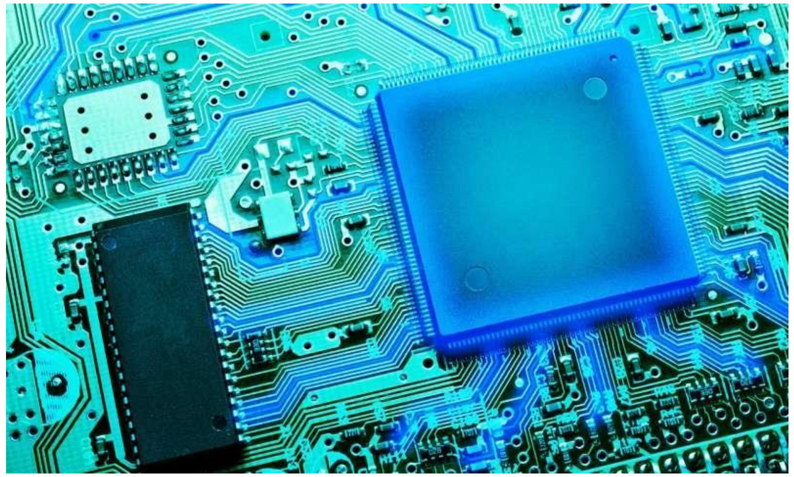
Outline

- ❑ Introduction to Logic
- ❑ Logic Gates
  - ▪ OR
  - ▪ AND
  - ▪ NOT
  - ▪ EX-OR/XOR
  - ▪ NOR
  - ▪ NAND
- ❑ Universal Gates
- ❑ Solved Examples and Exercises

APPLICATIONS OF LOGIC GATES



Microprocessors

Random Access Memory (RAM)

Flash Memories

Digital Data Transmission and Internet

Introduction to Logic

- **Digital logic** circuits use predefined **voltage levels** to represent the binary states e.g. '**1**' is represented by **high**, say 5 V and '**0**' is represented by **low**, say 0 V.

- As long as the variation in the voltage levels is not sufficient to change the binary state, the transmitted information is preserved (more **immune to noise** than analog representation).

- A **logic circuit** is one that behaves like a voltage-controlled **switch**, i.e. a two-position device with **ON** and **OFF** states

- This is termed a **binary device**, in which the **ON** state is represented by '**1**' and the **OFF** state by '**0**'

Boolean Logic

- Named after English mathematician George Boole (1815 – 1864)
- Boolean algebra uses symbols to represent a logical expression that has one of **two possible values**: **True/False** (**1/0**; ON/OFF; High/Low)
- The main purpose of these logical expressions is to describe the **relationship** between a logic circuit's **output (the decision)** and its **inputs (the circumstances), both** of which are **binary values**.
- Boolean **constants** and **variables** are allowed to have only two possible values, **0 or 1**
- A Boolean **variable** is a quantity that may, **at different times**, be equal to **either 0 or 1**

Boolean Logic

- Boolean **0 and 1** do not represent actual numbers but instead represent the **state of a voltage variable**, or what is called its "**logic level**"
- A voltage in a digital circuit is said to be at the logic **0 level** or the logic **1 level**, depending on its actual numerical value (e.g. **low** or **high**)
- The **inputs** are considered **logic variables** whose logic **levels** at any time **determine** the **output levels**
- We use **letter symbols** to represent **logic variables**. For example, the letter A might represent a certain digital circuit input or output, and at any time either A = 0 or A = 1

Logic Gates

- Logic gates are the most basic logic circuits
- These are the **fundamental building blocks** from which all other logic circuits and digital systems are constructed
- In Boolean algebra there are **only three basic operations**: **OR**, **AND**, and **NOT**, called logic operations
- Logic **gates** can be **constructed** from **diodes**, **transistors**, and resistors connected so that the circuit output is the result of a basic logic operation (OR, AND, NOT) performed on the inputs.

Logic Gates

Classification of Gates

Basic Gates
1. OR Gate
2. AND Gate
3. NOT (Invertor) Gate

Universal Gates
1. NAND Gate
2. NOR Gate

Special Purpose
1. EX-OR Gate
2. EX-NOR Gate

Boolean Algebra and Logic Gates

- Boolean Algebra and Logic Gates is an algebra developed by George Boole to argue the truth and falsity of statements
- Boolean Algebra and Logic Gates now serves as the mathematical basis for designing and building electronic circuits for computing devices
- Boolean Algebra and Logic Gates: The fundamental building blocks are logic gate circuits

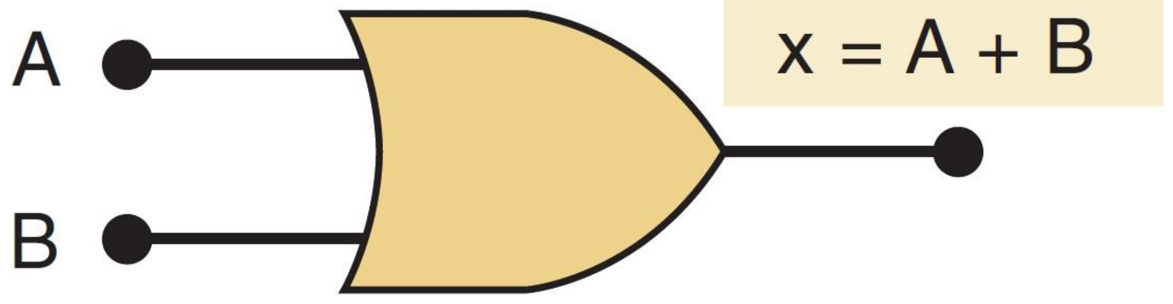
"OR" Logic Gate

- Two logic inputs, A and B, are combined using the OR operation (denoted by '+' symbol) to produce the output x (x, A, B are bits).
- A **truth table** is a means for describing how a logic circuit's **output depends** on the logic levels present at the circuit's **inputs**

OR

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

Truth Table



OR Gate

Circuit Symbol

“OR” LOGIC GATE

- Example: **The bus will go to A or B**. The success (truth) of the bus going to one or other can be represented by x; thus ‘x’ occurs when the bus goes to **either A or B or both** (it might travel through A to get to B or vice versa)
- The Boolean expression for the OR operation is **$x = A + B$**
- The **positive (+) sign** is **not** the **additive** function, but means OR in logic

x = A OR B

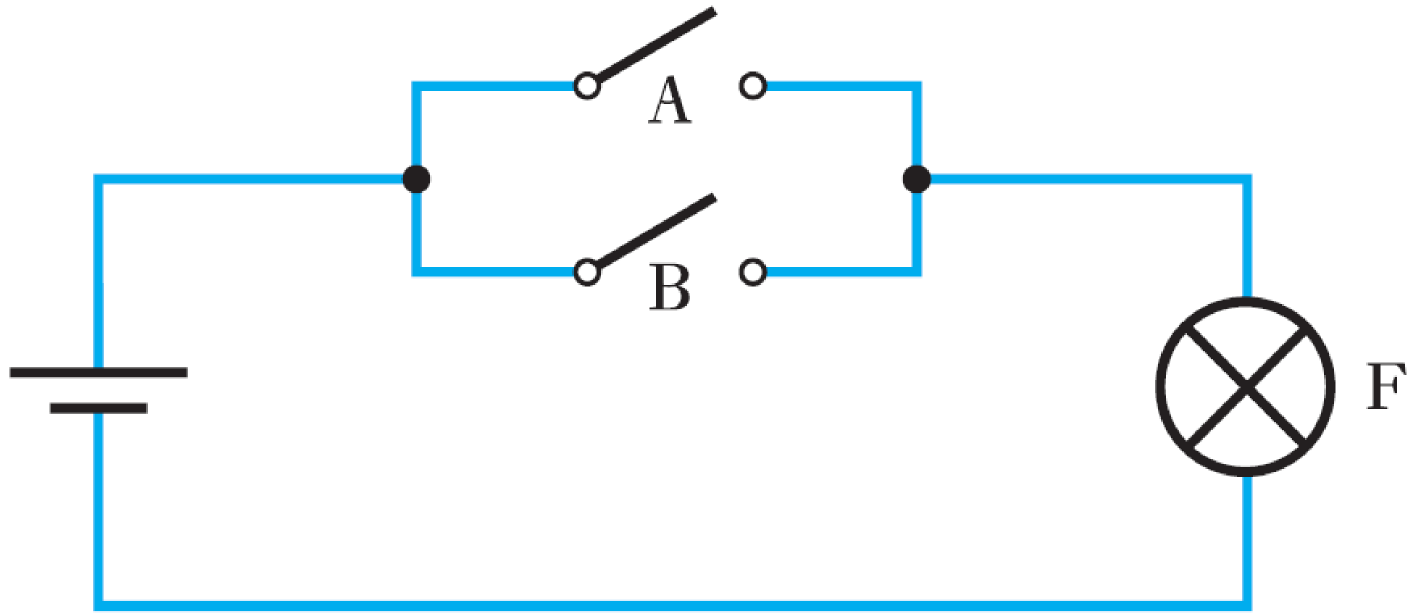
OR

A	B	$x = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

“OR” Logic Gate

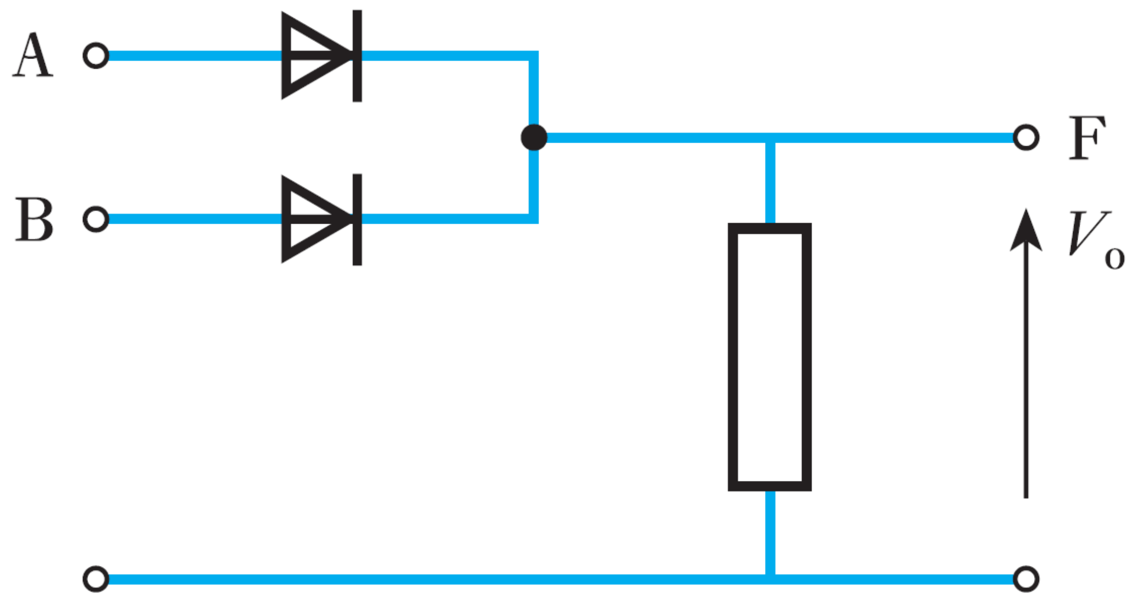
- In an electrical circuit, **OR** operation is equivalent to **two switches in parallel**
- The lamp F lights (F = 1) when either switch or both switches are closed (closed switch is represented by logic level 1)

i.e. F = A + B



“OR” Logic Gate

- If no (zero) voltage (logic level 0) is applied to both inputs then the output voltage V_o is also zero (logic level 0) (both diodes are reverse biased i.e. open circuit)
- If, however, a positive voltage of, say, 5 V (logic level 1) is applied to either one or both inputs then at least one diode turns on (forward biased), and the output voltage is also ~5 V

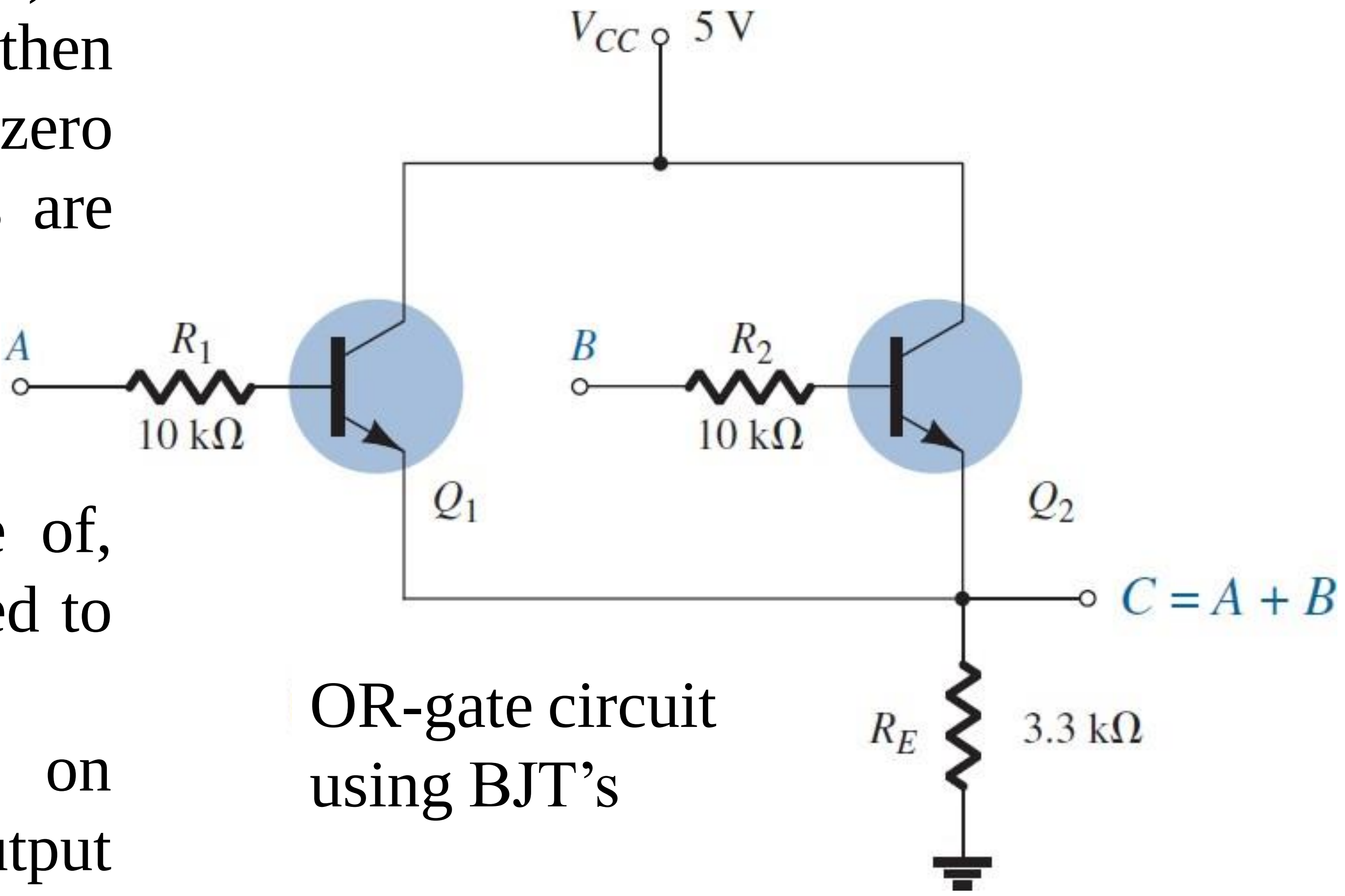


Diode-based OR-gate circuit

“OR” LOGIC GATE

If no (zero) voltage (logic level 0) is applied to both inputs *A* and *B* then the output voltage V_o is also zero (logic level 0) (both transistors are in cutoff i.e. open circuit)

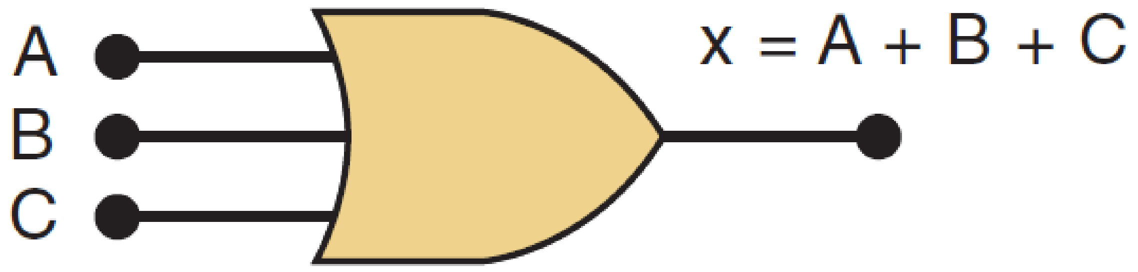
If, however, a positive voltage of, say, 5 V (logic level 1) is applied to either one or both inputs then at least one transistor turns on (saturation mode), and the output voltage *C* is also ~5 V (logic level 1)



