JUNAID . GIRKAR
60004190057

DIGITAL ELECTRONICS
EXPERIMENT - 2

* AIM: To implement the logic functions i.e. AND, OR, NOT, Ex-OR, Ex-NOR and a logical expression with the help of NAND and NOR universal gates respectively.

* THEORY: logic gates are electronic circuits which perform logical functions on one or more inputs to produce one output. There are seven logic gates. when all the input combinations of a logic gate are written in a series and their corresponding outputs written along them, the input combination is called Truth Table.

1 NAND GATE AS UNIVERSAL GATE:

- NAND gate is actually a combination of two logic gates (AND followed by NOT). So, the output is the complement of the AND gate. This gate can have minimum 2 inputs; output is always one.
- By using only NAND gates, we can realize all logic functions: AND, OR, NOT, X-OR, X-NOR, NOR. So this gate is also called a universal gate.

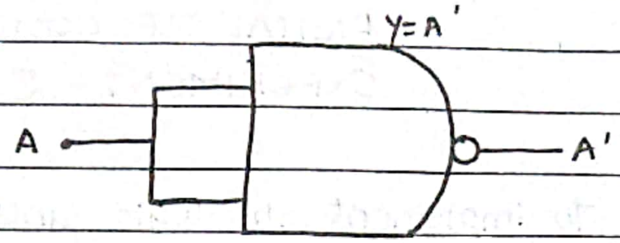
→ NAND gates as NOT gate

- A NOT produces the complement of the input. It can have only one input, tie the inputs of a NAND gate together. It will work as a NOT gate with the output.

- EXPRESSION: $Y = (A \cdot A)'$
  $= A'$

Page:
Date: / /

- DIAGRAM :



NOT (inverter)

TRUTH TABLE:

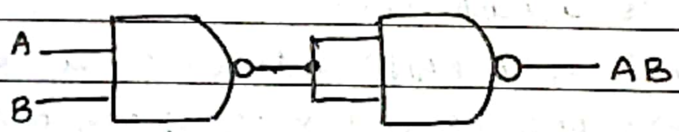
Input	Output
A	A'
0	1
1	0

→ NAND gates as AND gate.

- A NAND produces complement of the AND gate. So, if the output of a NAND gate is inverted, overall output will be that of AND gate.
- EXPRESSION : $Y = ((A \cdot B)')' = A \cdot B$

$Y = A \cdot B$

- DIAGRAM :



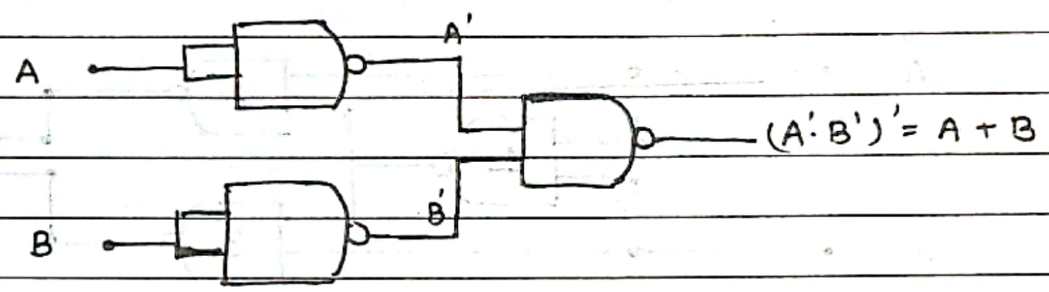
AND

TRUTH TABLE :

Input		Output
A	B	F=A·B
0	0	0
0	1	0
1	0	0
1	1	1

→ NAND gates as OR gate.

- From De Morgan's theorem : $(A \cdot B)' = A' + B'$
  $(A'B')' = A'' + B'' = A + B$
- So give the inverted outputs to the NAND gate, obtain OR operation at output.
- EXPRESSION : $Y = A + B$
- DIAGRAM :



OR

TRUTH TABLE :

A	B	X = A + B
0	0	0
0	1	1
1	0	1
1	1	1

→ NAND gates as Ex-OR gate.

