNCTION: USING NOR GATE

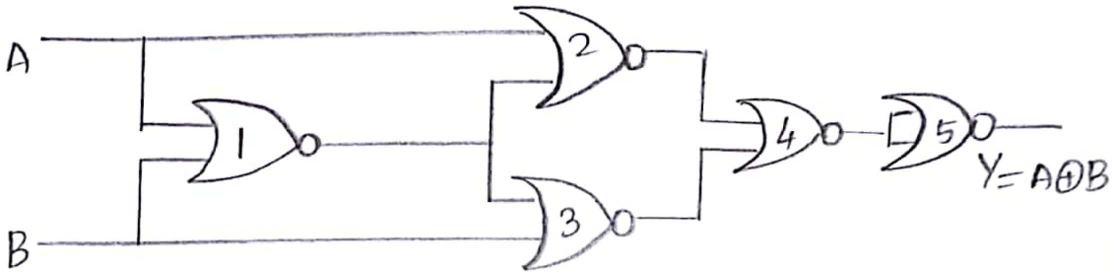


Truth table:

A	B	Y=A·B
0	0	0
0	1	0
1	0	0
1	1	1

Scanned by CamScanner

4. EX-OR FUNCTION: USING NOR GATE



Truth table:

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

RESULT:

Thus the realisation of NOT, OR, AND, EX-OR gates using NOR gate is verified.

EX. NO: 4

DATE:

K-MAP REDUCTION AND LOGIC CIRCUIT IMPLEMENTATION

AIM:

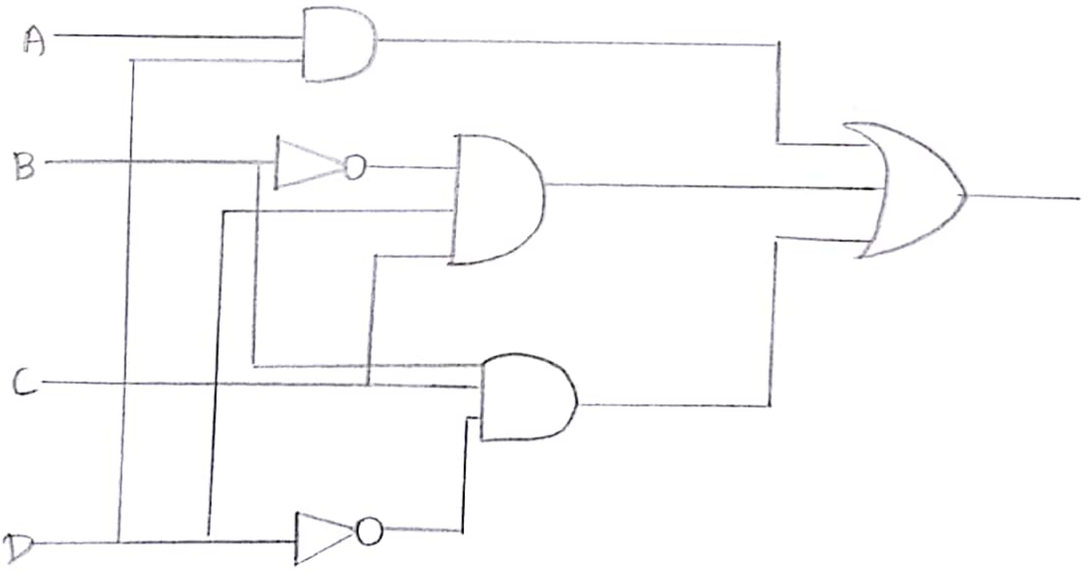
To design a simplified logic network using K-MAP.