OR-gate circuit using BJT’s

“OR” Logic Gate

For three inputs, the OR-gate circuit symbol, and corresponding truth table is given below. The Boolean expression is 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

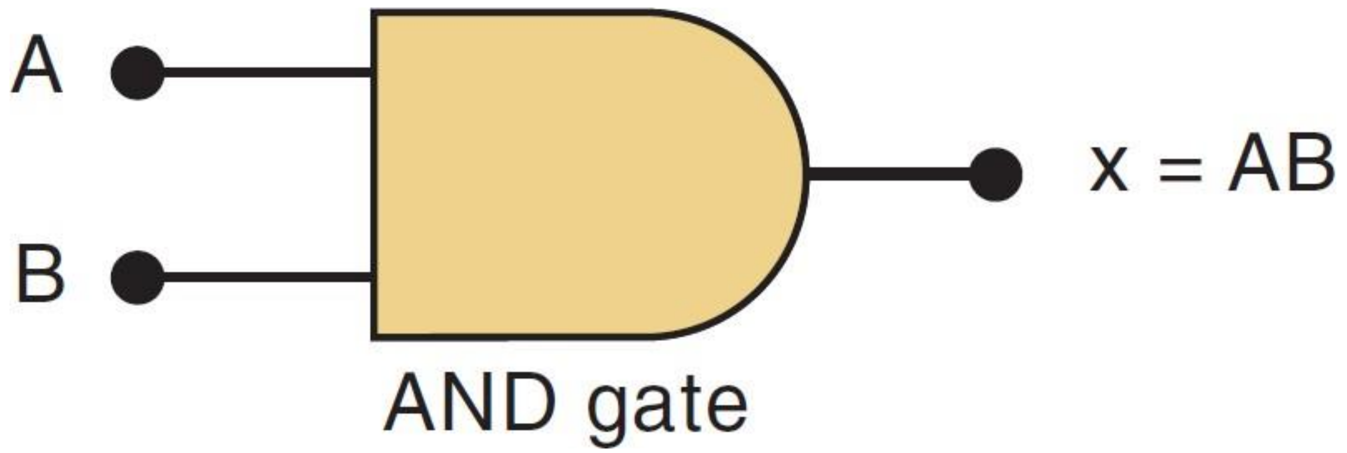
“AND” LOGIC GATE

- Two logic inputs, A and B, are combined using the AND operation (denoted by ‘.’ symbol) to produce the output x (x, A, B are bits)
- The table shows that x is a logic 1 only when both A and B are at logic 1

AND

A	B	$x = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

Truth Table



Circuit Symbol

"AND" Logic Gate

- Example: **The bus goes to A and B**. The success (truth) of the bus going to both places can be represented by x; thus 'x' occurs **only when** the bus goes to **both A and B**
- The Boolean expression for the AND operation is **F = A . B**
- The **period (.) sign** is **not** the **multiplicative** function, but means AND in logic

F = A AND B

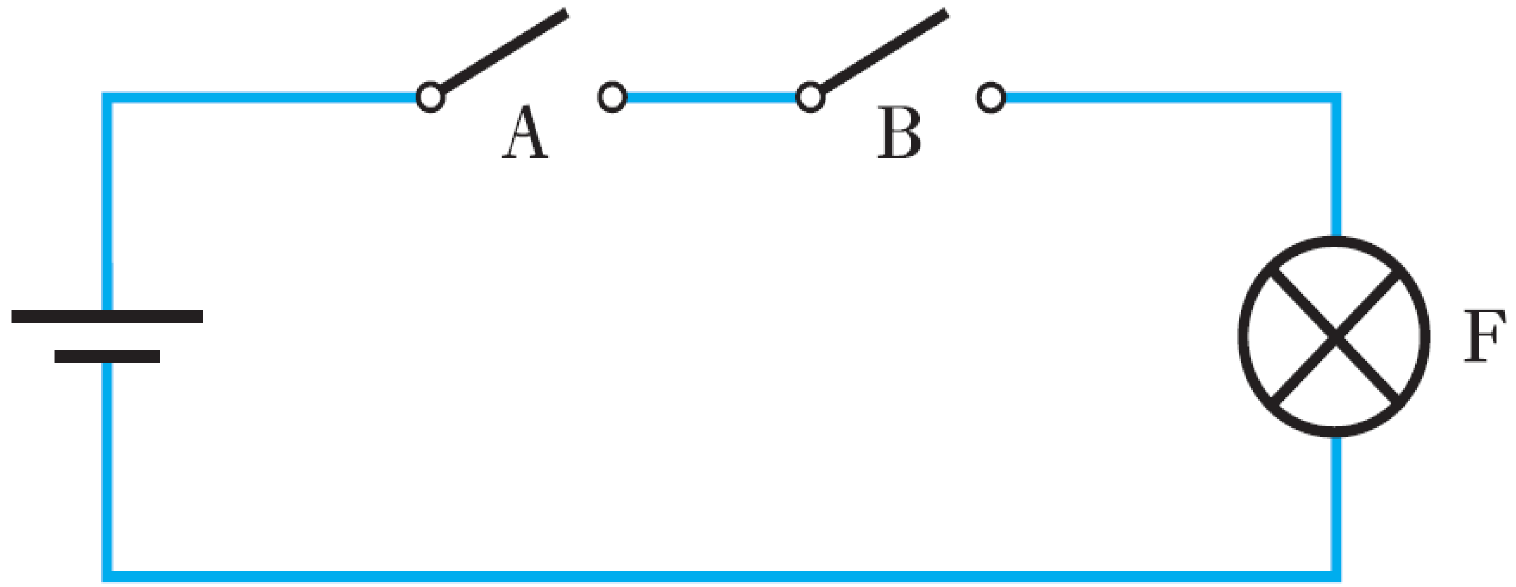
AND

A	B	$x = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

“AND” Logic Gate

- In an electric circuit, **AND** operation is equivalent to **two switches in series**
- The lamp **F lights** (i.e. F = 1) only when **both switches are closed** (closed switch is represented by logic level 1)

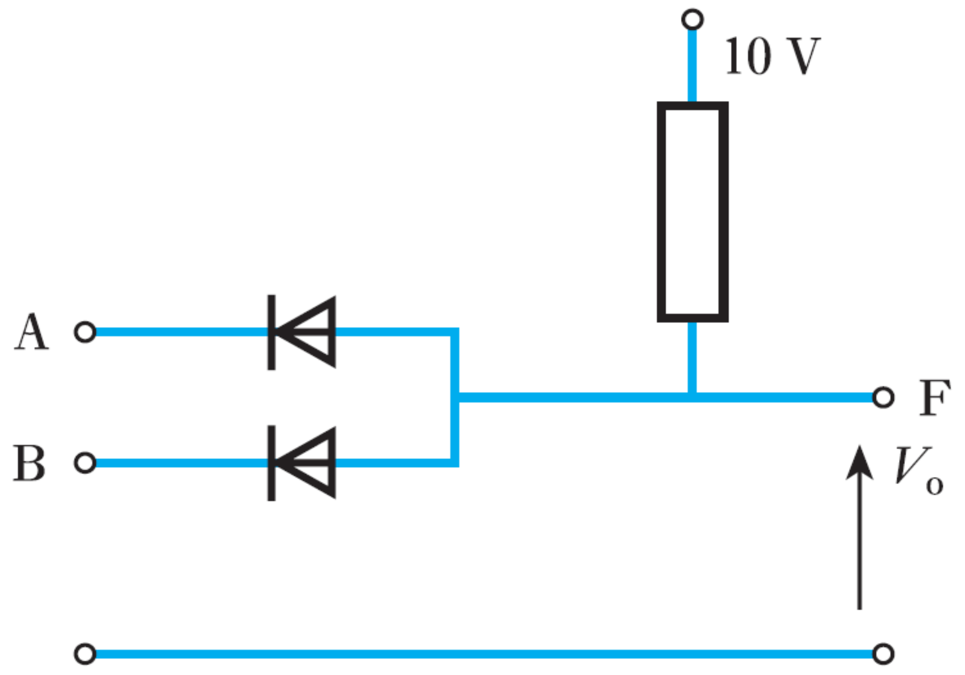
i.e. **F = A. B**



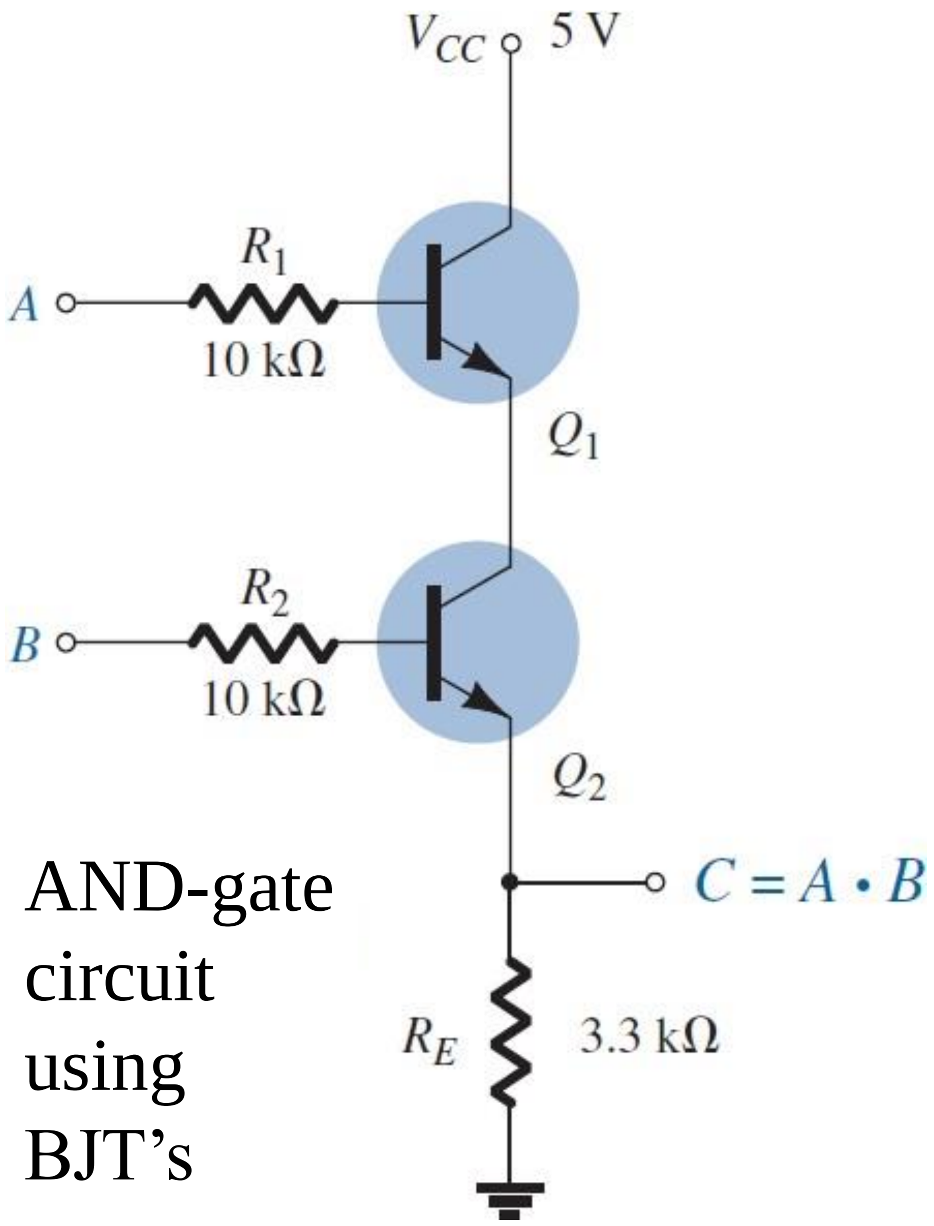
“AND” Logic Gate

- If zero voltage (logic level 0) is applied to either A or B or both, then that respective diode(s) is forward-biased (i.e. short-circuit), resulting in the output voltage becoming zero (logic level 0).
- If 10 V (logic level 1) is applied to both A and B then the both diodes are reverse-biased (i.e. open-circuit) and the output voltage rises to the 10 V (logic level 1) supply.

Diode-based
AND-gate circuit



“AND” Logic Gate



AND-gate circuit using BJT’s

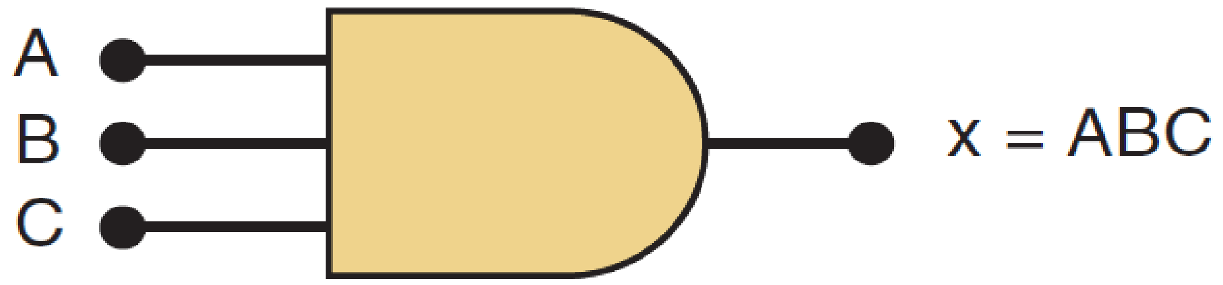
If no (zero) voltage (logic level 0) is applied to either input *A* or *B* or both, then the output voltage V_o is also zero (logic level 0) (at least one transistor is in cutoff i.e. open circuit so no current flows)

If, however, a positive voltage of, say, 5 V (logic level 1) is applied to both inputs then both transistors turn on (saturation mode), and the output voltage *C* is also ~5 V (logic level 1)

"AND" Logic Gate

For three inputs, the AND-gate circuit symbol and the truth table is given below. The Boolean expression is x = A . B . C

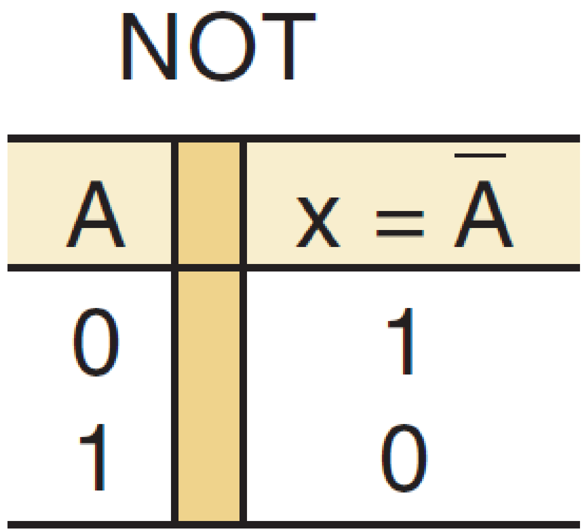
A	B	C	x = ABC
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



"NOT" Logic Gate

- The **NOT** operation is unlike the OR and AND operations because it is performed on a **single input** variable
- If the variable A is subjected to the NOT operation (also called inversion or complementation), the result x is expressed as $x = \bar{A}$ or $x = A'$