- The output of two input Ex-OR gate is shown by $Y = A'B + AB'$

Gate No	INPUT	OUTPUT
1	A, B	$(AB)'$
2	A, $(AB)'$	$(A(AB)')'$
3	$(AB)'$, B	$(B(AB)')'$
4	$(A(AB)')'$, $(B(AB)')'$	$A'B + AB'$

The output from gate 4 is the overall output of the configuration.

Page:
Date: / /

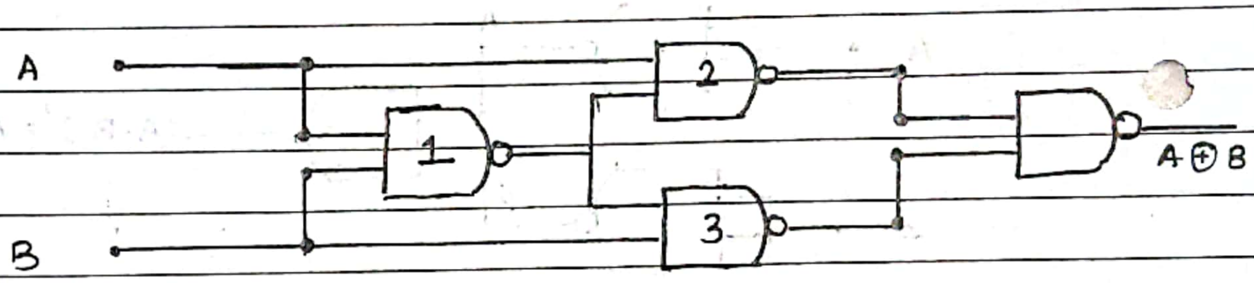
$$Y = ((A(AB)')' \cdot (B(AB)')')' = (A(AB)')'' + (B(AB)')''$$

$$Y = A(AB)' + B(AB') = (A(A'+B)') + (B(A'+B'))$$

$$Y = AA' + AB' + BA' + BB' = 0 + AB' + BA' + 0$$

$$Y = AB' + A'B$$

- EXPRESSION : $Y = AB' + A'B$
- DIAGRAM



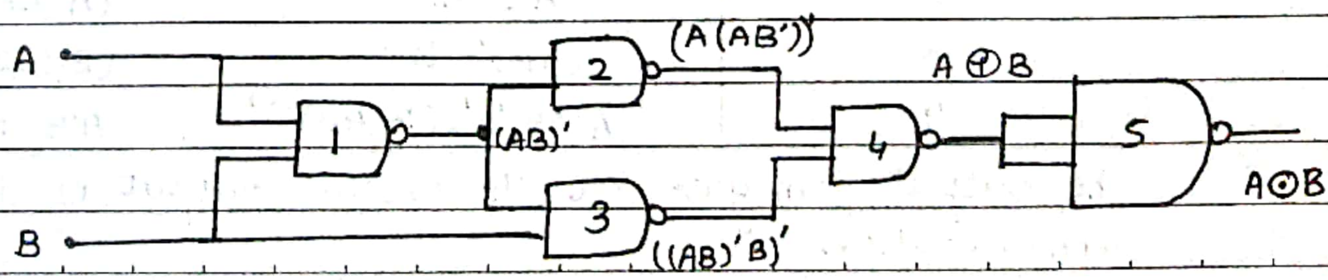
X-OR

TRUTH TABLE

A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

→ NAND gates as Ex-NOR gate

- Ex-NOR gate is the Ex-OR gate followed by the NOT gate. so given the output of Ex-OR gate to a NOT gate, overall output is that of Ex-NOR gate.
- EXPRESSION : $Y = AB + A'B'$



X-NOR

- TRUTH TABLE :

