CBCS

EL-1: Basics of Electronics

Digital Electronics

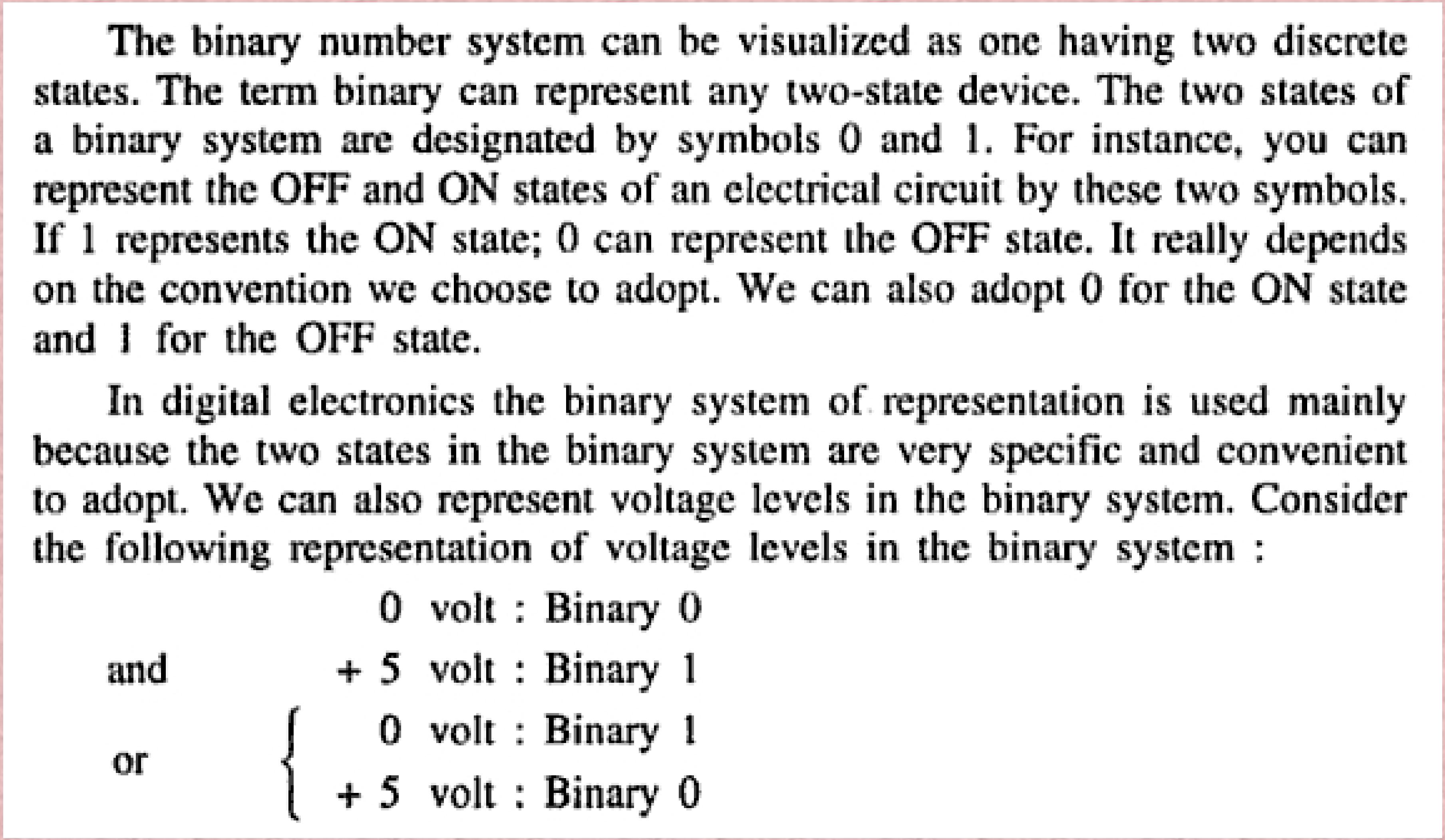
Presented By:

Dr. Anil Kumar Verma

SOS in Electronics & Photonics

Pt. Ravishankar Shukla University, Raipur (CG)

Introduction



The binary number system can be visualized as one having two discrete states. The term binary can represent any two-state device. The two states of a binary system are designated by symbols 0 and 1. For instance, you can represent the OFF and ON states of an electrical circuit by these two symbols. If 1 represents the ON state; 0 can represent the OFF state. It really depends on the convention we choose to adopt. We can also adopt 0 for the ON state and 1 for the OFF state.

In digital electronics the binary system of representation is used mainly because the two states in the binary system are very specific and convenient to adopt. We can also represent voltage levels in the binary system. Consider the following representation of voltage levels in the binary system :

$$\begin{array}{lrl} & 0 \text{ volt} : & \text{Binary } 0 \\ \text{and} & +5 \text{ volt} : & \text{Binary } 1 \end{array}$$

$$\text{or} \quad \left\{ \begin{array}{rl} 0 \text{ volt} : & \text{Binary } 1 \\ +5 \text{ volt} : & \text{Binary } 0 \end{array} \right.$$

Logic Plarity

We have, earlier on, briefly referred to logic polarity. While mostly positive logic is normally employed in designing logic circuits, there are, however, some instances when negative logic is found more convenient. There are also some systems which use both positive and negative logic. By and large most of the systems use positive logic. This implies that + 5 V means logic 1 level and 0 V represents logic 0 level. In actual practice, however, it is not possible to achieve these voltage levels precisely. Logic circuits are therefore so designed that voltages exceeding 2.5 V are taken to be High (logic 1), and voltages less than 0.8 V are considered to be Low (logic 0).

Figure 1.1 shows the voltage level requirements for positive logic and Fig. 1.2 shows the voltage level requirements for negative logic.

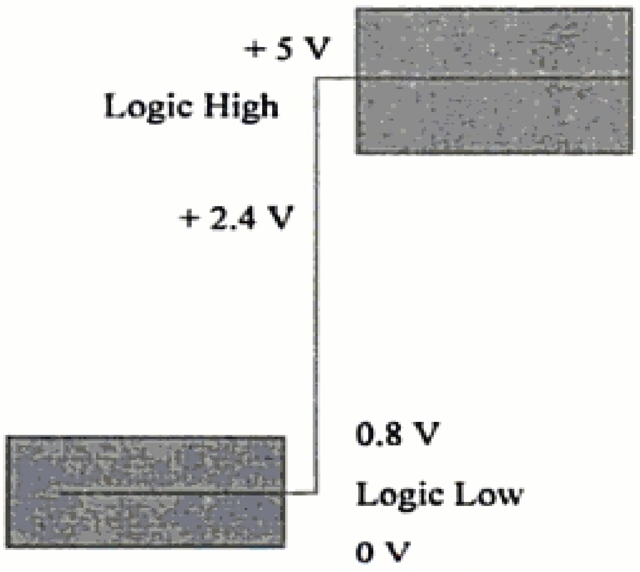


Fig. 1.1. Logic levels for positive logic.

