

Portas Lógicas

Algebra de Boole

1. Introdução

- Em 1854, o matemático inglês George Boole apresentou o sistema matemático conhecido como álgebra de Boole, que serve como base até hoje para eletrônica digital.
- Apesar de ter sido criada em 1854, a álgebra de Boole foi utilizada pela primeira vez apenas em 1938, para implementação de circuitos de telefonia.
- A eletrônica digital emprega em seus sistemas um pequeno grupo de circuitos básicos conhecidos como portas lógicas.
- Por meio destas portas lógicas conseguimos implementar todas as expressões geradas pela álgebra de Boole, eu constitui a base da eletrônica digital, e por consequência, dos sistemas computacionais atuais.

2. Funções Lógicas

- As funções lógicas derivam dos postulados da álgebra de Boole, e sempre possuem apenas dois estados distintos: estado 0 e estado 1.
- O estado 0 representa portão fechado (ausência de tensão) e estado 1 representa porta aberto. Uma porta lógica nunca terá valor 0 e 1 simultaneamente.
- As principais portas lógicas são AND, OR, NOT, NAND, NOR, XOR e XNOR. Combinando estas portas é possível implementar os principais componentes eletrônicos do computador, como a ULA por exemplo.

3. Função E ou AND

- A função AND representa a multiplicação entre duas variáveis. Se uma for 0, o resultado é 0.

Sua representação algébrica para 2 variáveis é $S = A \cdot B$, onde se lê $S = A$ e B .

Para melhor compreensão, vamos utilizar e analisar o circuito representativo da função E visto na Figura 2.1.

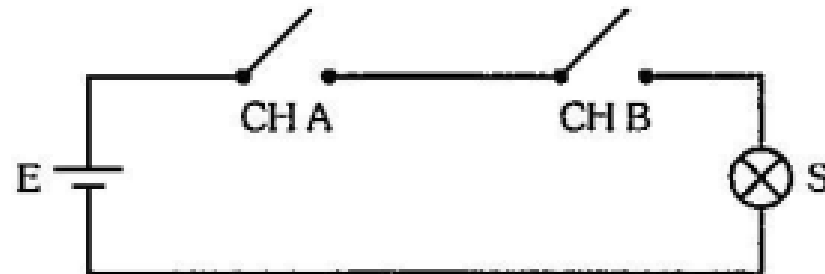
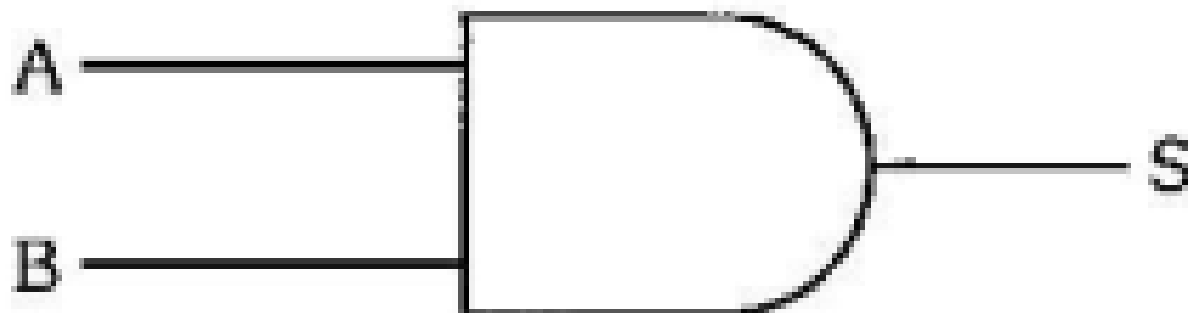


Figura 2.1

Convenções:	chave aberta = 0	chave fechada = 1
	lâmpada apagada = 0	lâmpada acesa = 1

