Digital Arithmetic & Logic gates

Digital Arithmetic

Binary Addition (+)

With this background, we can write the basic rules of binary addition as follows:

1. 0 + 0 = 0.
2. 0 + 1 = 1.
3. 1 + 0 = 1.
4. 1 + 1 = 0 with a carry of '1' to the next more significant bit.
5. 1 + 1 + 1 = 1 with a carry of '1' to the next more significant bit.

Example 1:

$$\begin{array}{r} 011\ (3) \\ +\ \ 110\ (6) \\ \hline 1001\ (9) \end{array} \qquad \begin{array}{r} 1001\ (9) \\ +\ \ 1111\ (15) \\ \hline 11000\ (24) \end{array} \qquad \begin{array}{r} 11.011\ (3.375) \\ +\ \ 10.110\ (2.750) \\ \hline 110.001\ (6.125) \end{array}$$

Binary Subtraction (-)

The basic principles of binary subtraction include the following:

1. 0 − 0 = 0.
2. 1 − 0 = 1.
3. 1 − 1 = 0.
4. 0 − 1 = 1 with a borrow of 1 from the next more significant bit.

Example 2:

$$\begin{array}{r} 110\ (6) \\ -\ \ 010\ (2) \\ \hline 100\ (4) \end{array} \qquad \begin{array}{r} 11011\ (27) \\ -\ \ 01101\ (13) \\ \hline 1110\ (14) \end{array} \qquad \begin{array}{r} 1000.10\ (8.50) \\ -\ \ 0011.01\ (3.25) \\ \hline 101.01\ (5.25) \end{array}$$

H.W

1. Add the following pairs of binary numbers:

(a) 10110 + 00111 (b) 011.101 + 010.010 (c) 10001111 + 00000001

2. Subtract the following pairs of binary numbers:

(a) 101101 – 010010 (b) 10001011 – 00110101 (c) 10101.1101 - 01110.0110

Logic gates

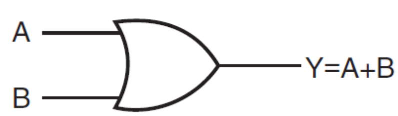
Logic gates are "elementary bricks" used in the construction of digital circuits. In logic, only two possible conditions exist for any input or output: true and false. The binary number system uses only two digits, 1 and 0, so it is perfect for representing logical relationships.

OR Gate

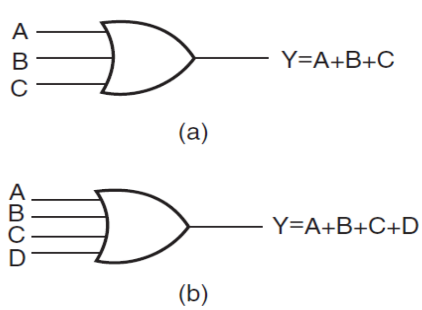
An OR gate performs an OR operation on two or more than two logic variables. The OR operation on two independent logic variables A and B is written as Y = A+B and reads as Y equals A OR B and not as A plus B. An OR gate is a logic circuit with two or more inputs and one output. The output of an OR gate is LOW only when all of its inputs are LOW. For all other possible input combinations, the output is HIGH.

The operation of a two-input OR gate is explained by the logic expression

$$Y = A+B$$



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



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

(c)

Figure 1: (a) Three-input OR gate, (b) four-input OR gate and (c) the truth table of a three-input OR gate.

Example: How would you hardware-implement a four-input OR gate using two-input OR gates only?

Figure 2 (a) shows one possible arrangement of two-input OR gates that simulates a four-input OR gate. A, B, C and D are logic inputs and Y3 is the output. **Figure 2 (b)** shows another possible arrangement.

Y1 = (A+B).

Y2 = (Y1+C) = (A+B+C).

Y3 = (Y2+D) = (A+B+C+D).

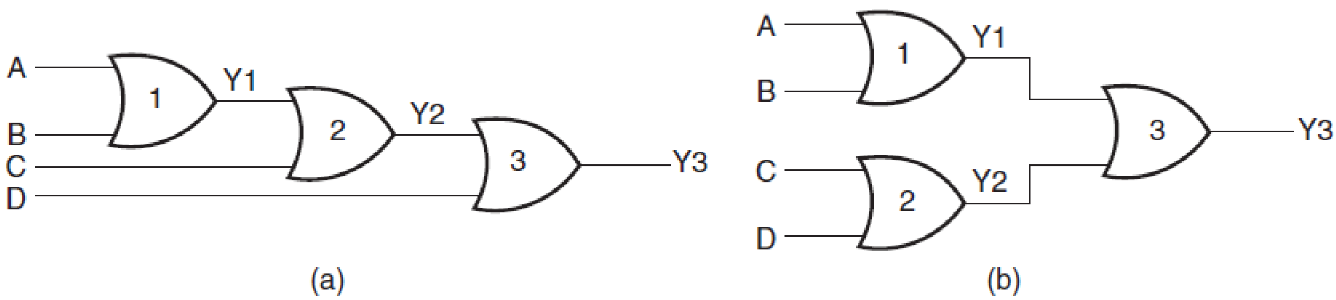
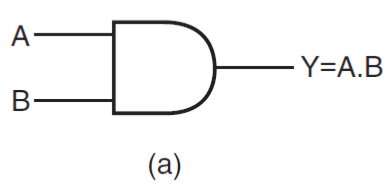


Figure 2: shows one possible arrangement of two-input OR gates that simulates a four-input OR gate.

