GRAY CODE:

The bits in a binary count sequence, it is clear that there are often several bits that must change states at the same time. Consider when 3 - bit binary number for 3 change to 4 : all three bits must change state.

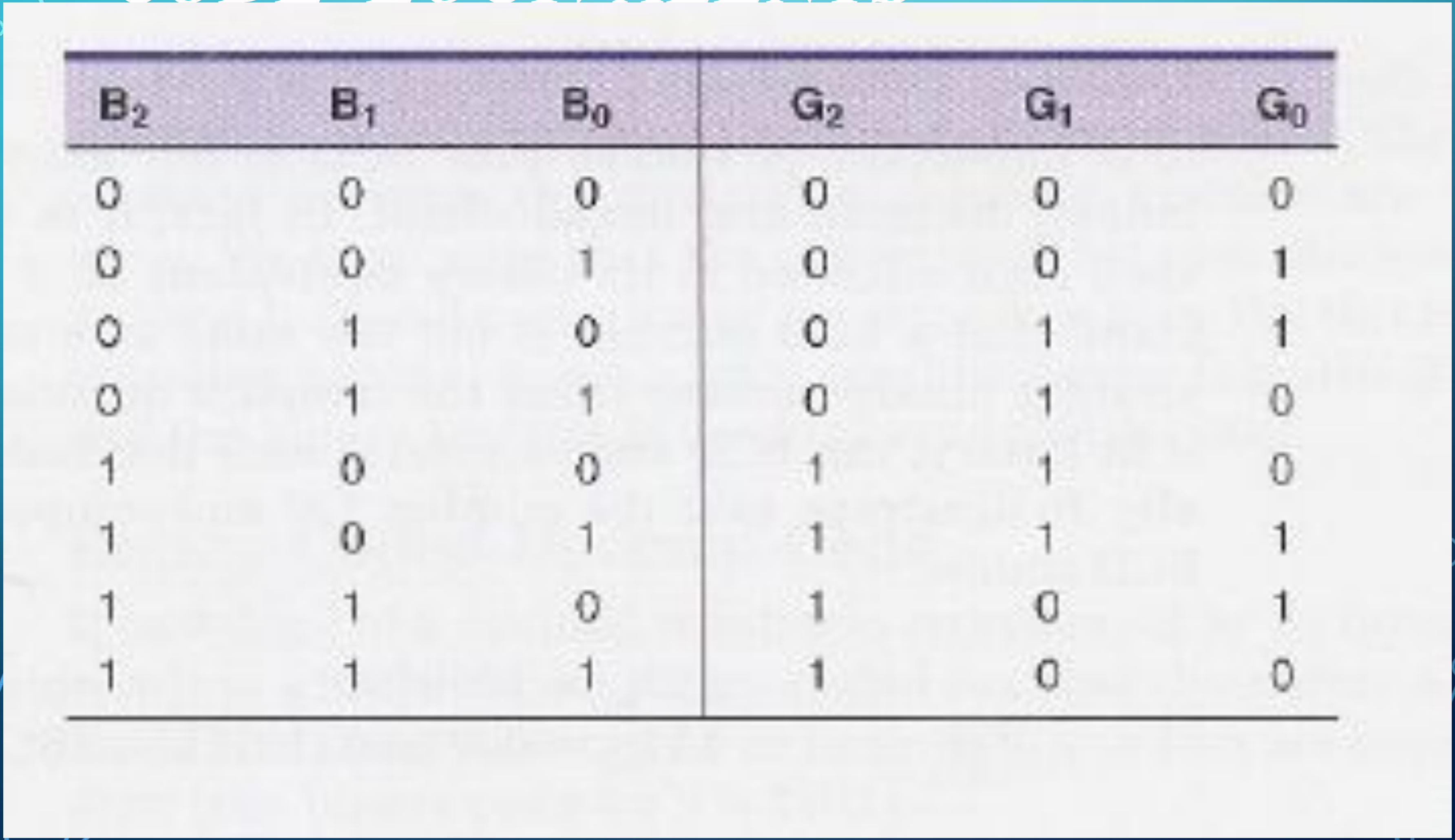
011	*(decimal 3)*
100	*(decimal 4)*

In order to reduce the likelihood of a digital circuit misinterpreting a changing input , the Gray code has been developed as way to represent a sequence of numbers.

The unique aspect of the Gray code is that only one bit ever changes between two successive numbers in sequence.

The Gray code is also known as The Reflected Code

THREE BIT BINARY AND GRAY CODE EQUIVALENTS



B_2	B_1	B_0	G_2	G_1	G_0
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

AN APPLICATION
THE CONCEPT OF A 3-BIT SHAFT

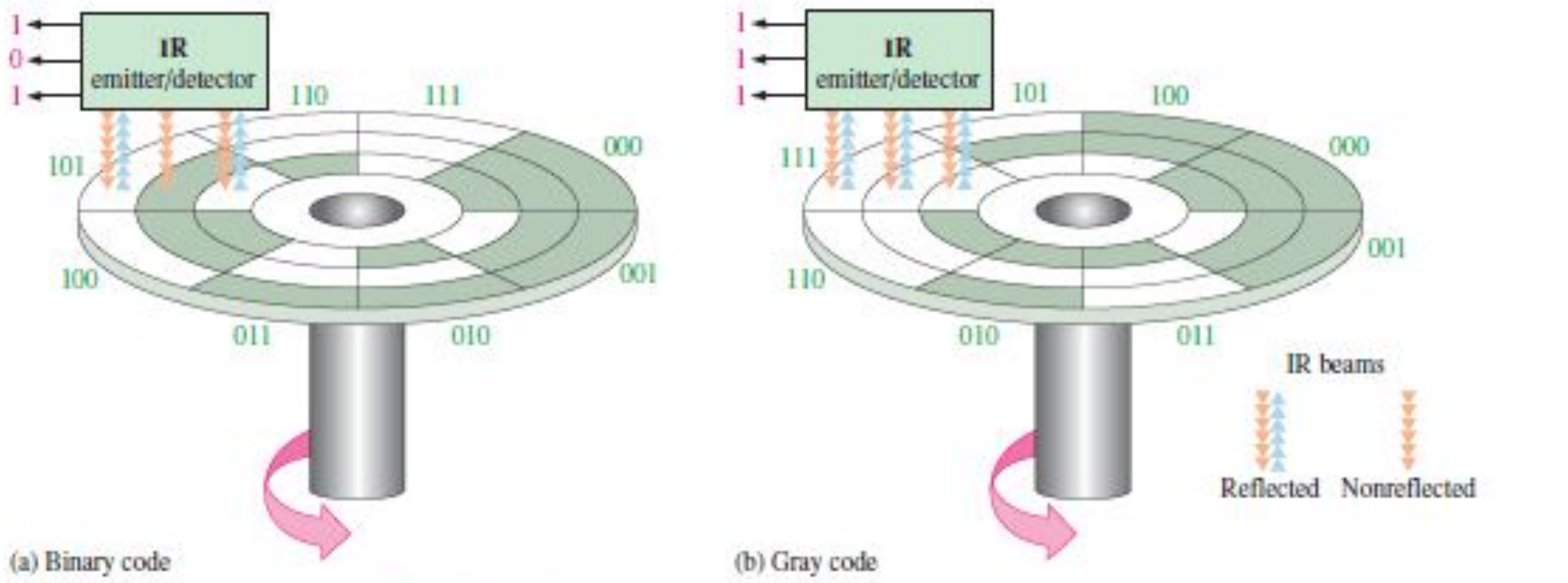
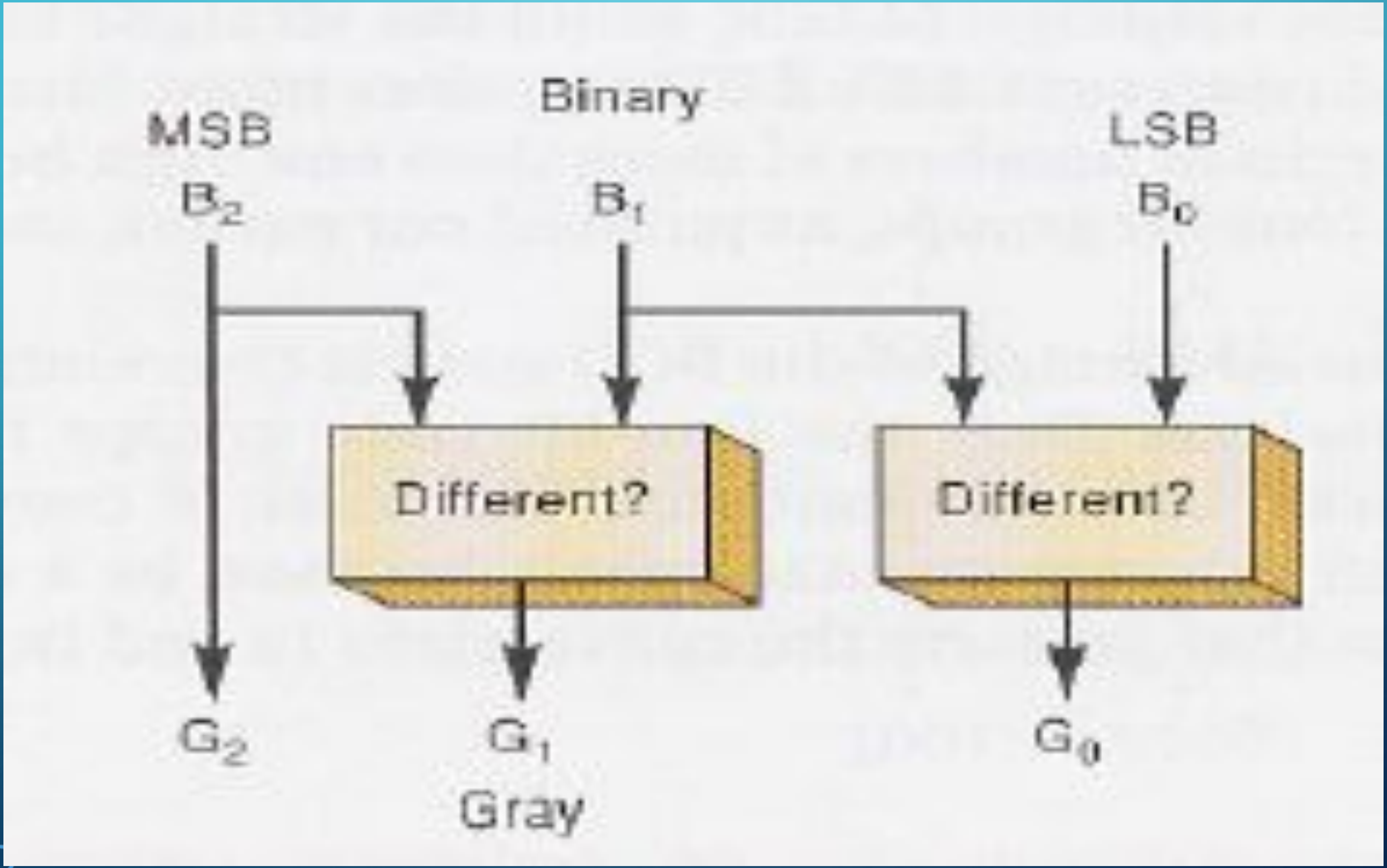


FIGURE 2–7 A simplified illustration of how the Gray code solves the error problem in shaft position encoders. Three bits are shown to illustrate the concept, although most shaft encoders use more than 10 bits to achieve a higher resolution.

BINARY TO GRAY CODE CONVERSION

- *MSB Binary use as Gray MSB*
- *Compare MSB Binary with next binary bit (B1)*
- *If they are same   G1 = 0*
- *If they are different G1 = 1*
- *G0 can be found by comparing B1 and B0*