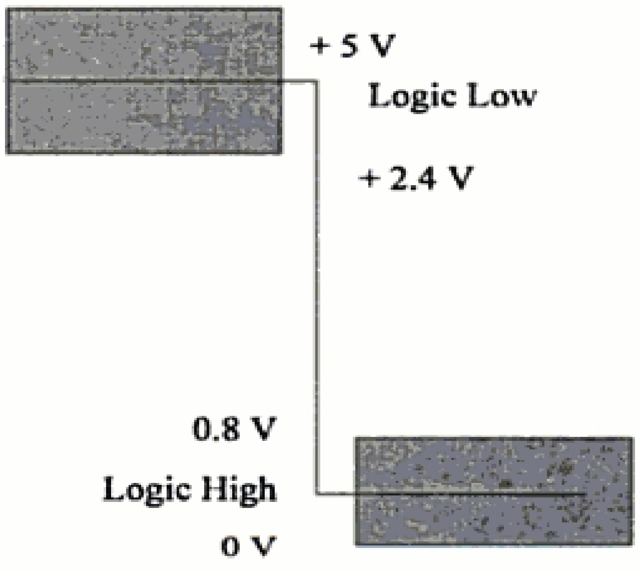


Fig. 1.2. Logic levels for negative logic.

What is Logic Gate?

- ❖Logic gates are electronic digital circuit that can perform logic functions. Commonly expected logic functions are already having the corresponding logic circuits in Integrated Circuit (I.C.) form.

- ❖I.C. may be SSI,LSI,MSI,VLSI etc. depending upon no. of component used.

Types

- ✓ Basic gates:

  AND , OR , NOT

- ✓ Universal gates:

  NAND , NOR

- ✓ Arithmetic/Advanced gates:

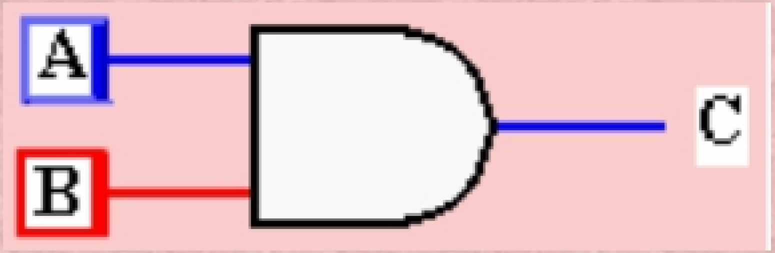
  Exclusive-OR(Ex-OR /XOR)

  Exclusive-NOR(Ex-NOR /XNOR)

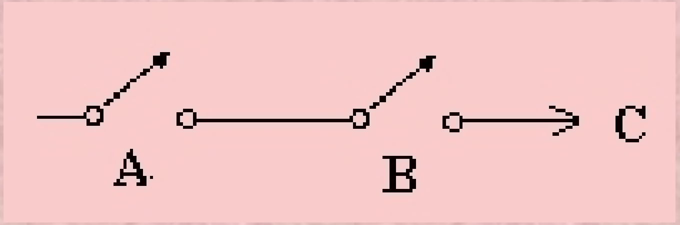
AND Gate

❖ In order for current to flow, both switches must be closed.

- ¤ Logic notation C=A•B or C=AB)
- ¤ **Symbol**

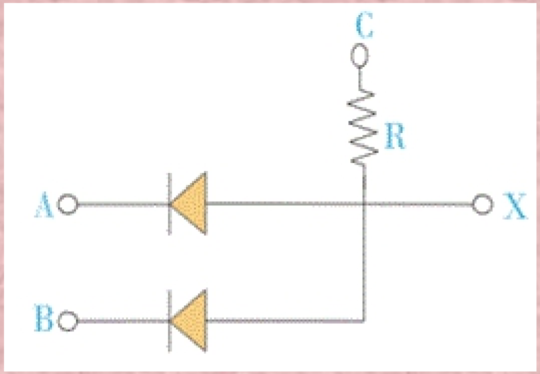


- ¤ **Switching Circuit:**

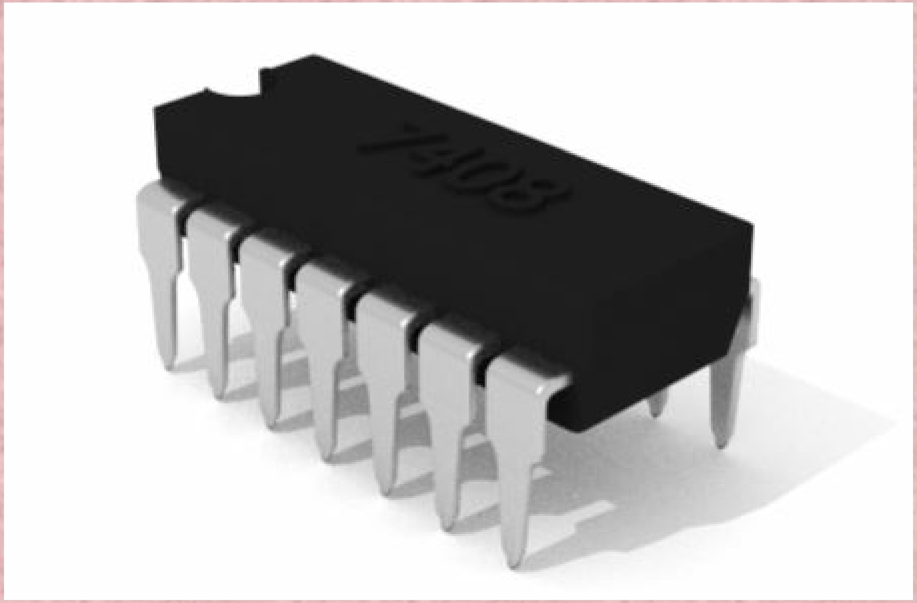


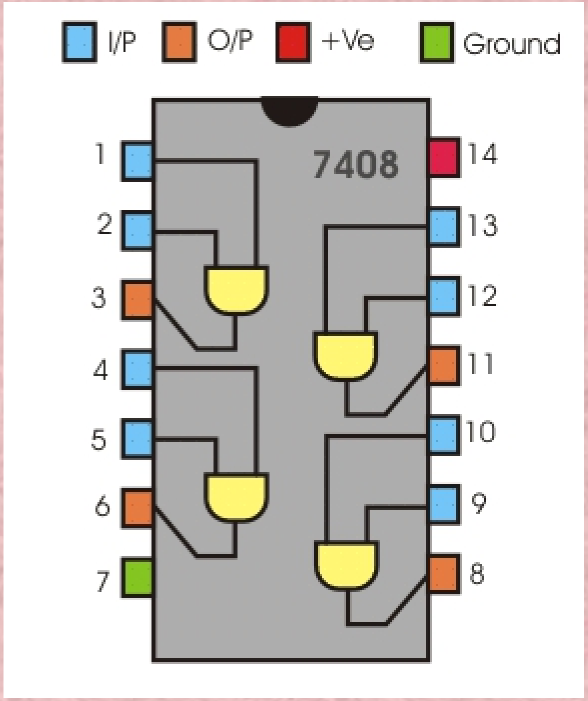
Truth Table(T.T.)