K-MAP:

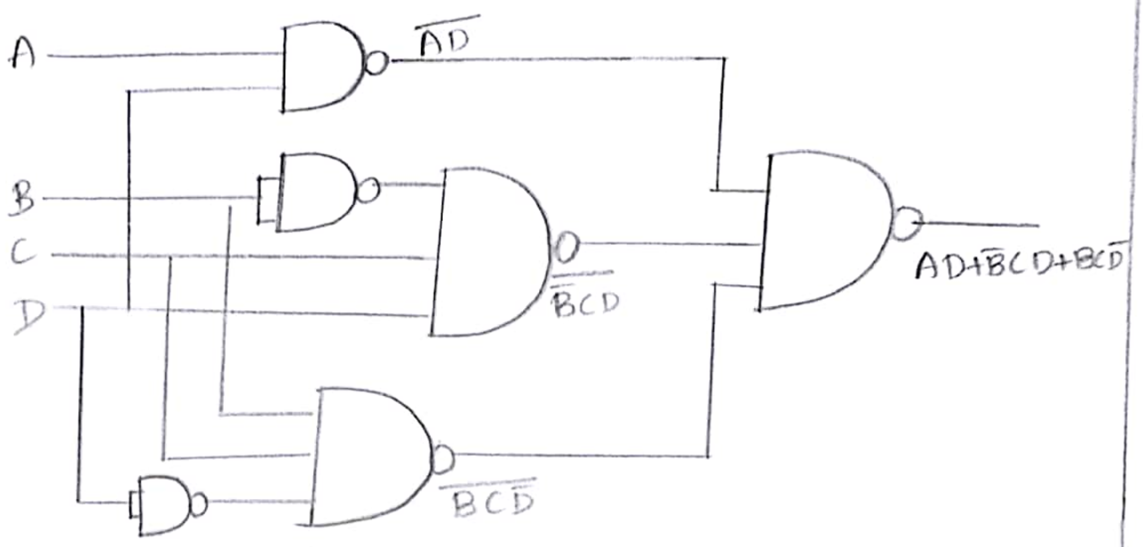
AB\CD	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	1	0
$\bar{A}B$	0	0		1
AB	X	X	X	X
$A\bar{B}$	0	1	X	X

$F = AD + \bar{B}CD + BC\bar{D}$

LOGIC DIAGRAM:



LOGIC DIAGRAM: USING NAND GATES



Truth table:

A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	x 0 d
1	0	1	1	x 1 d
1	1	0	0	x 0 d
1	1	0	1	x 1 d
1	1	1	0	x 1 d
1	1	1	1	x 1 d

(dont care condition)

Scanned by CamScanner

RESULT:

Thus the simplified logic ciruit using K-map is veified successfully.

Scanned by CamScanner

Ex.No: 5

DATE:

VERIFICATION OF DEMORGAN'S THEOREM

AIM:

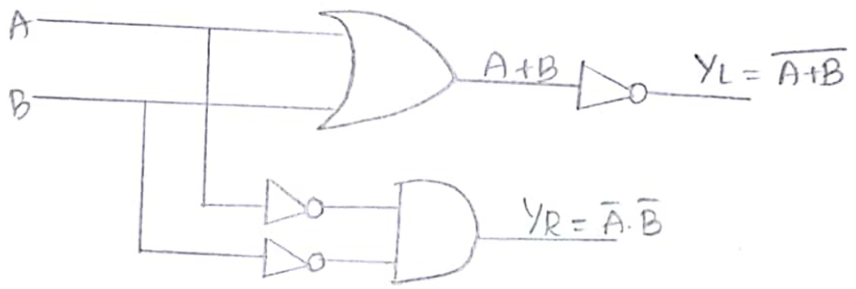
To verify demorgan's law.

FORMULA:

- * The complement of sum equals the product of the complement $\overline{A+B} = \bar{A}\cdot\bar{B}$
- * The complement of product equals the sum of the complement $\overline{A\cdot B} = \bar{A}+\bar{B}$

LOGIC DIAGRAM:

1. $\overline{A+B} = \bar{A}\cdot\bar{B}$

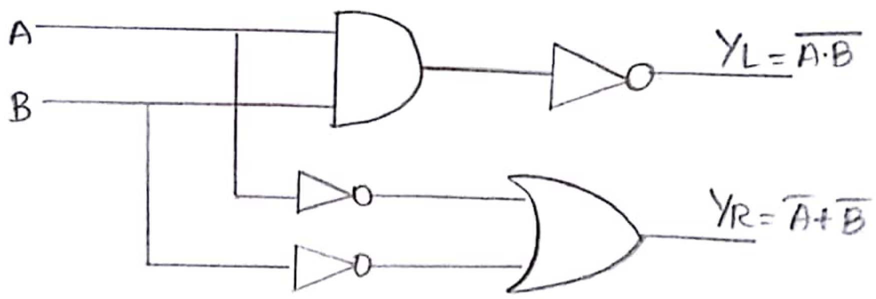


Truth table:

A	B	$\bar{A}$	$\bar{B}$	A+B	$Y_L = \overline{A+B}$	$Y_R = \bar{A}\cdot\bar{B}$
0	0	1	1	0	1	1
0	1	1	0	1	0	0
1	0	0	1	1	0	0
1	1	0	0	1	0	0

Scanned by CamScanner

2. $\overline{A \cdot B} = \overline{A} + \overline{B}$


A
B
$Y_L = \overline{A \cdot B}$
$Y_R = \overline{A} + \overline{B}$

A	B	$\overline{A}$	$\overline{B}$	$A \cdot B$	$Y_L = \overline{A \cdot B}$	$Y_R = \overline{A} + \overline{B}$
0	0	1	1	0	1	1
0	1	1	0	0	1	1
1	0	0	1	0	1	1
1	1	0	0	1	0	0

RESULT:

Thus the De-morgan's law is Verified.

EX.No:6

DATE:

VERIFICATION OF ASSOCIATIVE LAW

AIM:

To veify the associative law

FORMULAS:

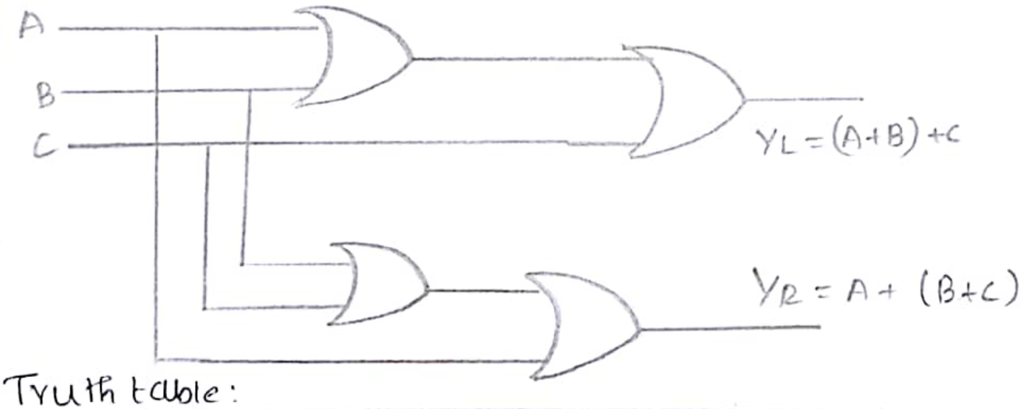
ASSOCIATIVE LAWS:

OR FUNCTION: $(A+B)+C = A+(B+C)$

AND FUNCTION: $A\cdot(B\cdot C) = (A\cdot B)\cdot C$

LOGIC DIAGRAM FOR ASSOCIATIVE LAW:

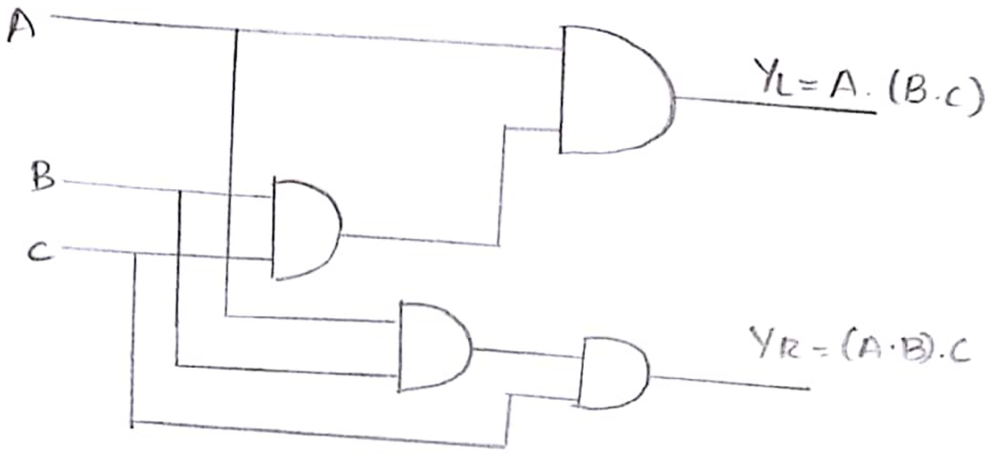
1. OR FUNCTION: $(A+B)+C = A+(B+C)$



Truth table:

A	B	C	A+B	B+C	$Y_L=(A+B)+C$	$Y_R=A+(B+C)$
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	1	0	1	1
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

2. AND FUNCTION: $A.(B.C) = (A.B).C$



Truth table:

A	B	C	B.C	A.B	$Y_L = A.(B.C)$	$Y_R = (A.B).C$
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	0	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	0	1	0	0
1	1	1	1	1	1	1

RESULT:

The associative law for OR, AND function is verified successfully.

Ex. No: 7 DATE:	VERIFICATION OF DISTRIBUTIVE LAW

AIM:

To verify the distributive law.

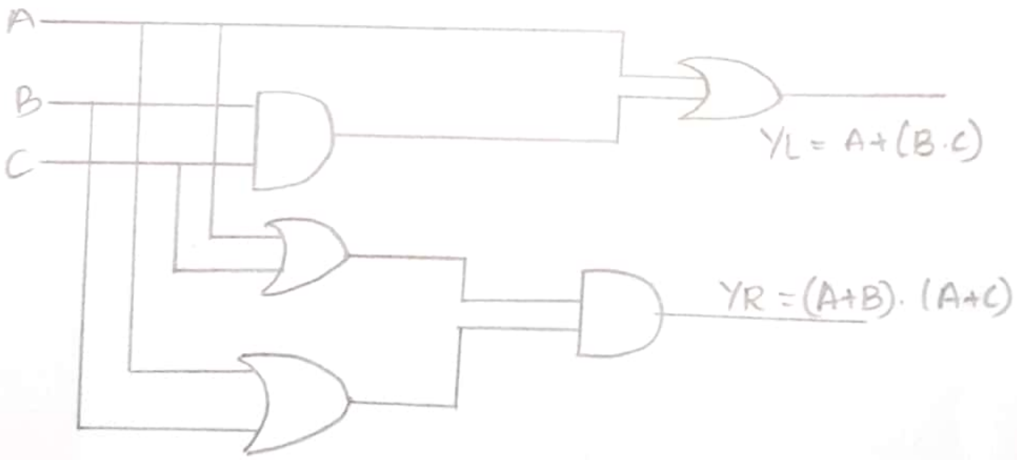
FORMULA:

DISTRIBUTIVE LAW:

1. OR FUNCTION: $A+(B \cdot C) = (A+B) \cdot (A+C)$.
2. AND FUNCTION: $A \cdot (B+C) = (A \cdot B)+(A \cdot C)$

LOGIC DIAGRAM FOR DISTRIBUTIVE LAW:

1. OR FUNCTION: $A+(B \cdot C)= (A+B) \cdot (A+C)$

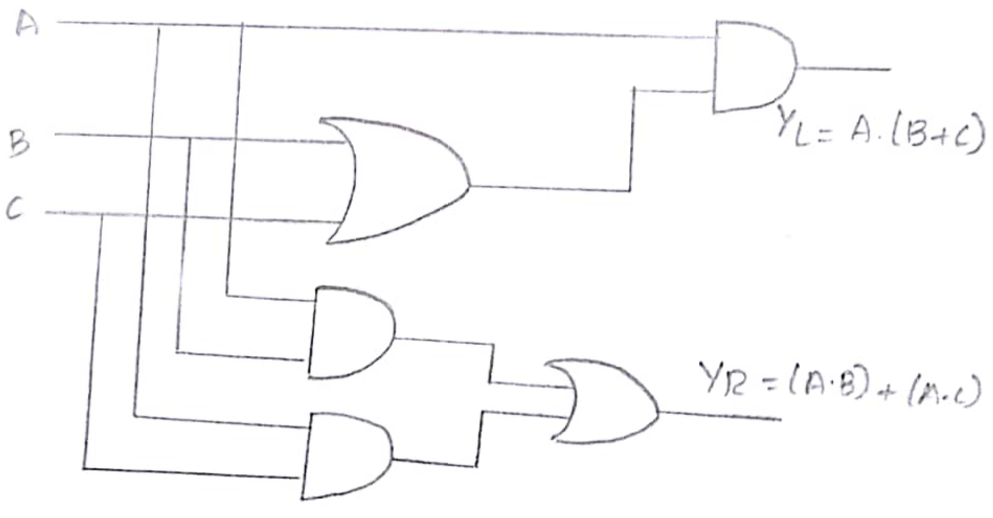


Scanned by CamScanner

Truth table:

A	B	C	BC	A+B	A+C	Y_L	Y_R
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	0	0	1	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

2. AND FUNCTION: $A.(B+C) = (A.B)+(A.C)$



Truth table:

A	B	C	B+C	A·B	A·C	YL	YR
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	0	1	1	1
1	1	0	1	1	0	1	1
1	1	1	1	1	1	1	1

RESULT:

The distributive law for OR, AND is veified successfully.

Scanned by CamScanner

Ex. No: 8 DATE:	IMPLEMENTATION OF HALF ADDER AND FULL ADDER

AIM:

To implement the half adder and full adder using basic gates and Ex-OR gates.

FORMULA:

1. HALF ADDER:

$$Sum = x'y + xy'$$

$$Carry = xy$$

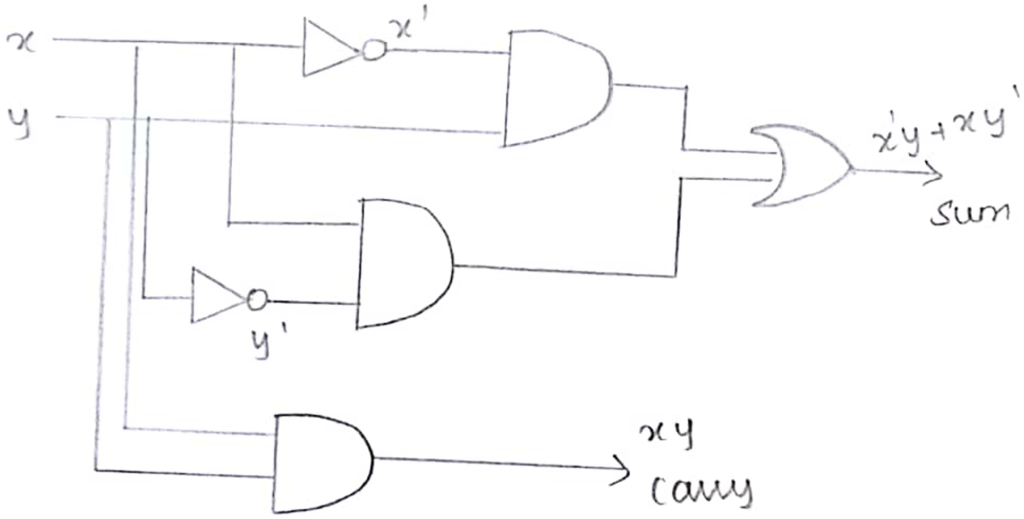
2. FULL ADDER:

$$Sum = z \oplus (x \oplus y)$$

$$Carry = z(x \oplus y) + xy$$

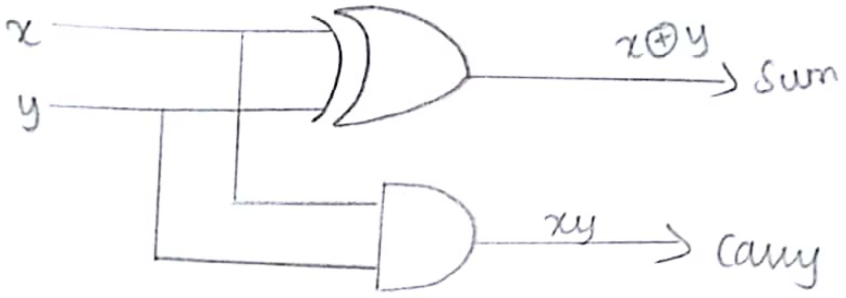
LOGIC GATE FOR HALF ADDER:

1. USING BASIC GATES:



Scanned by CamScanner

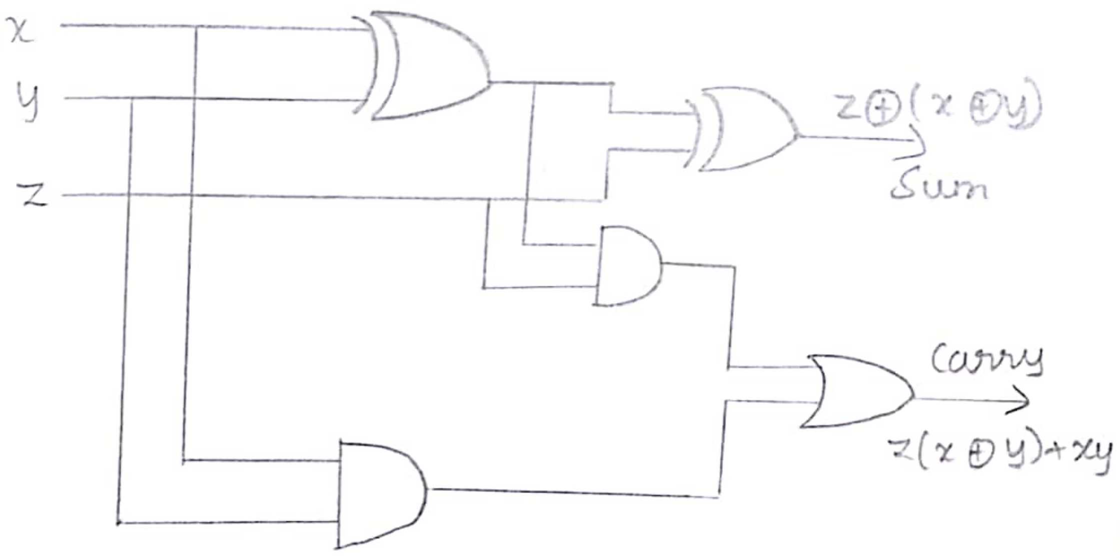
2. USING EX-OR gate:



Input		Output	
x	y	x	y
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

FULL ADDER:

1. USING BASIC AND EX-OR GATES:



Scanned by CamScanner

Truth table:

Input			Output	
x	y	z	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	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

RESULT:

Thus the implementation of half adder and full adder using basic gates and EX-OR gate are verified successfully.

Scanned by CamScanner

EX.No: 9

DATE:

IMPLEMENTATION OF HALF SUBTRACTOR AND FULL SUBTRACTOR

AIM:

To implement the half subtractor and full subtractor using basic gates and EX-OR gates.

FORMULA:

HALF SUBTRACTOR:

$$\text{Difference} = x'y + xy'$$

$$\text{Borrow} = x'y$$

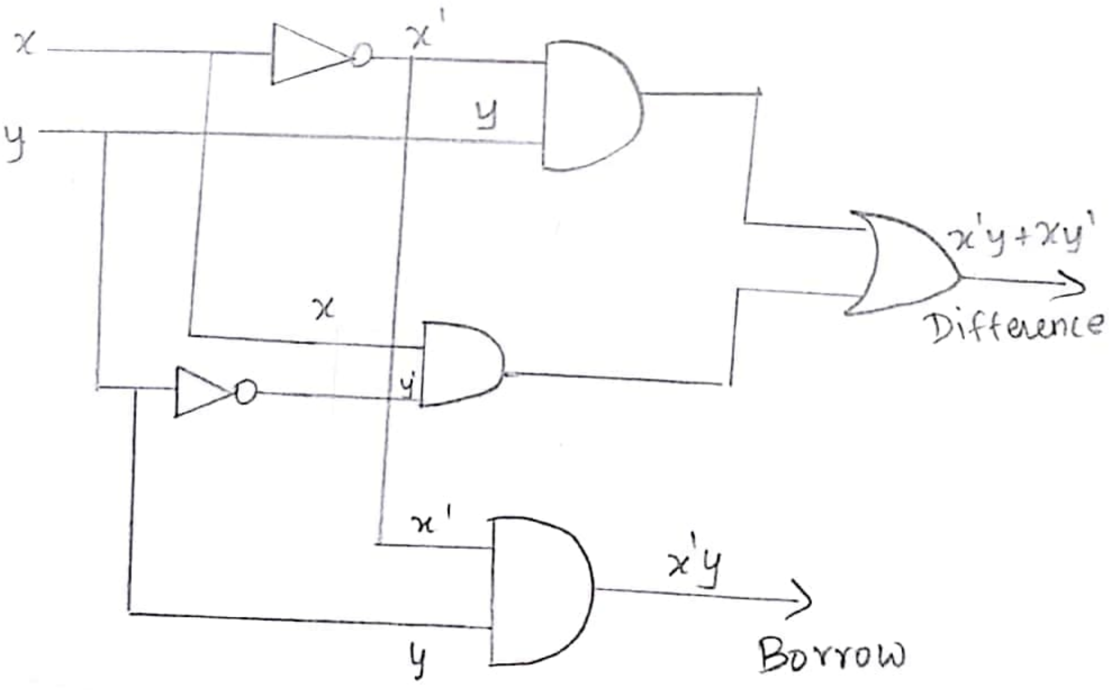
FULL SUBTRACTOR:

$$\text{Difference} = z \oplus (x \oplus y$$

$$\text{Borrow} = x'(y \oplus z) + yz$$

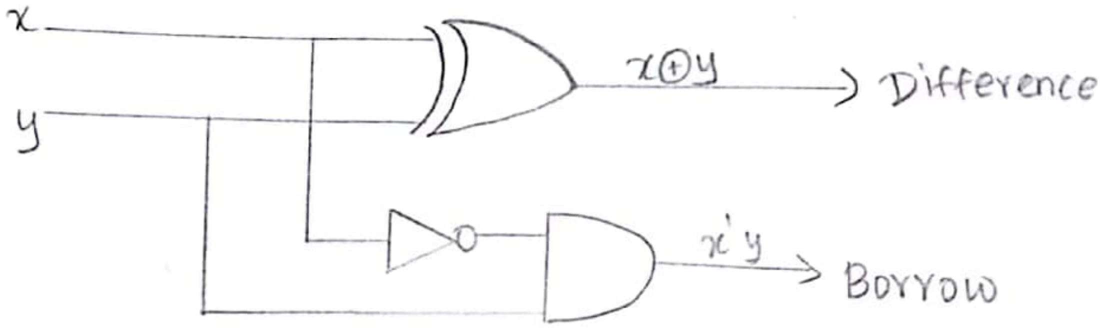
LOGIC DIAGRAM:

HALF SUBTRACTOR USING BASIC GATES :