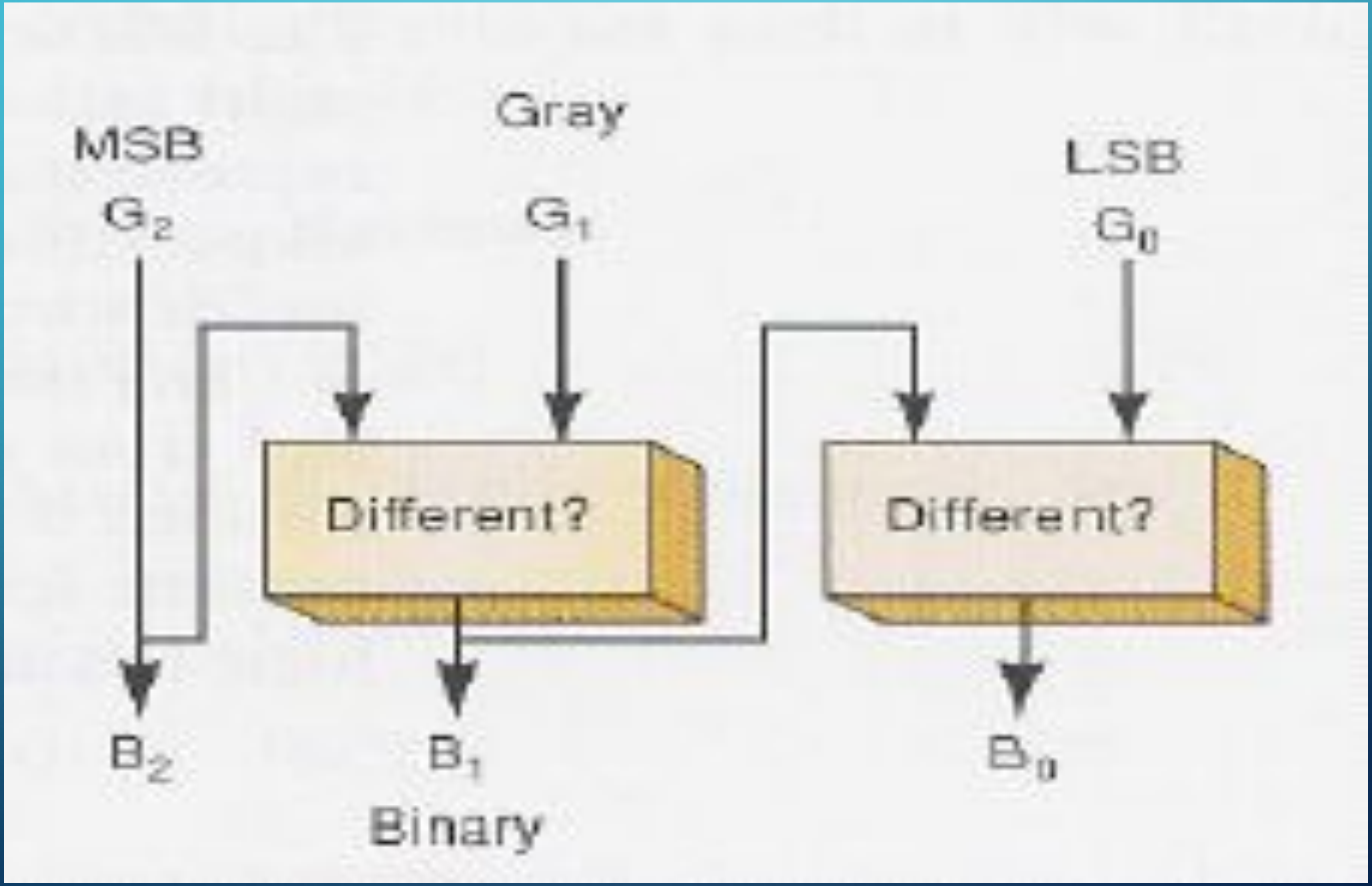
CONVERTING BINARY TO GRAY CODE



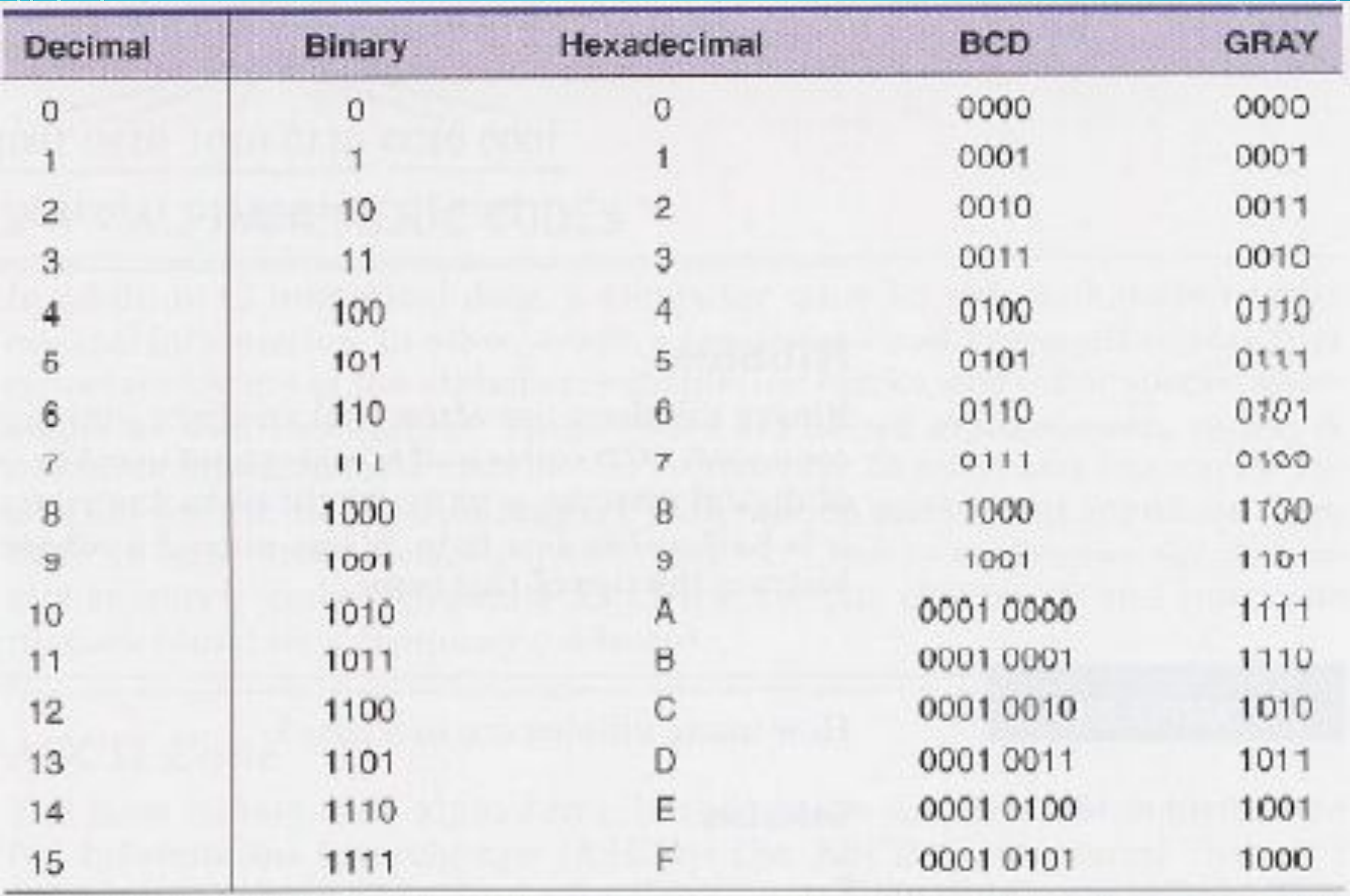
GRAY CODE TO BINARY CONVERSION

- *Gray MSB used as MSB binary*
- *Compare Gray MSB with binary bit (G1)*
- *If they are same   B1 = 0*
- *If they are different B1 = 1*
- *B0 can be found by comparing B1 and G0*

CONVERTING GRAY TO BINARY CODE



PUTTING ALL TOGETHER



Decimal	Binary	Hexadecimal	BCD	GRAY
0	0	0	0000	0000
1	1	1	0001	0001
2	10	2	0010	0011
3	11	3	0011	0010
4	100	4	0100	0110
5	101	5	0101	0111
6	110	6	0110	0101
7	111	7	0111	0100
8	1000	8	1000	1100
9	1001	9	1001	1101
10	1010	A	0001 0000	1111
11	1011	B	0001 0001	1110
12	1100	C	0001 0010	1010
13	1101	D	0001 0011	1011
14	1110	E	0001 0100	1001
15	1111	F	0001 0101	1000

CHAPTER OUTLINE

- The Inverter
- The AND Gate
- The OR Gate
- The NAND Gate
- The NOR Gate
- The Exclusive-OR and Exclusive-NOR Gates

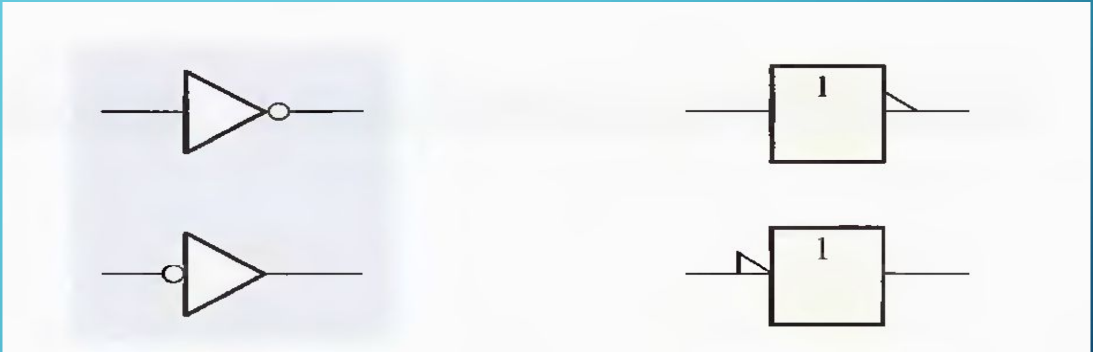
NOT OPERATION

Boolean Expression :

$x = \overline{A}$

$0 = \overline{1}$ because 0 is not 1

$1 = \overline{0}$ because 1 is not 0

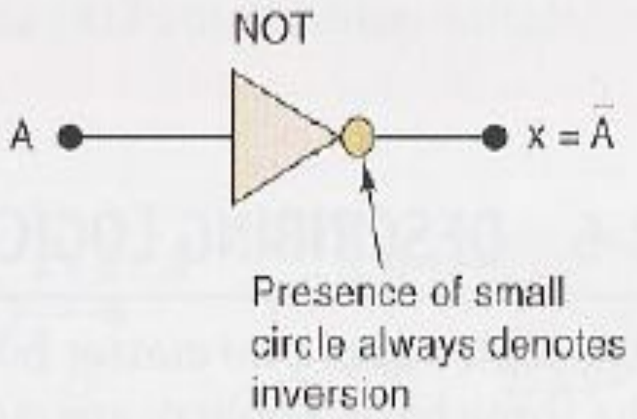


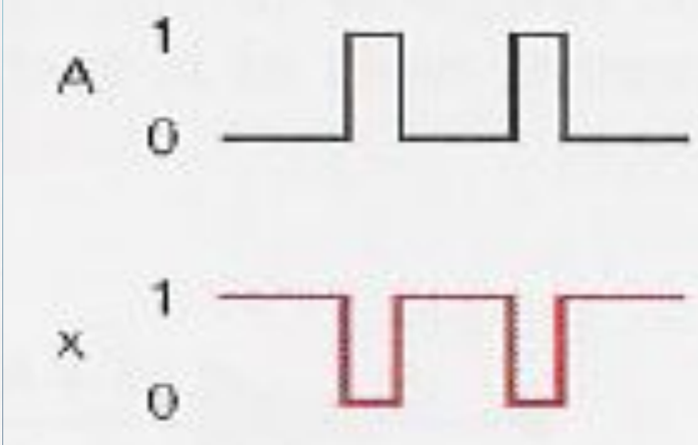
(a) Distinctive shape symbols with negation indicators

(b) Rectangular outline symbols with polarity indicators

NOT

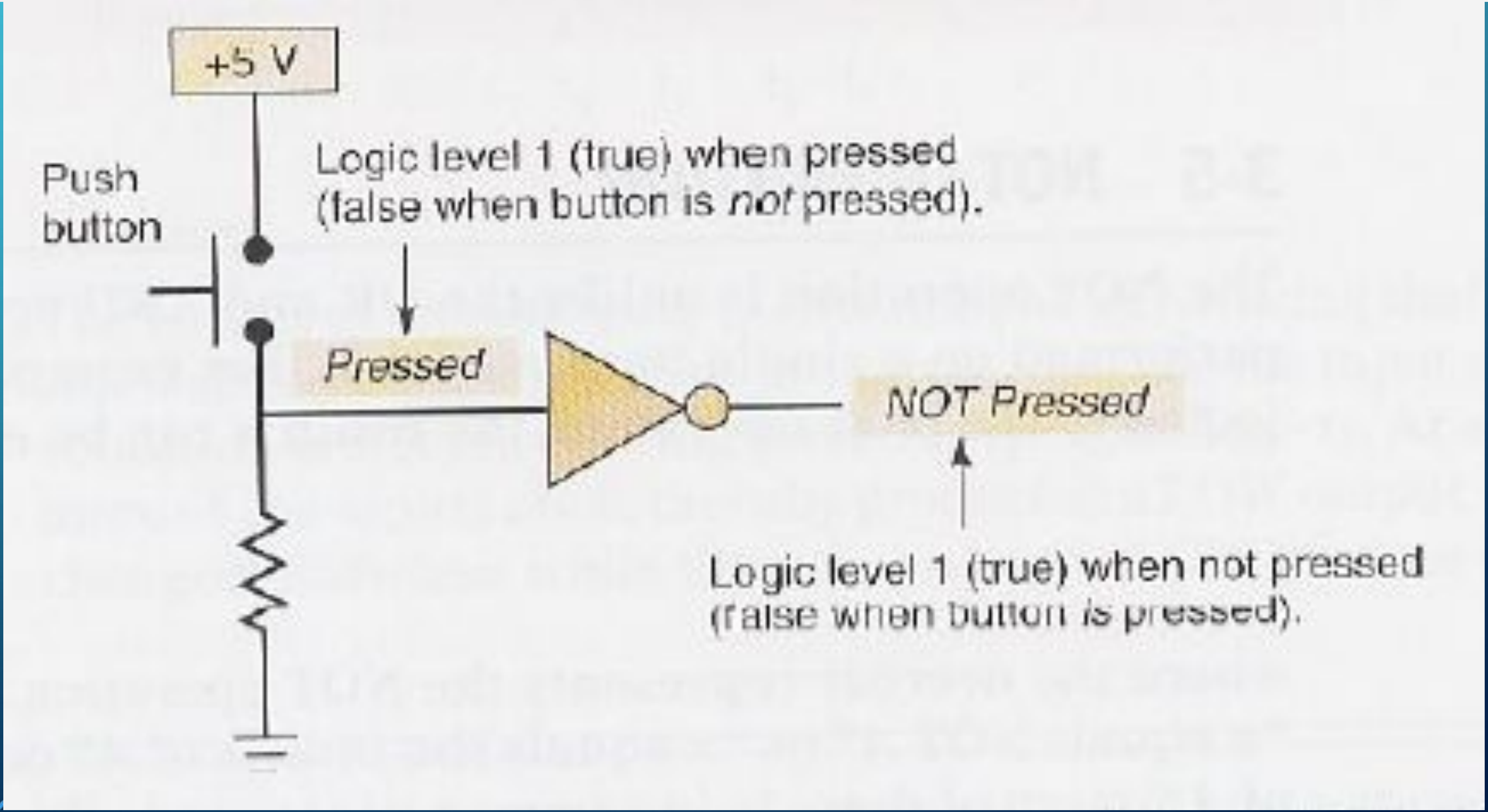
A	$x = \overline{A}$
0	1
1	0





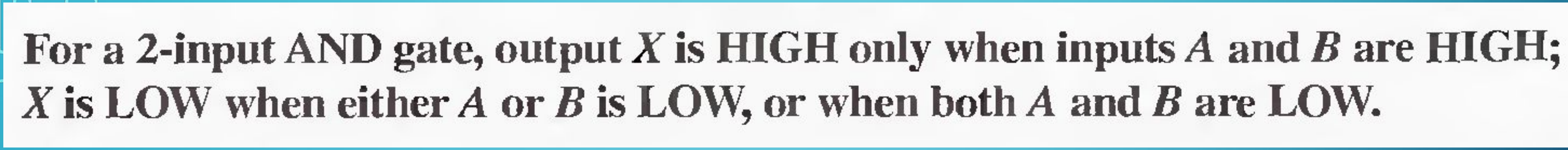
EXAMPLE

Figure 3-12 shows a typical application of the NOT gate. The push button is wired to produce a logic 1 (true) when it is pressed. Sometimes we want to know if the push button is not being pressed, and so this circuit provides an expression that is true when the button is not pressed.