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



IC 7408

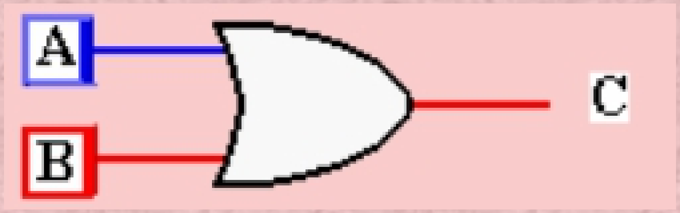




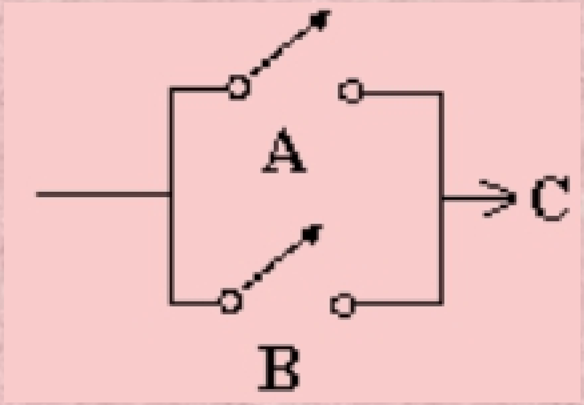
OR Gate

- ❖Current flows if either switch is closed
  - ¤Logic notation C= A + B

Symbol:

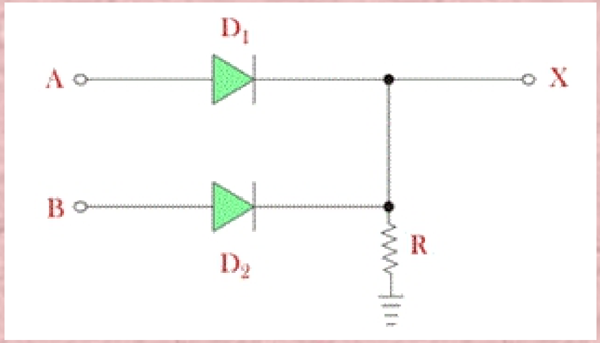


Switching Circuit:

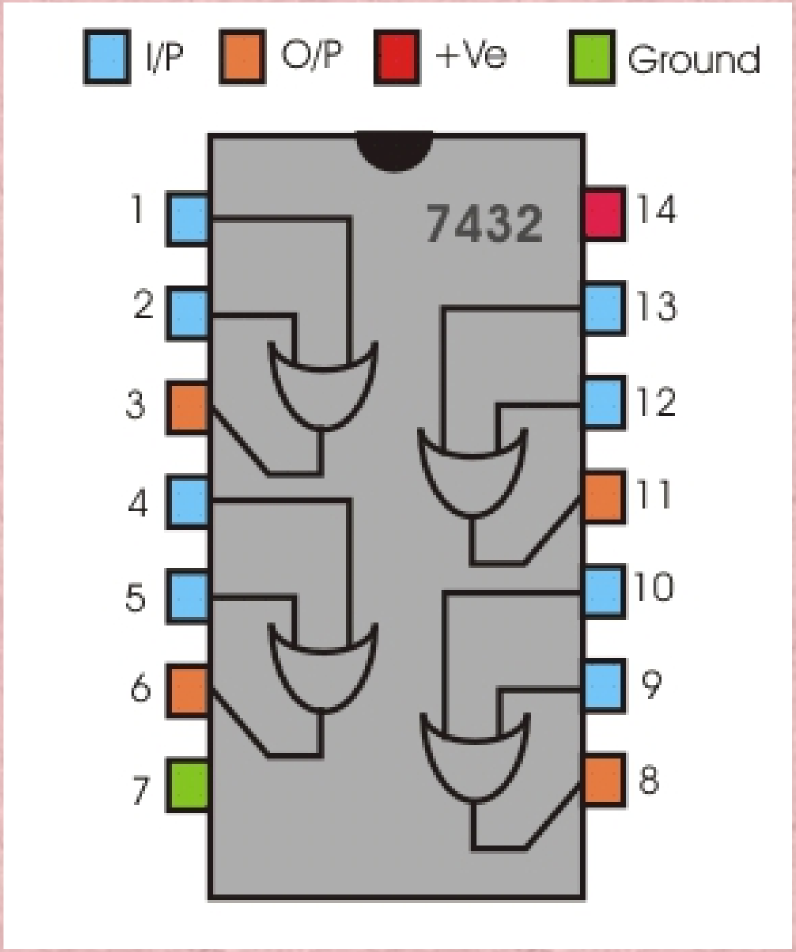


Truth Table(T.T.)