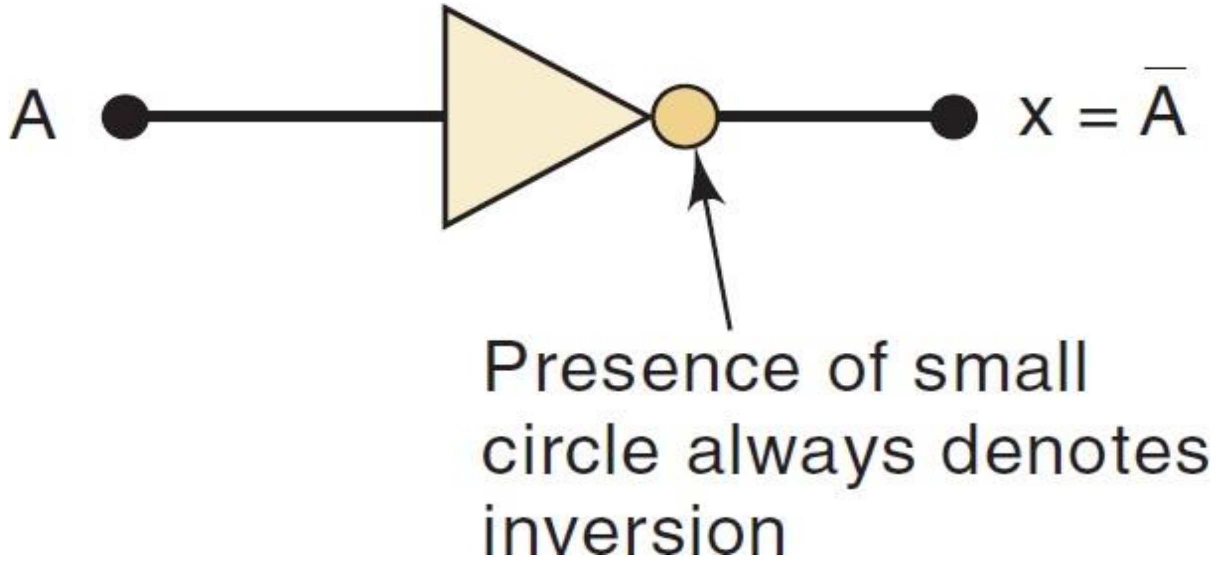
Truth Table



NOT

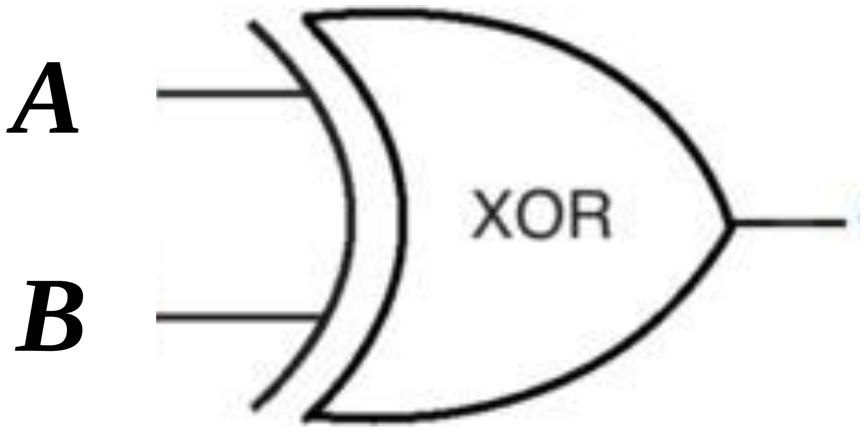
A	$x = \bar{A}$
0	1
1	0

NOT Circuit Symbol



“EX-OR/XOR” Logic Gate

- Two logic inputs, **A** and **B**, are combined using the Exclusive-OR (EX-OR or simply XOR) operation (denoted by symbol ⊕) to produce the output **F**
- The truth table shows that **F** is a logic 1 only when **A** and **B** are different



$$F = A \oplus B$$

Circuit Symbol

Truth Table

A	**B**	**F**
0	0	0
0	1	1
1	0	1
1	1	0

“EX-OR/XOR” Logic Gate

- Example: **The bus goes to either A or B but not to both**. The success (truth) of the bus going to one or other can be represented by F; thus ‘F’ occurs when the bus goes **only to A (and not B) or only to B (and notA)**

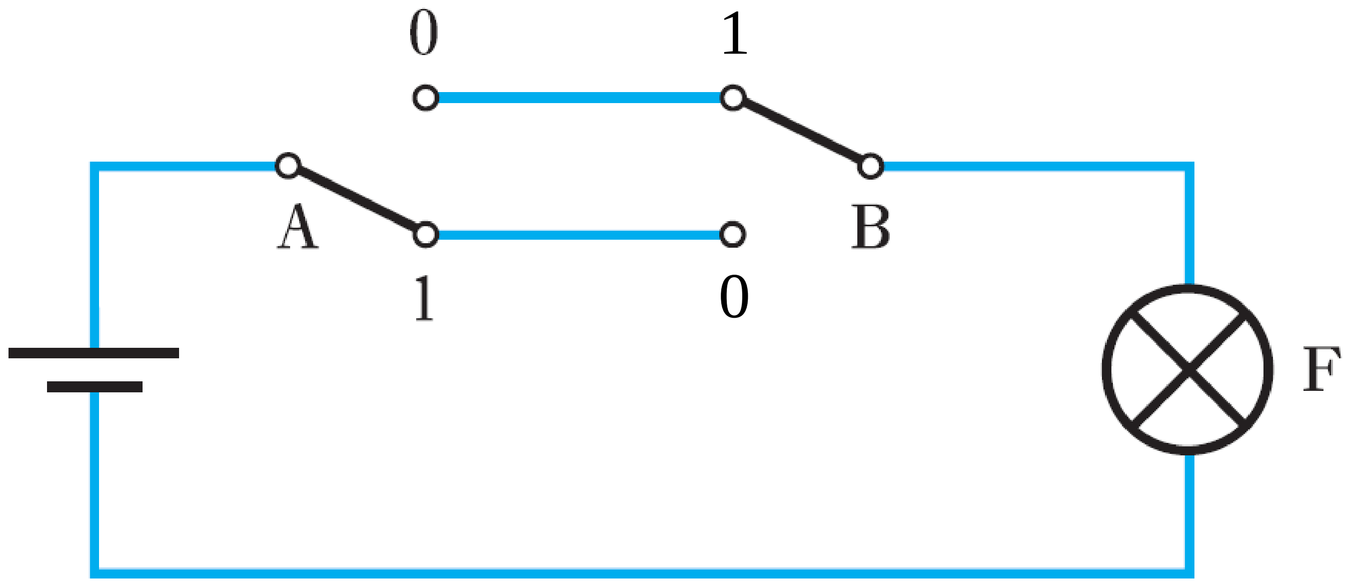
- The Boolean expression for the EX-OR operation is
  $\mathbf{F = A \oplus B = (A . B) + (A . B)}$
  (in terms of the fundamental gates OR, AND, NOT)

A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

“EX-OR/XOR” Logic Gate

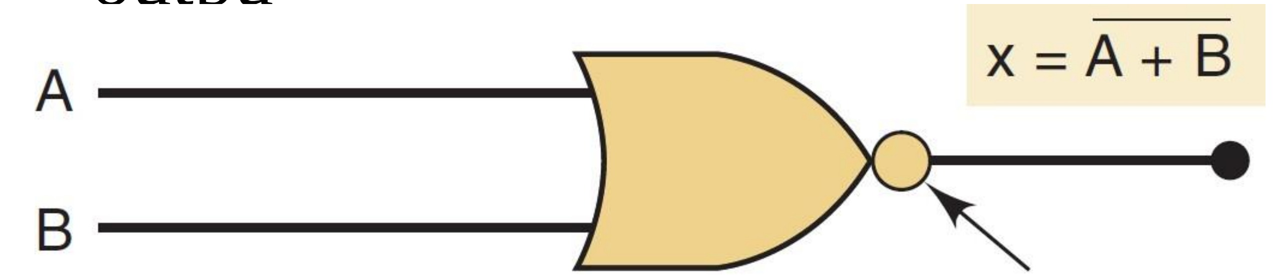
- The **XOR** operator is typically represented in an electric circuit by the **two-way switching** associated with a stair light

- The lamp **F lights** (i.e. F = 1) **only** when **one of the switches is ON** (1) AND **the other is OFF** (0) i.e. **F = A ⊕ B = (A . B) + (A. B)**



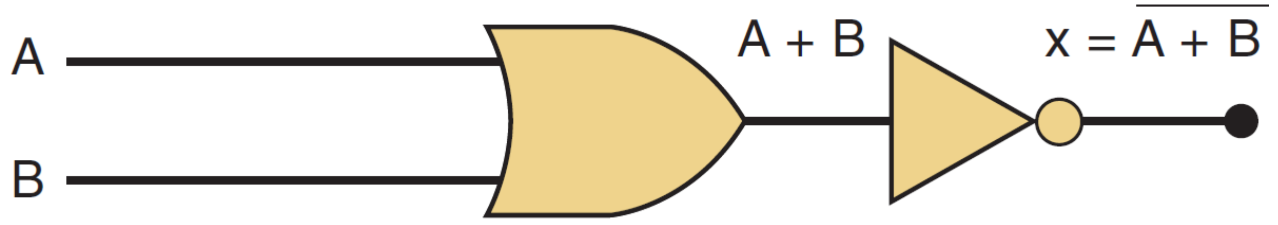
“NOR” LOGIC GATE

- NOR gate operates like an OR-gate followed by an Inverter (NOT gate)
- The NOR-gate output ($x = \overline{A + B}$) is the exact inverse of the OR-gate outpu



Circuit Symbol

Equivalent Circuit

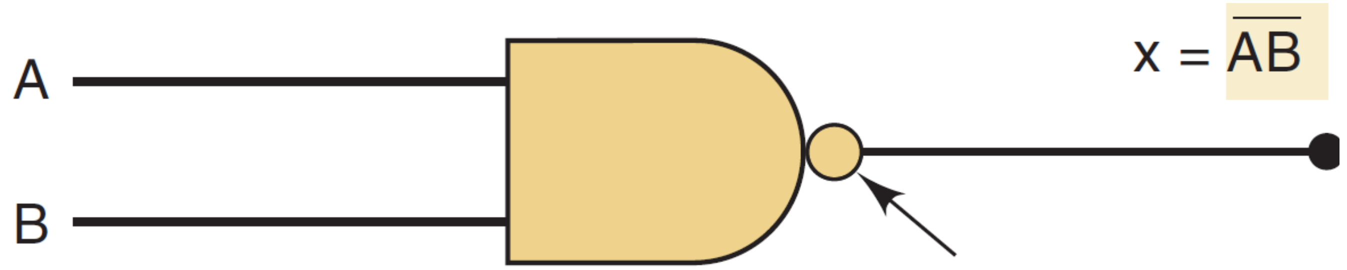


Truth Table

		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

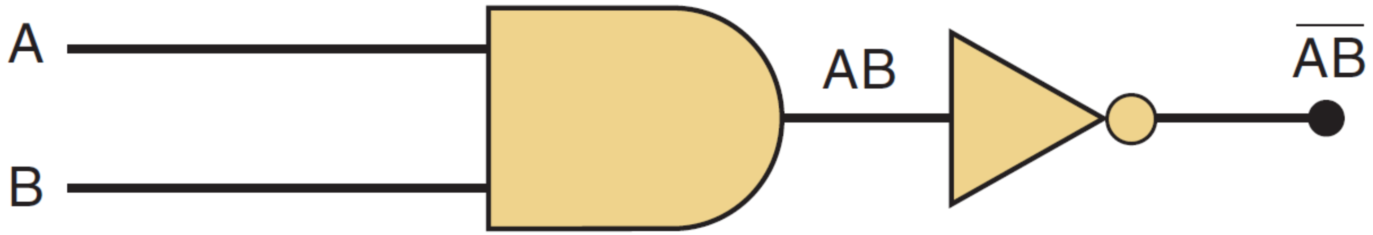
“NAND” Logic Gate

- NAND-gate operates like an AND-gate followed by an Inverter (NOT gate)
- The NAND-gate output ($x = \overline{\mathbf{A}.\ \mathbf{B}}$) is the exact inverse of the AND-gate



Circuit 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

OPERATOR PRECEDENCE

- If an **expression contains both AND and OR operations**, the **AND** operations are **performed first**, unless there are parentheses in the expression, in which case the **operation inside the parentheses is to be performed first**

- Example: F = A . B + C

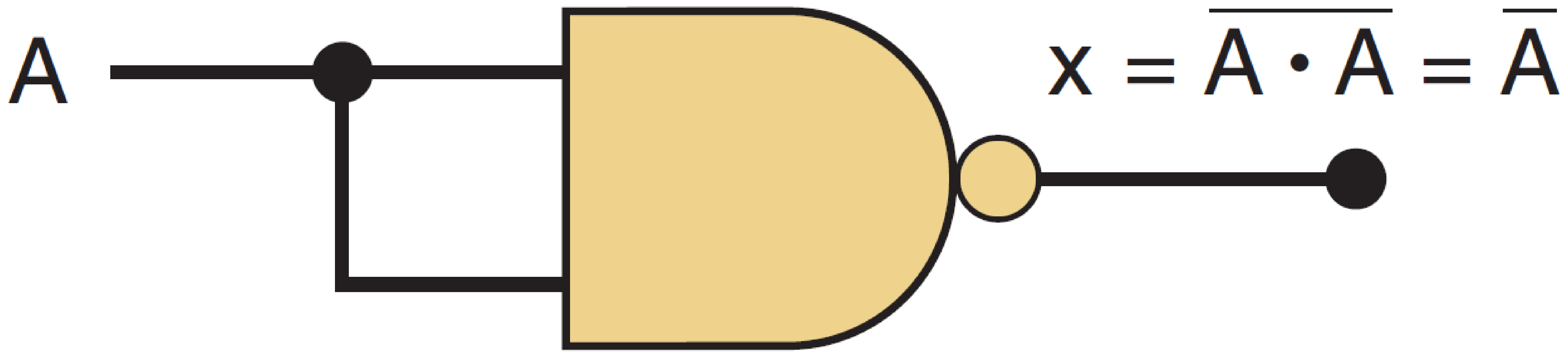
  First AND operation is performed between A and B

  The result of above operation is then ORed with C

UNIVERSAL GATES: NAND/NOR

- All Boolean expressions consist of various combinations of the basic operations of OR, AND, and NOT (or INVERT).
- Any expression can be implemented using combinations of OR gates, AND gates, and INVERTERs (NOT gates).
- NAND and NOR gates, in the proper combination, can be used to perform each of the Boolean operations OR, AND, and INVERT (NOT)
- Thus, it is **possible to implement any logic expression** using **NAND** or **only NOR gates** and no other type of gate
- Thus, **NAND and NOR** gates are called "**universal gates**"
- Easier/simpler to standardize by using a single type of gate. However, the number of gates that need to be used increases.