+5 V
Push button
Logic level 1 (true) when pressed (false when button is *not* pressed).
Pressed
NOT Pressed
Logic level 1 (true) when not pressed (false when button *is* pressed).

AND OPERATION WITH AND GATES

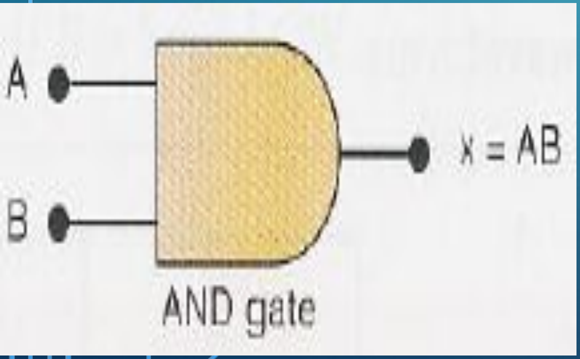


For a 2-input AND gate, output *X* is HIGH only when inputs *A* and *B* are HIGH; *X* is LOW when either *A* or *B* is LOW, or when both *A* and *B* are LOW.

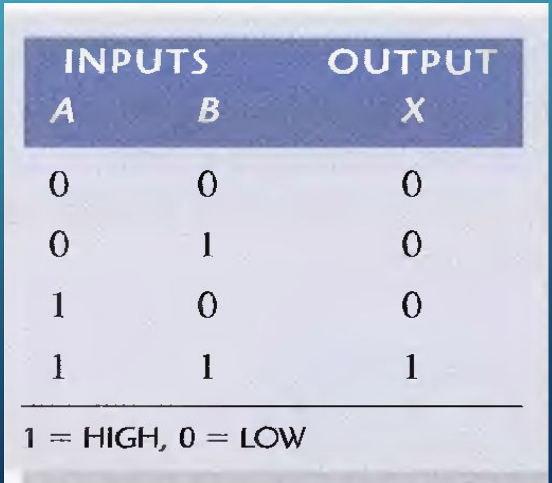
The Boolean expression for AND operation :

$$X = A.B$$

Circuit symbol

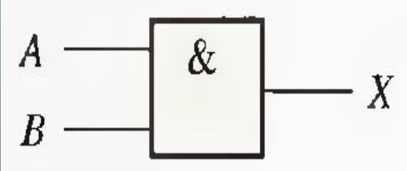


Truth Table



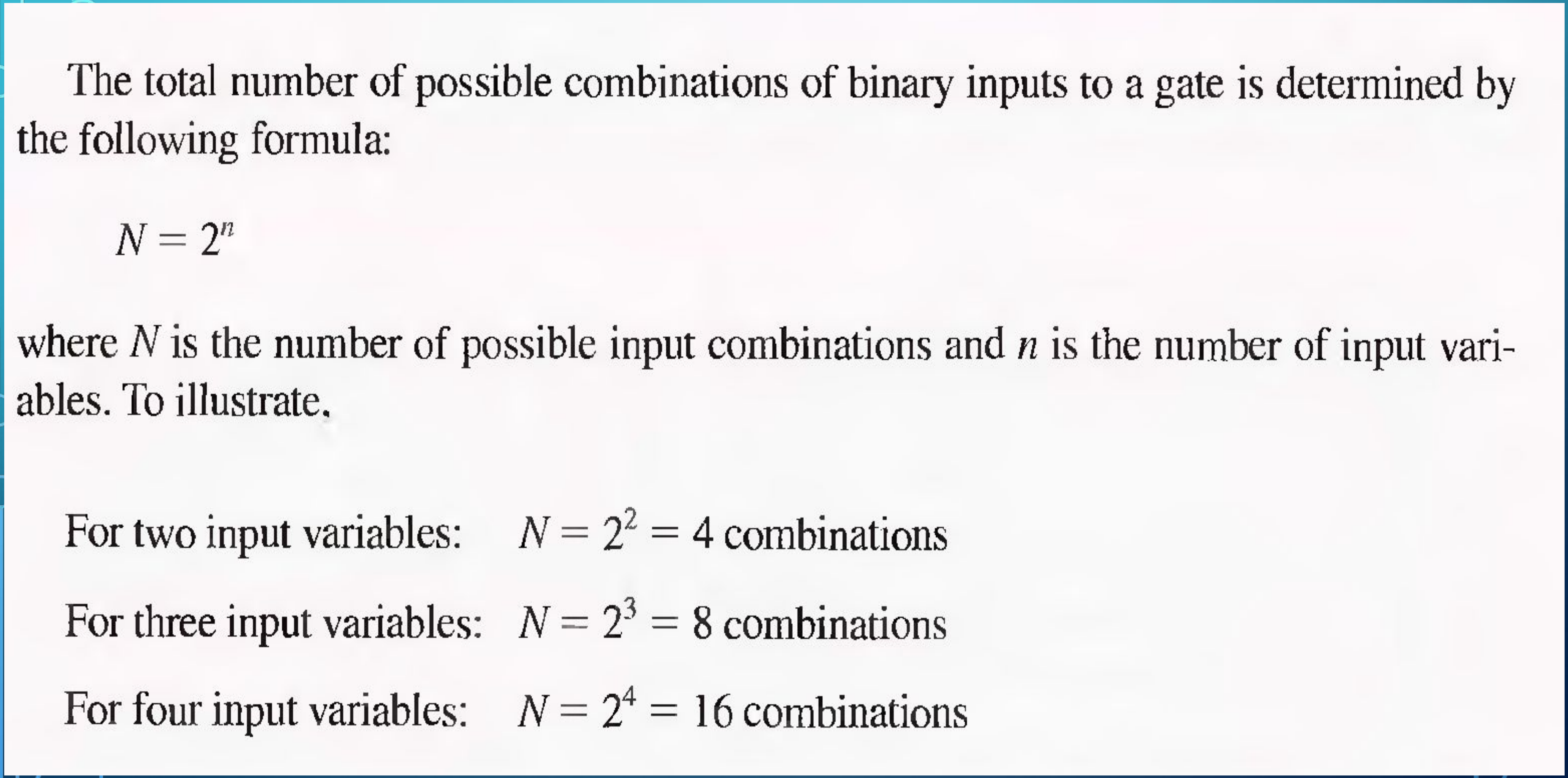
INPUTS		OUTPUT
A	*B*	*X*
0	0	0
0	1	0
1	0	0
1	1	1

1 = HIGH, 0 = LOW



(b) Rectangular outline with the AND (&) qualifying symbol

Possible Combinations of binary input to a gate



The total number of possible combinations of binary inputs to a gate is determined by the following formula:

$$N = 2^n$$

where N is the number of possible input combinations and n is the number of input variables. To illustrate,

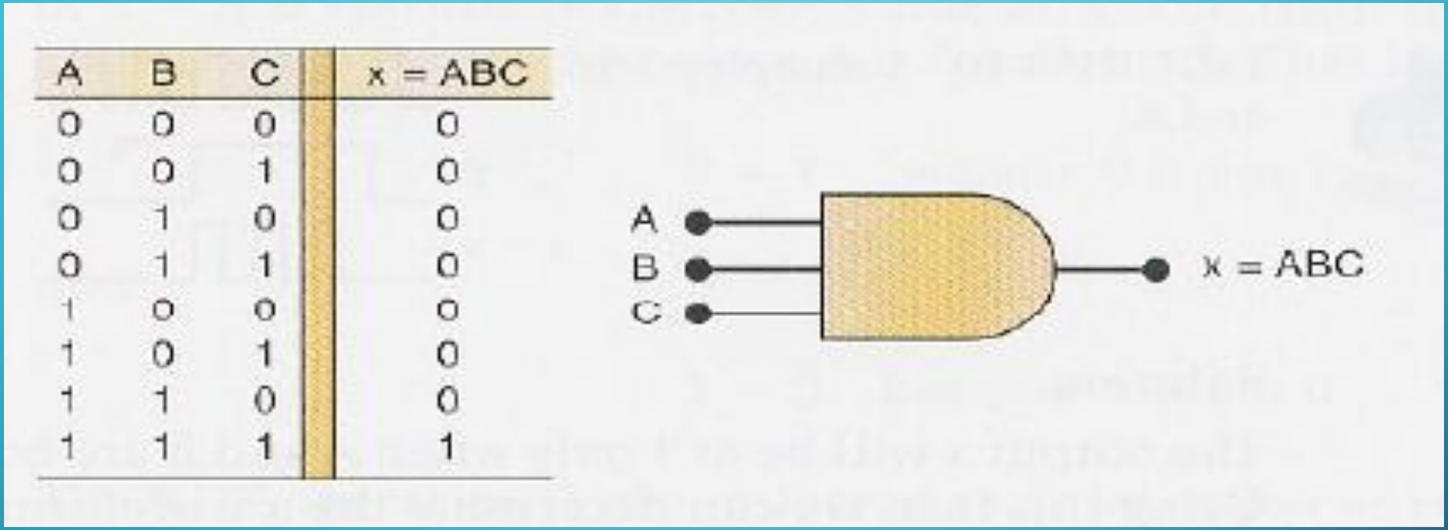
For two input variables: $N = 2^2 = 4$ combinations

For three input variables: $N = 2^3 = 8$ combinations

For four input variables: $N = 2^4 = 16$ combinations

AND OPERATION FOR THREE INPUT

X = A.B.C



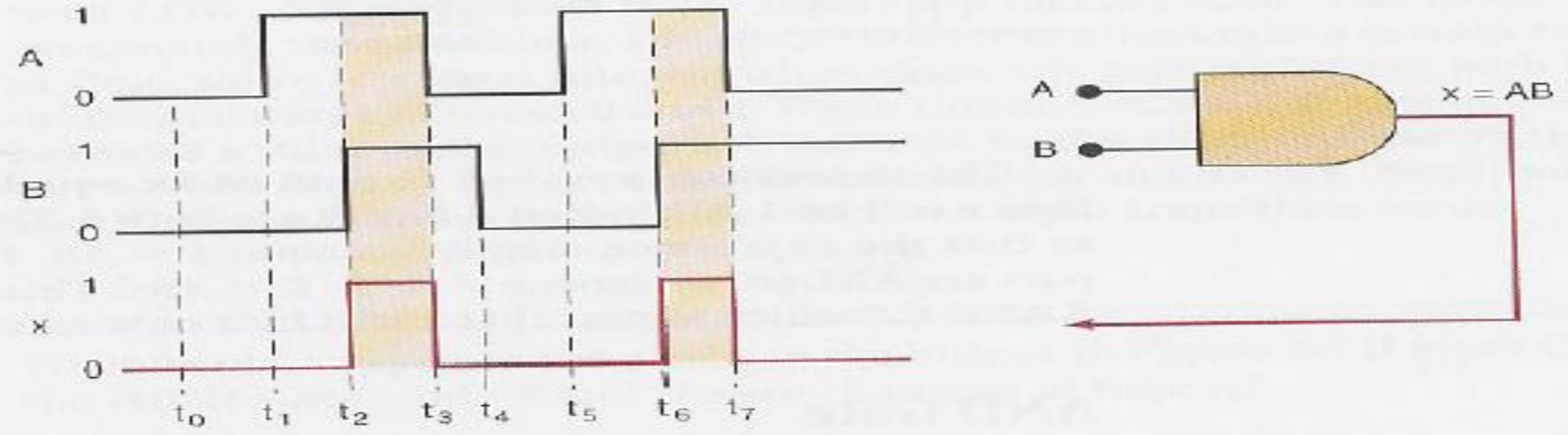
A	B	C	x = ABC
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Important point :

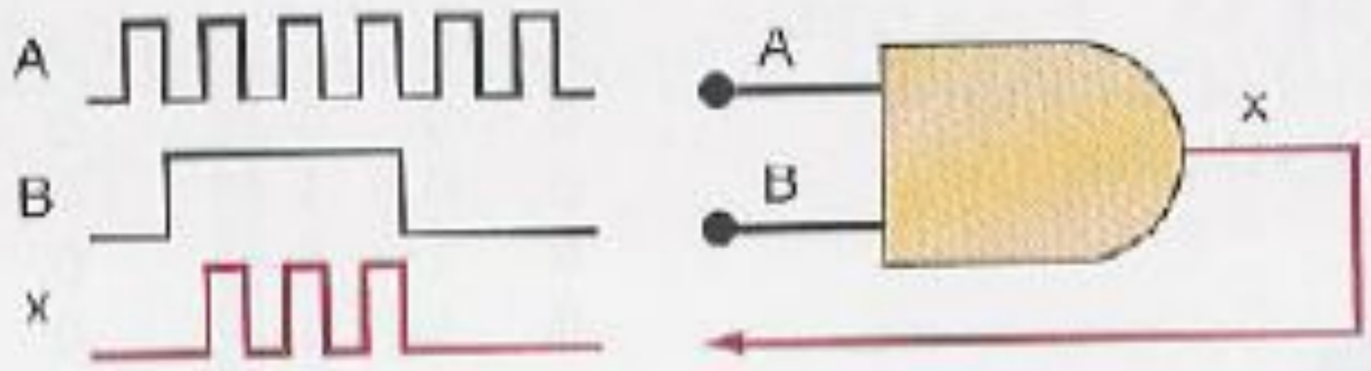
1. The AND operation is performed the same as ordinary multiplication of 1s and 0s.
2. An AND gate is a logic circuit that performs the AND operation on the circuit's inputs.
3. An AND gate output will be 1 *only* for the case when *all* inputs are 1; for all other cases, the output will be 0.
4. The expression $x = AB$ is read as "x equals A AND B."