3. Função E ou AND

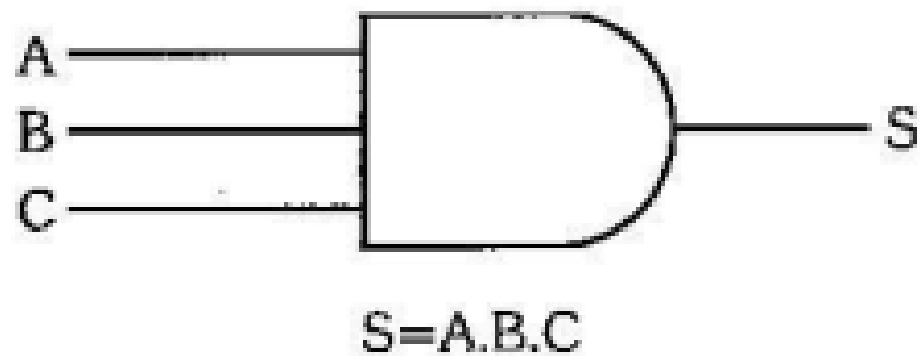
- A porta AND é um componentes que executa a função AND da álgebra da Boole. O símbolo abaixo representa a porta AND:



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

3. Função E ou AND

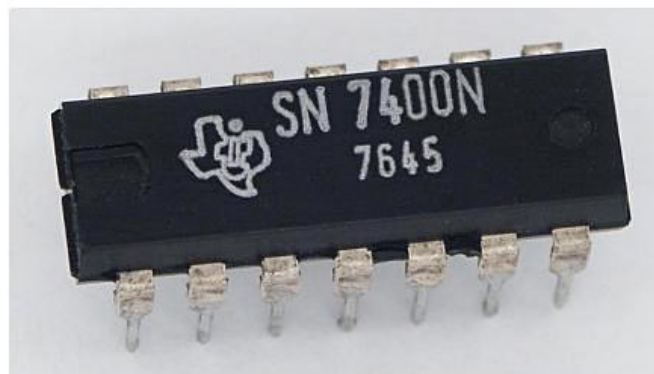
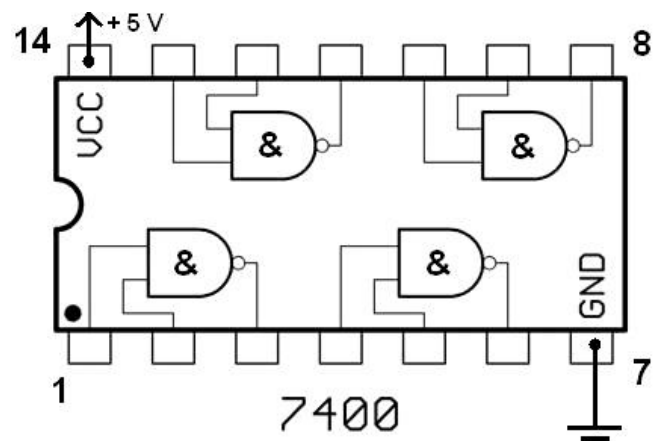
- A porta AND pode ter mais de duas entradas:



A	B	C	S
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

3. Função E ou AND

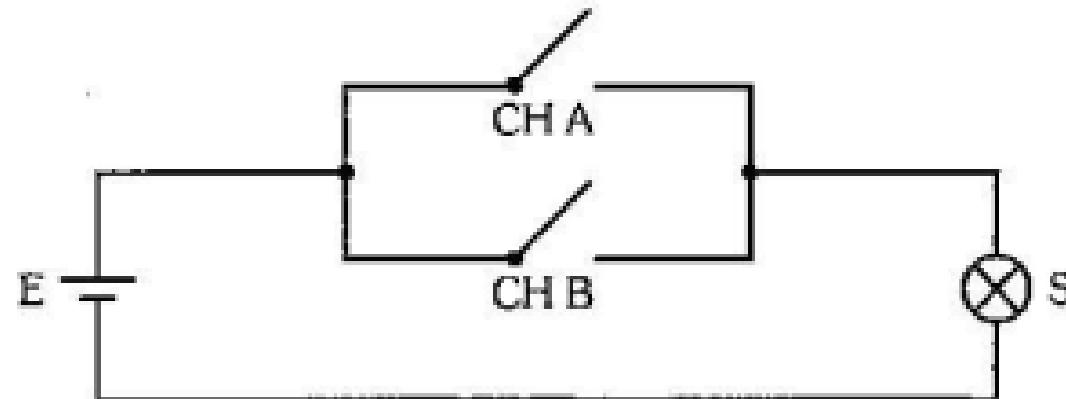
- A imagem abaixo mostra uma porta lógica AND real:



4. Função OU ou OR

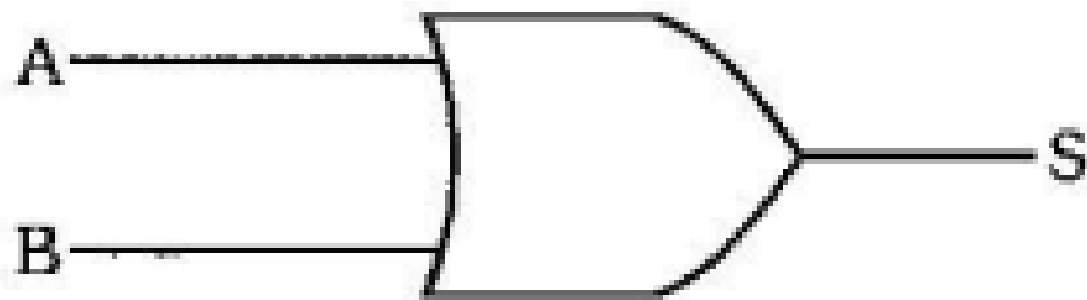
- A função OR retorna 1 se uma ou mais entradas for 1.

A função OU é aquela que assume valor 1 quando uma ou mais variáveis da entrada forem iguais a 1 e assume valor 0 se, e somente se, todas as variáveis de entrada forem iguais a 0. Sua representação algébrica para 2 variáveis de entrada é $S = A + B$, onde se lê $S = A$ ou B .



4. Função OU ou OR

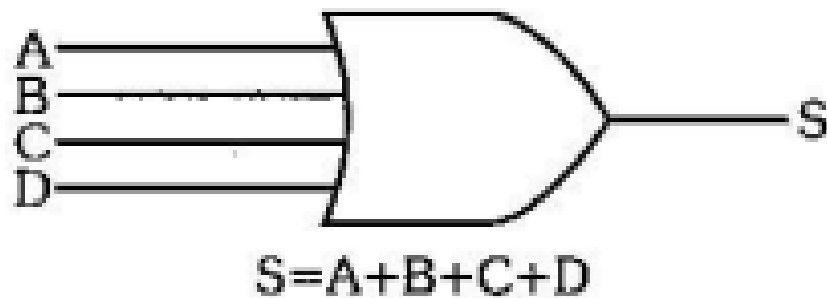
- A porta OR é representada na imagem abaixo:



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

4. Função OU ou OR

- A porta OR pode ter mais variáveis:

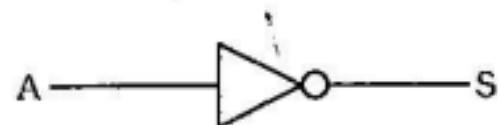


A	B	C	D	S
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

5. Função Não ou NOT

- A porta NOT inverte o valor.

A função NÃO é aquela que inverte ou complementa o estado da variável, ou seja, se a variável estiver em 0, a saída vai para 1, e se estiver em 1, a saída vai para 0. É representada algebricamente da seguinte forma: $S = \bar{A}$ ou $S = A'$, onde se lê **A barra** ou **NÃO A**.



—○ (antes de um outro bloco lógico)

A	S
0	1
1	0

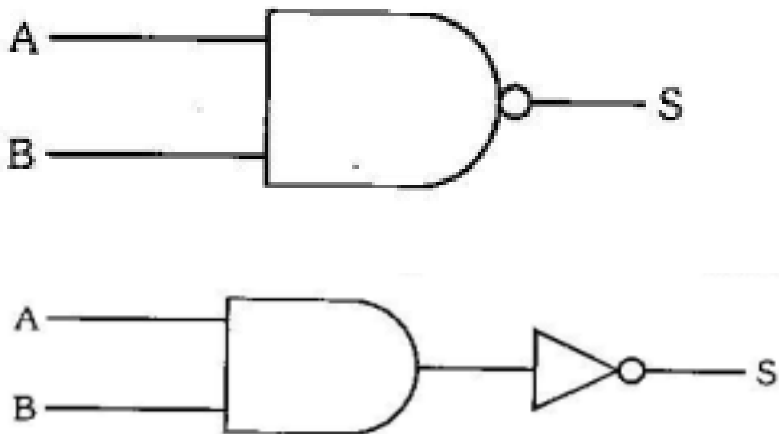
100 101 1020 1021 1022 1023

6. Função Não E, NE ou NAND

- A NAND é basicamente a função AND invertida. Podemos representar de duas formas:

Como o próprio nome “NÃO E” diz: essa função é uma composição da função E com a função NÃO, ou seja, teremos a função E invertida. É representada algebricamente da seguinte forma:

$S = \overline{(A \cdot B)}$, onde o traço indica que temos a inversão do produto A.B.



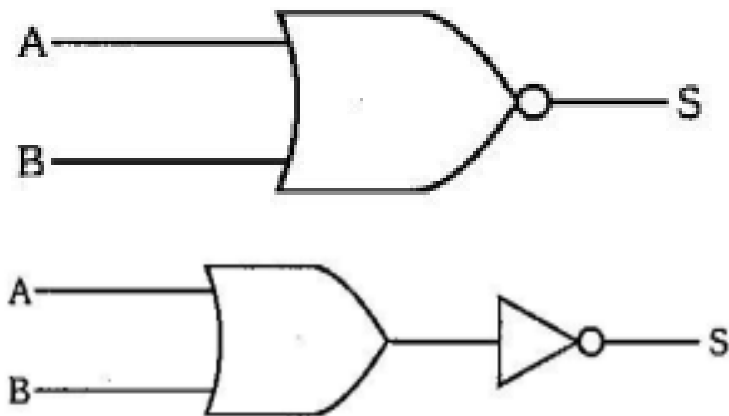
A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

7. Função Não OU, NOU ou NOR

- A NOR é basicamente a função OR invertida. Podemos representar de duas formas:

Analogamente à função NE, a função NOU é a composição da função NÃO com a função OU, ou seja, a função NOU será o inverso da função OU. É representada da seguinte forma:

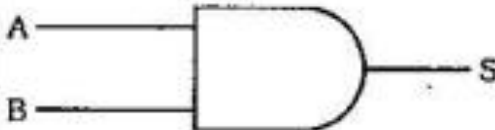
$S = \overline{(A + B)}$, onde o traço indica a inversão da soma booleana $A + B$.



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

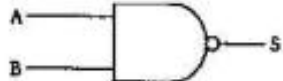
8. Resumo

- A imagem resume as portas lógicas:

BLOCOS LÓGICOS BÁSICOS																			
Porta	Símbolo Usual	Tabela da Verdade	Função Lógica	Expressão															
E AND		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	S	0	0	0	0	1	0	1	0	0	1	1	1	Função E: assume 1 quando todas as <u>variáveis</u> forem <u>1 e 0</u> nos outros casos.	$S = AB$
A	B	S																	
0	0	0																	
0	1	0																	
1	0	0																	
1	1	1																	
OU OR		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	S	0	0	0	0	1	1	1	0	1	1	1	1	Função OU: assume 0 quando todas as variáveis forem 0 e 1 nos outros casos.	$S = A + B$
A	B	S																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	1																	

