NAND and NOR

As a universal gates

Eng. : Eman Abu Hani

What are a Universal Gate And why NAND and NOR are known as universal gates?

- A gate which can be use to create any Logic gate is called Universal Gate

- NAND and NOR are called Universal Gates because all the other gates can be created by using these gates

Proof for NAND gates

- Any Boolean function
  can be implemented using AND, OR and NOT gates

- In the same way AND, OR and NOT gates can be implemented using NAND gates only,

Implementation of NOT using NAND

A NOT gate is made by joining the inputs of a NAND gate together.

Desired NOT Gate

NAND Construction

Input	Output
0	1
1	0

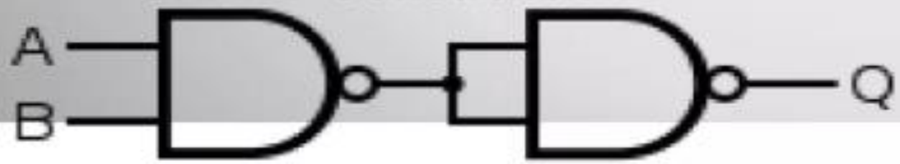
Implementation of AND using NAND

- A NAND gate is an inverted AND gate.

- An AND gate is made by following a NAND gate with a NOT gate

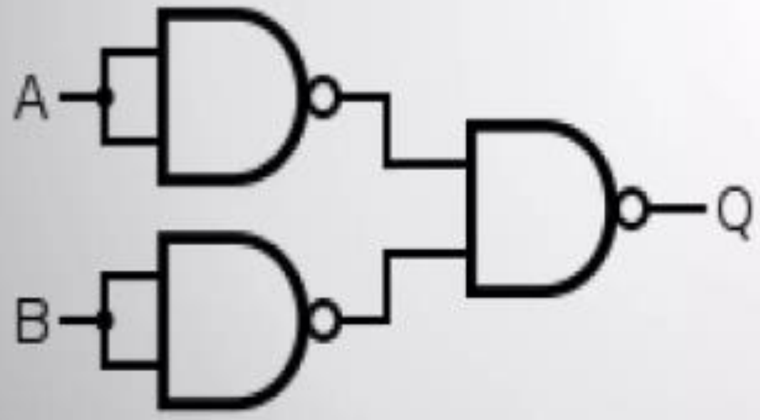
Input A	Input B	Output Q
0	0	0
0	1	0
1	0	0
1	1	1

NAND Construction



Implementation of OR gate using NAND

- If the truth table for a NAND gate is examined or by applying De Morgan's Laws, it can be seen that if any of the inputs are 0, then the output will be 1. To be an OR gate,

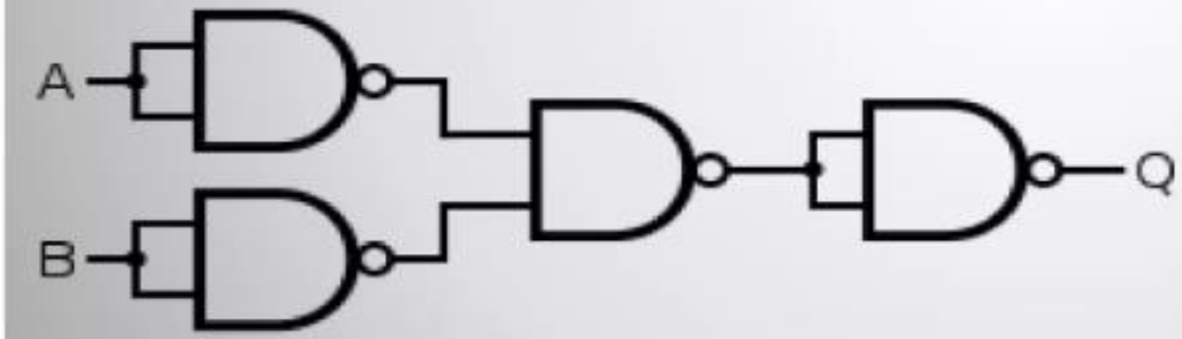


NAND Construction

Input A	Input B	Output Q
0	0	0
0	1	1
1	0	1
1	1	1

Implementation of NOR gate using NAND

- A NOR gate is simply an inverted OR gate. Output is high when neither input A nor input B is high:

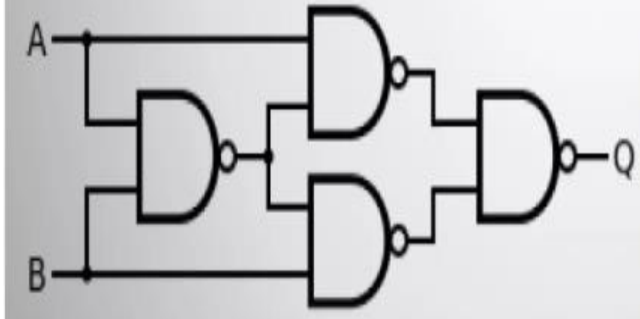


NAND Construction

Input A	Input B	Output Q
0	0	1
0	1	0
1	0	0
1	1	0

Implementation of XOR gate using NAND

- An XOR gate is constructed similarly to an OR gate, except with an additional NAND gate inserted such that if both inputs are high, the inputs to the final NAND gate will also be high,