EXAMPLES

Determine the output x from the AND gate in Figure 3-9 for the given input waveforms.

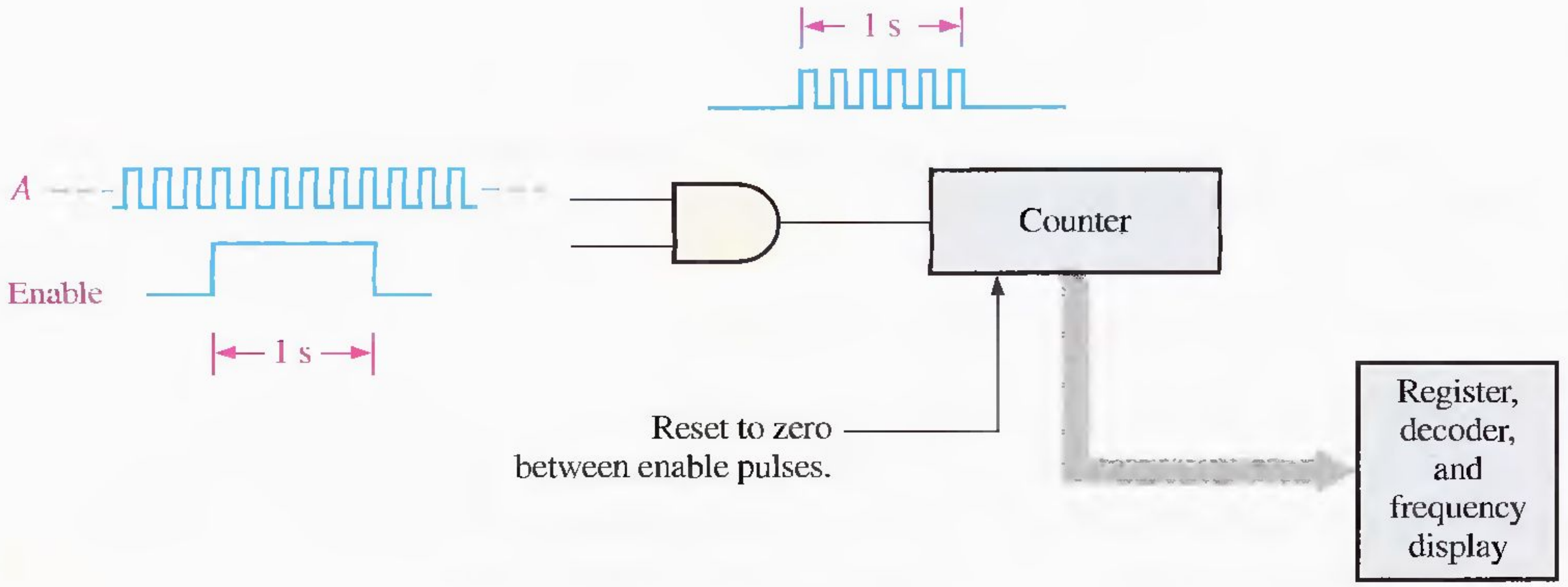


Determine the output waveform for the AND gate shown in Figure 3-10.



Applications

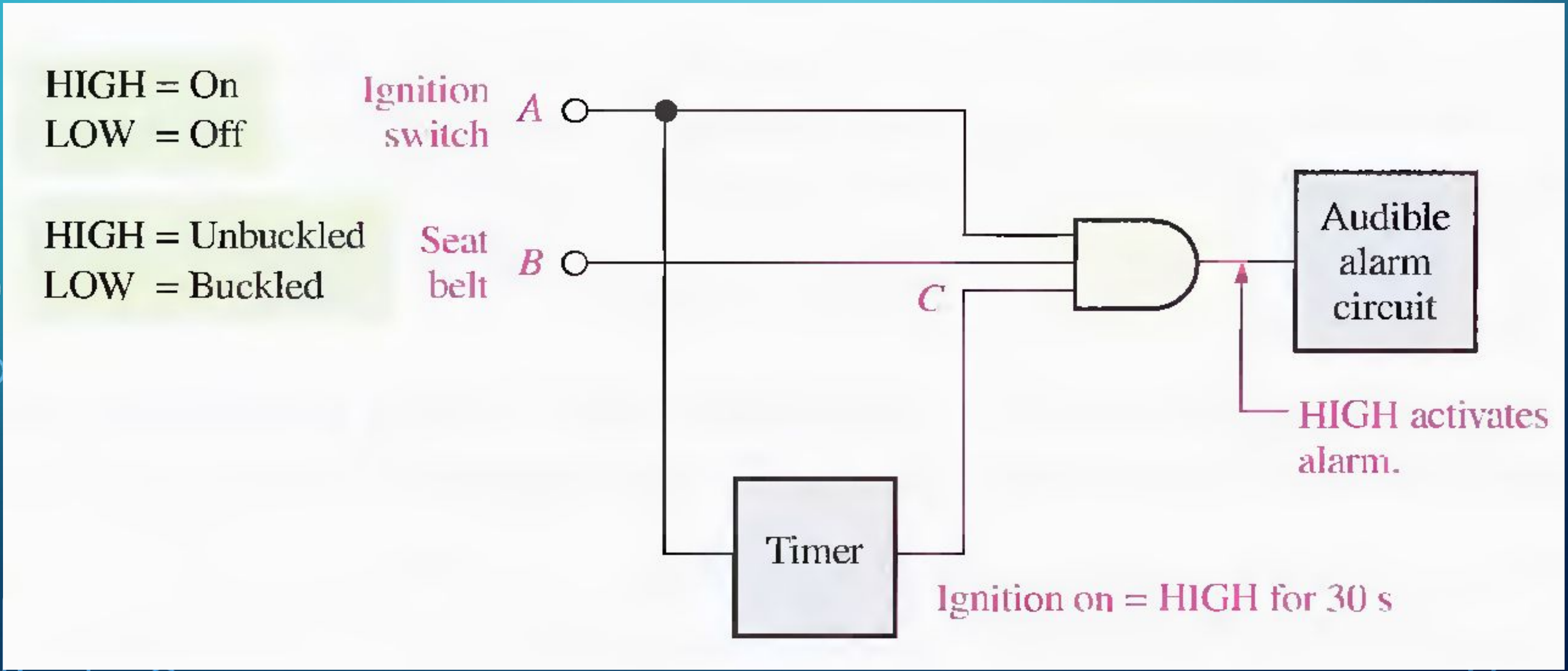
The AND Gate as an Enable/Inhibit Device


1 s
A
Enable
1 s
Counter
Reset to zero between enable pulses.
Register, decoder, and frequency display

An AND gate performing an enable/inhibit function for a frequency counter.

Applications

A Seat Belt Alarm System

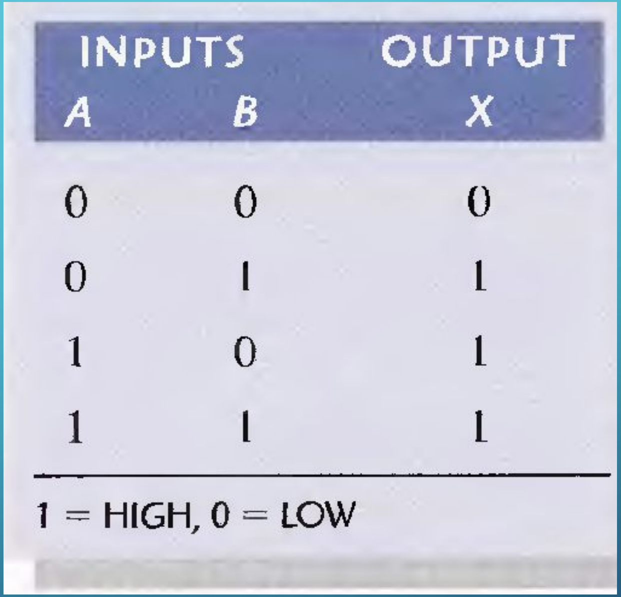


OR OPERATION WITH OR GATES

The Boolean expression for OR operation :

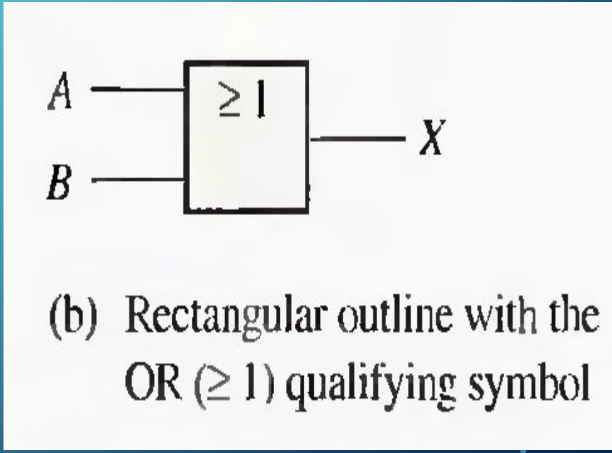
$$X = A + B$$

The Truth Table

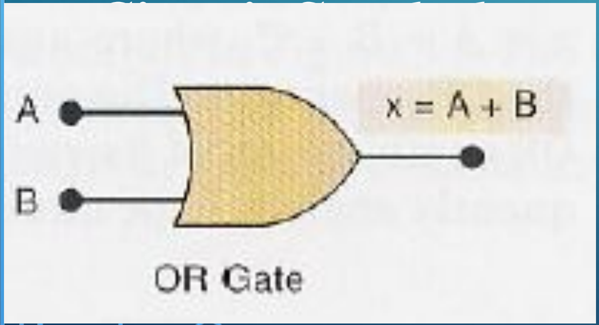


INPUTS		OUTPUT
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

1 = HIGH, 0 = LOW

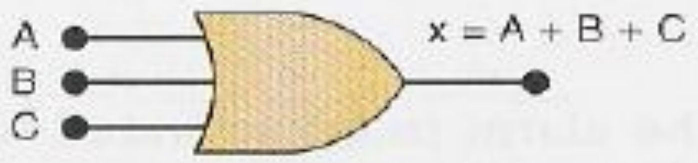


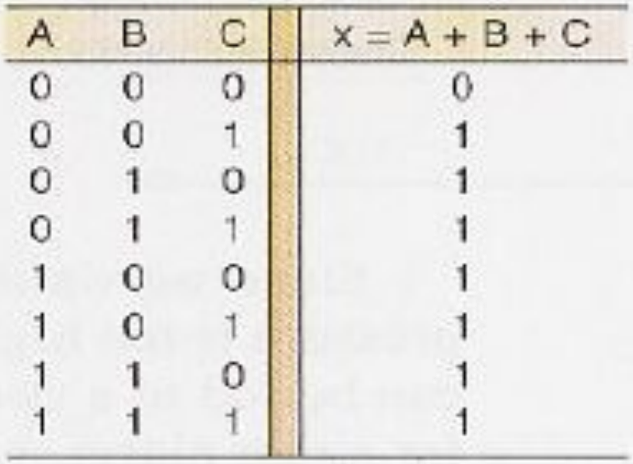
(b) Rectangular outline with the OR (≥ 1) qualifying symbol