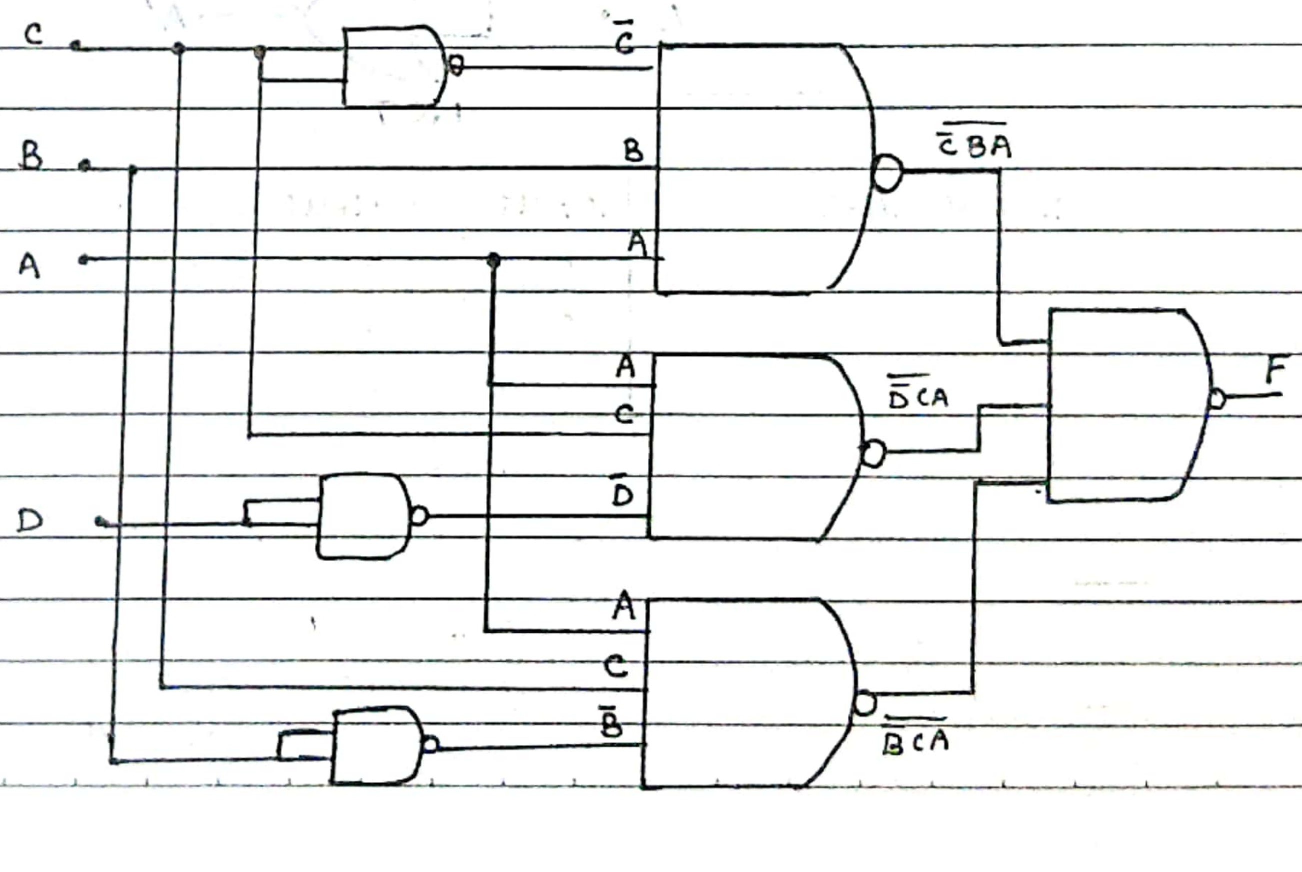
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