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

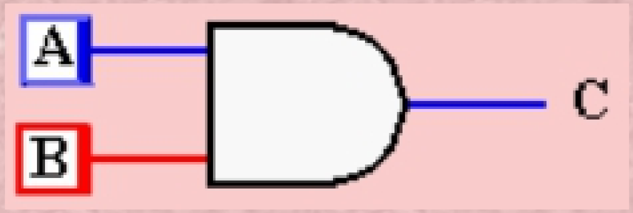


IC 7432

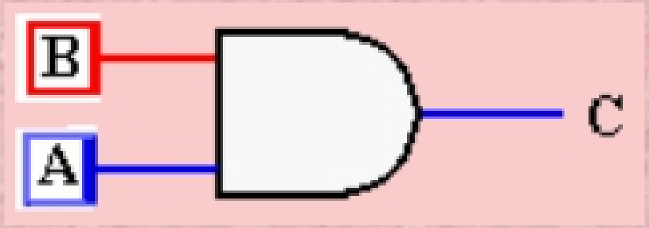


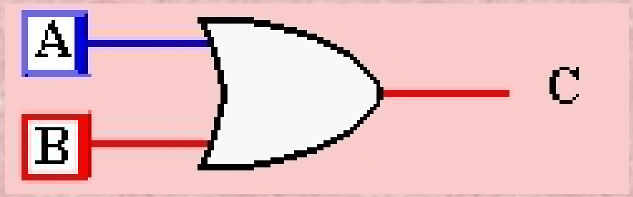
Properties of AND and OR

❖ Commutation
  - ¤ A + B = B + A
  - ¤ A • B = B • A

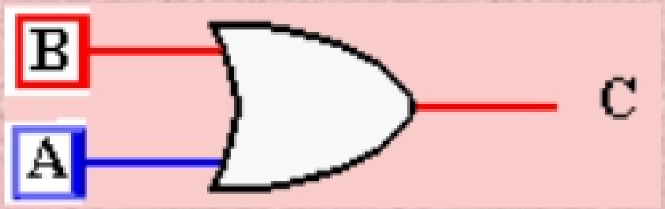


Same as



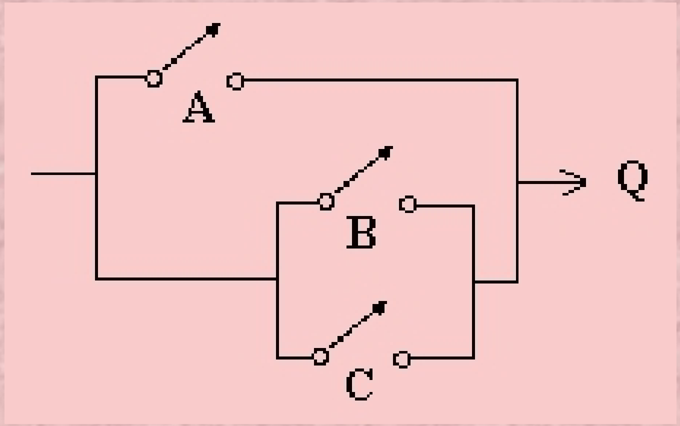


Same as

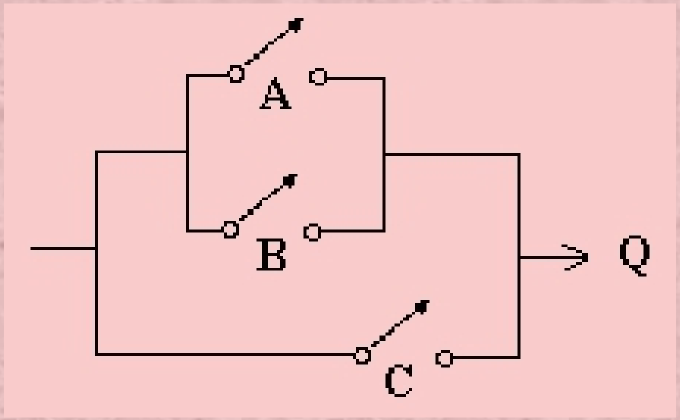


Properties of AND and OR

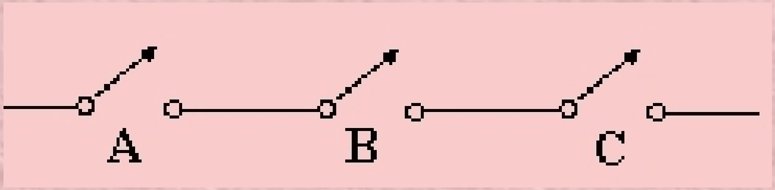
- Associative Property
  - ¤ A + (B + C) = (A + B) + C



=

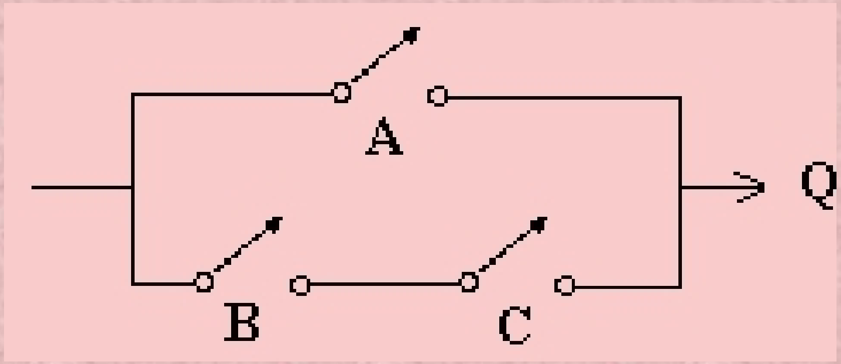


  - ¤ A • (B • C) = (A • B) • C



Properties of AND and OR

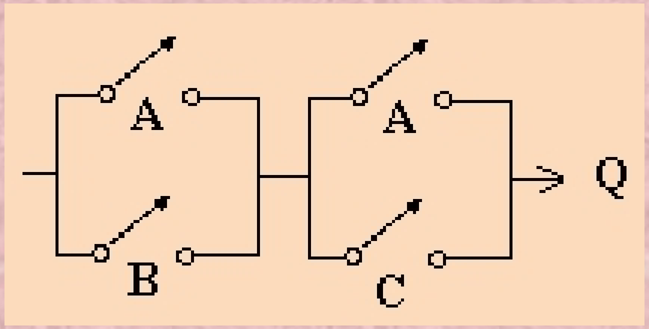
- Distributive Property
  - ¤ A + B • C = (A + B) • (A + C)
  - ¤ A + B • C



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

Distributive Property

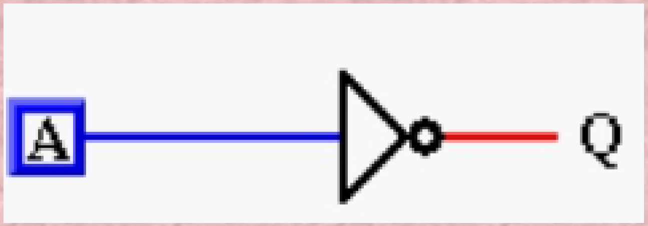
(A + B) • (A + C)



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

Inversion (NOT)

Logic Symbol:

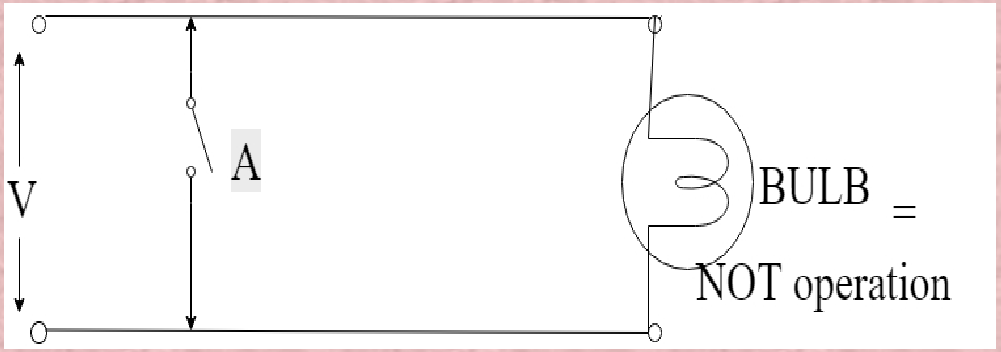


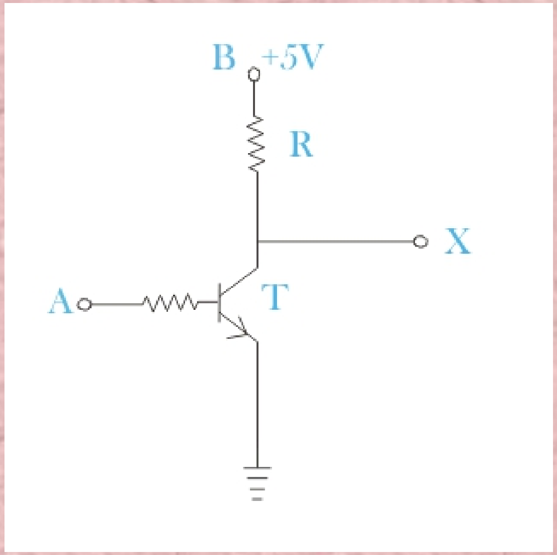
Logic Expression:

$$Q = \overline{A}$$

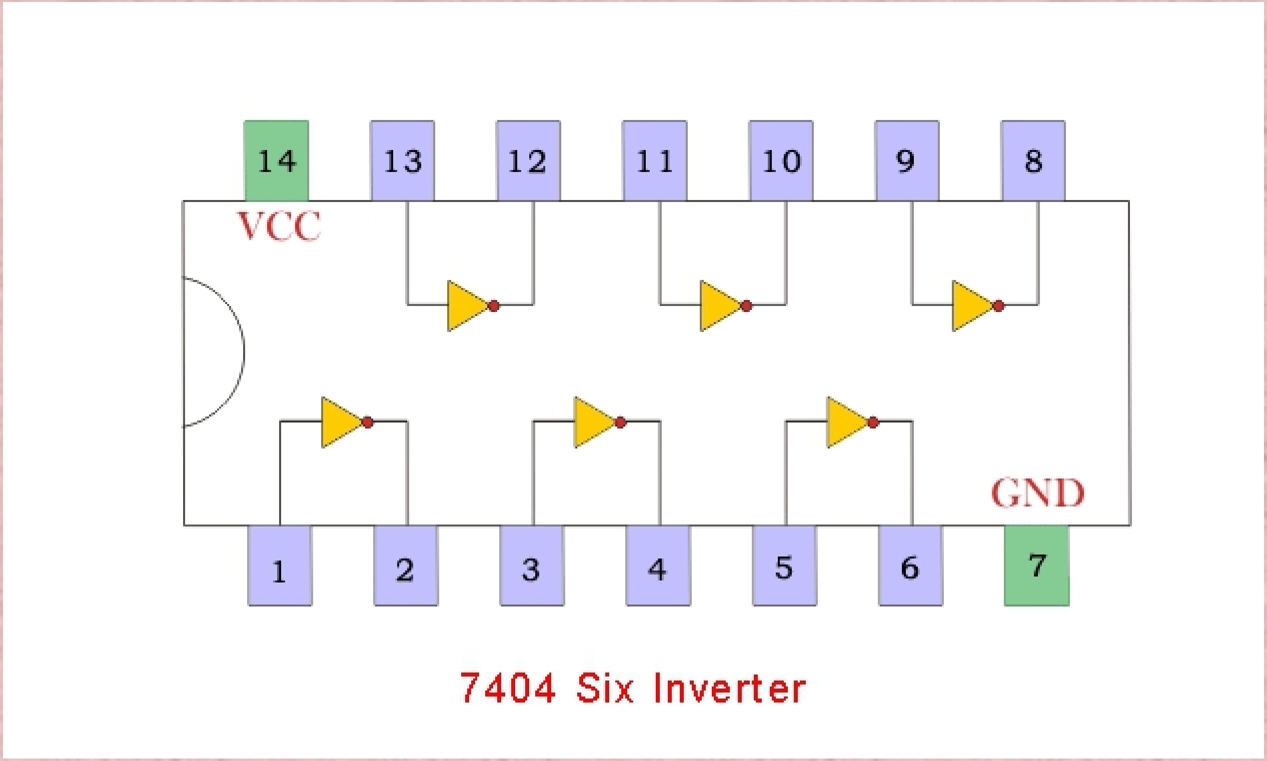
Truth Table:

A	Q
0	1
1	0



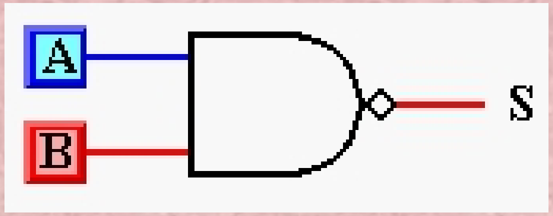


IC 7404



7404 Six Inverter

NAND (NOT+AND)

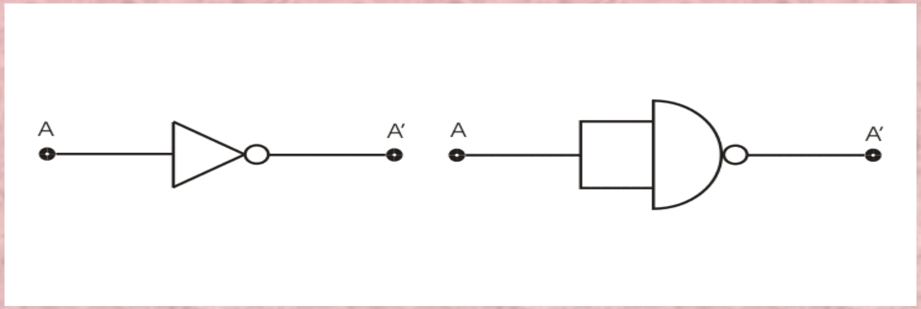


$$Q = \overline{A \cdot B}$$

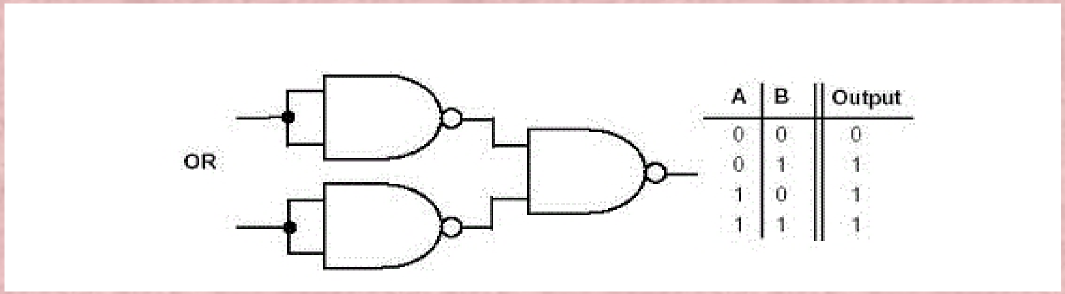
A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0

Universality of NAND gate

As Not gate:

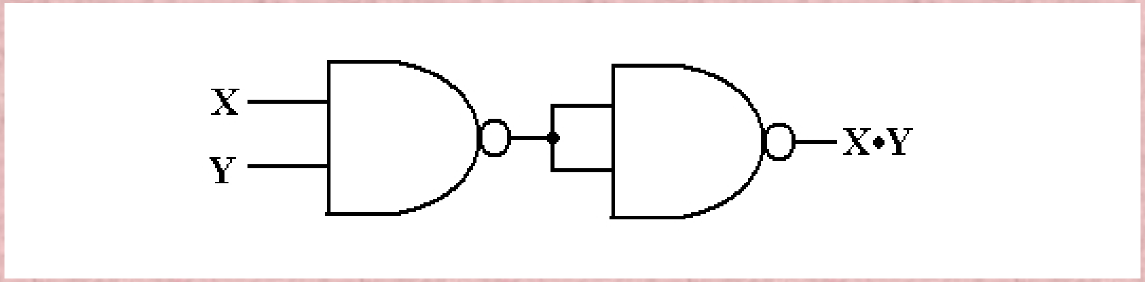


As OR gate:

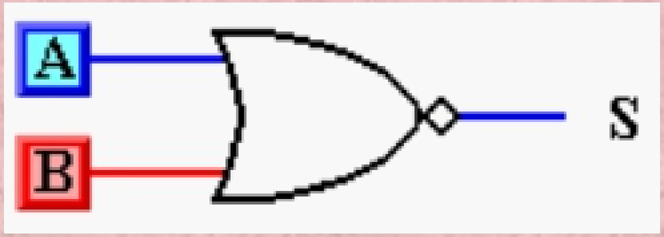


A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

As AND gate:



NOR (NOT+OR)

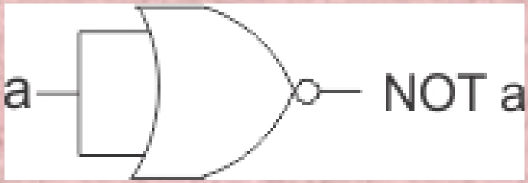


$$Q = \overline{A + B}$$

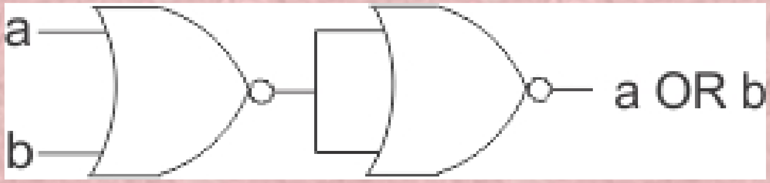
A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

Universality of NOR gate

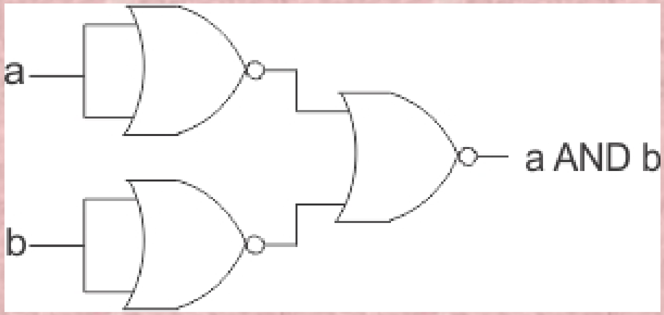
As Not gate:



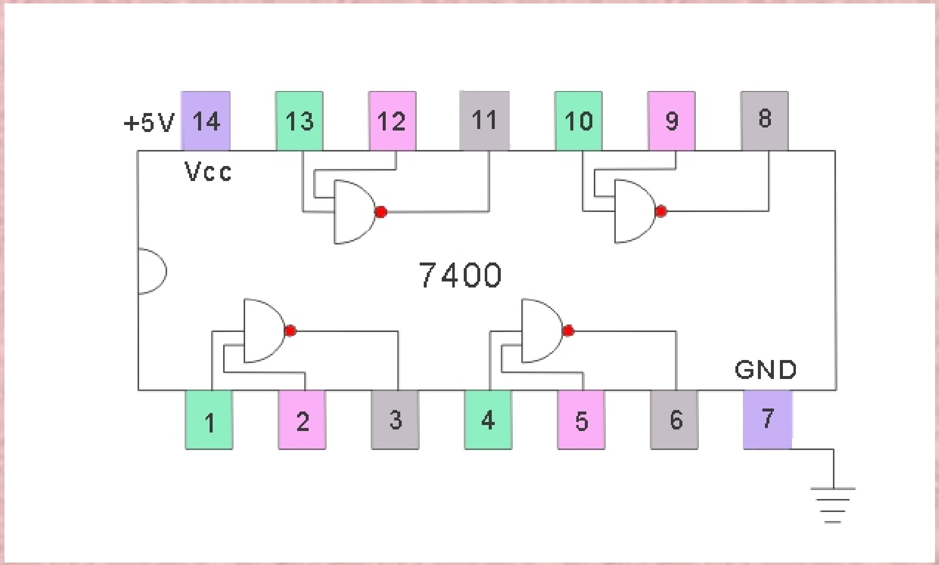
As OR gate:



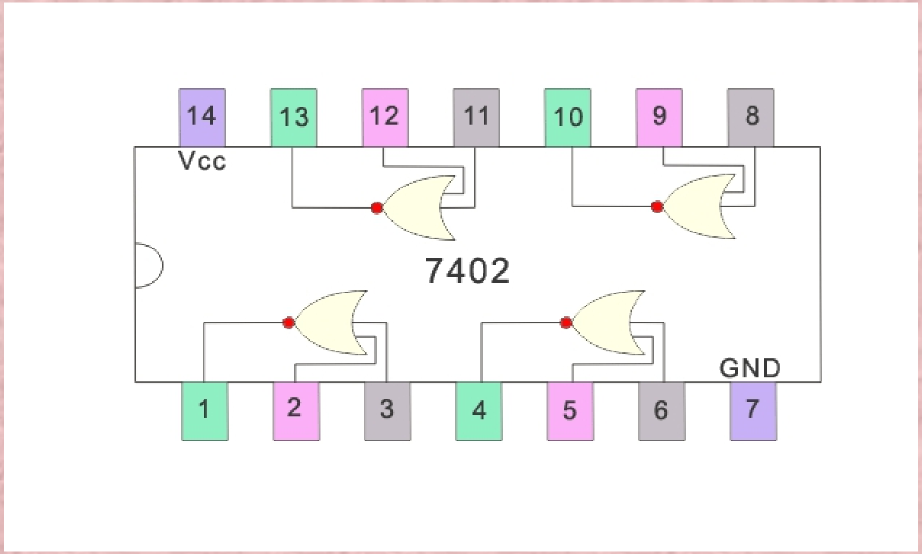
As AND gate:



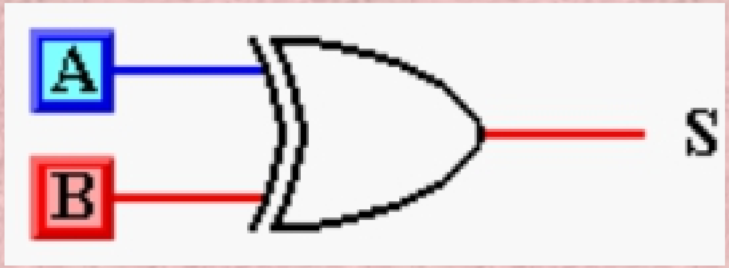
NAND IC:



NOR IC:



Exclusive OR (XOR)



- ❖Either A or B, but not both
- ❖This is sometimes called the **inequality detector**, because the result will be 0 when the inputs are the same and 1 when they are different.
- ❖The truth table is the same as for S on Binary Addition. S = A   B

A	B	S
0	0	0
1	0	1
0	1	1
1	1	0

Getting the XOR

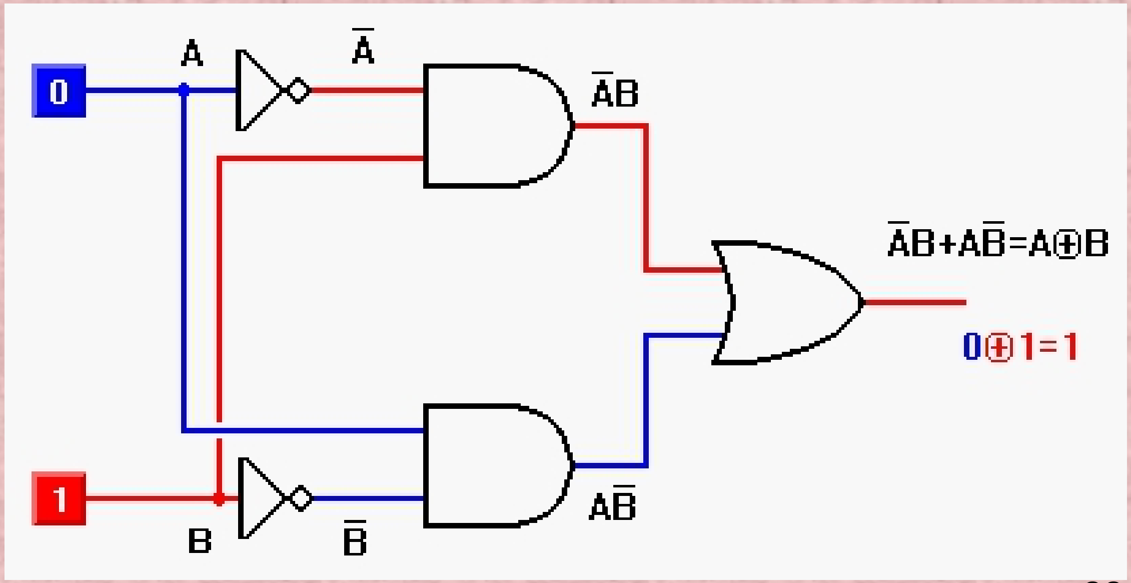
Two ways of getting S = 1

$$A \cdot \overline{B} \text{ or } \overline{A} \cdot B$$

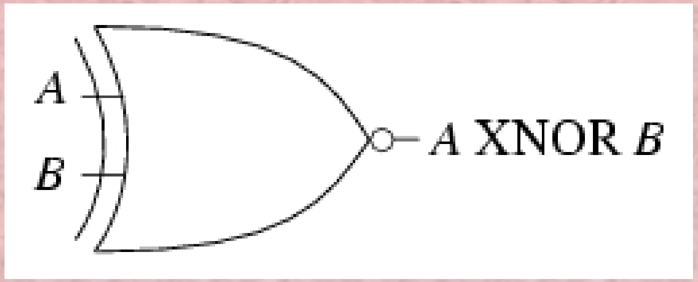
A	B	S
0	0	0
1	0	1
0	1	1
1	1	0

Circuit for XOR

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



Exclusive NOR

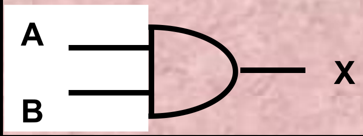


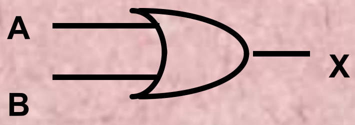
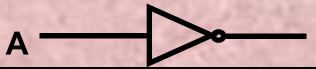
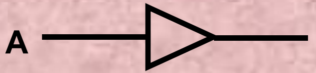
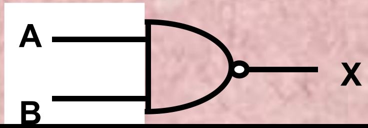
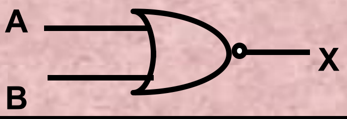
$$Q = \overline{A \oplus B}$$

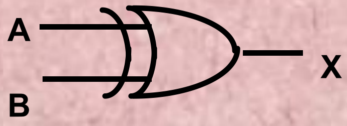
A	B	Q
0	0	1
0	1	0
1	0	0
1	1	1

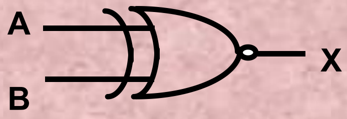
SUMMARY TABLE OF LOGIC GATES

Name	Symbol	Function	Truth Table

AND		X = A • B
		or
		X = AB
			A B X
			0 0 0
			0 1 0
			1 0 0
			1 1 1

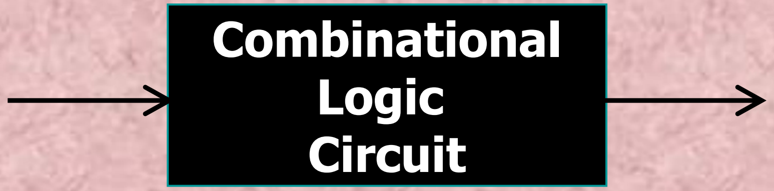
OR		X = A + B	A B X 0 0 0 0 1 1 1 0 1 1 1 1
NOT		X = A’	A X 0 1 1 0
Buffer		X = A	A X 0 0 1 1
NAND		X = (AB)’	A B X 0 0 1 0 1 1 1 0 1 1 1 0
NOR		X = (A + B)’	A B X 0 0 1 0 1 0 1 0 0 1 1 0

XOR
Exclusive OR
		X = A ⊕ B
		or
		X = A’B + AB’
			A B X
			0 0 0
			0 1 1
			1 0 1
			1 1 0

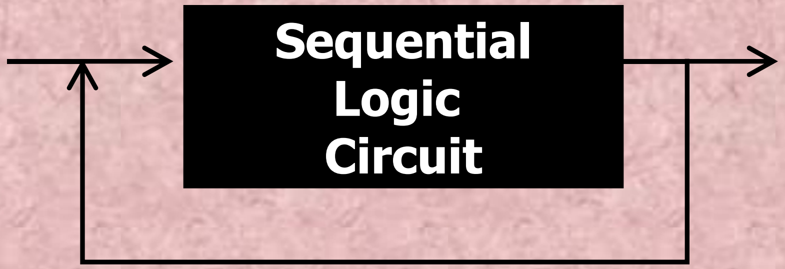
XNOR
Exclusive NOR
or Equivalence
		X = (A ⊕ B)’
		or
		X = A’B’+ AB
			A B X
			0 0 1
			0 1 0
			1 0 0
			1 1 1

Two types of digital logic circuits

- Combinational logic circuits(e.g. HA,FA etc.)
  - Output depends only on its input at that time.