OR OPERATION FOR THREE INPUT

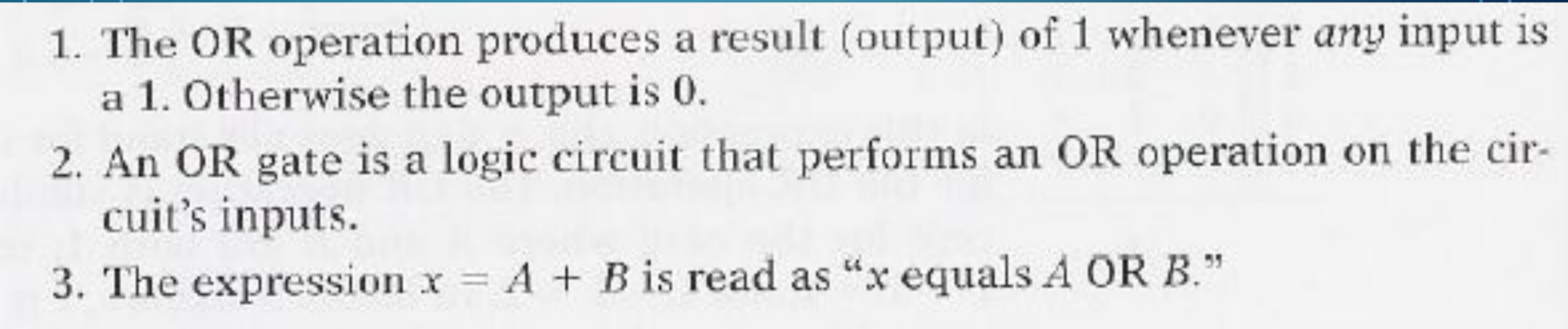
X = A+B+C





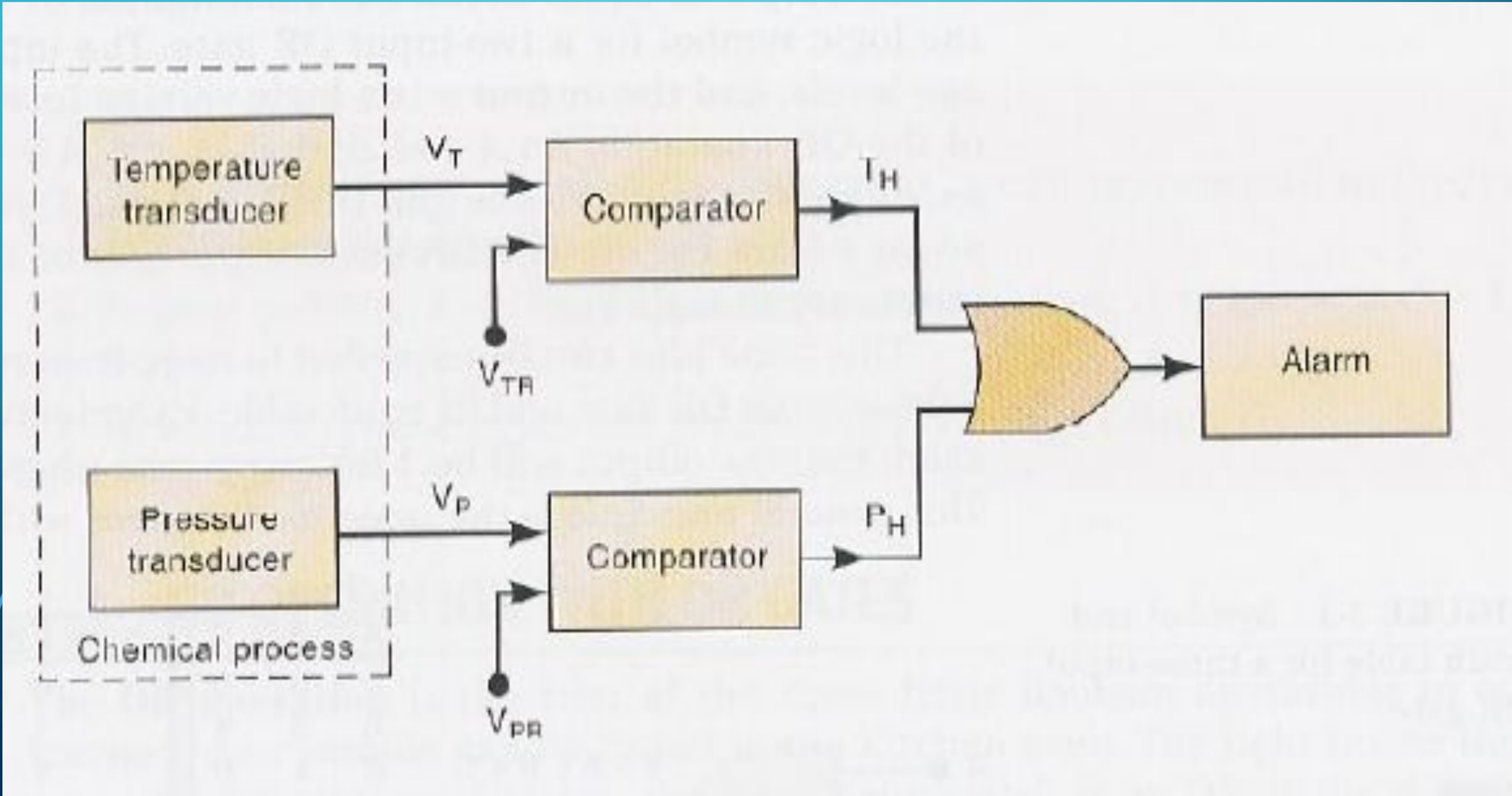
A	B	C	x = A + B + C
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Important point :



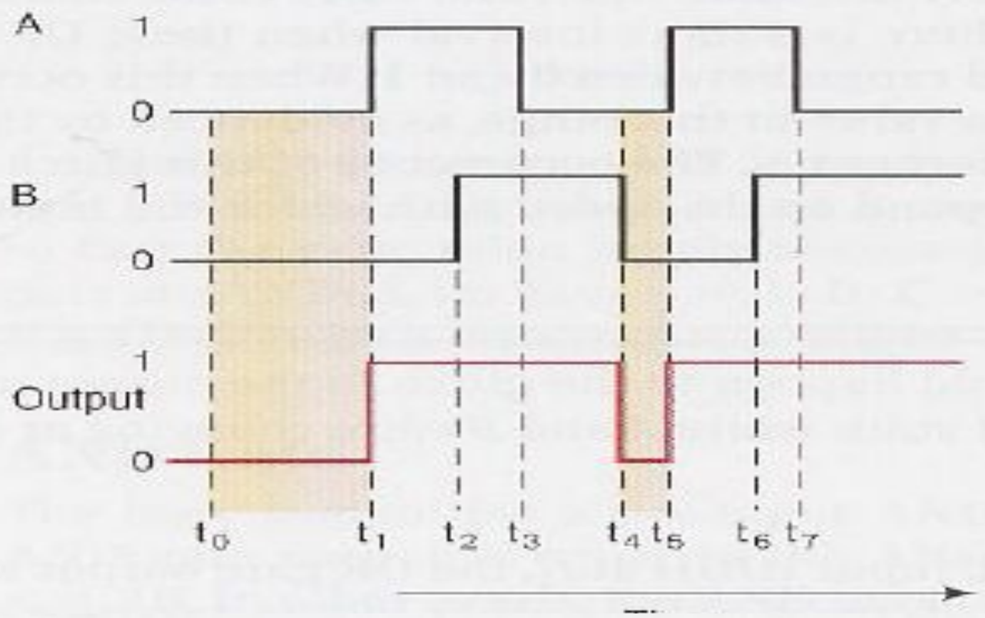
1. The OR operation produces a result (output) of 1 whenever *any* input is a 1. Otherwise the output is 0.
2. An OR gate is a logic circuit that performs an OR operation on the circuit's inputs.
3. The expression $x = A + B$ is read as "x equals A OR B."

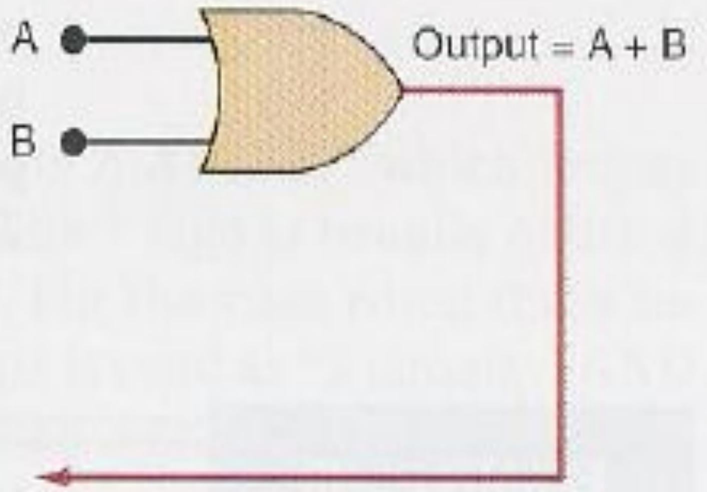
APPLICATION OF USE OF OR GATE IN AN ALARM SYSTEM



EXAMPLES

Determine the OR gate output in Figure 3-5. The OR gate inputs A and B are varying according to the timing diagrams shown. For example, A starts out LOW at time t_0, goes HIGH at t_1, back to LOW at t_3, and so on.

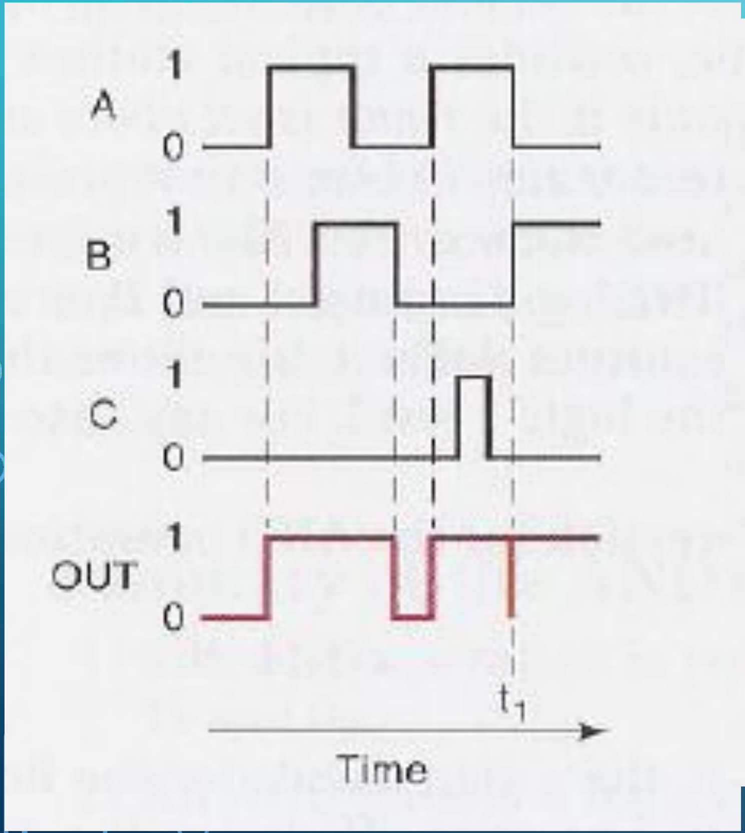


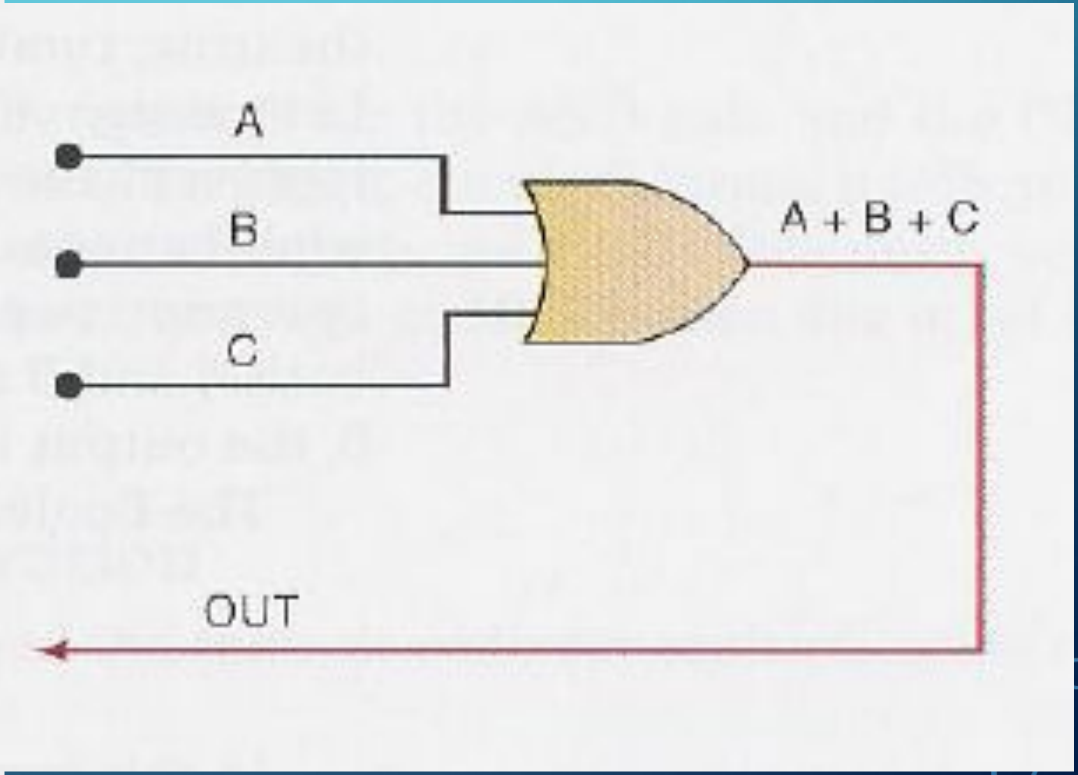


The OR gate output will be HIGH whenever *any* input is HIGH. Between time t_0 and t_1, both inputs are LOW, so OUTPUT = LOW. At t_1, input A goes HIGH while B remains LOW. This causes OUTPUT to go HIGH at t_1 and stay HIGH until t_4 because, during this interval, one or both inputs are HIGH. At t_4, input B goes from 1 to 0 so that now both inputs are LOW, and this drives OUTPUT back to LOW. At t_5, A goes HIGH, sending OUTPUT back HIGH, where it stays for the rest of the shown time span.

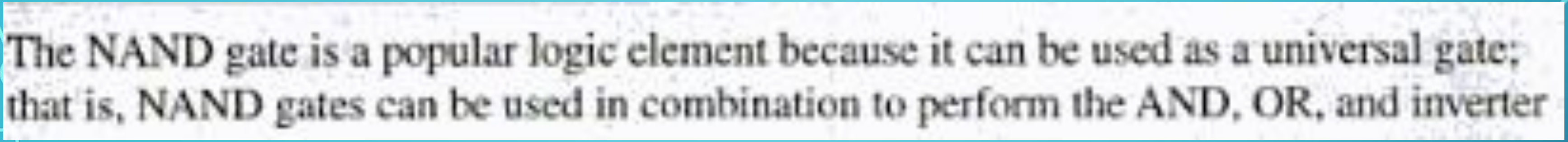
EXAMPLES

For the situation depicted in Figure 3-6, determine the waveform at the OR gate output.