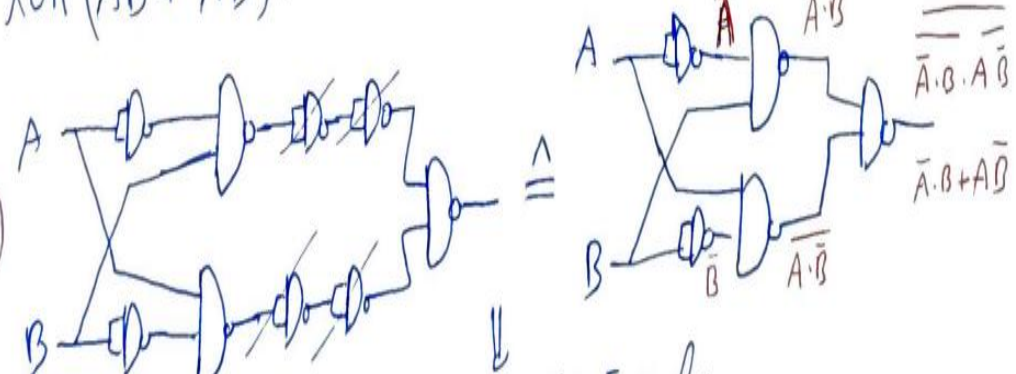


NAND Construction

Input A	Input B	Output Q
0	0	0
0	1	1
1	0	1
1	1	0

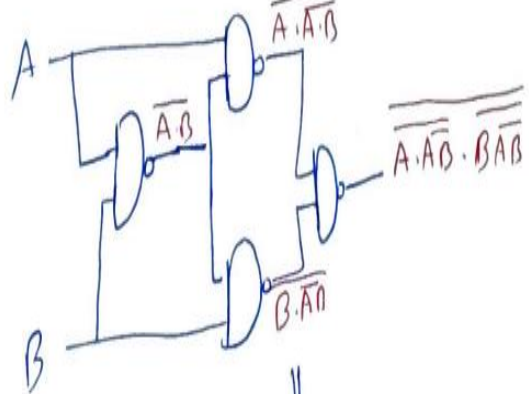
5) XOR $(A\bar{B} + \bar{A}B)$:-

①



⇓ we use 5 gates

②



$\overline{\overline{A\cdot\overline{AB}}\cdot\overline{B\overline{AB}}} = \overline{\overline{A\cdot\overline{AB}}} + \overline{\overline{B\overline{AB}}}$

$= A\cdot\overline{AB} + B\cdot\overline{AB}$

$= A(\bar{A}+\bar{B}) + B(\bar{A}+\bar{B})$

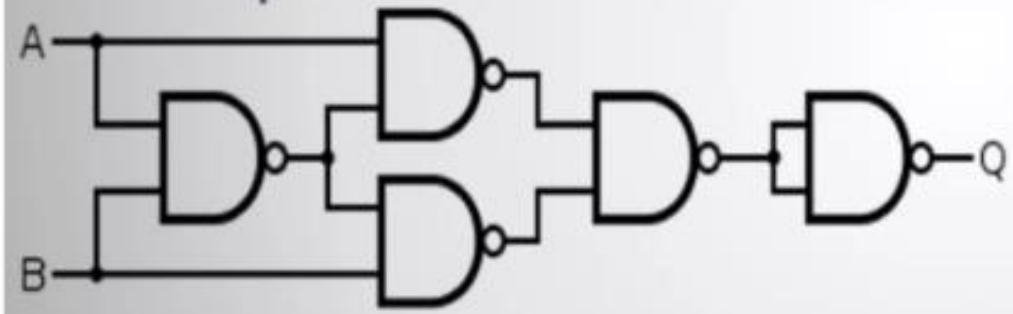
$= A\bar{A} + A\bar{B} + \bar{A}B + B\bar{B}$

$= A\cdot\bar{B} + \bar{A}B \Rightarrow$ XOR

⇓ we use 4 gates

Implementation of XNOR gate using NAND

- An XNOR gate is simply an XOR gate with an inverted output:



NAND Construction

Input A	Input B	Output Q
0	0	1
0	1	0
1	0	0
1	1	1

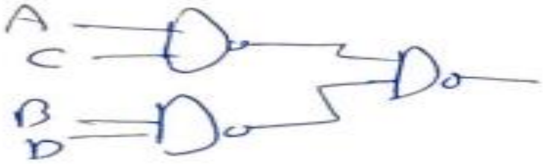
- AB + AB ⟶ X nor

** We can use calculation method to convert the gates into NAND gate

Example (1): Convert AC+BD by using NAND gate only?