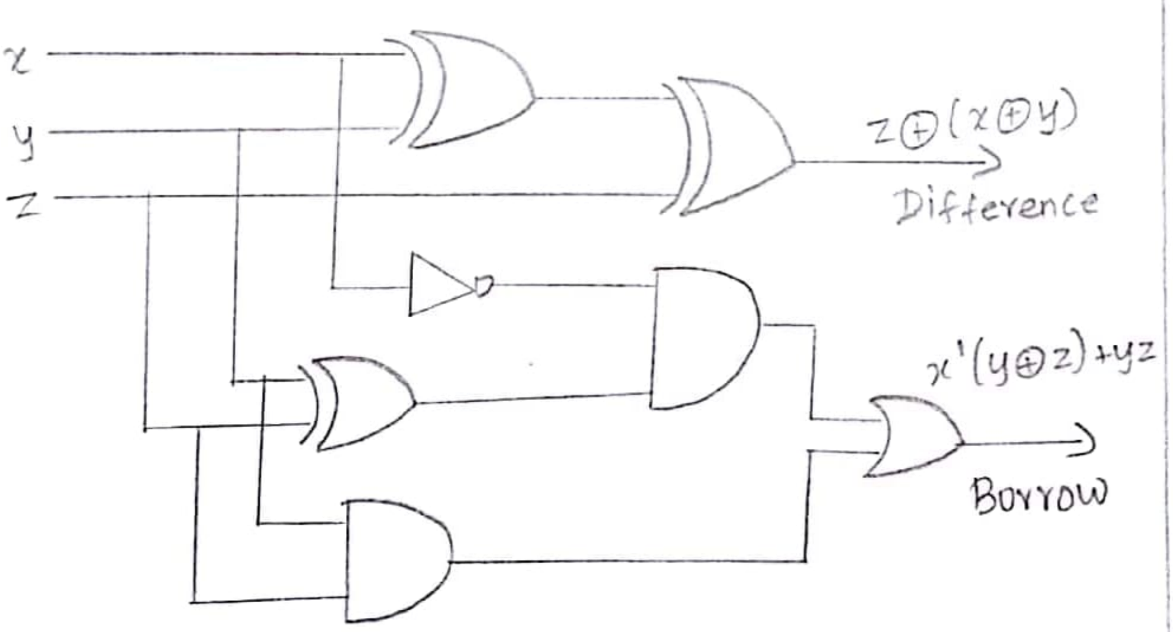
Scanned by CamScanner

2. USING EX-OR GATE:



Input		Output	
x	y	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

FULL SUBTRACTOR:



Truth table:

Input			Output	
x	y	z	Difference	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

RESULT:

Thus the implementation of half Subtractor and full subtractor using basic gates and EX-OR gates are verified successfully.

Scanned by CamScanner

EX.NO: 10

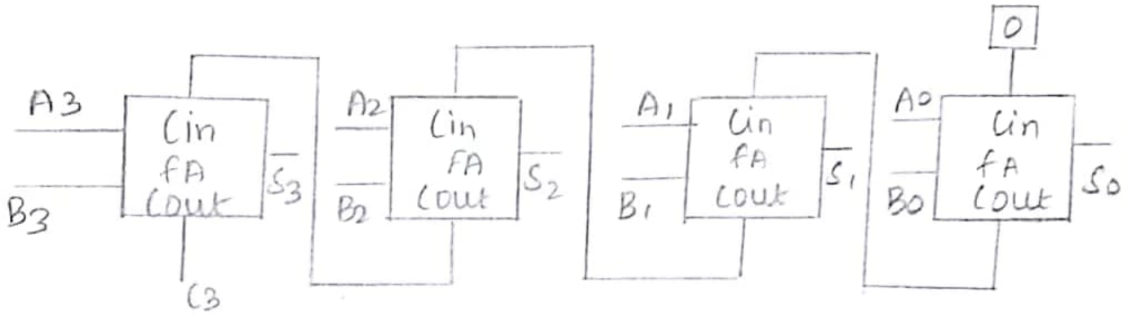
DATE:

FOUR BIT BINARY ADDER

AIM:

To Construct a four bit binary adder.

LOGIC DIAGRAM:



Truth table:

A_3	A_2	A_1	A_0	B_3	B_2	B_1	B_0	C_3	S_3	S_2	S_1	S_0
1	0	0	0	0	0	1	1	0	1	0	1	1
0	1	1	1	0	0	1	0	0	1	0	0	1
0	1	0	1	0	1	0	1	0	1	0	1	0
0	1	0	0	0	0	1	1	0	0	1	1	1
1	1	0	0	0	1	0	0	1	0	0	0	0

RESULT:

Thus the four bit binary adder is constructed and verified successfully.