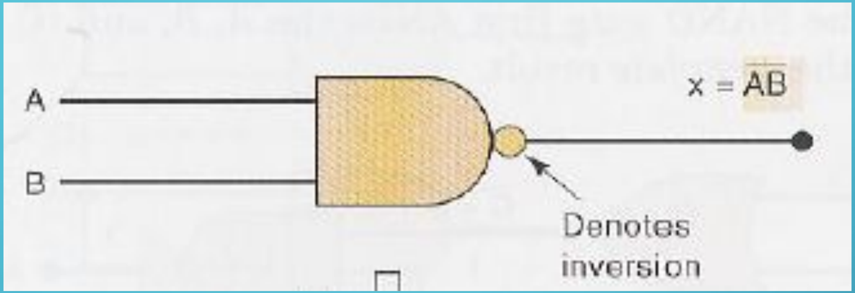


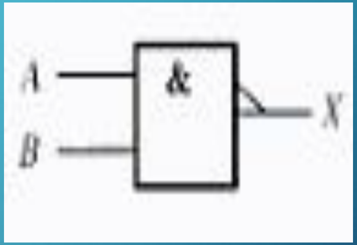
NAND GATE:



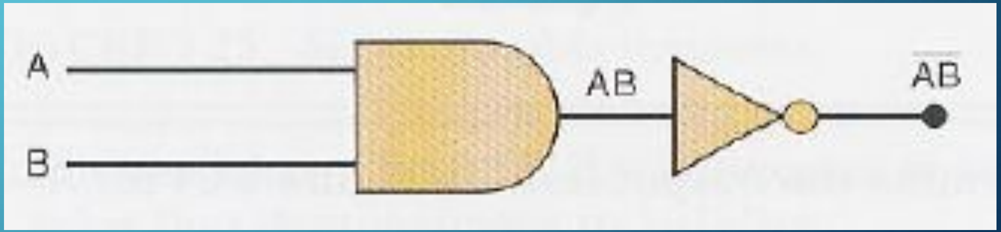
The NAND gate is a popular logic element because it can be used as a universal gate; that is, NAND gates can be used in combination to perform the AND, OR, and inverter

Symbol :

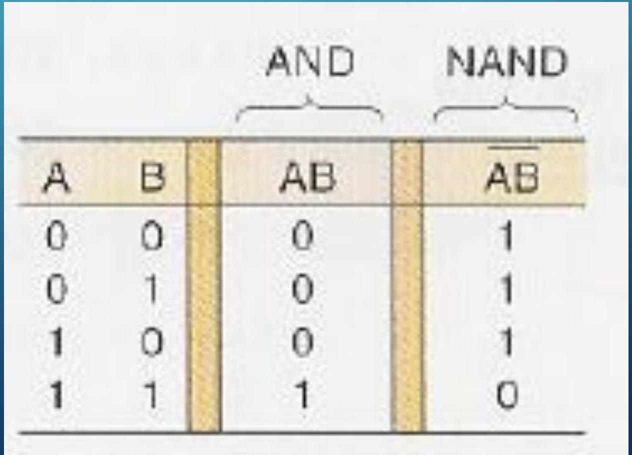




Equivalent Circuit :

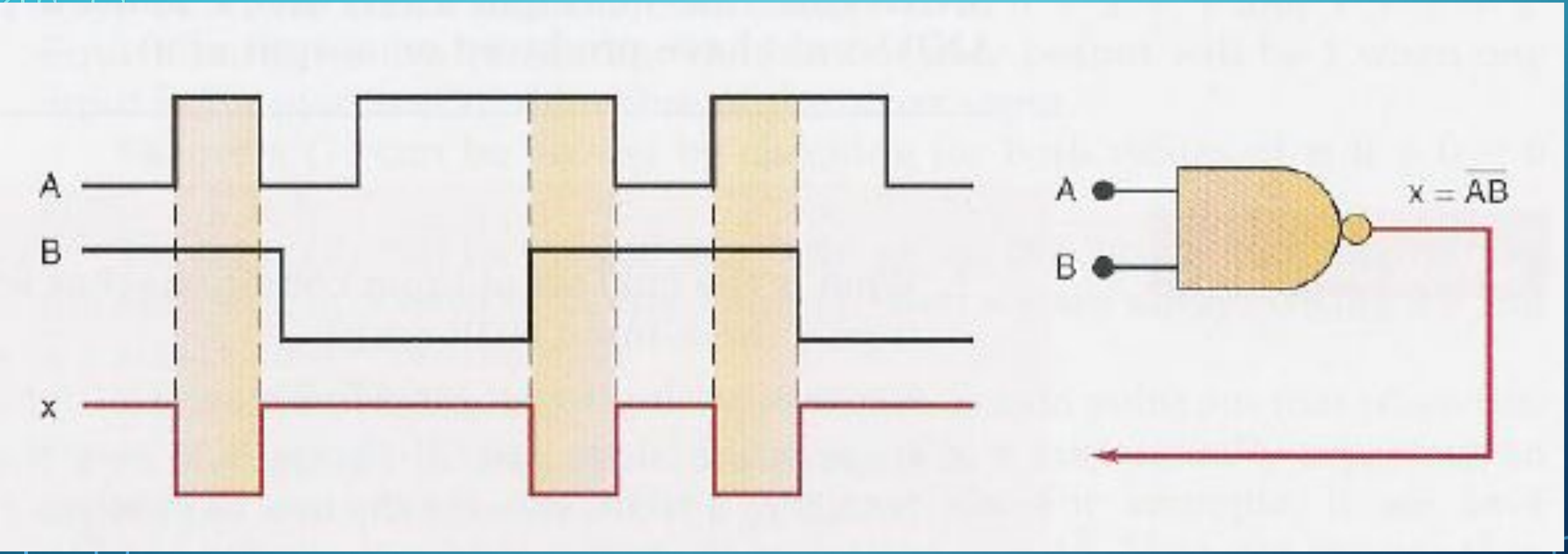


Truth Table :



		AND	NAND
A	B	AB	$\overline{AB}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

WAVE FORM AT THE OUT PUT OF NAND GATE

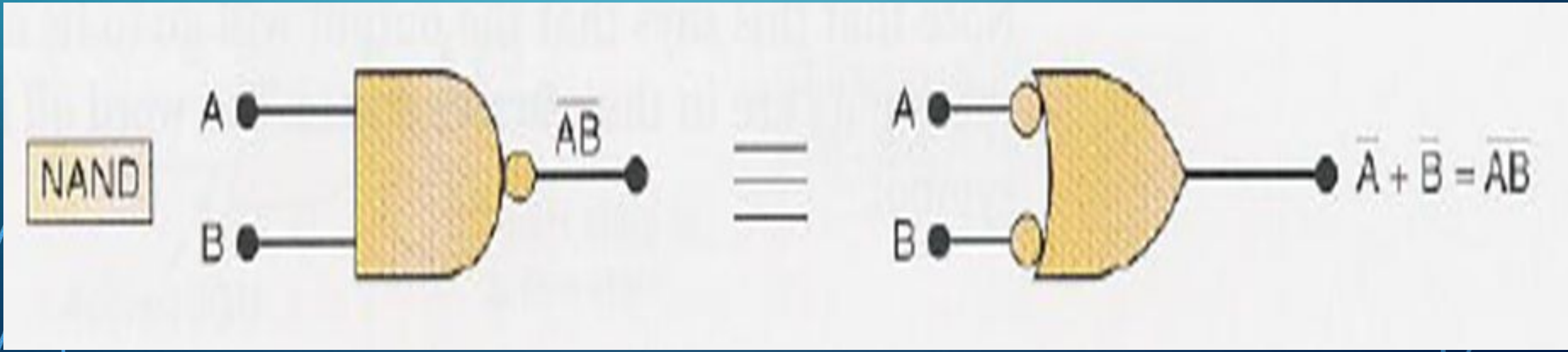


Operation of NAND Gate

For a 2-input NAND gate, output *X* is LOW if inputs *A* and *B* are HIGH; *X* is HIGH if either *A* or *B* is LOW, or if both *A* and *B* are LOW.

NEGATIVE –OR EQUIVALENT OPERATION OF A NAND GATE

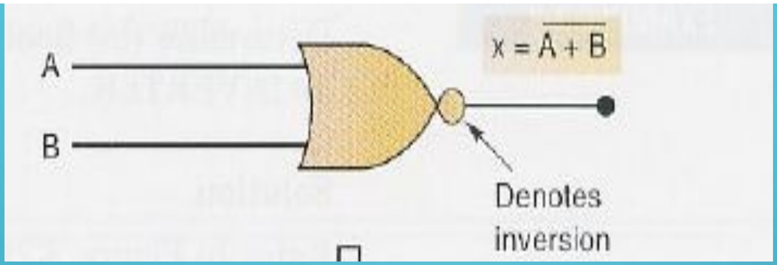
For a 2-input NAND gate performing a negative-OR operation, output *X* is HIGH if either input *A* or input *B* is LOW, or if both *A* and *B* are LOW.

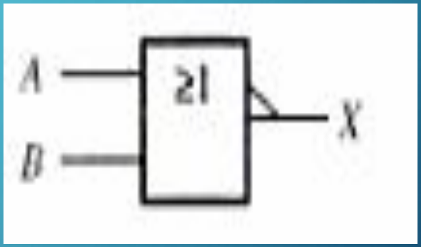


NOR GATE

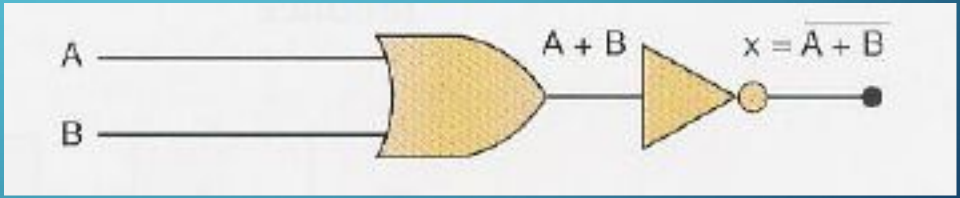
The NOR gate, like the NAND gate, is a useful logic element because it can also be used as a universal gate; that is, NOR gates can be used in combination to perform the AND, OR, and inverter operations.

Symbol :

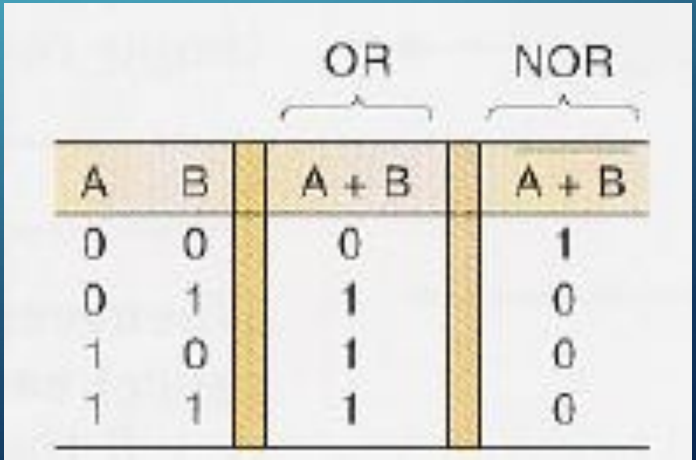




Equivalent Circuit :

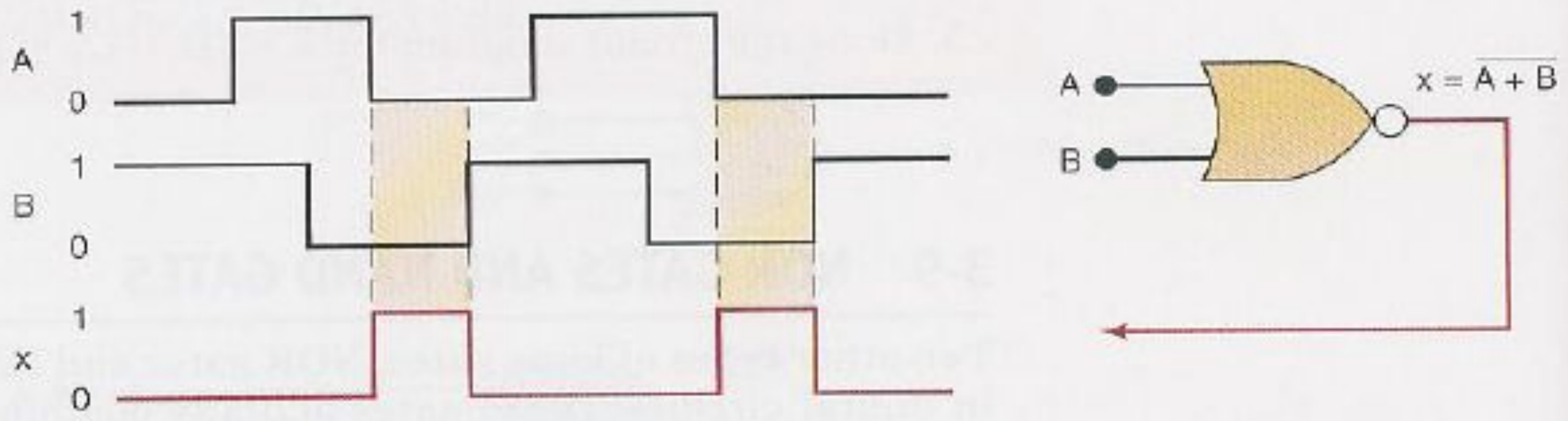


Truth Table :

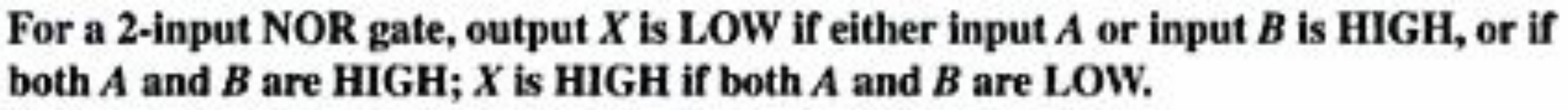


		OR	NOR
A	B	A + B	$\overline{A + B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

WAVE FORM AT THE OUT PUT OF NOR GATE

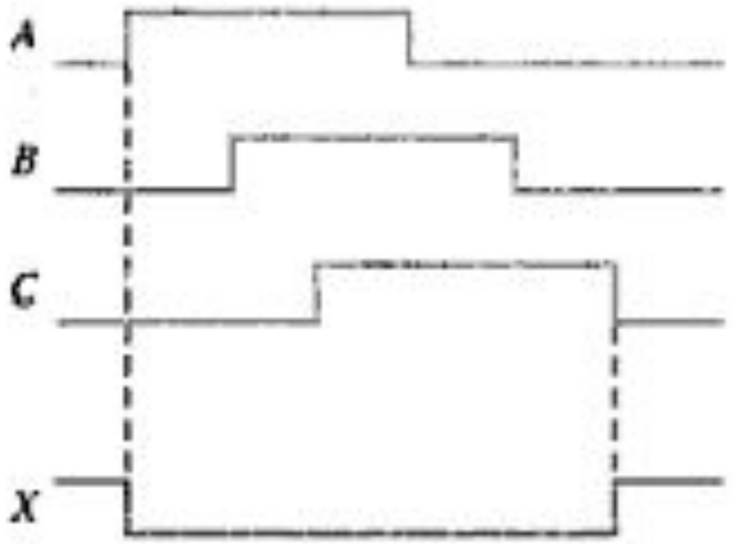


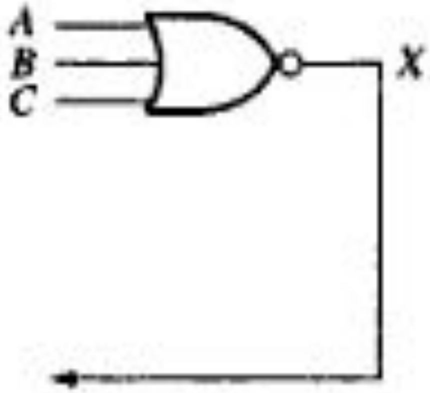
Operation of NOR Gate


For a 2-input NOR gate, output *X* is LOW if either input *A* or input *B* is HIGH, or if both *A* and *B* are HIGH; *X* is HIGH if both *A* and *B* are LOW.