$\overline{\overline{AC + BD}} = \overline{\overline{AC} \cdot \overline{BD}}$

1) using double negation
2) the expression must be in SOP form.

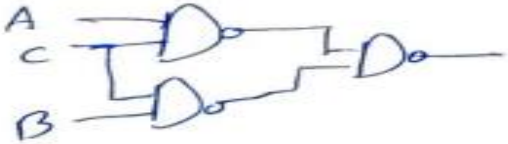


Example (2): (A+B) C

$C(A+B) = AC + BC$ ⟶ SOP form

$= \overline{\overline{AC + BC}}$ double negation

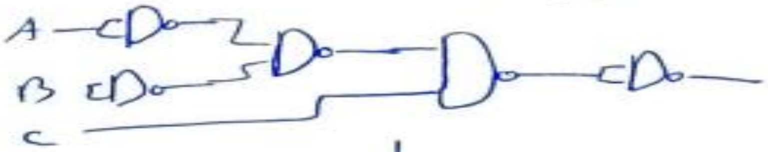
$= \overline{\overline{AC} \cdot \overline{BC}}$



⟶ 3 gates

* If we convert the expression by using NAND Realization

(A+B).C
↓ OR ↓ AND



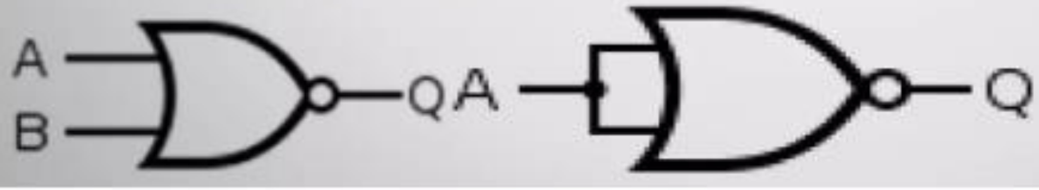
↓
5 gates not minimum gate

Proof for NOR gates

- Like NAND gates, NOR gates are so-called "universal gates" that can be combined to form any other kind of logic gate. A NOR gate is logically an inverted OR gate

Implementation of NOT gate using NOR

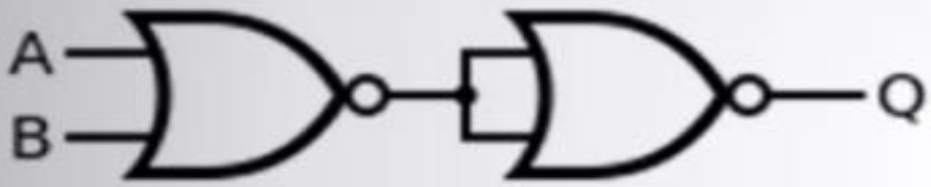
- NOT made by joining the inputs of a NOR gate.



Input	Output
0	1
1	0

Implementation of OR gate using NOR

- The OR gate is simply a NOR gate followed by another NOR gate

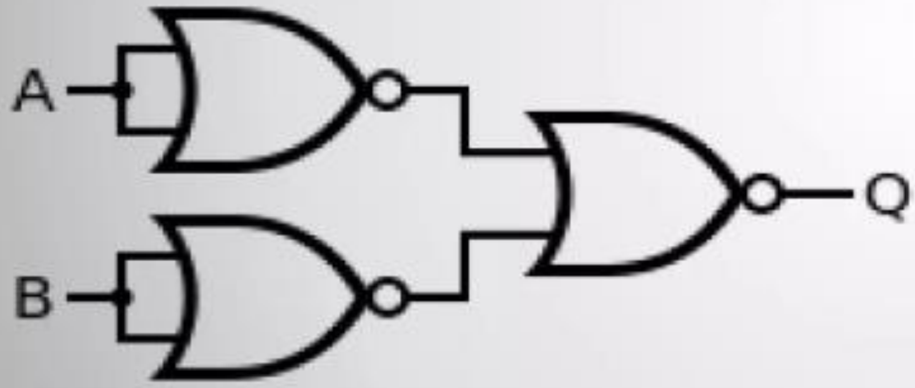


Desired Gate

Input A	Input B	Output Q
0	0	0
0	1	1
1	0	1
1	1	1

Implementation of AND gate using NOR

- an AND gate is made by inverting the inputs to a NOR gate.