AND gate

An **AND** gate is a logic circuit having two or more inputs and one output. The output of an AND gate is **HIGH** only when all of its inputs are in the **HIGH** state. In all other cases, the output is **LOW**.

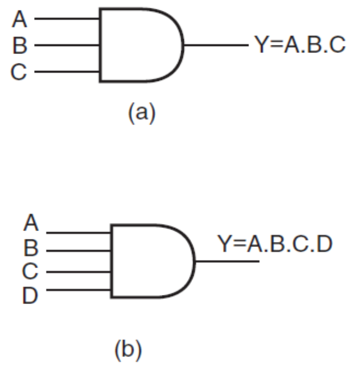
Y = A.B



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

(b)

Figure 3: Two-input AND gate.



A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

(c)

Figure 4: (a) Three-input AND gate, (b) four-input AND gate and (c) the truth table of a four-input AND gate.

Example

Determine the output x from the AND gate in **Figure 5** below for the given input waveforms.

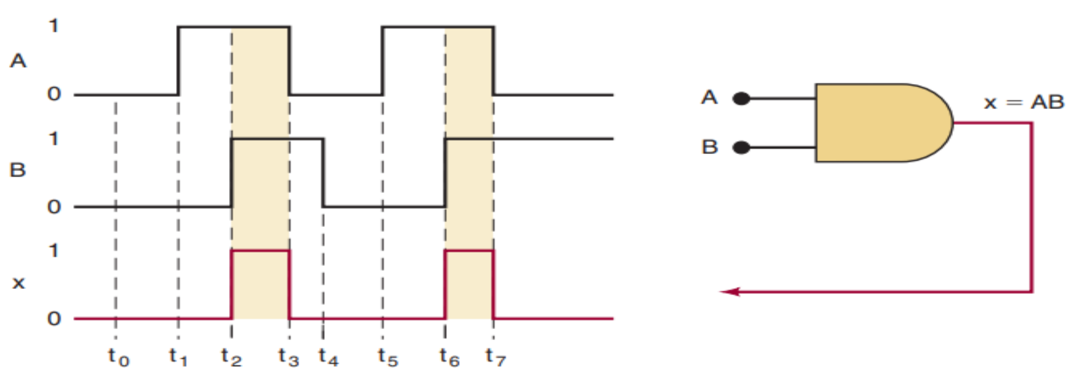
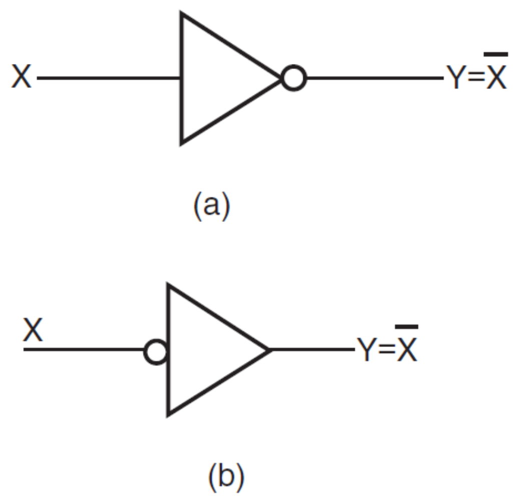


Figure 5: AND gate.

NOT Gate

A NOT gate is a one-input, one-output logic circuit whose output is always the complement of the input. That is, a LOW input produces a HIGH output, and vice versa.

$$A' = \overline{A}$$



X	Y
0	1
1	0

Figure 6: (a) Circuit symbol of a NOT circuit and (b) the truth table of a NOT circuit.

- **Describing logic circuits algebraically**

Any logic circuit, no matter how complex, can be described completely using the three basic Boolean operations because the OR gate, AND gate, and NOT circuit are the basic building blocks of digital systems. For example, consider the circuit in **Figure 7** below(a).