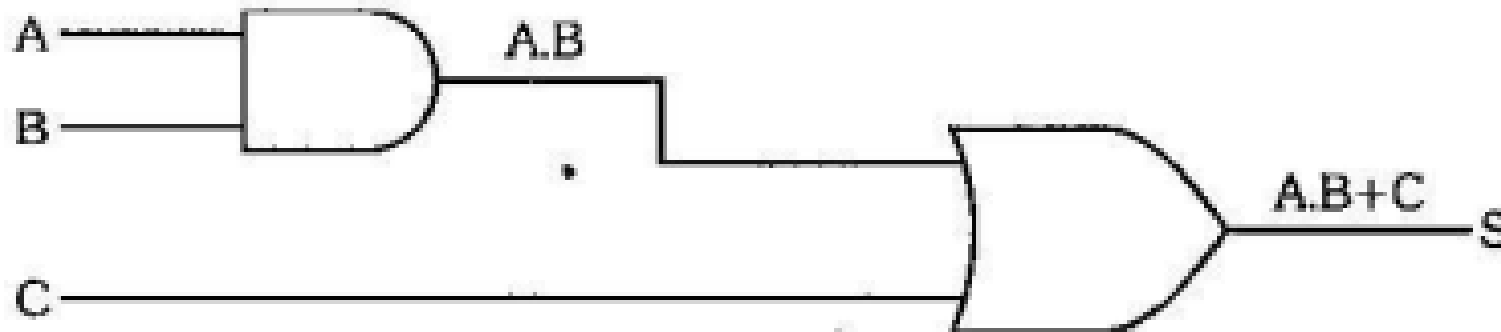
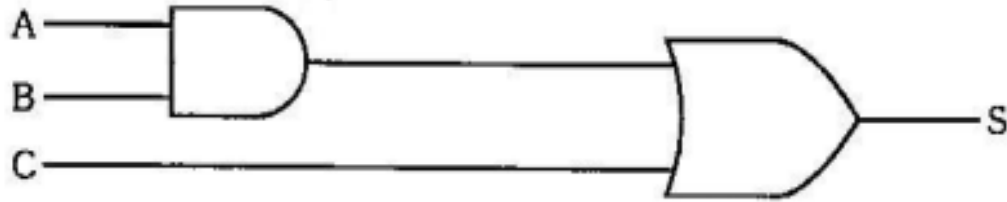
8. Resumo

- A imagem resume as portas lógicas:

BLOCOS LÓGICOS BÁSICOS																			
Porta	Símbolo Usual	Tabela da Verdade	Função Lógica	Expressão															
NÃO NOT INVERSOR		<table><tr><th>A</th><th>S</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	S	0	1	1	0	Função NÃO: inverte a variável aplicada à sua entrada.	$S = \overline{A}$									
A	S																		
0	1																		
1	0																		
NE NAND		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	S	0	0	1	0	1	1	1	0	1	1	1	0	Função NE: inverso da função E.	$S = \overline{AB}$
A	B	S																	
0	0	1																	
0	1	1																	
1	0	1																	
1	1	0																	
NOU NOR		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	S	0	0	1	0	1	0	1	0	0	1	1	0	Função NOU: inverso da função OU.	$S = \overline{A + B}$
A	B	S																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	0																	

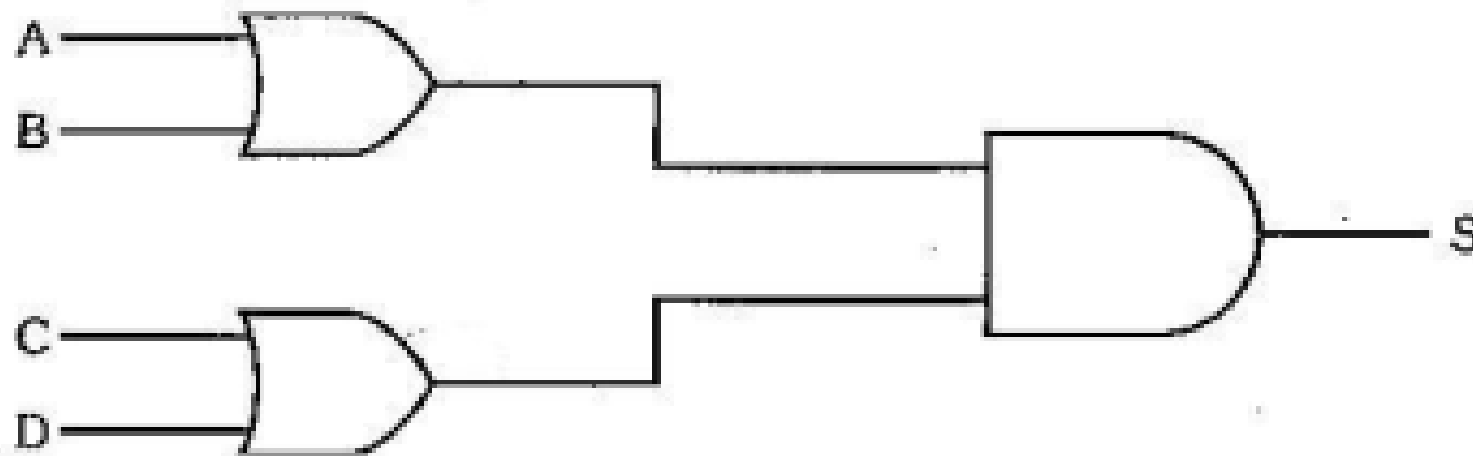
9. Representando Circuitos por Expressões