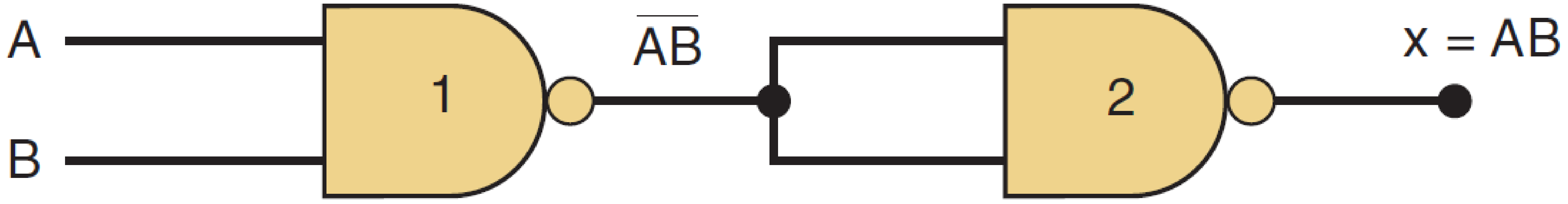
NOT Operation Using NAND Gate



$x = \overline{A \cdot A} = \overline{A}$

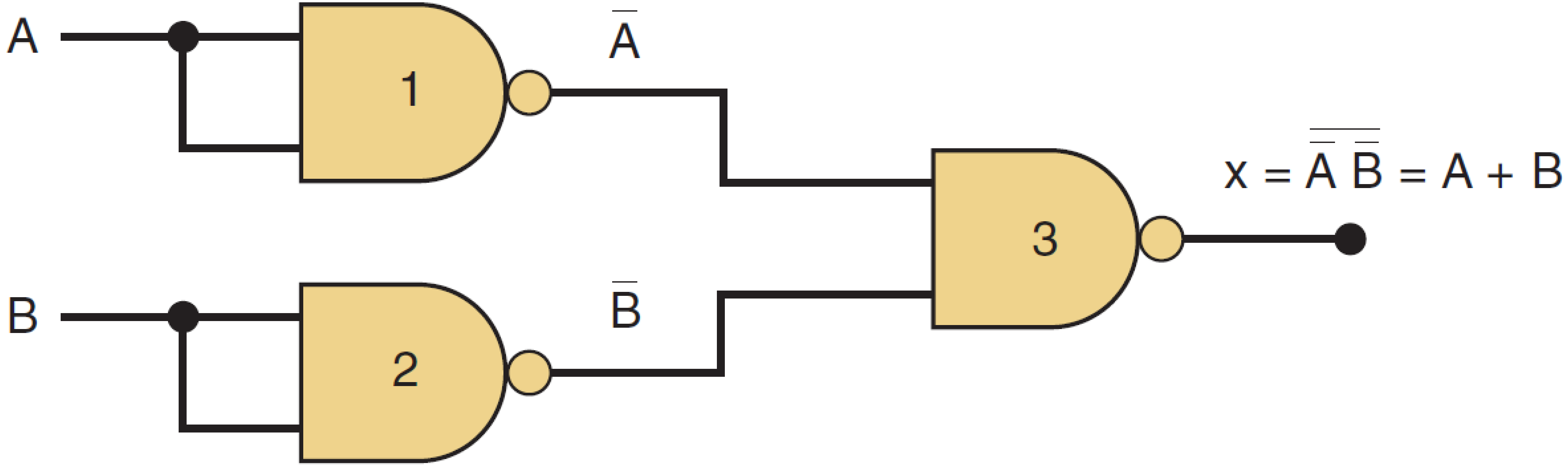
AND Operation Using NAND Gates

Two NAND gates are required to perform the operation of one AND gate

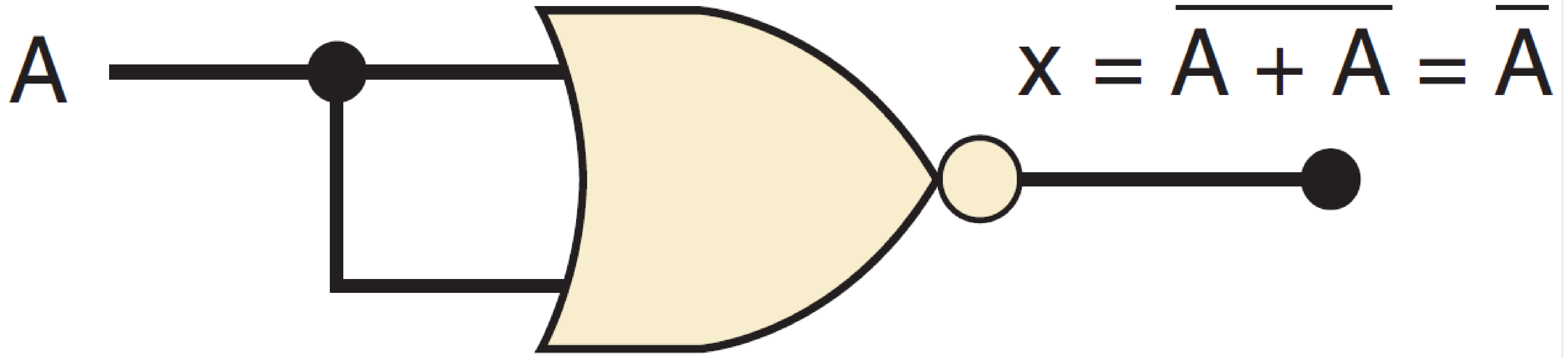


OR Operation Using NAND Gates

Three NAND gates are required to perform the operation of one OR gate



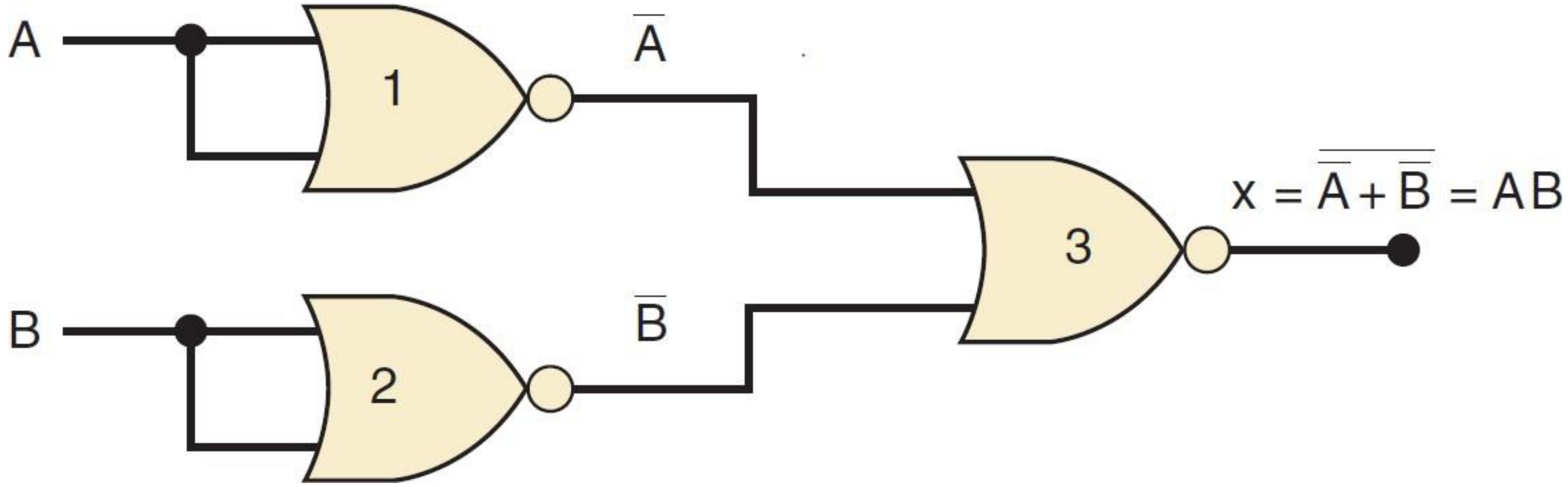
NOT Operation Using NOR Gate



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

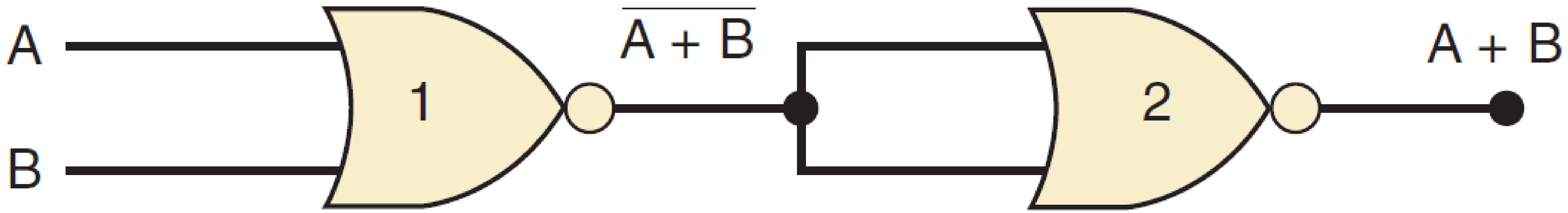
AND Operation Using NOR Gates

Three NOR gates are required to perform the operation of one AND gate



OR OPERATION USING NOR GATES

Two NOR gates are required to perform the operation of one OR gate



IC Technology-Terms

Datasheet

A printed specification giving details of the pin configuration, electrical properties, and mechanical profile of an electronic device.

Design Specifications

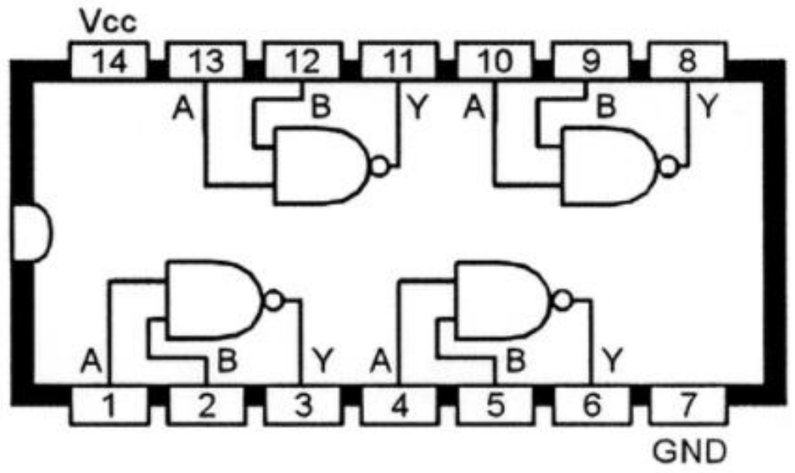
A detailed description, especially one providing information needed to make, build, or produce something.

Libraries

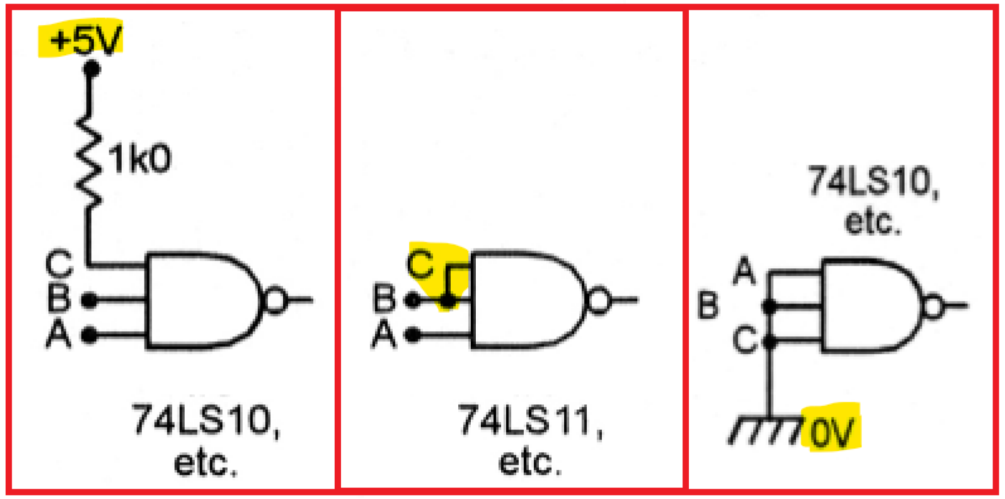
a collections of descriptions of commonly used hardware circuits that can be used as modules in a design file

Logic Gates IC

Device	Type	Description
4011B	CMOS	Quad 2-input NAND gate
4093B	CMOS	Quad 2-input Schmitt NAND gate
74LS00	LS TTL	Quad 2-input NAND gate
74HC00	CMOS	Quad 2-input NAND gate
74LS132	LS TTL	Quad 2-input Schmitt NAND gate
74HC132	CMOS	Quad 2-input Schmitt NAND gate
74LS10	LS TTL	Triple 3-input NAND gate
4023B	CMOS	Triple 3-input NAND gate
74LS20	LSTLL	Dual 4-input NAND gate
4012B	CMOS	Dual 4-input NAND gate
74LS30	LS TTL	8-input NAND gate
4068B	CMOS	8-input NAND gate
74HC133	CMOS	13-input NAND gate

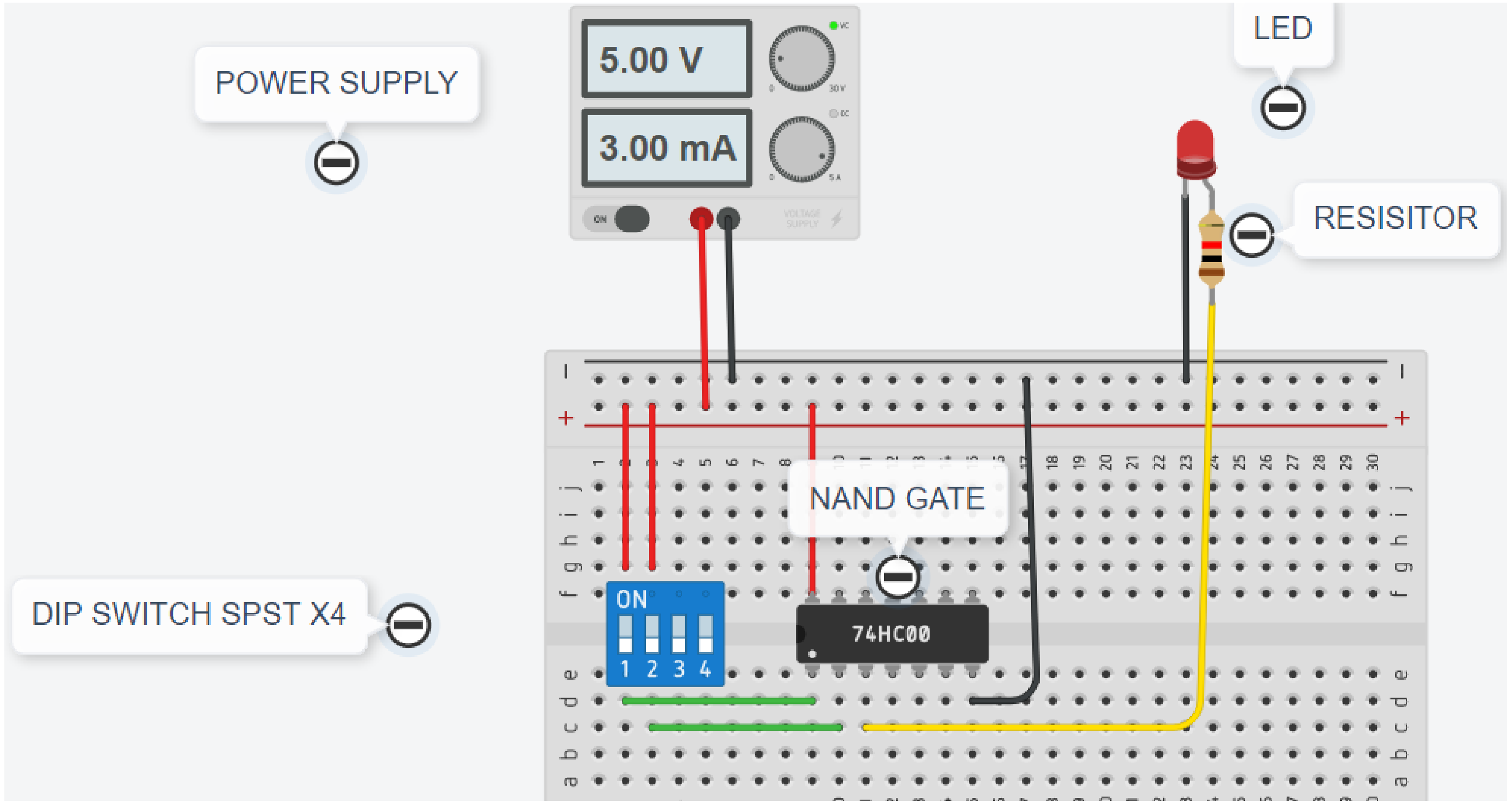


Functional diagram of the 74LS00 or 74HC00 quad two-input NAND gate IC.

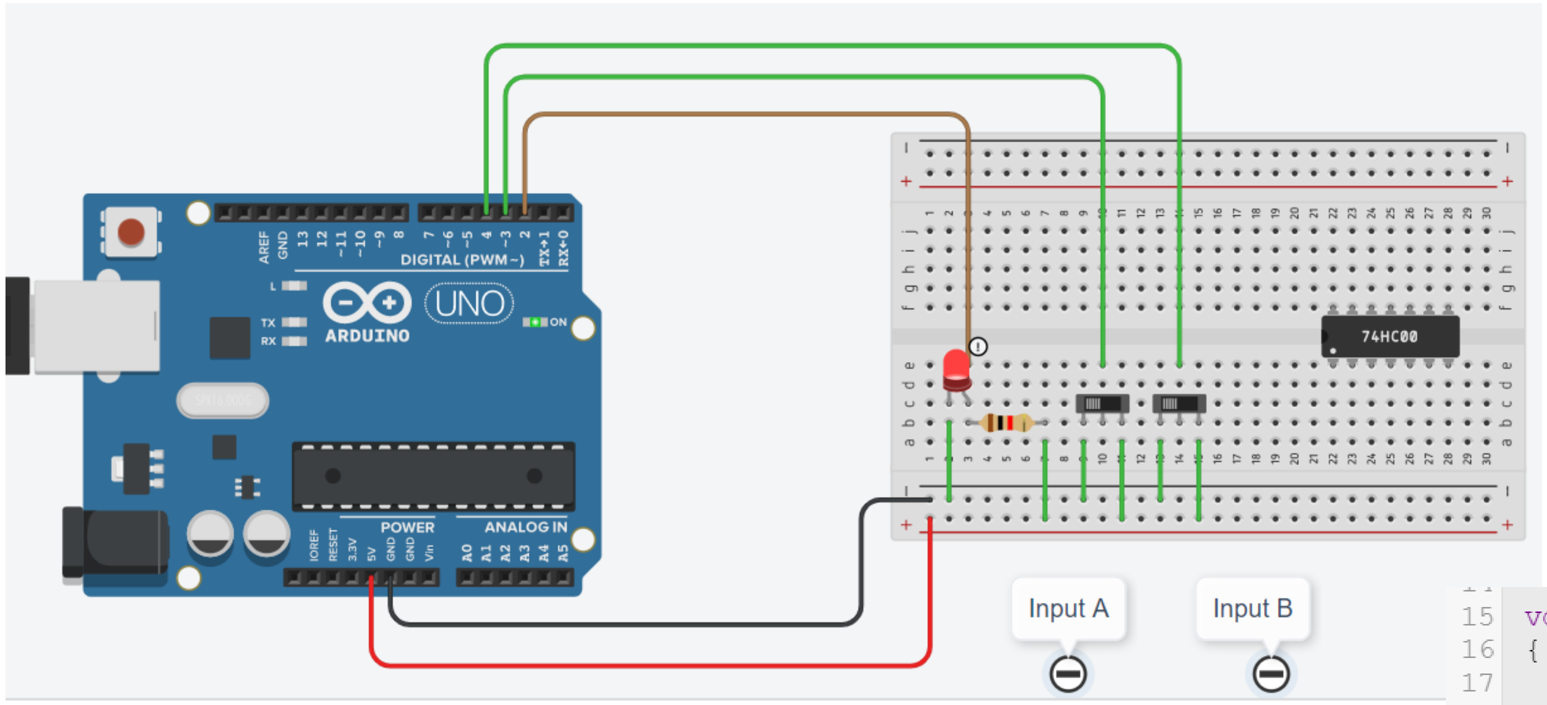


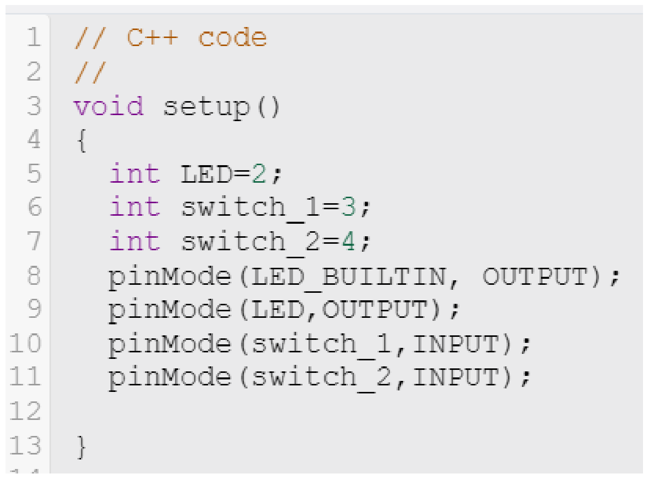
Basic method of disabling unwanted TTL NAND gate inputs