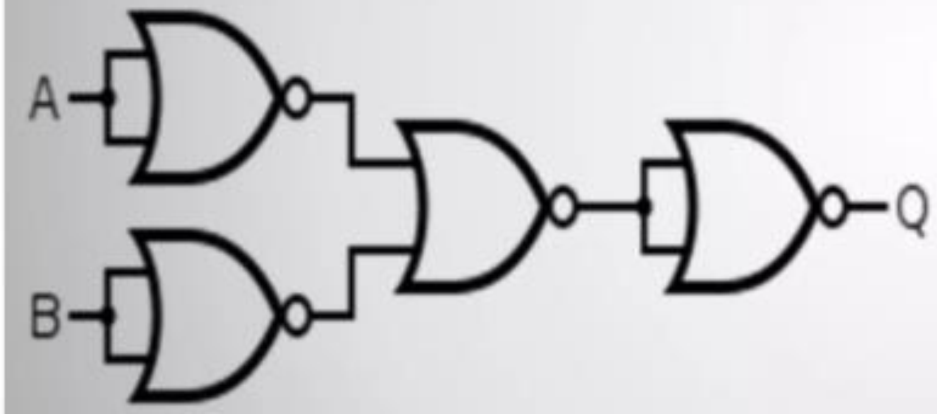


NOR Construction

Input A	Input B	Output Q
0	0	0
0	1	0
1	0	0
1	1	1

Implementation of NAND gate using NOR

- A NAND gate is made using an AND gate in series with a NOT gate:

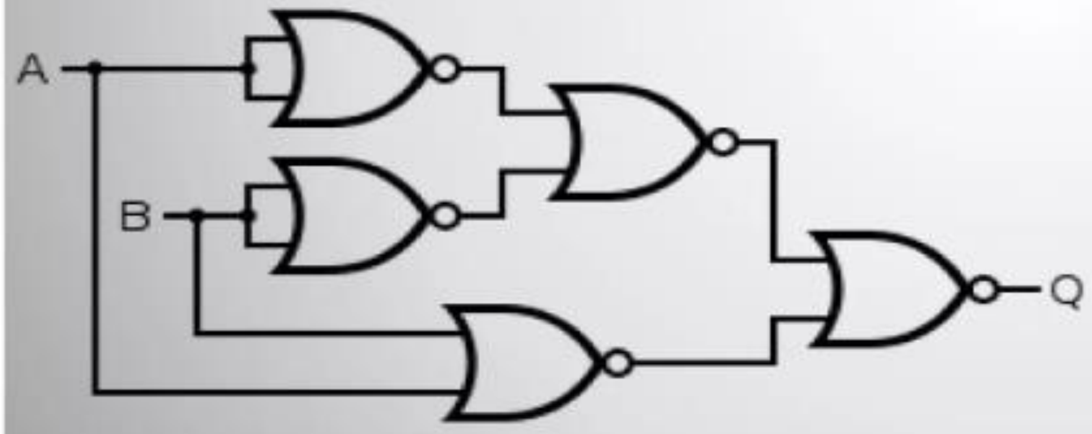


NOR Construction

Input A	Input B	Output Q
0	0	1
0	1	1
1	0	1
1	1	0

Implementation of XOR gate using NOR

- An XOR gate is made by connecting the output of 3 NOR gates (connected as an AND gate) and the output of a NOR gate to the respective inputs of a NOR gate.

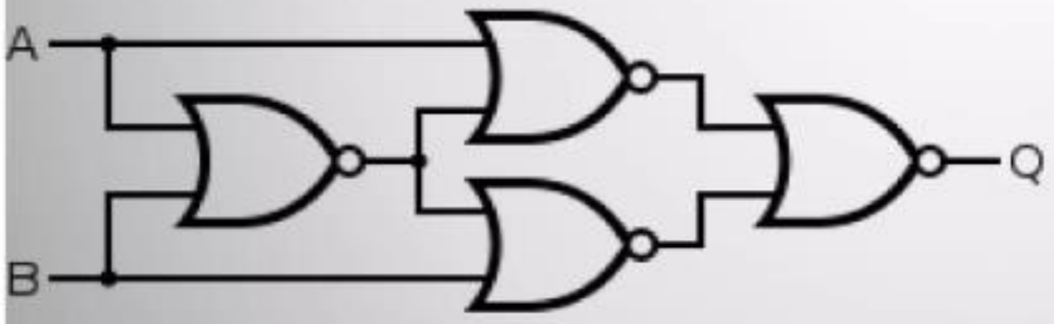


NOR Construction

Input A	Input B	Output Q
0	0	0
0	1	1
1	0	1
1	1	0

Implementation of XNOR gate using NOR

- An XNOR gate can be constructed from four NOR gates implementing the expression "(A NOR N) NOR (B NOR N) where N = A NOR B". This construction has a propagation delay three times that of a single NOR gate, and uses more gates.