- Todo circuito executa uma função booleana. Por mais complexa que a expressão seja, ela será executada pela interligação de portas lógicas. O circuito abaixo pode ser representado por uma expressão: $S=A.B+C$



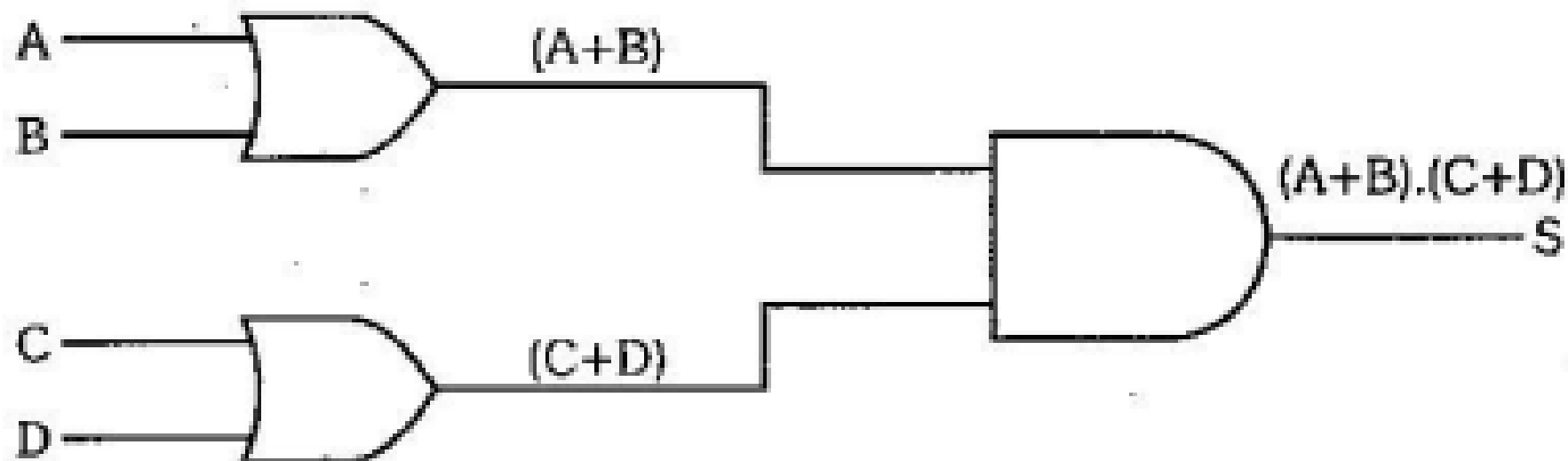
9. Representando Circuitos por Expressões