- Sequential logic circuits(e.g. Flip-flop etc)
  - Output depends both on
    - its previous output and
    - its input at that time



BASIC LOGIC BLOCK - GATE

❖**Types of Basic Logic Blocks**

✓ **Combinational Logic Block**

Logic Blocks whose output logic value depends only on the input logic values

e.g. Adder, Subtractors, Multiplexer, Demultiplexer etc.

✓**Sequential Logic Block**

Logic Blocks whose output logic value depends on the input values and the state (stored information) of the blocks

e.g. Flip Flop, Counter, Registers etc.

❖**Functions of Gates can be described by**

- Truth Table
- Boolean Function
- Karnaugh Map

Binary Addition: Half Adder(HA)

A	B	S	C(carry)
0	0	0	0
1	0	1	0
0	1	1	0
1	1	0	1

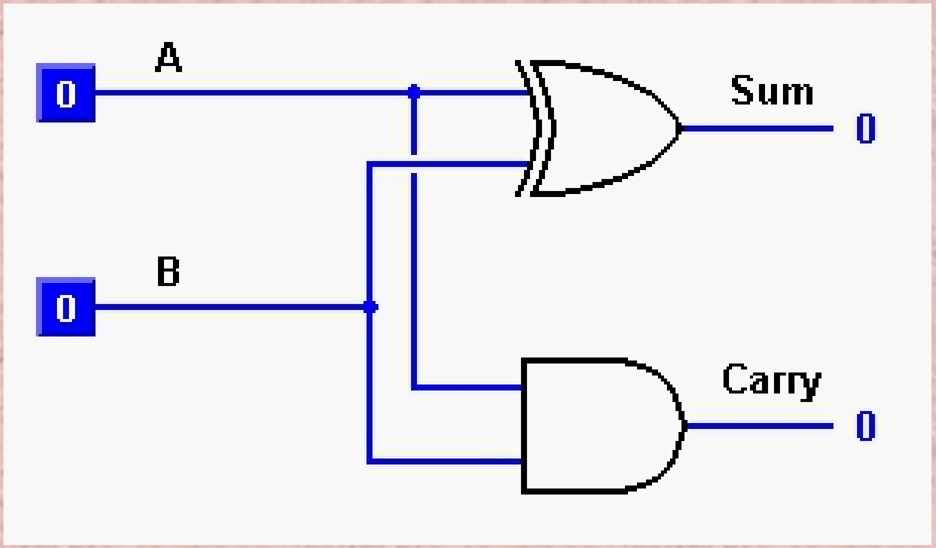
Notice that the carry results are the same as AND

$$C = A \bullet B$$

And Sum results are the same as XOR

S= A xor B

Circuit for a Half Adder



Called a half adder because we haven't allowed for any carry bit on input. In elementary addition of numbers, we always need to allow for a carry from one column to the next.

18

25

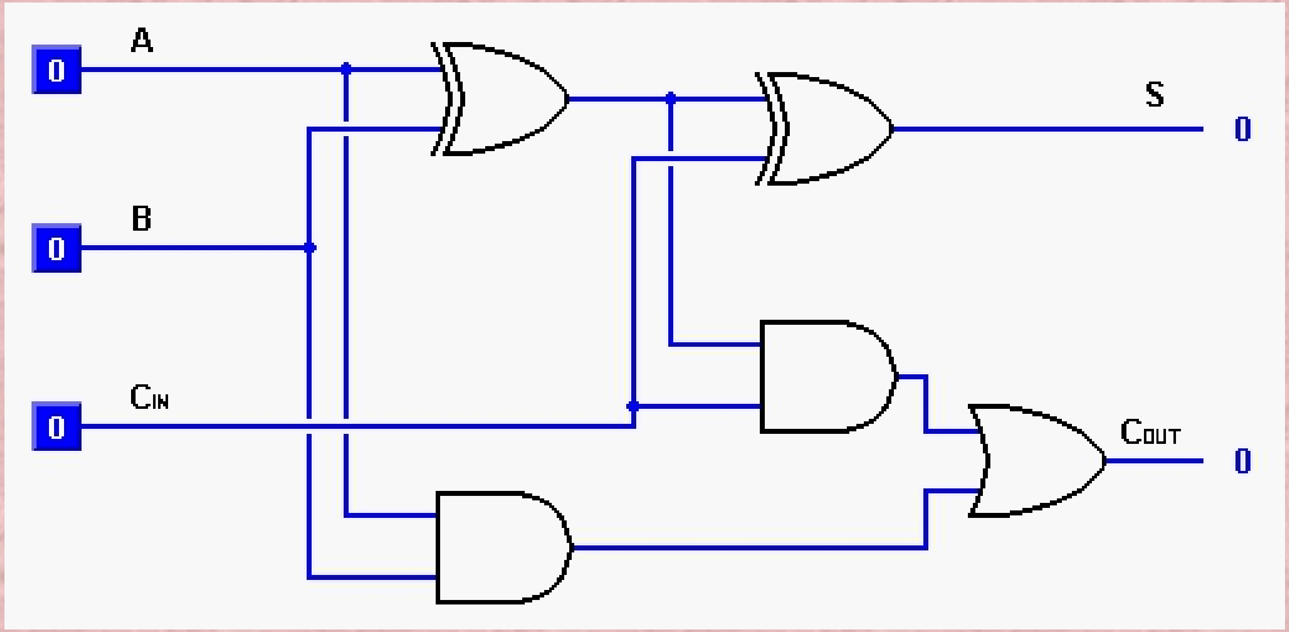
3 (plus a carry)

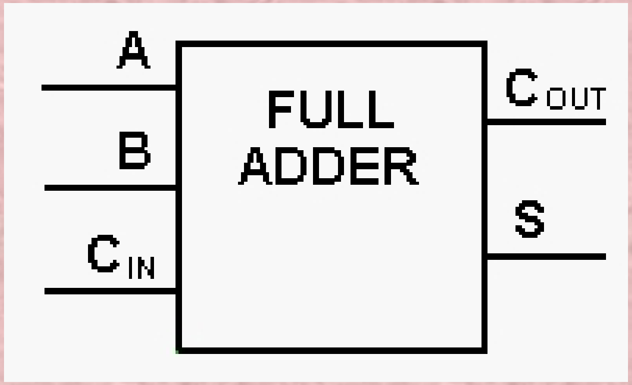
4

Full Adder

INPUTS			OUTPUTS	
A	B	C_{IN}	C_{OUT}	S
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

Full Adder Circuit





Thanks!