NOR Construction

Input A	Input B	Output Q
0	0	1
0	1	0
1	0	0
1	1	1

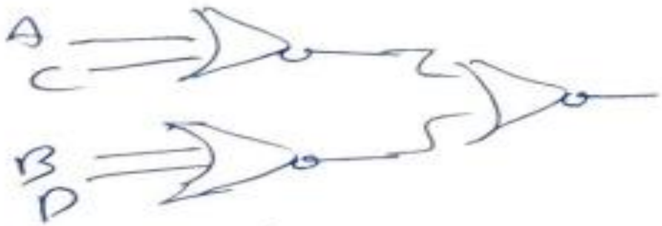
** we can use calculation method to convert the gates into NOR gate

1- the expression must be in POS form

example (1) $(A+C)(B+D)$ using calculation

$$\overline{\overline{(A+C)(B+D)}}$$

$$= \overline{\overline{A+C} + \overline{B+D}}$$



example (2) $(A+B)\,\bar{C}$

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

1.2.5 Representation of Switching Networks in NAND or NOR Technology

❐ Experiment 1: Pseudo-tetrade monitoring

For tetradic codes (e. g. 8421-BCD code), a decimal number is converted into a 4-digit binary number (tetrade). Sixteen different combinations can be represented with one tetrade of which only 10 combinations are required. The combinations which cannot be assigned to decimal numbers are referred to as **pseudo-tetrades**.

In the experiment, a circuit is to be designed which produces a 1-signal at output Q when a pseudo-tetrade is applied to the inputs (pseudo-tetrade monitoring).

❐ Experiment 1: Pseudo-tetrade monitoring

Decimal number	Dual number				Q
	A (8)	B (4)	C (2)	D (1)	
0	0	0	0	0	*0*
1	0	0	0	1	*0*
2	***0***	***0***	***1***	***0***	***0***
3	***0***	***0***	***1***	***1***	***0***
4	***0***	***1***	***0***	***0***	***0***
5	***0***	***1***	***0***	***1***	***0***
6	***0***	***1***	***1***	***0***	***0***
7	***0***	***1***	***1***	***1***	***0***
8	***1***	***0***	***0***	***0***	***0***
9	***1***	***0***	***0***	***1***	***0***
Pseudo-tetrades	***1***	***0***	***1***	***0***	***1***
	1	***0***	***1***	***1***	***1***
	1	***1***	***0***	***0***	***1***
	1	***1***	***0***	***1***	***1***
	1	***1***	***1***	***0***	***1***
	1	***1***	***1***	***1***	***1***

$$Q = A\bar{B}C\bar{D} \vee A\bar{B}CD \vee AB\bar{C}\bar{D} \vee AB\bar{C}D \vee ABC\bar{D} \vee ABCD$$

$$= A\bar{B}C(\bar{D} \vee D) \vee AB\bar{C}(\bar{D} \vee D) \vee ABC(\bar{D} \vee D)$$

$$= A(\bar{B}C \vee B\bar{C} \vee BC)$$

$$= A(\bar{B}C \vee B\bar{C} \vee BC \vee B\bar{C})$$

$$= A[C(B \vee \bar{B}) \vee B(\bar{C} \vee C)]$$

$$Q = \mathbf{A(C \vee B)}$$

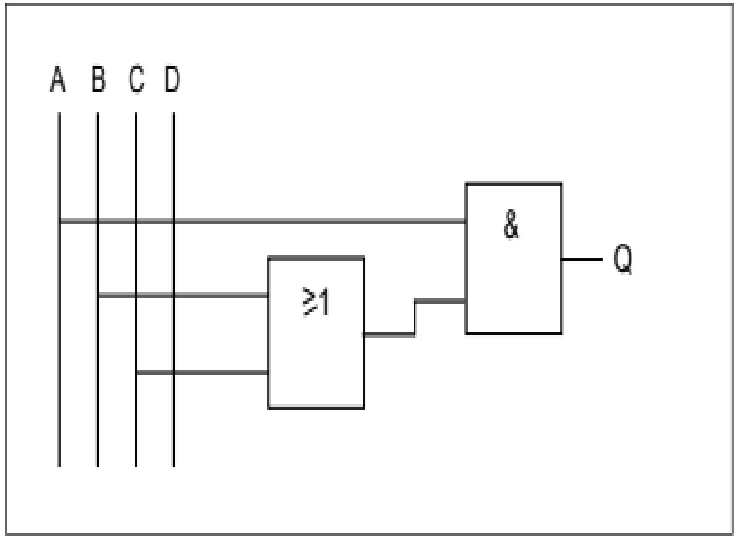
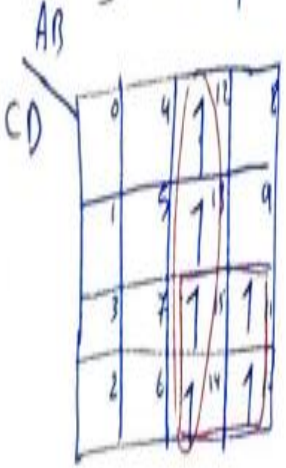


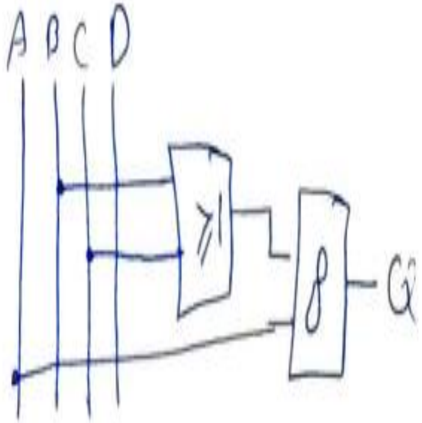
Fig. 1.2.5.1 Circuit

by using k-map:



$Q = AB + AC$

$A(B+C)$



❐ Experiment 2: NAND technology

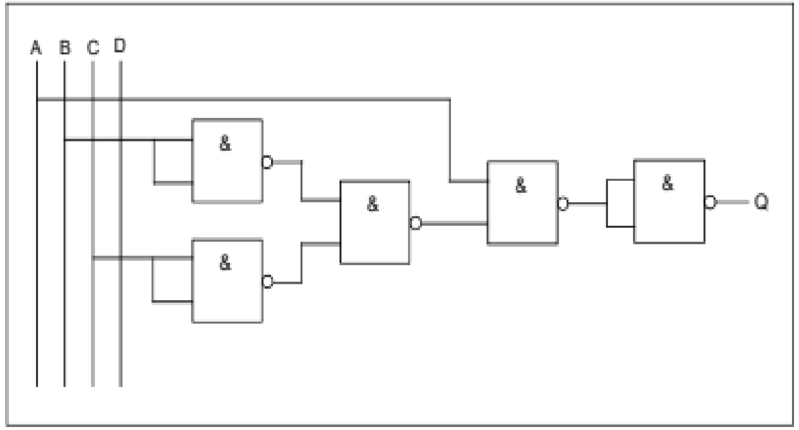


Fig. 1.2.5.2 Circuit in NAND technology

$Q = A\,(C \vee B) = A\,C \vee A\,B$

$\bar{Q} = \overline{A\,C \vee A\,B} = \overline{A\,C} \wedge \overline{A\,B}$

$\bar{\bar{Q}} = Q = \overline{\overline{A\,C} \wedge \overline{A\,B}}$

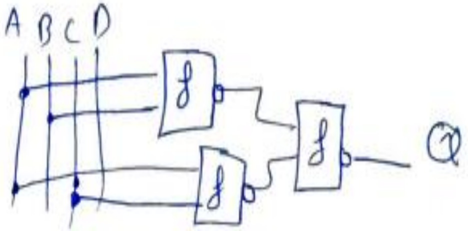
Experiment(2) NAND technology P.20

$Q = AB + AC \Rightarrow$ SOP form

$\bar{Q} = \overline{AB + AC}$

$\bar{\bar{Q}} = \overline{\overline{AB + AC}}$

$= \overline{\overline{AB} \cdot \overline{AC}}$



❐ Experiment 3: NOR technology

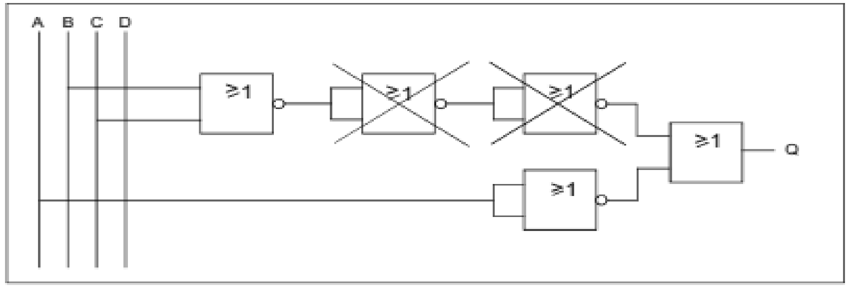


Fig. 1.2.5.3 Circuit in NOR technology

$Q = A\,(B \vee C)$

$\bar{Q} = \overline{A\,(B \vee C)} = \bar{A} \vee \overline{B \vee C}$

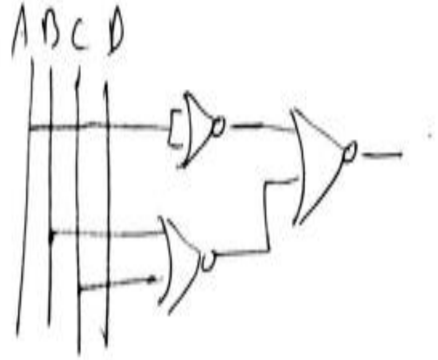
$\bar{\bar{Q}} = Q = \overline{\bar{A} \vee \overline{B \vee C}}$

Experiment3: NOR technology P.21

$Q = A(B+C) \Rightarrow$ POS form

$\bar{Q} = \overline{A(B+C)}$

$\bar{\bar{Q}} = \overline{\overline{A(B+C)}} = \overline{\bar{A} + \overline{B+C}}$



1.2.6 Equivalence

❐ Experiment 1: Fundamental principles

Examine the circuit for equivalence.