EXAMPLE

Show the output waveform for the 3-input NOR gate in Figure 3–35 with the proper time relation to the inputs.

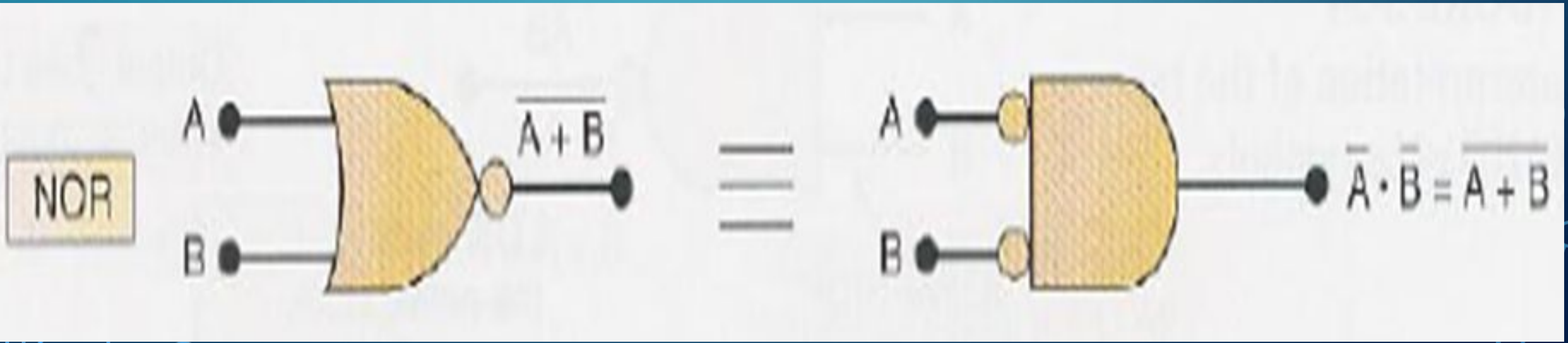




The output *X* is LOW when any input is HIGH as shown by the output waveform *X* in the timing diagram.

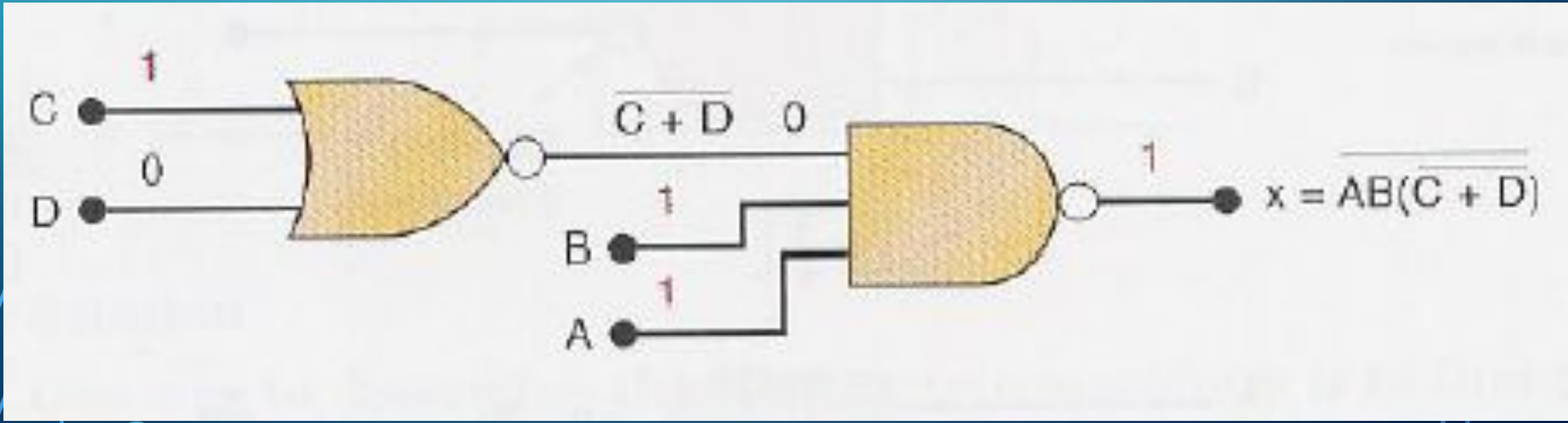
NEGATIVE –AND EQUIVALENT OPERATION OF THE NOR GATE

For a 2-input NOR gate performing a negative-AND operation, output *X* is HIGH if both inputs *A* and *B* are LOW.



EXAMPLE

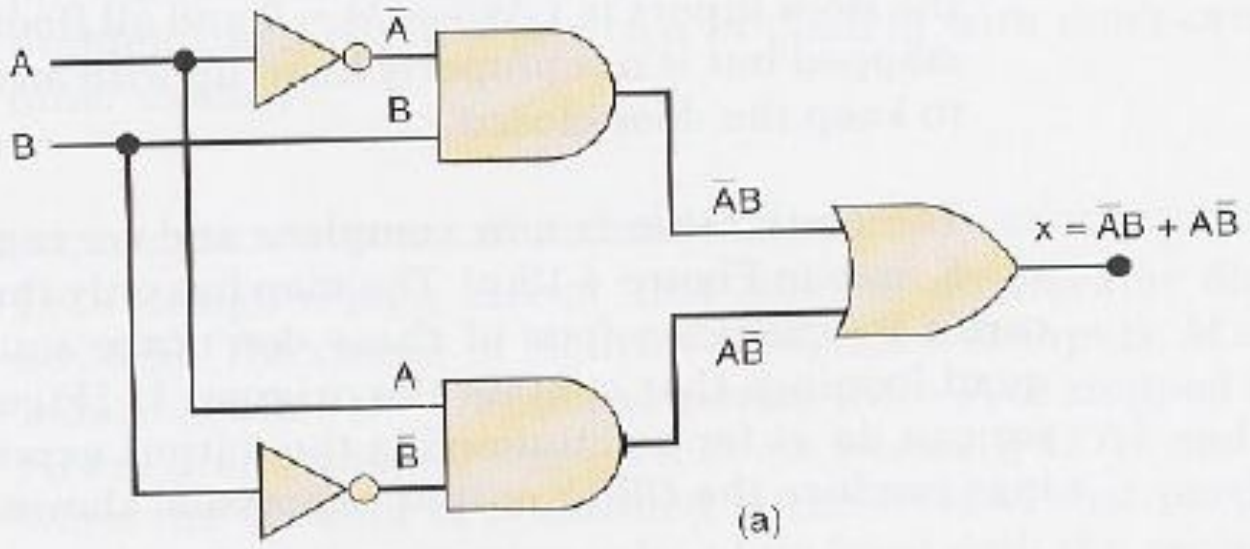
Implement the logic circuit that has the expression $x = \overline{AB \cdot (\overline{C + D})}$ using only NOR and NAND gates.



EXCLUSIVE – OR

This circuit produces a HIGH output whenever the two inputs are at opposite levels.

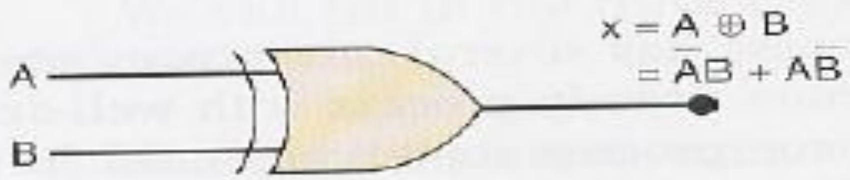
$$x = \bar{A}B + A\bar{B}$$

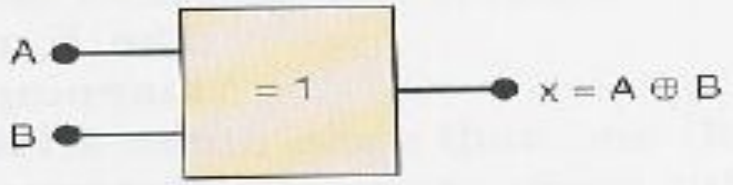


(a)

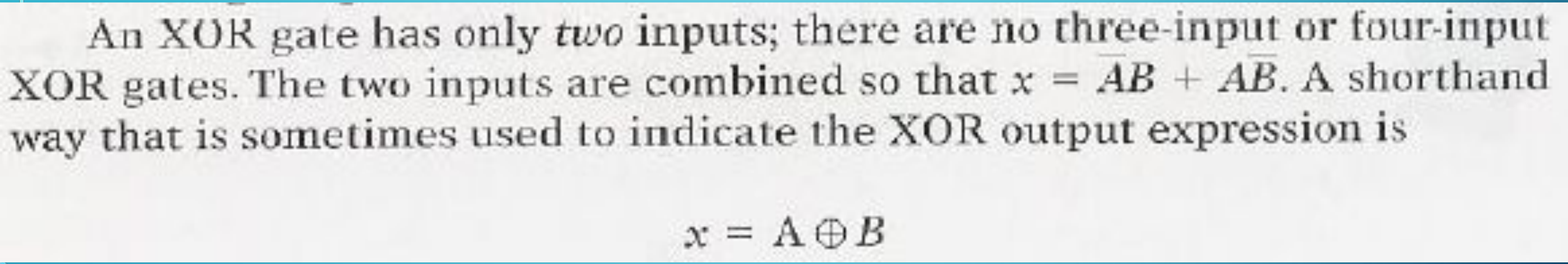
A	B	x
0	0	0
0	1	1
1	0	1
1	1	0

XOR gate symbols



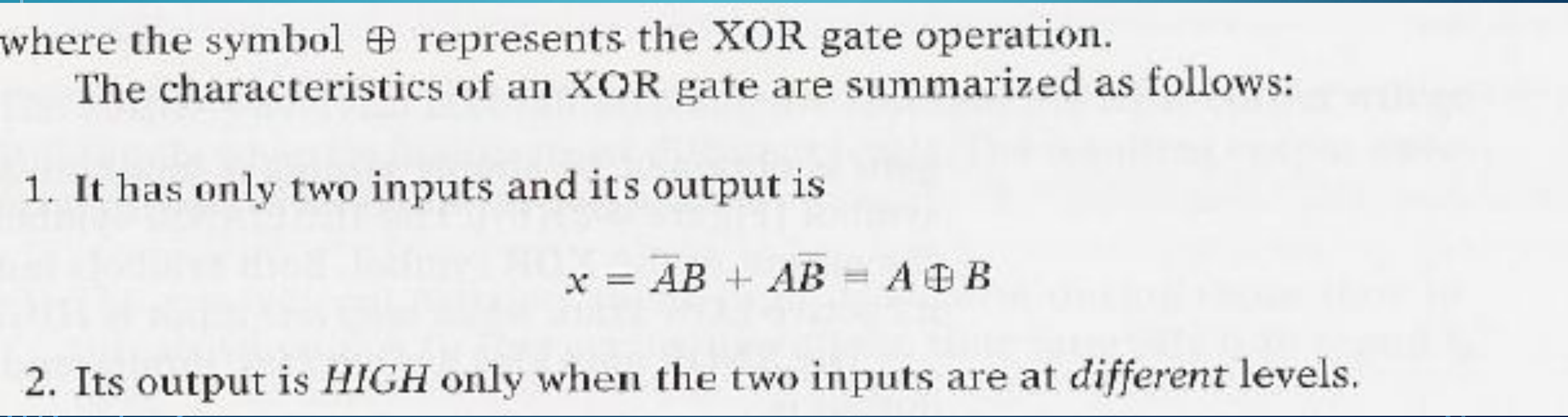


EXCLUSIVE – OR



An XOR gate has only *two* inputs; there are no three-input or four-input XOR gates. The two inputs are combined so that $x = \overline{A}B + A\overline{B}$. A shorthand way that is sometimes used to indicate the XOR output expression is

$$x = A \oplus B$$



where the symbol $\oplus$ represents the XOR gate operation.