- Exemplo: Escreva a expressão do circuito



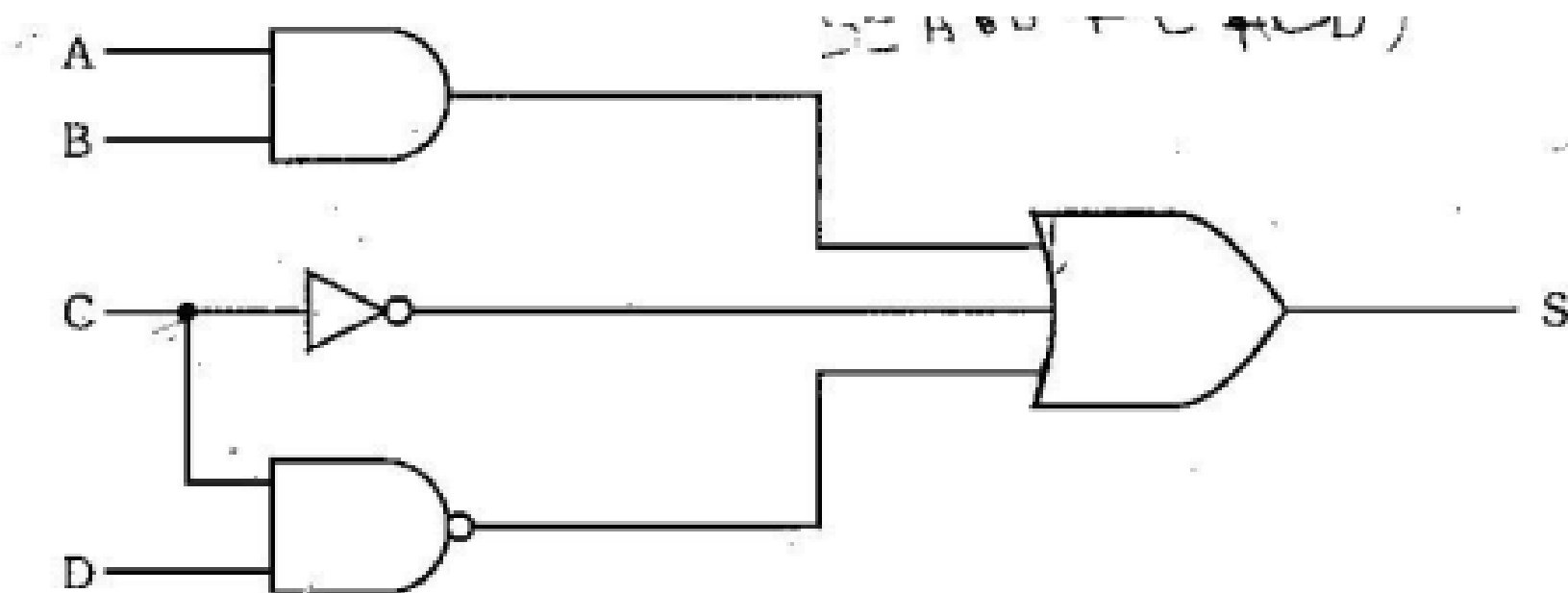
9. Representando Circuitos por Expressões

- O resultado é: $S = (A+B).(C+D)$



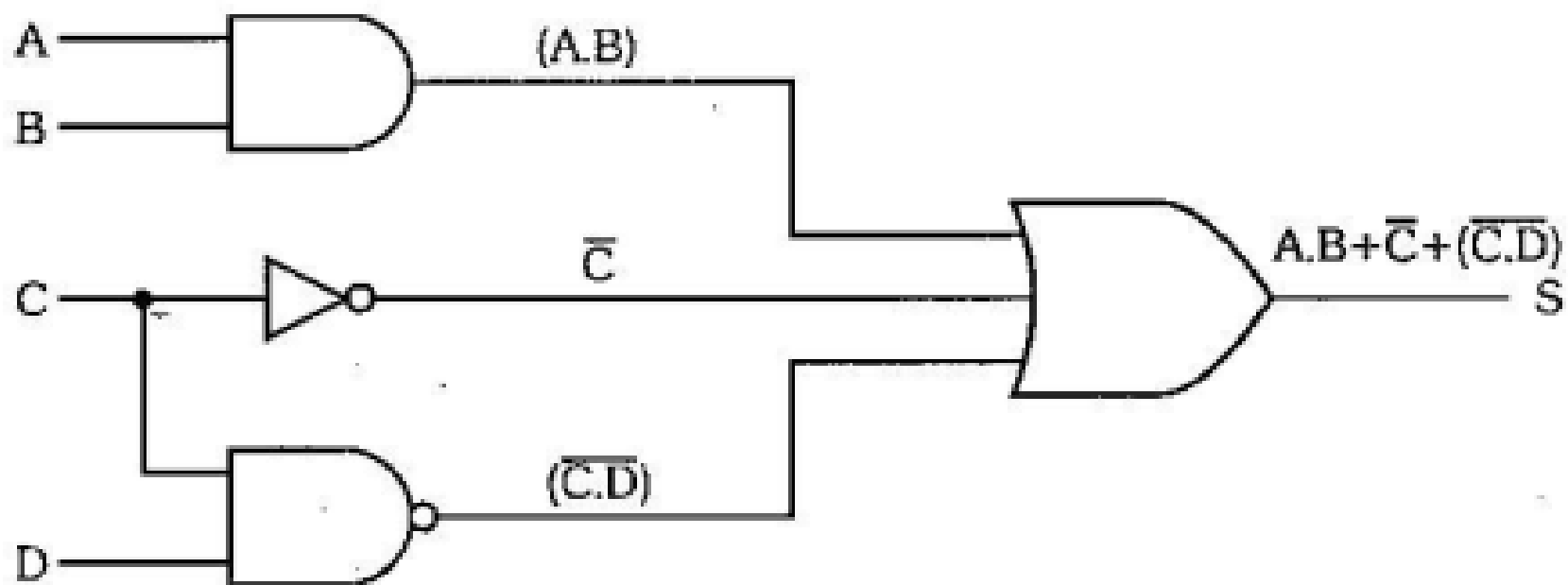
9. Representando Circuitos por Expressões