→ IMPLEMENTING THE SIMPLIFIED FUNCTIONS WITH NAND GATES ONLY

- EXPRESSION: $F = ((C'.B.A)'(D'.C.A)'(C.B'.A)')'$

- The entire expression is inverted and we have three terms ANDed. This means that we must use a 3-input NAND gate. Each of the three terms is itself a NAND expression. Finally, negated single terms can be generated with a 2-input NAND gate acting inverted.

$$F = \overline{(\overline{\bar{C}\cdot B\cdot A})(\overline{\bar{D}\cdot C\cdot A})(\overline{C\cdot\bar{B}\cdot A})}$$



Page:
Date: / /

2. NOR GATE AS UNIVERSAL GATE

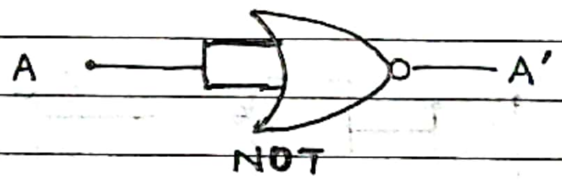
- NOR gate is actually a combination of two logic gates: OR gate followed by NOT gate. So its output is complement of the output of an OR gate. This gate can have minimum two inputs; output is always one. By using NOR gates, we can realize all logic functions: AND, OR, NOT, EX-OR, EX-NOR, NAND. So, this gate is also called a universal gate.

→ NOR gates as NOT gate:

- A NOT produces complement of the input. It can have only one input, and after trying the inputs of a NOR gate together, it will work as a NOT gate.

- EXPRESSION: $Y = (A + A)' = A'$

- DIAGRAM:

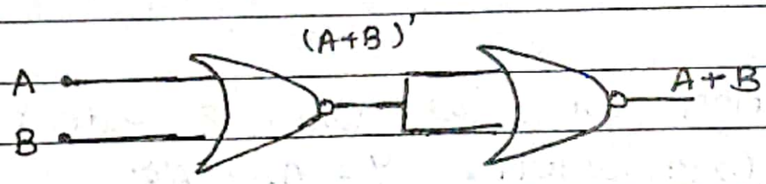


TRUTH TABLE:

INPUT	OUTPUT
A	A'
0	1
1	0

→ NOR GATES AS OR GATE :

- A NOR produces complement of OR gate. so, if the output of a NOR gate is inverted, overall output will be that of an OR gate.
- EXPRESSION : $Y = ((A+B)')' = A+B$
- DIAGRAM :



OR

- TRUTH TABLE

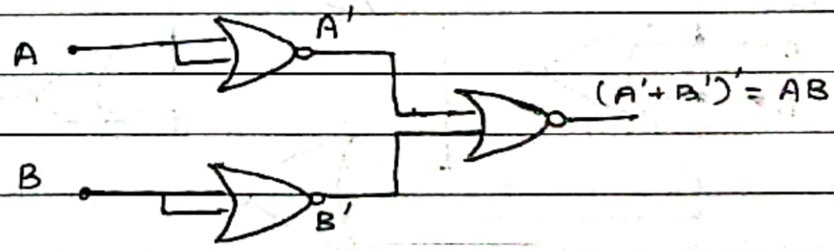
A	B	X=A+B
0	0	0
0	1	1
1	0	1
1	1	1

→ NOR GATES AS AND GATE :

- By De Morgan's Law : $(A+B)' = A'B'$
  $(A'+B')' = A \cdot B$

so give the inverted inputs to a NOR gate, obtain AND operation at output

- EXPRESSION : $Y = A \cdot B$
- DIAGRAM :



AND

TRUTH TABLE :

INPUT		OUTPUT
A	B	$F = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

→ NOR gates as Ex-NOR gate :

- EXPRESSION : $Y = AB + A'B'$

Gate No	Inputs	output
1	$A \cdot B$	$(A+B)'$
2	$A \cdot (A+B)'$	$(A + (A+B)')'$
3	$(A+B)' \cdot B$	$(B + (A+B)')'$
4	$(A+(A+B)')' \cdot (B+(A+B)')'$	$AB + A'B'$