TinkerCAD-Explore NAND gate (Hardware)



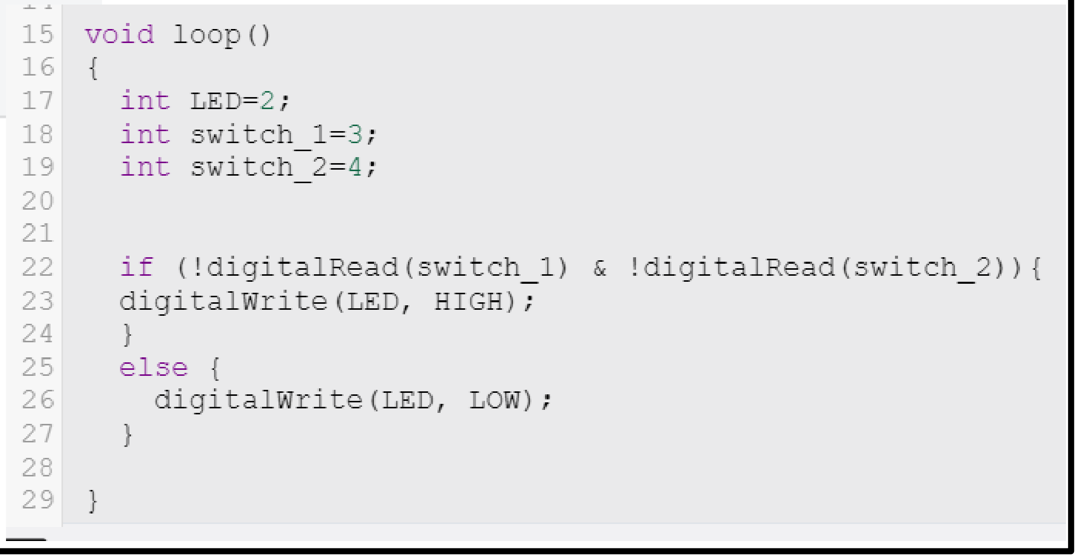
TinkerCAD-Explore NOR gate (Hardware + Software library)





```
// C++ code
//
void setup()
{
  int LED=2;
  int switch_1=3;
  int switch_2=4;
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(LED,OUTPUT);
  pinMode(switch_1,INPUT);
  pinMode(switch_2,INPUT);

}
```



```
void loop()
{
  int LED=2;
  int switch_1=3;
  int switch_2=4;


  if (!digitalRead(switch_1) & !digitalRead(switch_2)){
  digitalWrite(LED, HIGH);
  }
  else {
    digitalWrite(LED, LOW);
  }

}
```

Solved Examples

1. **Write the Boolean expression for a 3-input NAND gate.**

Sol. The output Y of the NAND gate with inputs A, B, and C is $Y = \overline{A.B.C}$

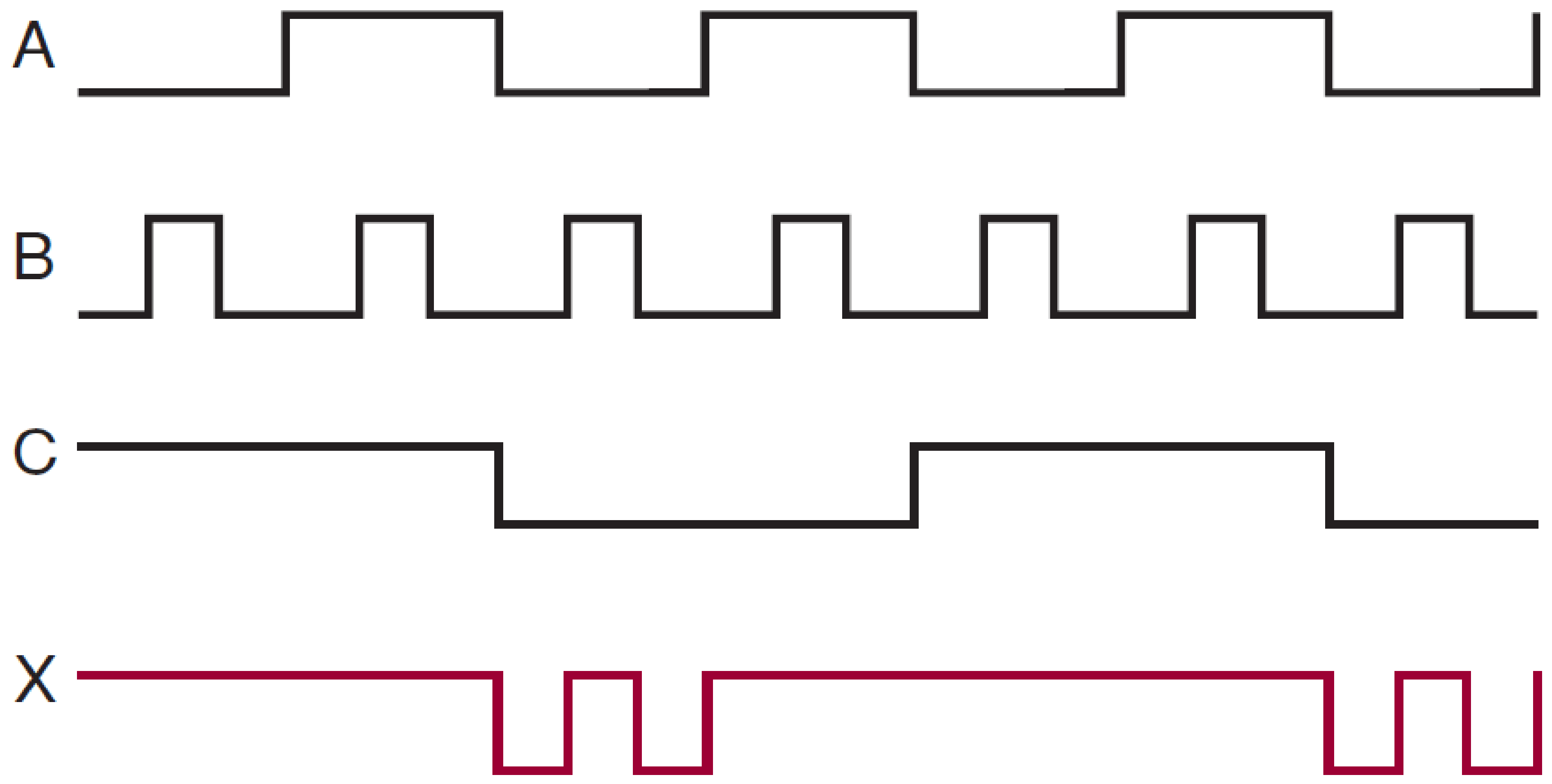
2. **Which logic gate generates a HIGH output when an odd number of inputs are HIGH?**

Sol. Exclusive-OR gate (EX-OR/XOR Gate)

Solved Examples

3. For the given input signals A, B, C, sketch the output for an OR gate.

Sol.



The output X = A + B + C is high when any of the inputs is high.

SOLVED EXAMPLES

4.How many different sets of input conditions will produce a HIGH output from a five-input OR gate?

Sol. The total number of sets of input conditions are $2^5 = 32$. The output is low only when all the inputs are low. This happens in only one case. Therefore, 31 different sets of input conditions will produce a HIGH output.

5. Write the Boolean expression for a 4-input XOR gate.

Sol. The output *Y* of the XOR gate with inputs *A*, *B*, *C* and *D* is

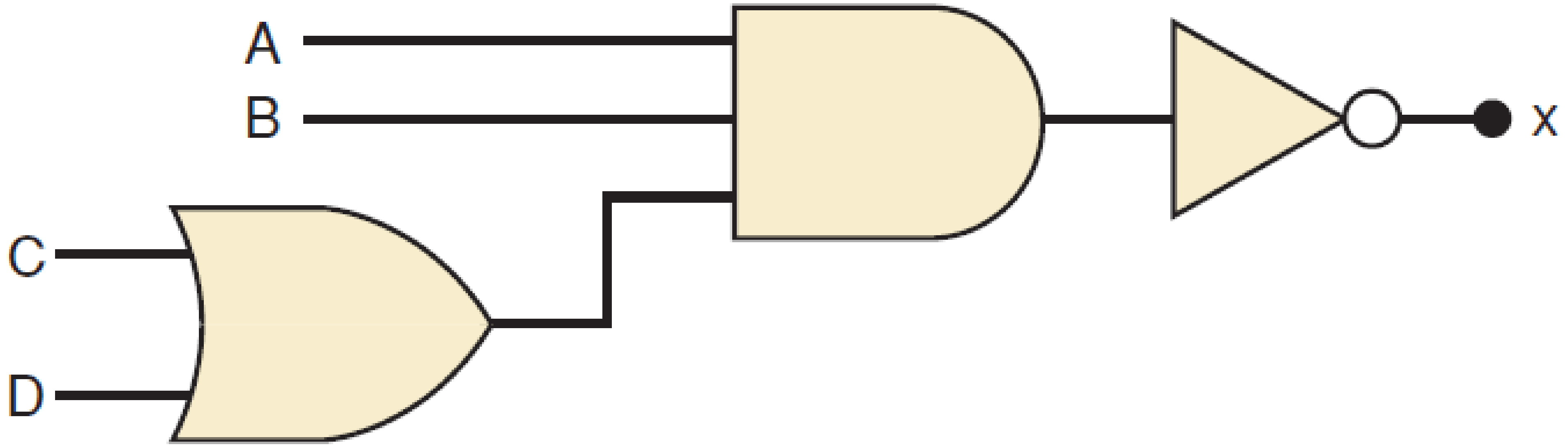
$$Y = A \oplus B \oplus C \oplus D$$

SOLVED EXAMPLES

6. For the given expression, draw the corresponding logic circuit, using AND, OR and NOT gates.

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

Sol.

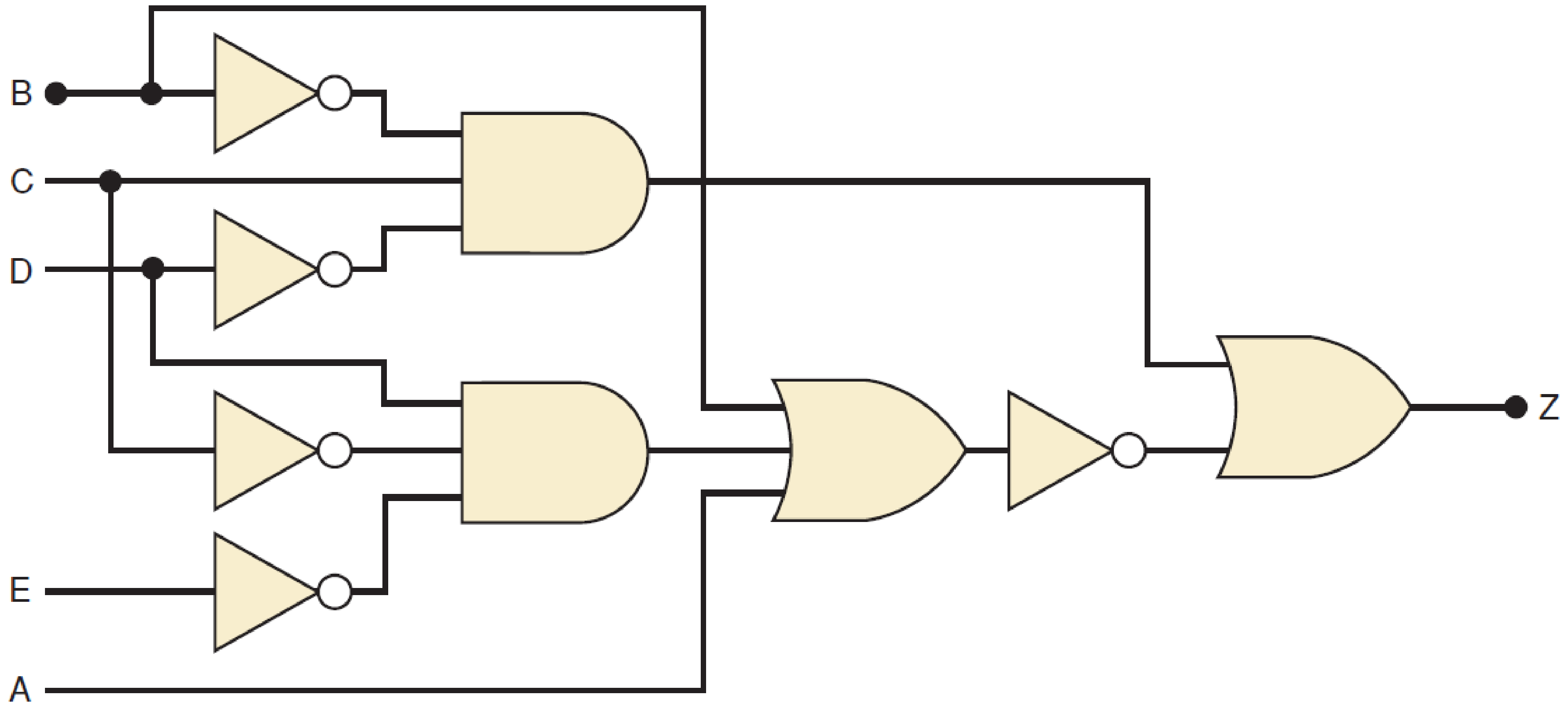


SOLVED EXAMPLES

7. For the given expression, draw the corresponding logic circuit, using AND, OR and NOT gates.

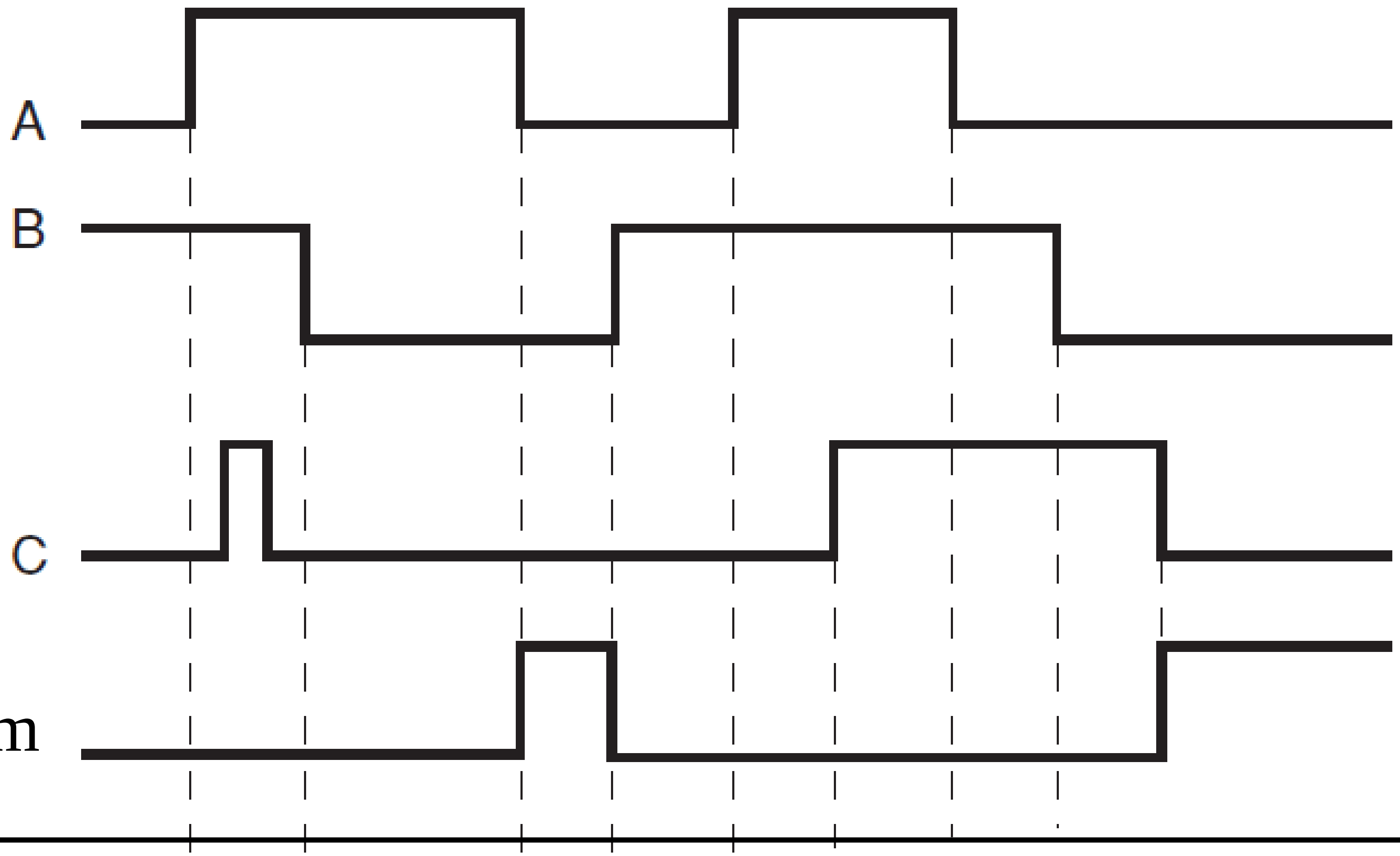
$$z = \overline{A + B + \overline{C}D\overline{E}} + \overline{B}C\overline{D}$$

Sol.