❐ Experiment 1: Fundamental principles

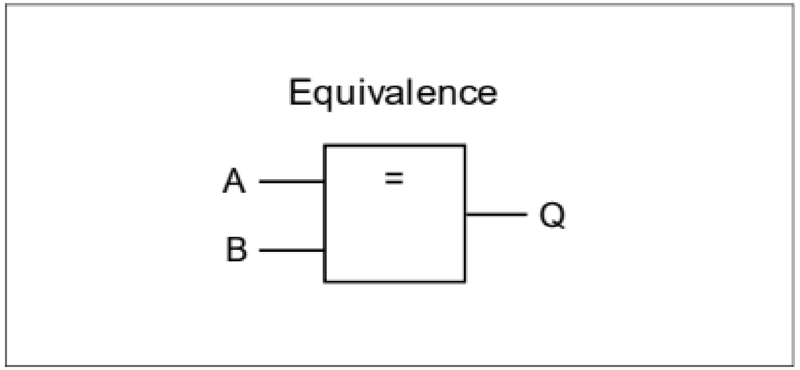


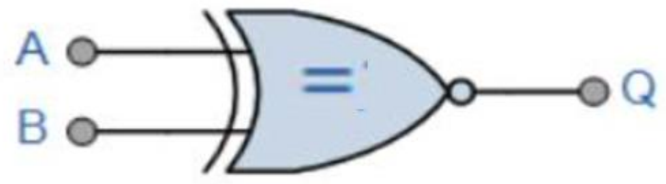
Fig. 1.2.6.1 Circuit symbol

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	1

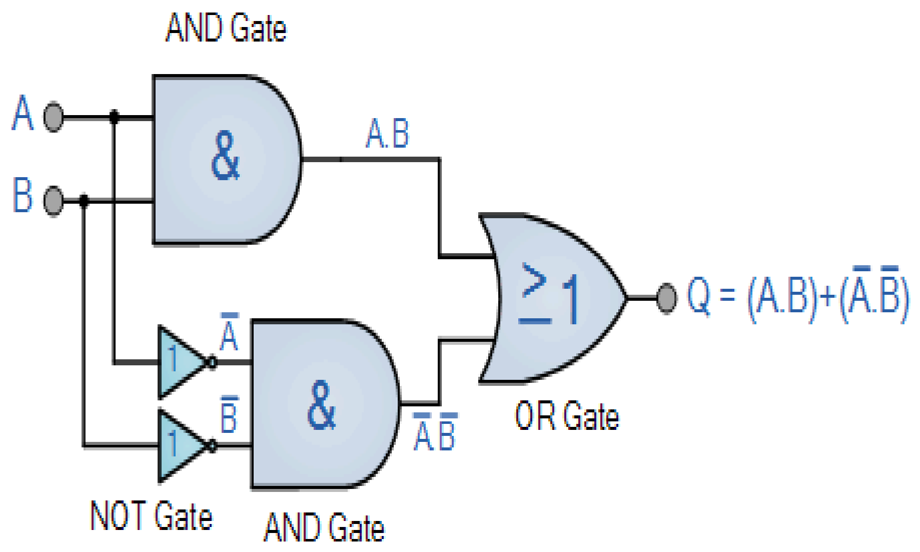
Table 1.2.6.1 Value table

$Q = \overline{A}\,\overline{B} \vee A\,B$

Symbol

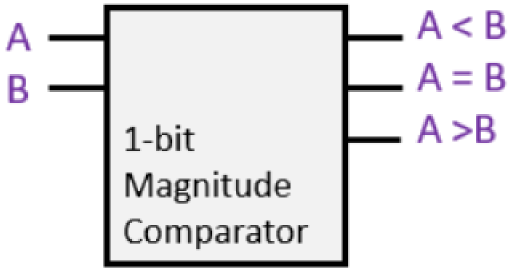


2-input Ex-NOR Gate



❒ Experiment 2: 1-bit number comparator

Design the circuit of a 1-bit number comparator in which a greater-smaller comparison is also carried out in addition to equality of two 1-digit dual numbers P and Q.