The characteristics of an XOR gate are summarized as follows:

1. It has only two inputs and its output is

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

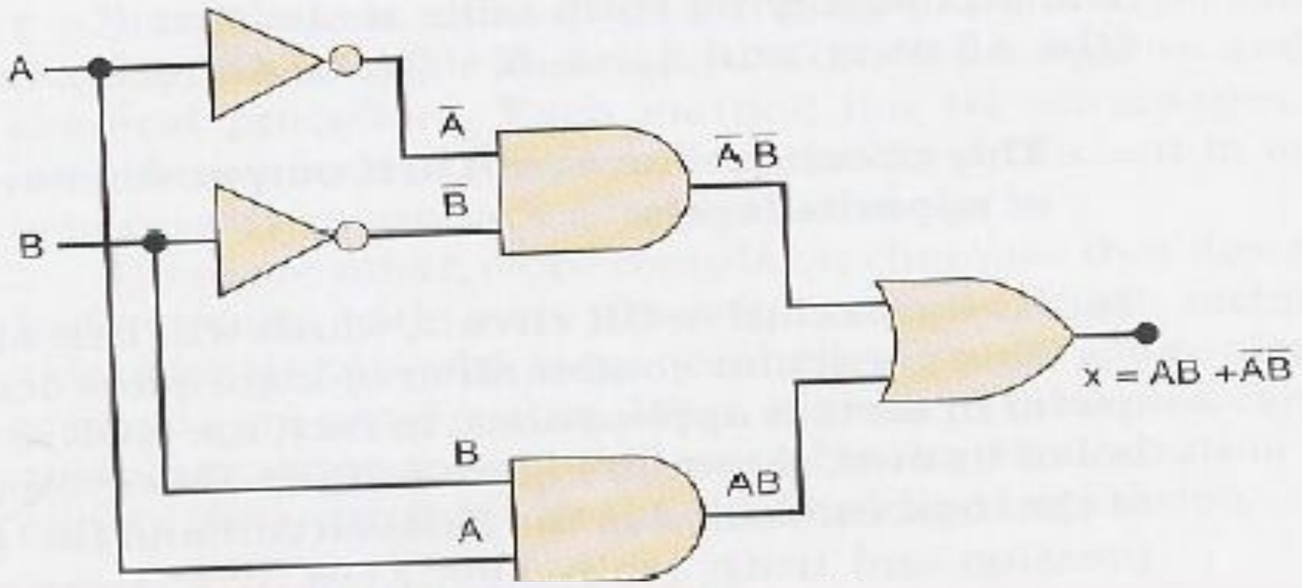
2. Its output is *HIGH* only when the two inputs are at *different* levels.

Exclusive – NOR

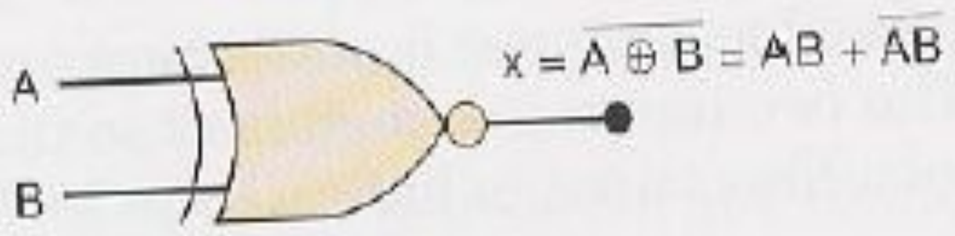
The XNOR produces a HIGH output whenever the two inputs are at the same level.

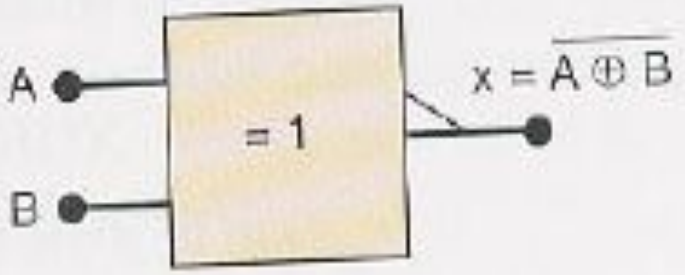
$x = AB + \overline{A}\,\overline{B}$

A	B	x
0	0	1
0	1	0
1	0	0
1	1	1



XNOR gate symbols





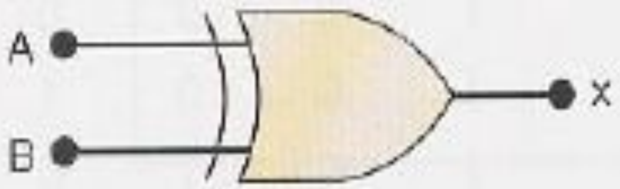
Exclusive – OR

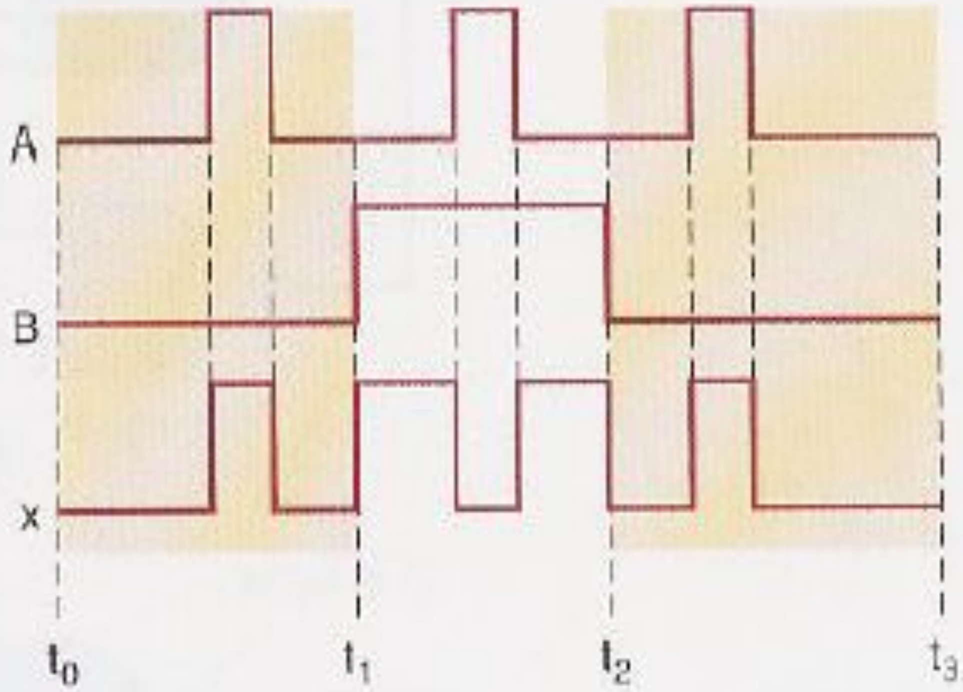
1. It has only two inputs and its output is

$$x = AB + \overline{A}\,\overline{B} = \overline{A \oplus B}$$

2. Its output is HIGH only when the two inputs are at the *same* level.

Determine the output waveform for the input waveforms given in Figure

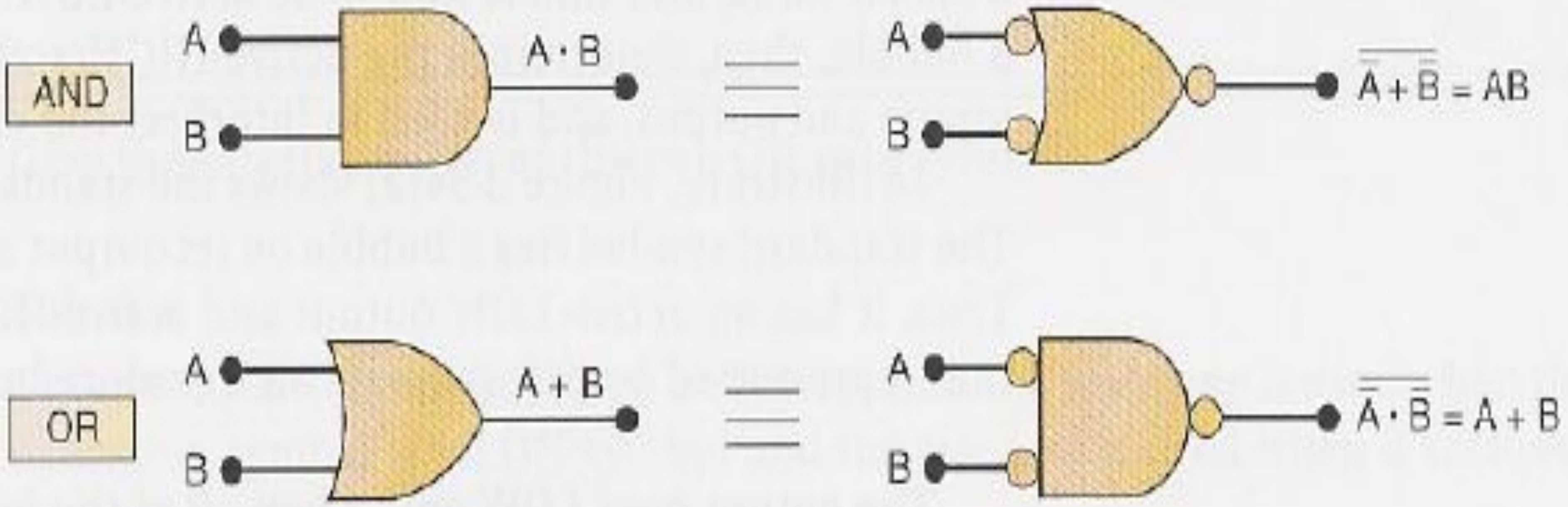




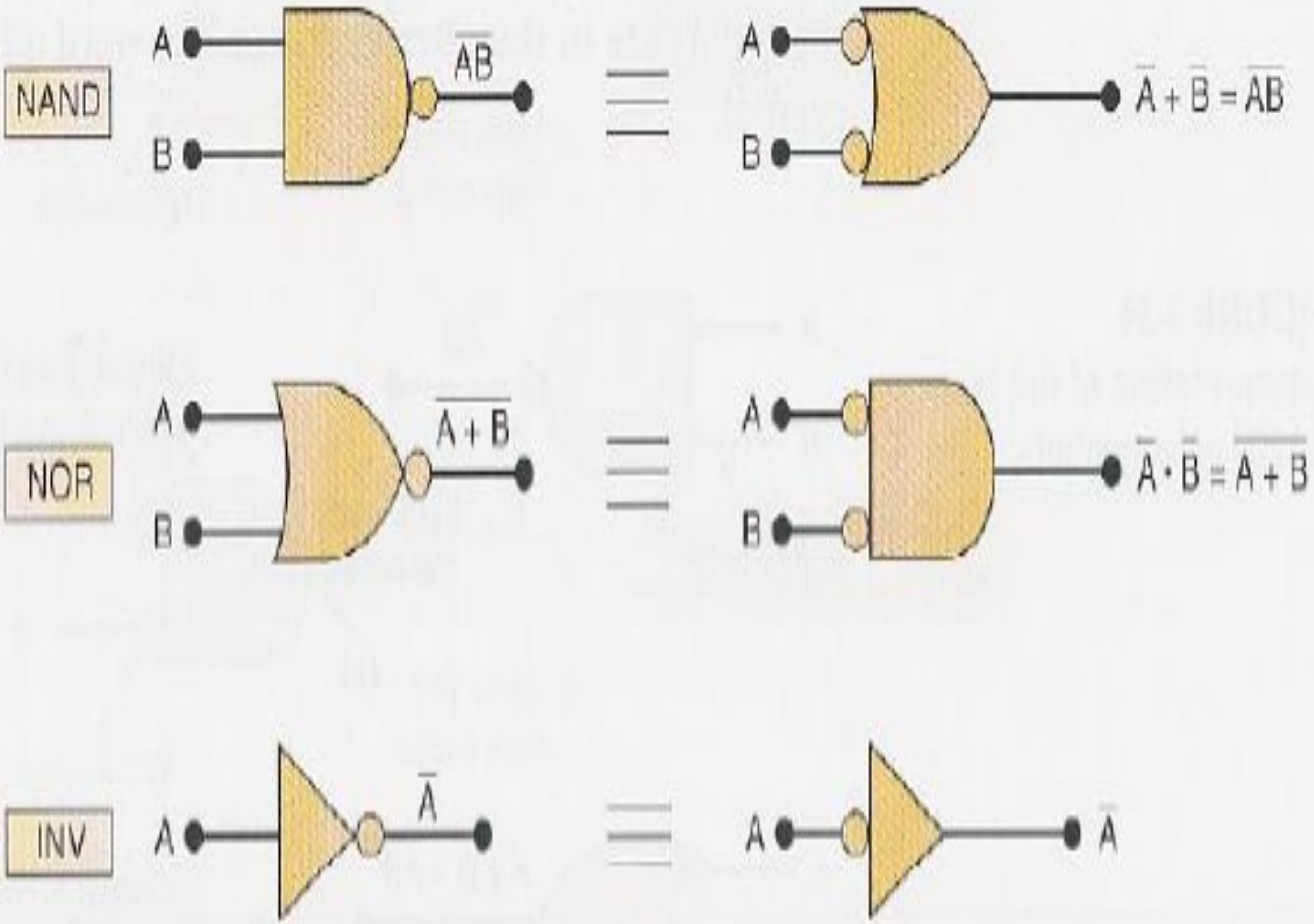
ALTERNATE LOGIC GATE REPRESENTATIONS

The alternate symbol for each gate is obtained from the standard

1. Invert each input and output of the standard symbol. This is done by adding bubbles (small circles) on input and output lines that do not have bubbles and by removing bubbles that are already there.
2. Change the operation symbol from AND to OR, or from OR to AND. (In the special case of the INVERTER, the operation symbol is not changed.)



AND
A
B
A · B
≡
A
B
$\overline{\overline{A}+\overline{B}} = AB$
OR
A
B
A + B
≡
A
B
$\overline{\overline{A}\cdot\overline{B}} = A + B$



NAND
A
B
$\overline{AB}$
A
B
$\overline{A} + \overline{B} = \overline{AB}$
NOR
A
B
$\overline{A + B}$
A
B
$\overline{A} \cdot \overline{B} = \overline{A + B}$
INV
A
$\overline{A}$
A
$\overline{A}$