Solved Examples

8. Apply the input waveforms A, B and C given below to a NOR gate, and draw the output waveform.



Sol. The output waveform

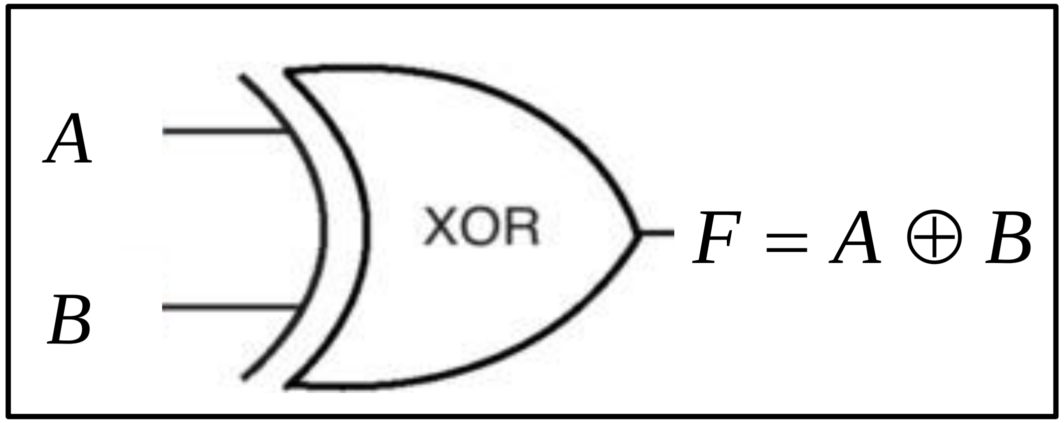
EXERCISES

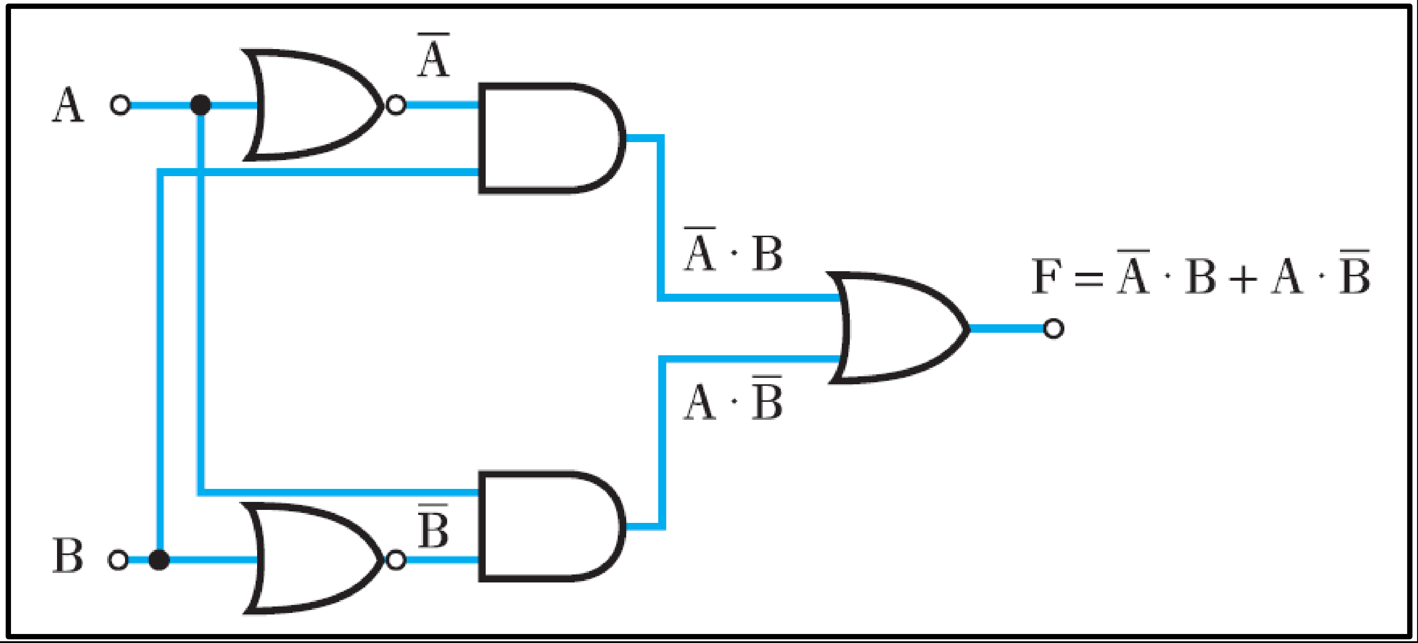
1. Draw a logic circuit, incorporating any gates of your choice, which will produce an output 1 when its two inputs are different. Also draw the same logic circuit incorporating only NOR gates.

EXERCISES

1. **Draw a logic circuit, incorporating any gates of your choice, which will produce an output 1 when its two inputs are different. Also draw the same logic circuit incorporating only NOR gates.**

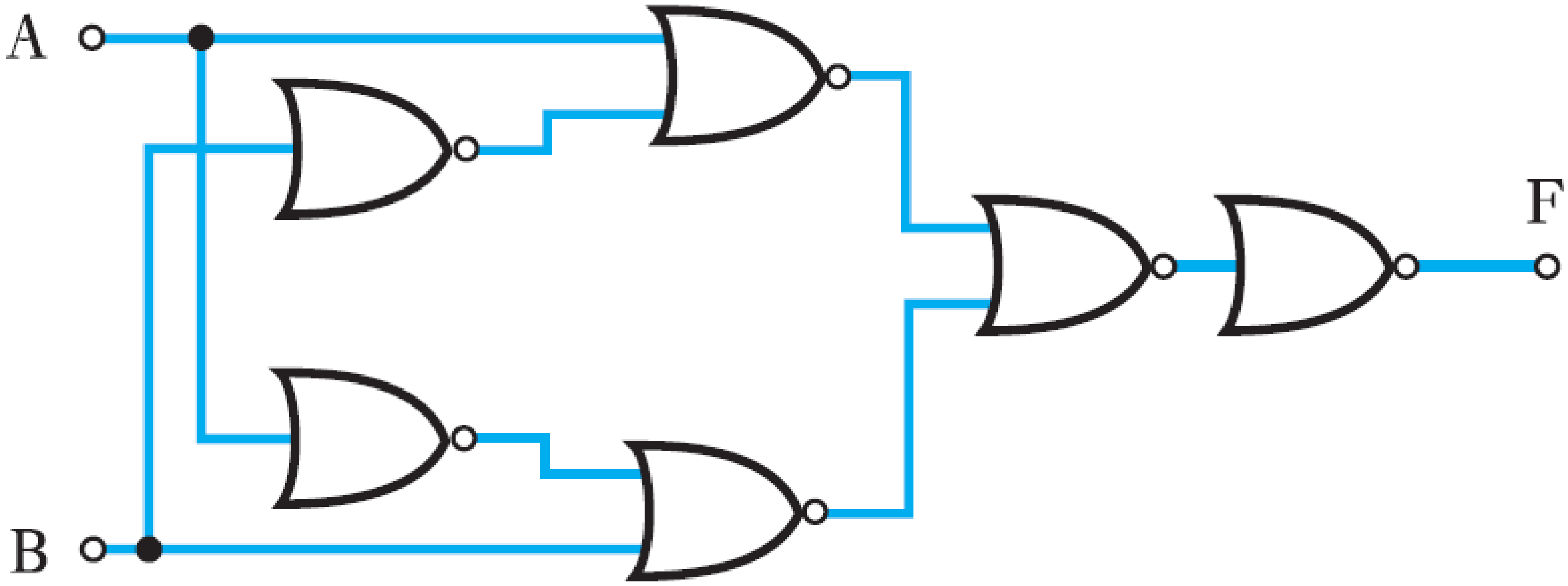
Ans. The required function is $\mathbf{F = A \oplus B = (A . B) + (A. B)}$





EXERCISES

Circuit for $\mathbf{F = A \oplus B = (A . B) + (A. B)}$ using only NOR gates (gates with single input imply that both inputs are same)



EXERCISES

2. **Draw the circuit diagram to implement the expression below**

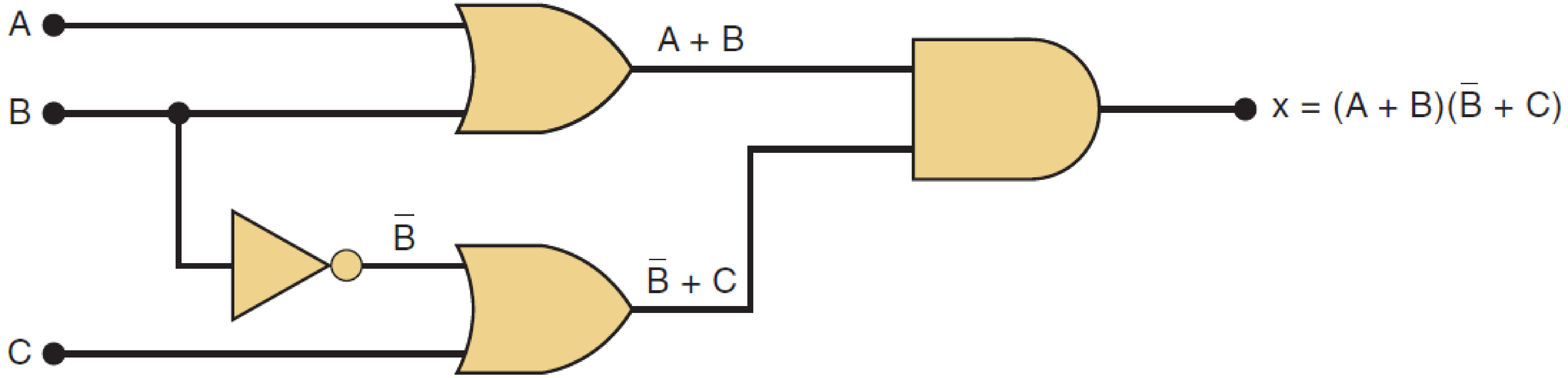
$$x = (A + B)(\overline{B} + C)$$

EXERCISES

2. Draw the circuit diagram to implement the expression below

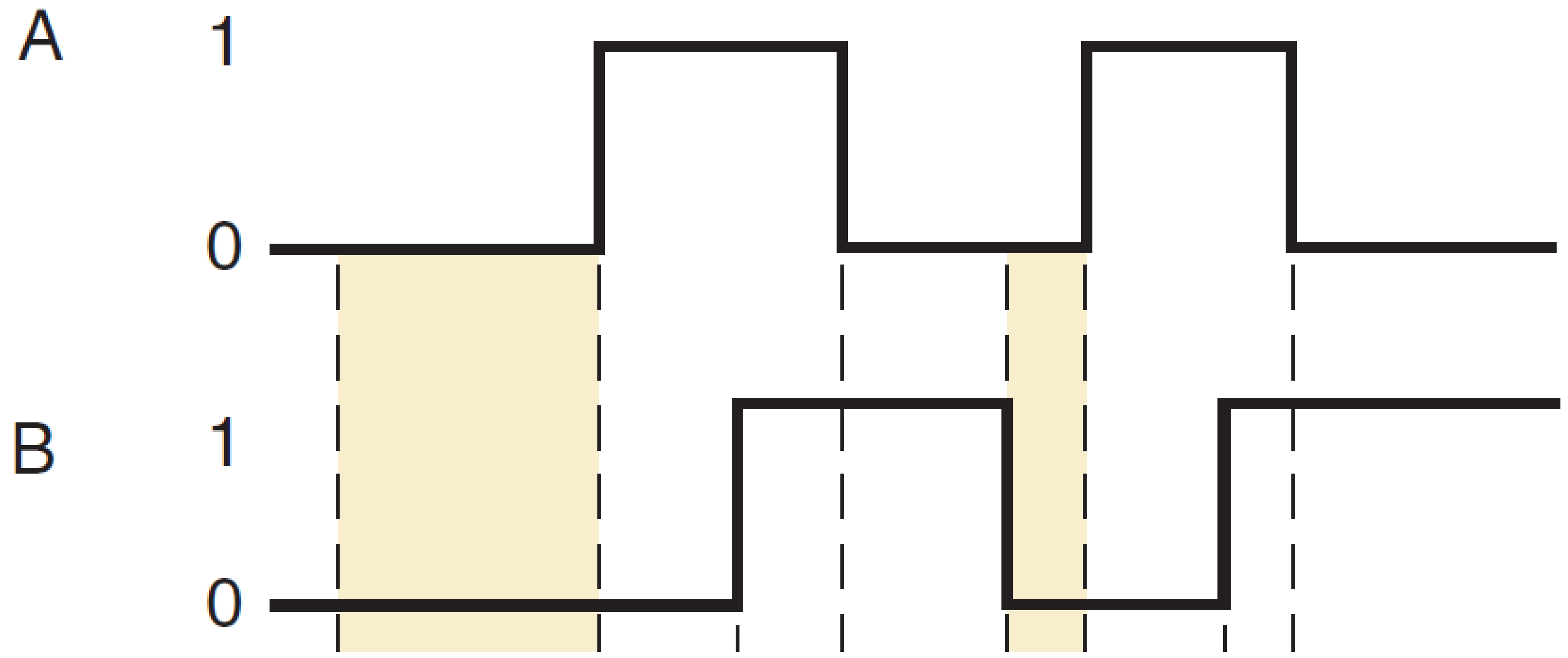
$$x = (A + B)(\overline{B} + C)$$

Ans.