❒ Experiment 2: 1-bit number comparator

P	Q	P > Q	P = Q	P < Q
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

Table 1.2.6.2 Value table

P > Q: $\mathbf{P\,\overline{Q}}$

P = Q: $\mathbf{\overline{P}\,\overline{Q} \vee P\,Q}$

P < Q: $\mathbf{\overline{P}\,Q}$

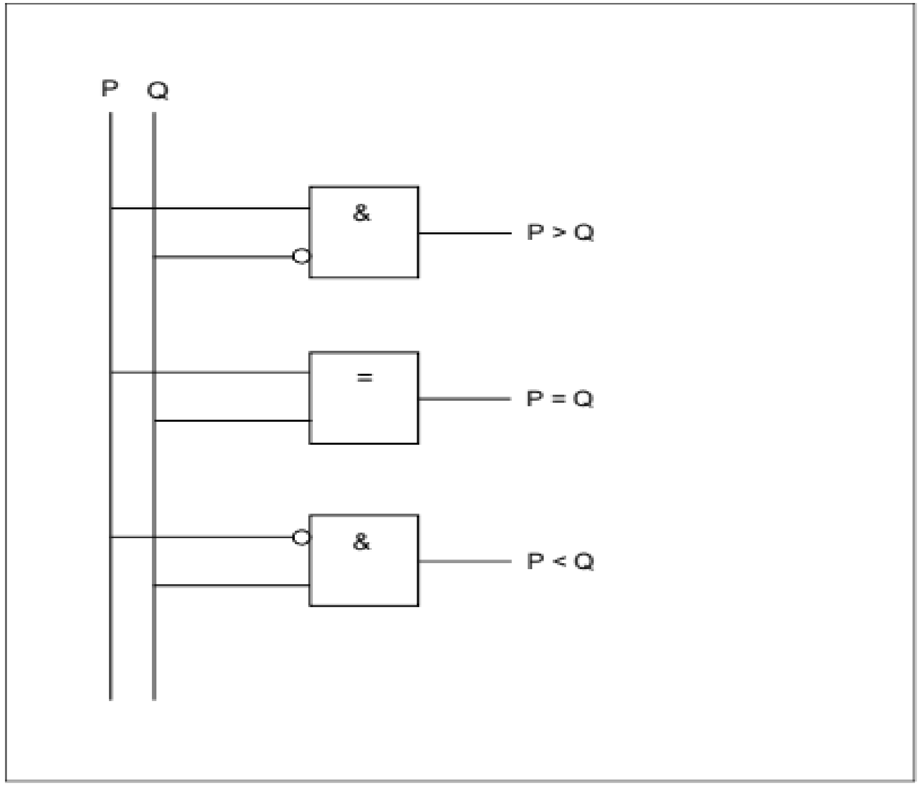


Fig. 1.2.6.2 Circuit

1.2.7 Antivalence

❒ Experiment 1: Fundamental principles

Experiment procedure:

- Complete the value table (table 1.2.7.1) for the circuit shown in fig. 1.2.7.1.

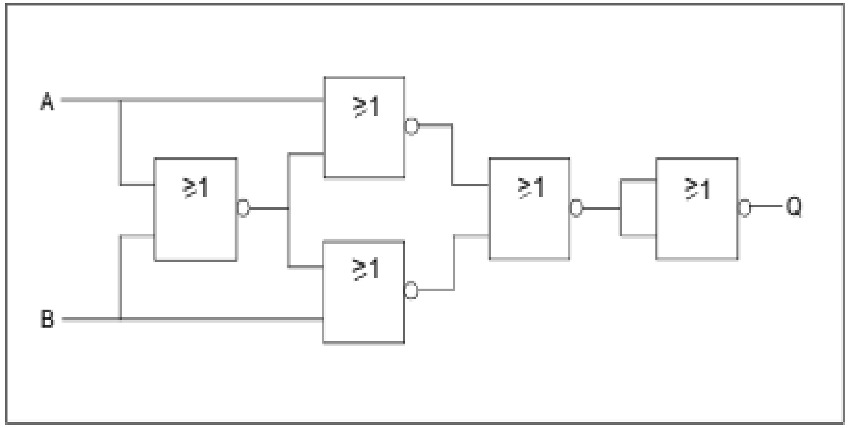


Fig. 1.2.7.1

$$Q = \overline{\overline{\overline{\overline{A \vee B} \vee A} \vee \overline{\overline{A \vee B} \vee B}}}$$

$$= \overline{\overline{A \vee B}} \wedge \overline{A} \vee \overline{\overline{A \vee B}} \wedge \overline{B}$$

$$= (A \vee B)\,\overline{A} \vee (A \vee B)\,\overline{B}$$

$$= A\overline{A} \vee \overline{A}B \vee A\overline{B} \vee B\overline{B}$$

$$Q = \overline{A}B \vee A\overline{B}$$

A	*B*	*Q*
0	*0*	*0*
0	*1*	*1*
1	*0*	*1*
1	*1*	*0*

Table 1.2.7.1 Value table

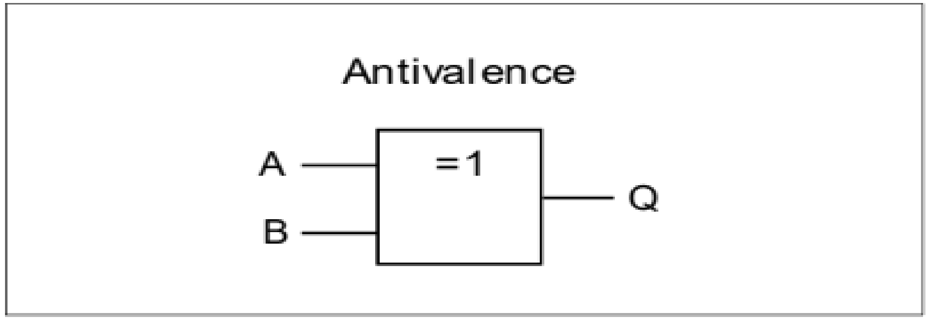


Fig. 1.2.7.2 Circuit symbol