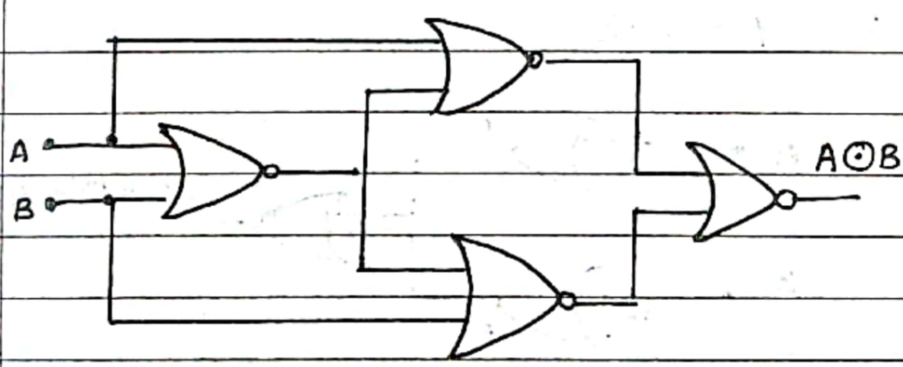
$$Y = ((A+(A+B)')' \cdot (B+(A+B)')')' = (A + (A+B)')(B+(A+B)')$$

$$Y = (A+A')(A+B')(B+A')(B+B') = A(B+A') + B'(B+A')$$

$$Y = AB + AA' + B'B + B'A' = AB + B'A'$$

$$Y = AB + A'B'$$

DIAGRAM



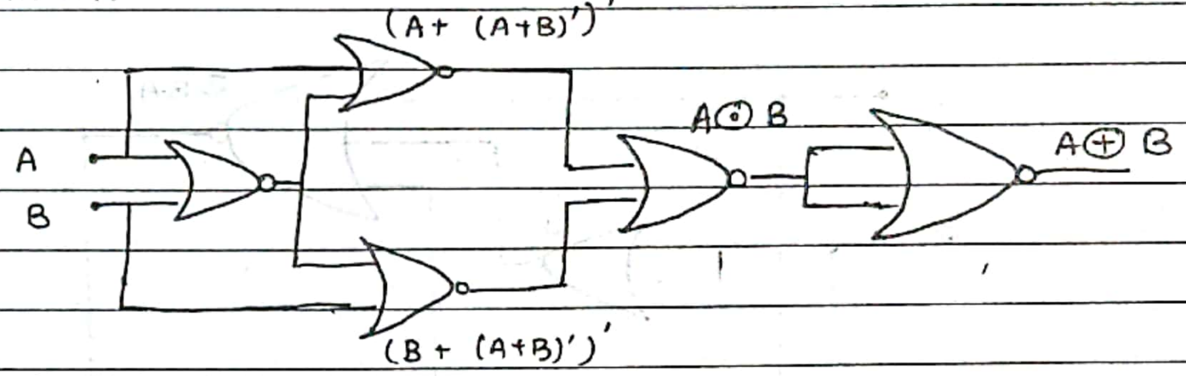
X-NOR

TRUTH TABLE

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

→ NOR gates as Ex-OR gate :

- Ex-OR gate is actually Ex-NOR gate followed by NOT gate. So give the output of Ex-NOR gate to a NOT gate, overall output is that of an Ex-OR gate.
- EXPRESSION : $Y = A'B + AB'$
- DIAGRAM:



X - OR.

- TRUTH TABLE

A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

→ CONSTRUCTING A CIRCUIT WITH NOR GATES ONLY

- $F = (((C\cdot\bar{B}\cdot A) + (D\cdot\bar{C}\cdot A) + (C\cdot\bar{B}\cdot A))')'$

- DIAGRAM :