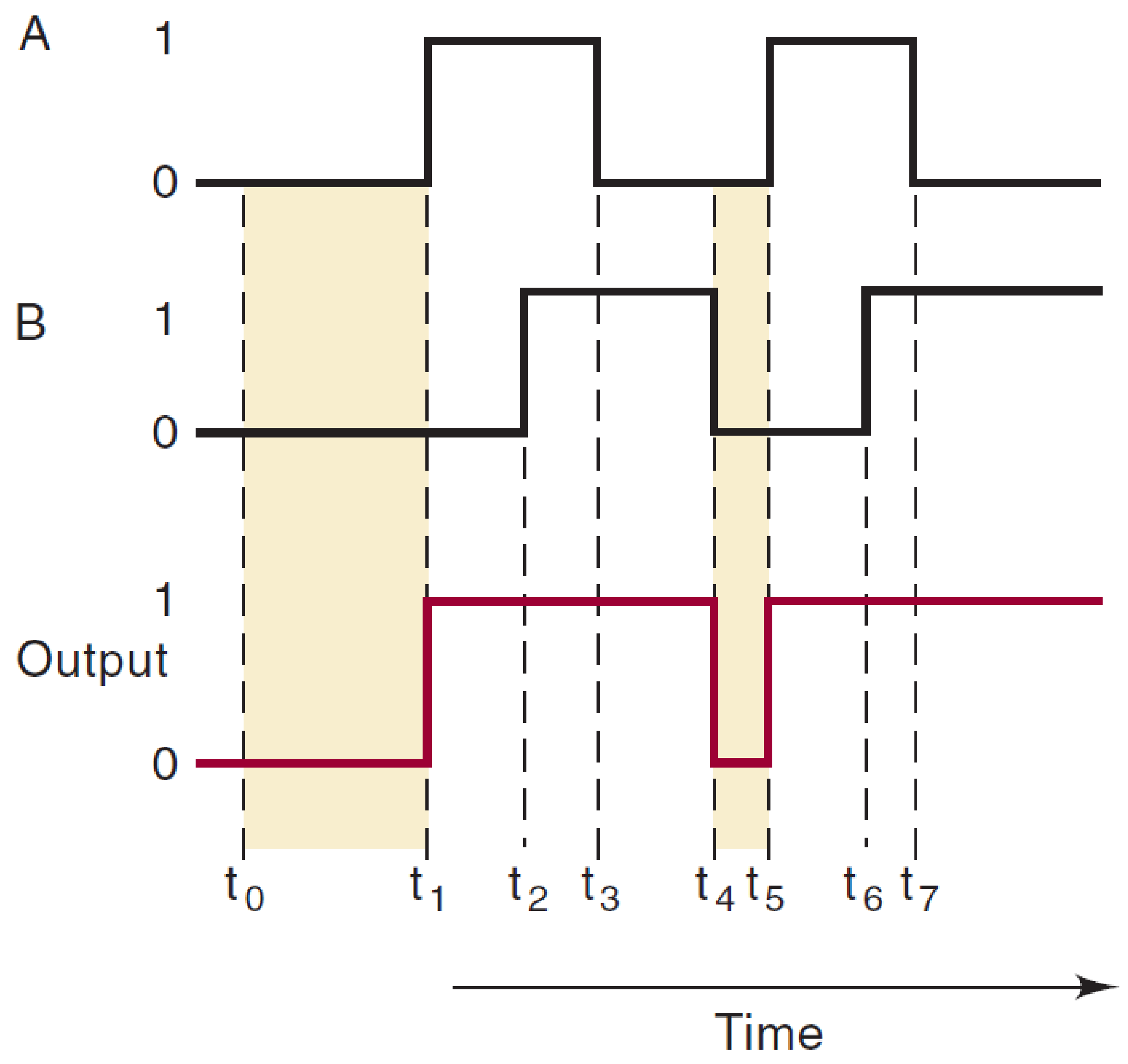
EXERCISES

3. **Determine the output waveform for an OR gate output with inputs A and B varying according to the timing diagrams shown below**



EXERCISES

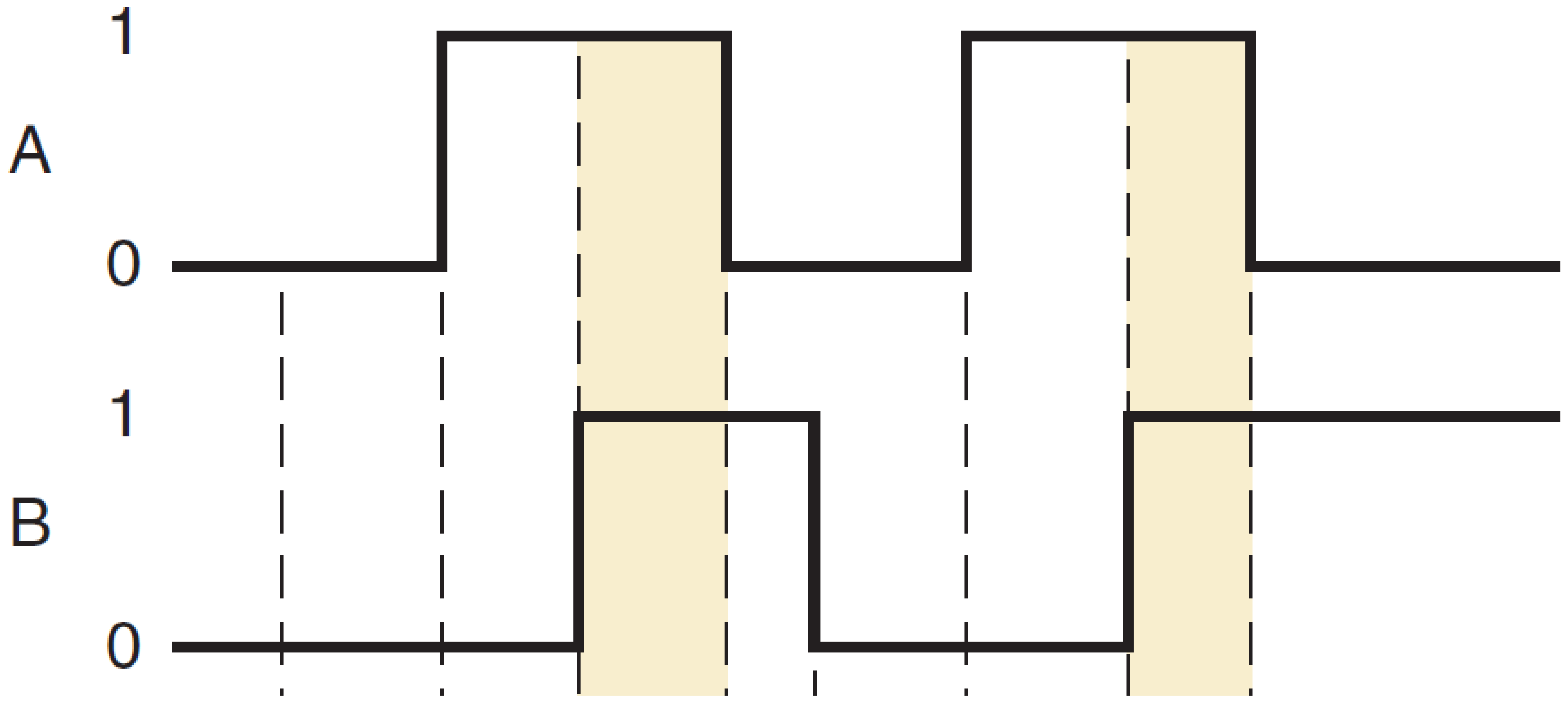
Ans.



A + B

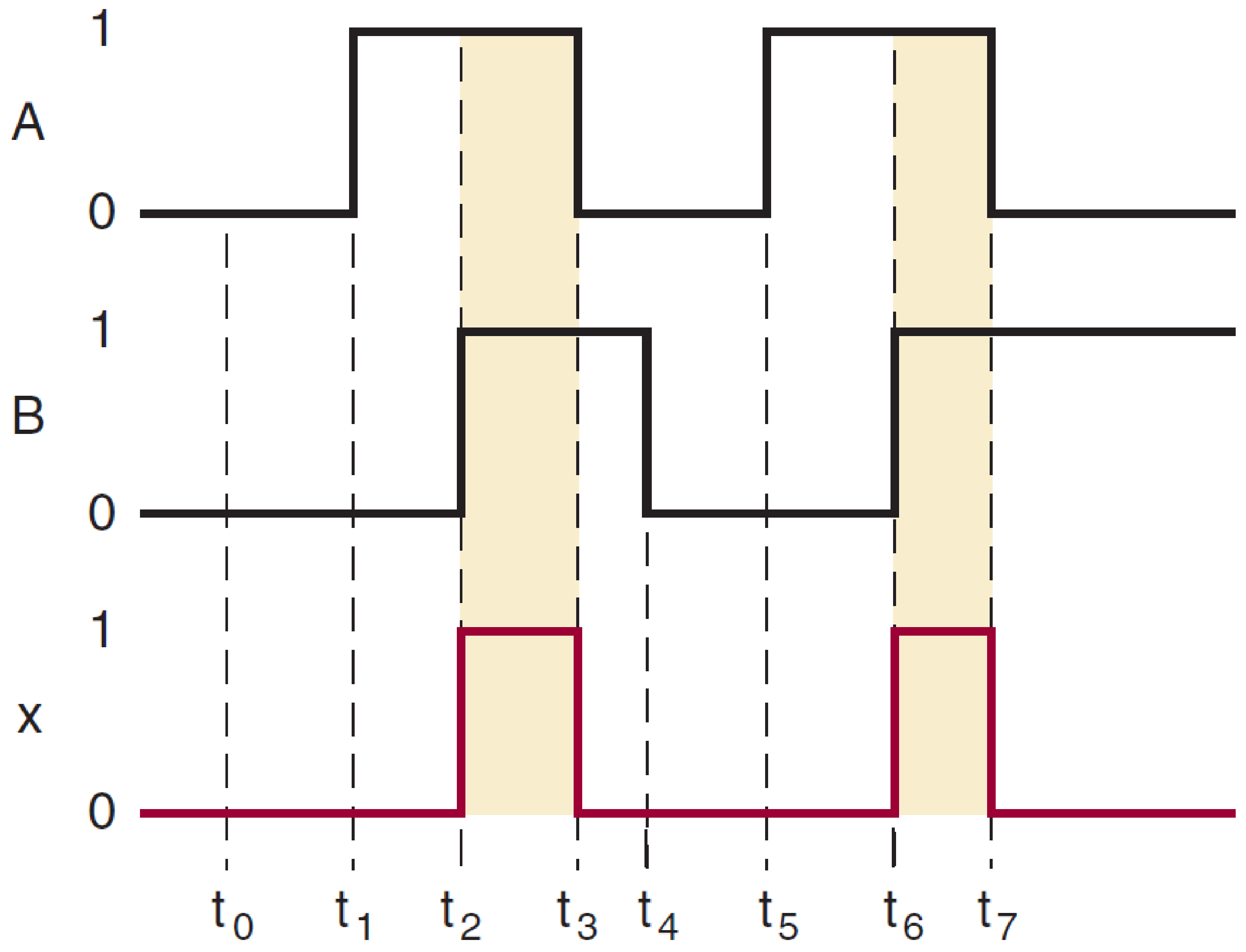
EXERCISES

4. **Determine the output waveform for an AND gate with inputs A and B varying according to the timing diagrams shown below**



EXERCISES

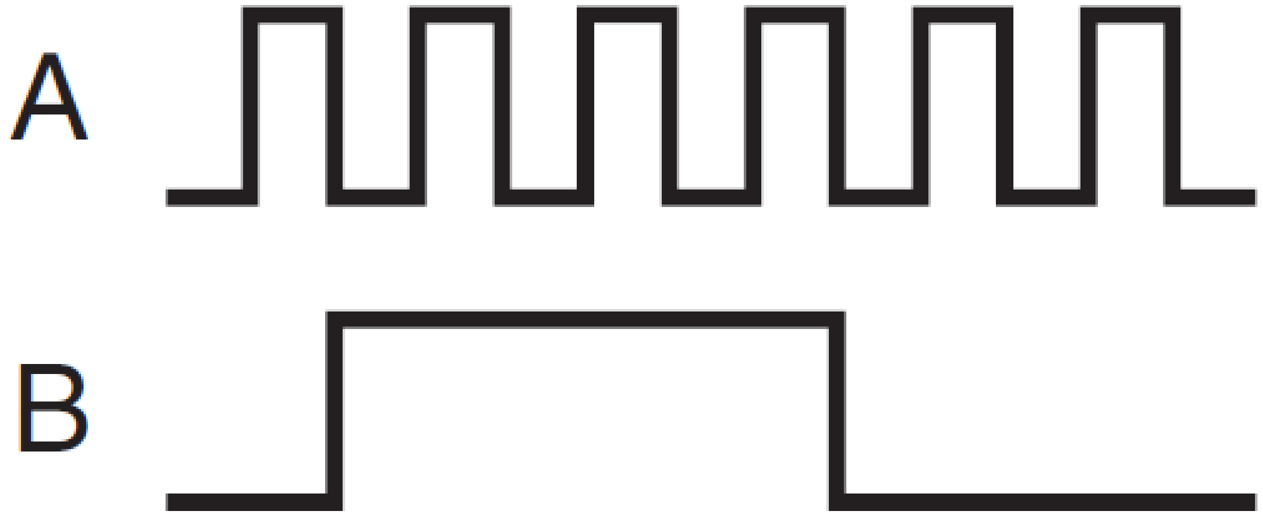
Ans.



A . B

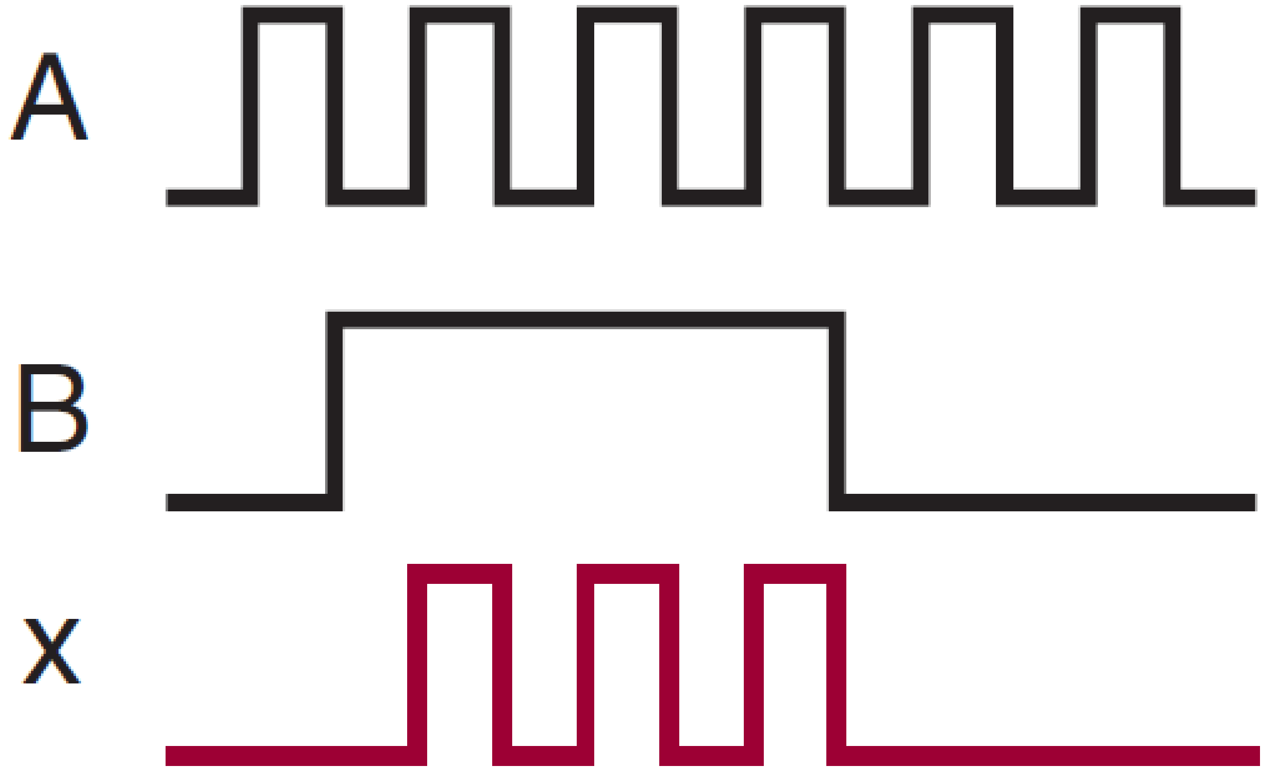
EXERCISES

5. Determine the output waveform for an AND gate with inputs A and B varying according to the timing diagrams shown below



EXERCISES

5. Determine the output waveform for an AND gate with inputs A and B varying according to the timing diagrams shown below



Ans. The output

A . B

Exercises

6. **Draw the circuit diagram to implement the expression**

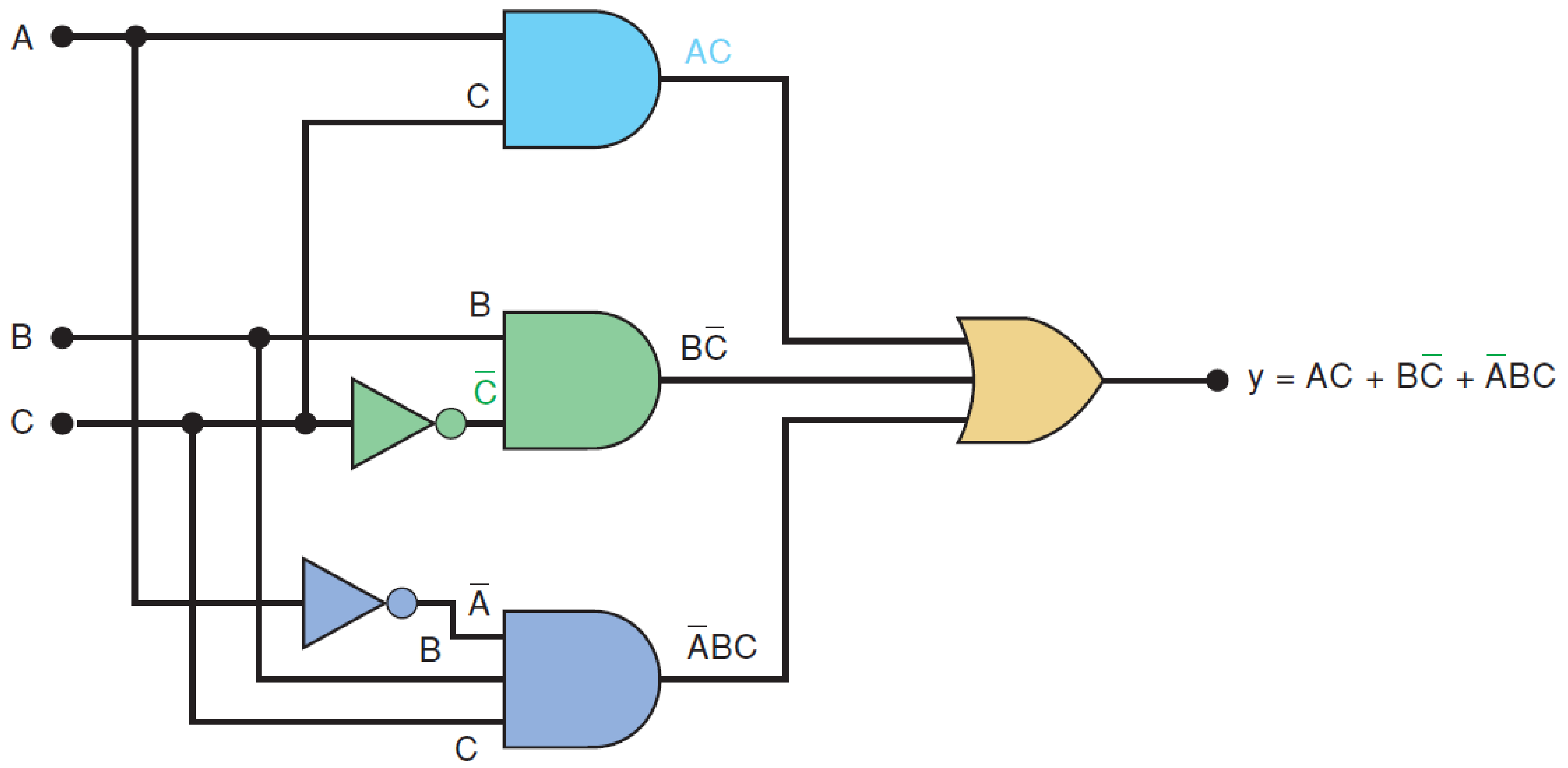
$$x = AC + B\overline{C} + \overline{A}BC$$

EXERCISES

6. Draw the circuit diagram to implement the expression

$$x = AC + B\overline{C} + \overline{A}BC$$

Ans.