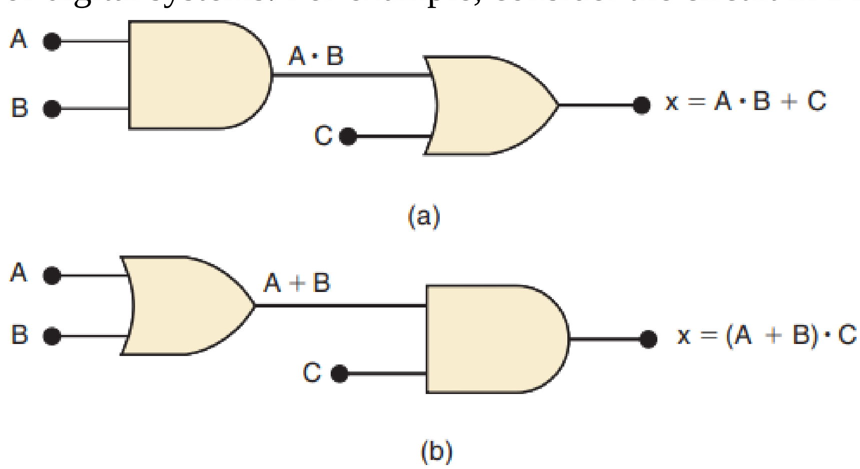


Figure 7: (a) Logic circuit with its Boolean expression; (b) logic circuit whose expression requires parentheses.

- Circuits Containing INVERTER

Whenever an **INVERTER** is present in a logic-circuit diagram, its output expression is simply equal to the input expression with a bar over it. **Figure 8** below shows two examples using **INVERTERs**.

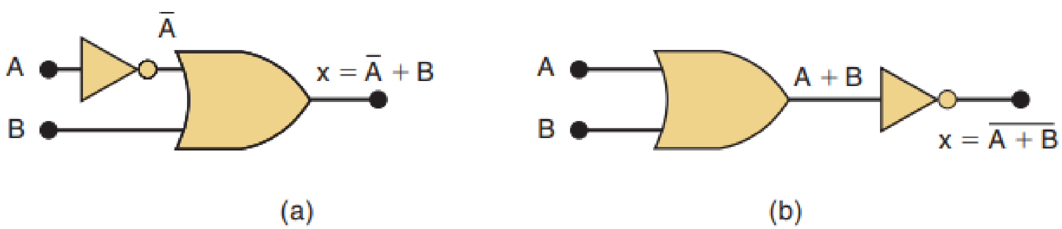


Figure 8: Circuits using INVERTERs.

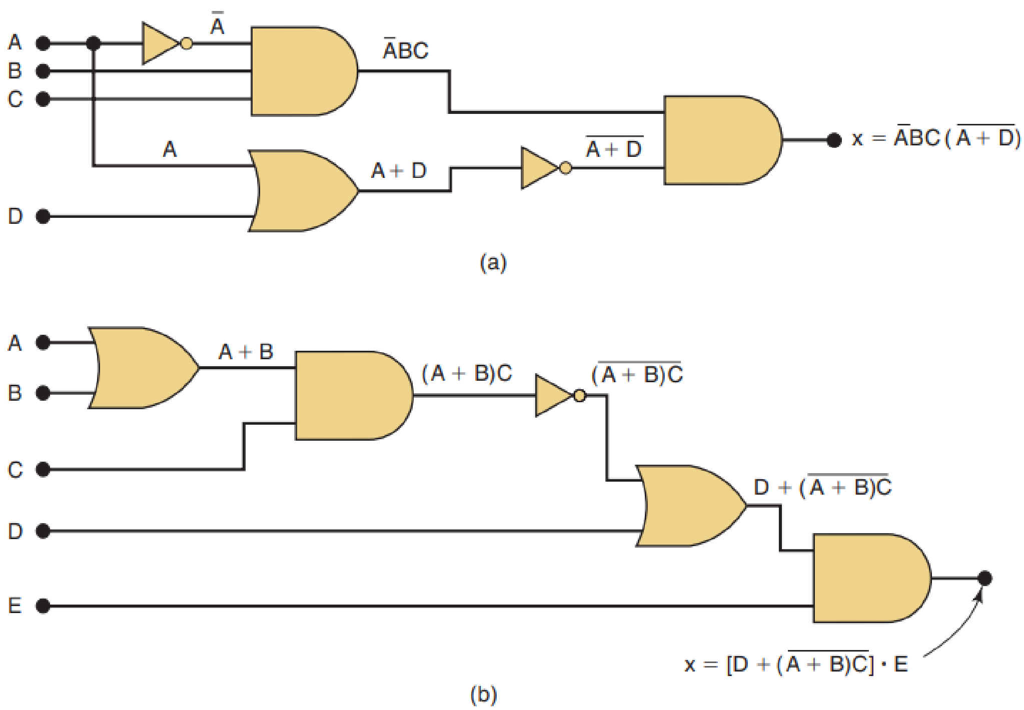


Figure 9: Also notice in Figure above (a) that the input variable A is connected as an input to two different gates.

Summary of boolean Operations

The rules for the OR, AND, and NOT operations may be summarized as follows:

OR	*AND*	*NOT*
$0 + 0 = 0$	$0 \cdot 0 = 0$	$\overline{0} = 1$
$0 + 1 = 1$	$0 \cdot 1 = 0$	$\overline{1} = 0$
$1 + 0 = 1$	$1 \cdot 0 = 0$	
$1 + 1 = 1$	$1 \cdot 1 = 1$	