EXERCISES

7. **Draw the circuit diagram to implement the expression below, using gates with no more than three inputs**

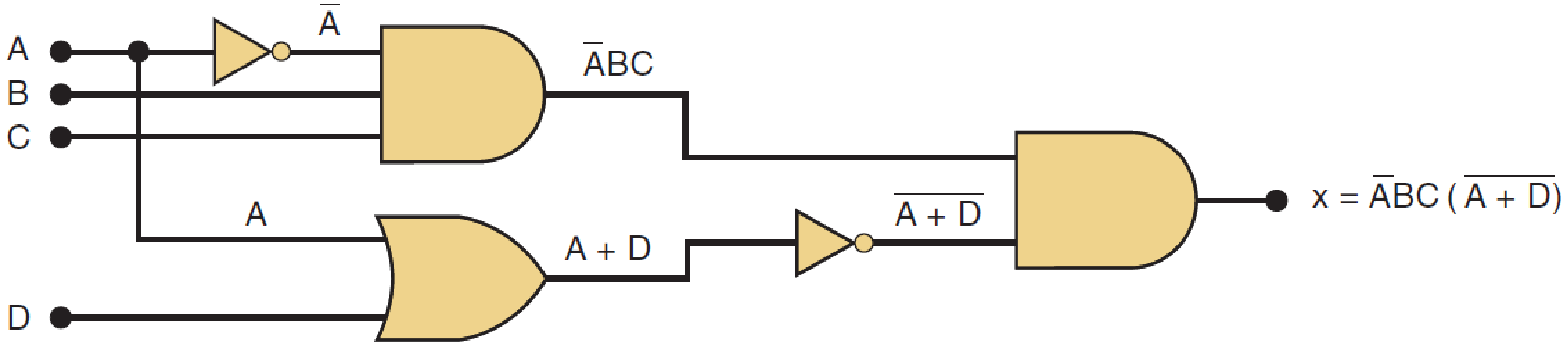
$$x = \overline{A}BC(\overline{A + D})$$

EXERCISES

7. Draw the circuit diagram to implement the expression below, using gates with no more than three inputs

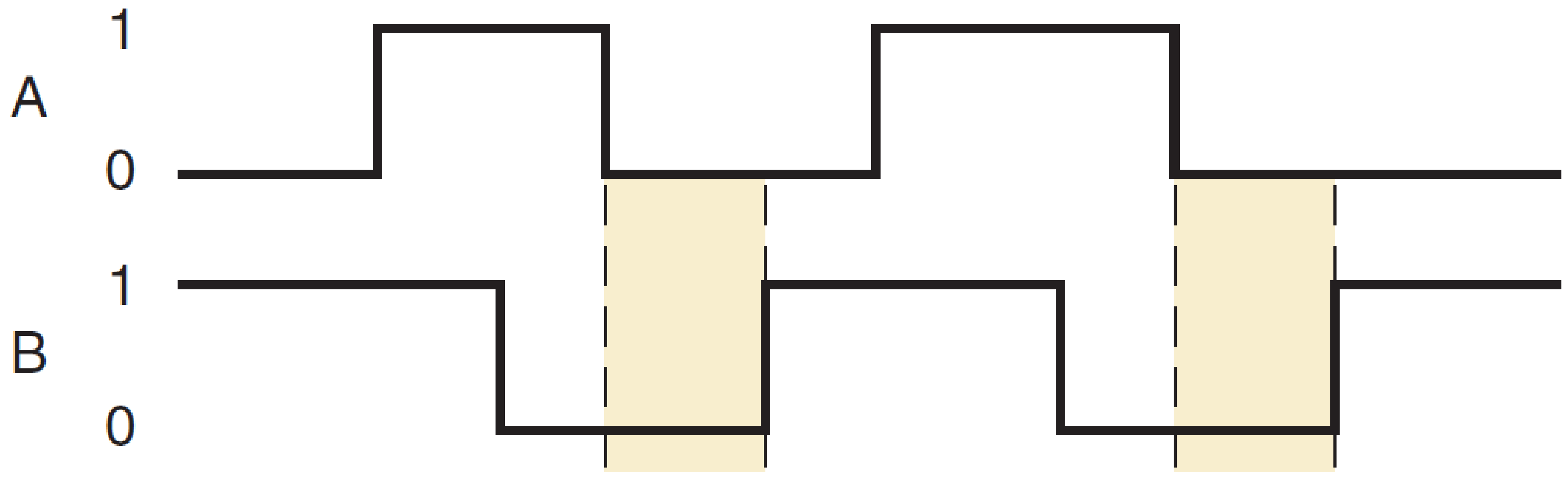
$$x = \overline{A}BC(\overline{A + D})$$

Ans.



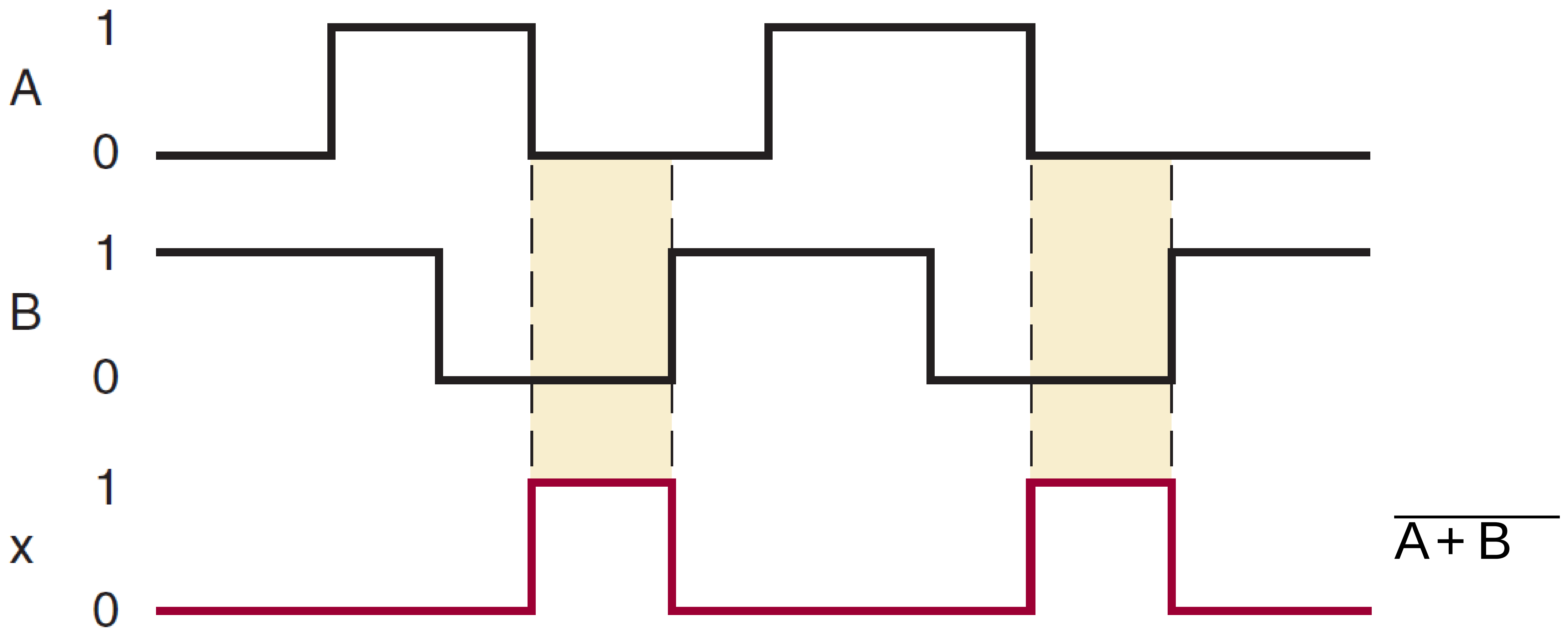
EXERCISES

8. **Determine the waveform at the output of a NOR gate for the input waveforms shown below**



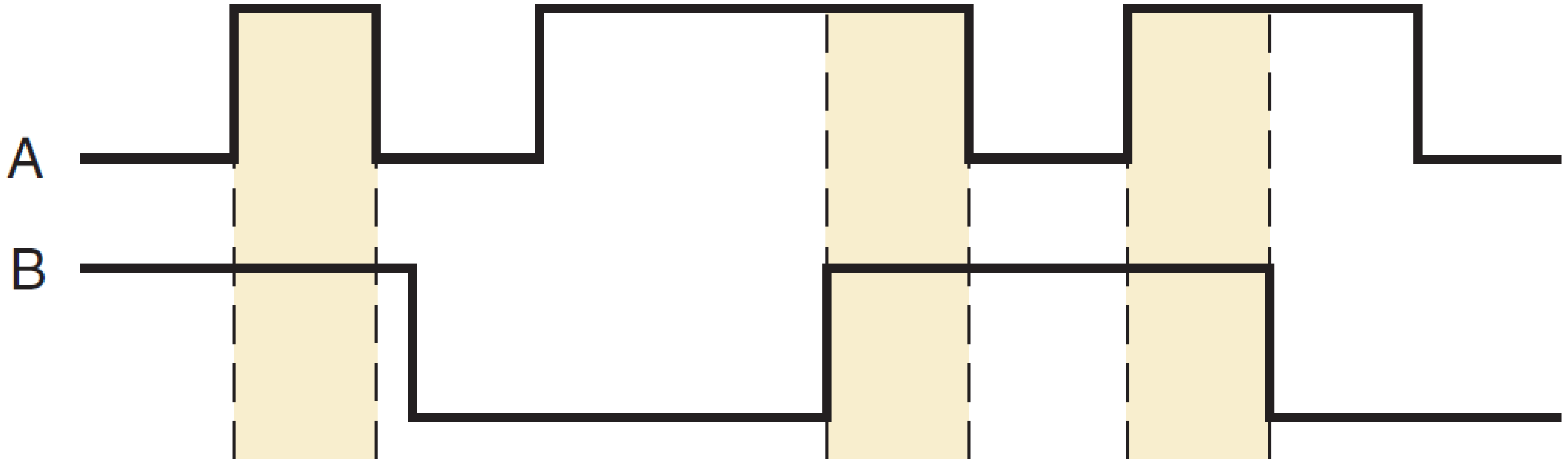
Exercises

Ans.



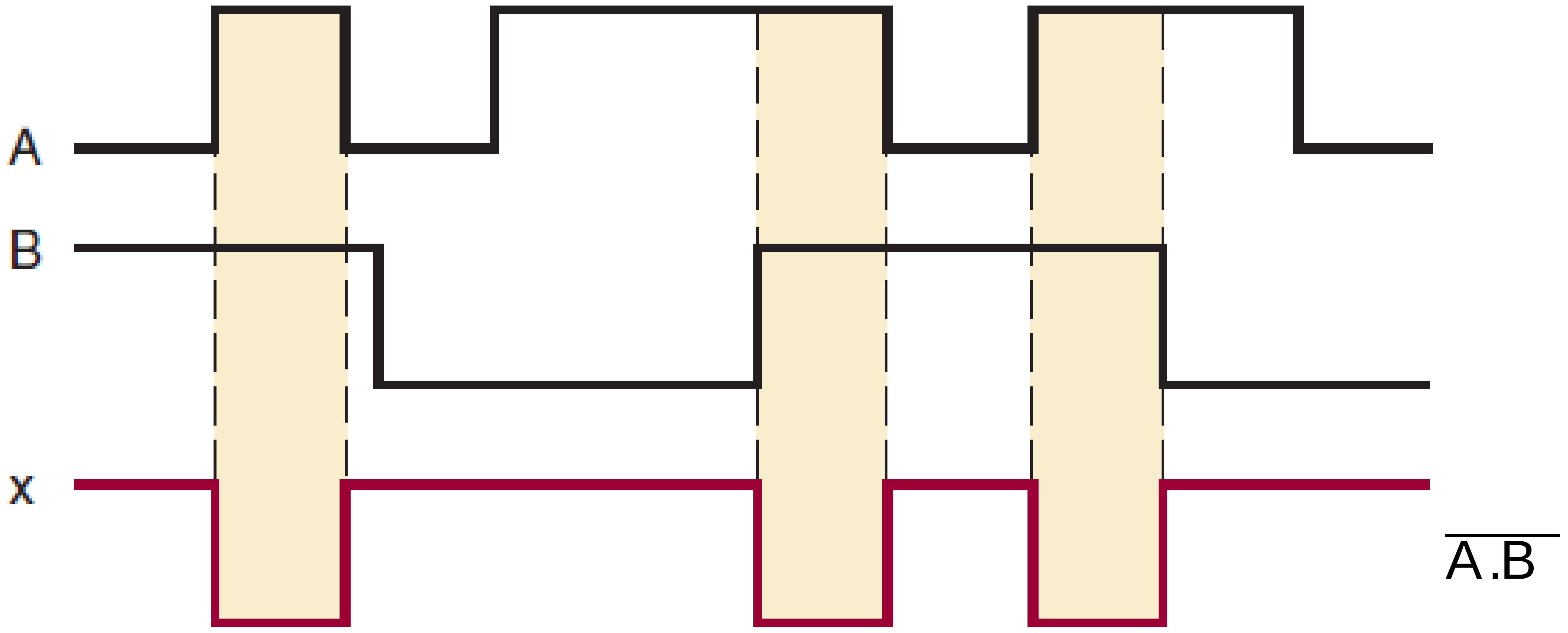
EXERCISES

9. **Determine the waveform at the output of a NAND gate for the input waveforms shown below**



EXERCISES

Ans.



Exercises

10. Implement the logic circuit for the expression below using only NOR and NAND gates

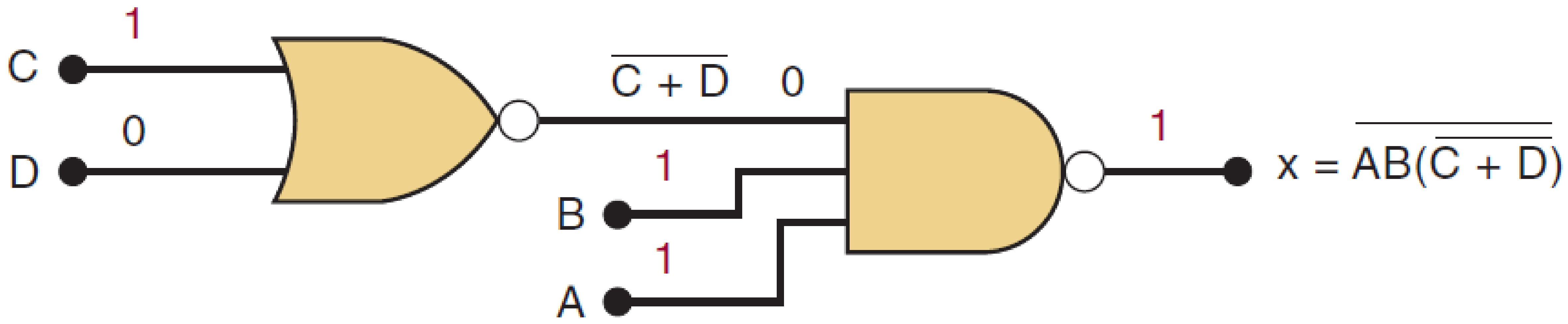
$$x = \overline{AB.(\overline{C + D})}$$

EXERCISES

10. Implement the logic circuit for the expression below using only NOR and NAND gates

$$x = \overline{AB.(\overline{C+D})}$$

Ans.



C
D
1
0
$\overline{C + D}$
0
1
B
1
A
1
$x = \overline{AB(\overline{C + D})}$