- Exemplo: Escreva a expressão do circuito



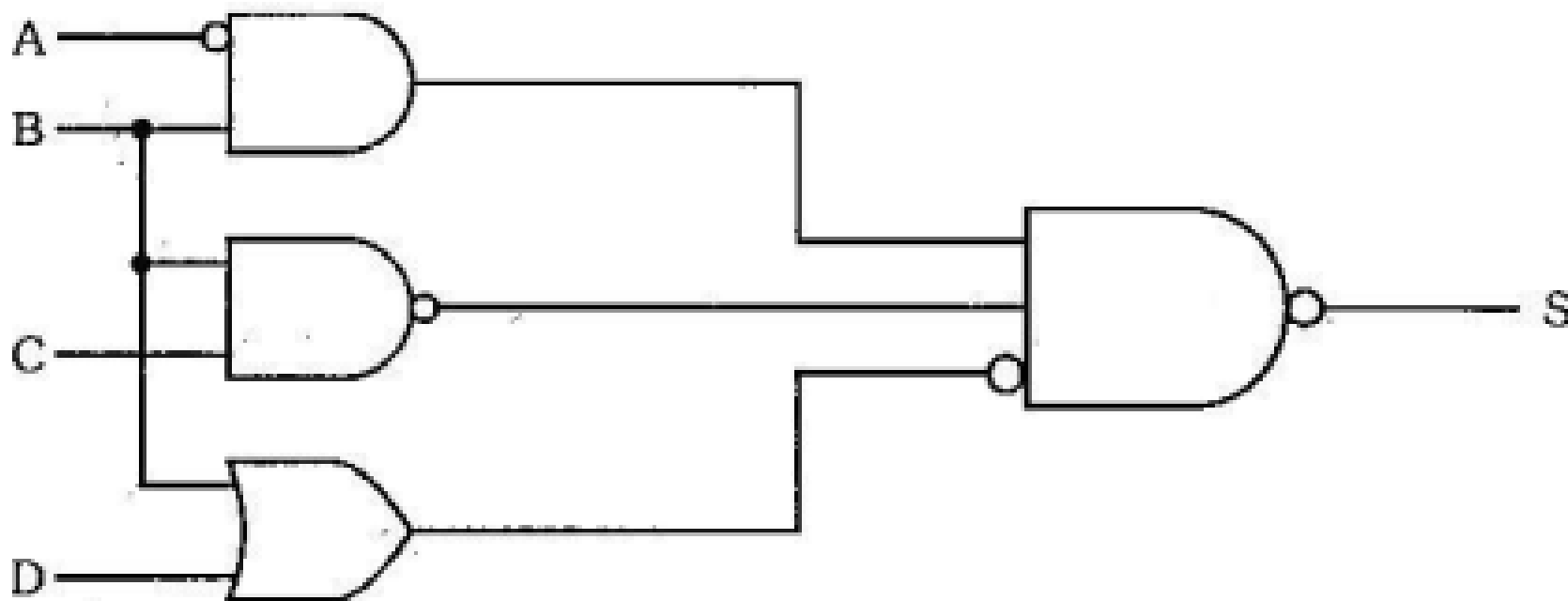
9. Representando Circuitos por Expressões

- Resposta:



9. Representando Circuitos por Expressões

- Exemplo: Escreva a expressão do circuito



9. Representando Circuitos por Expressões

- Resposta:

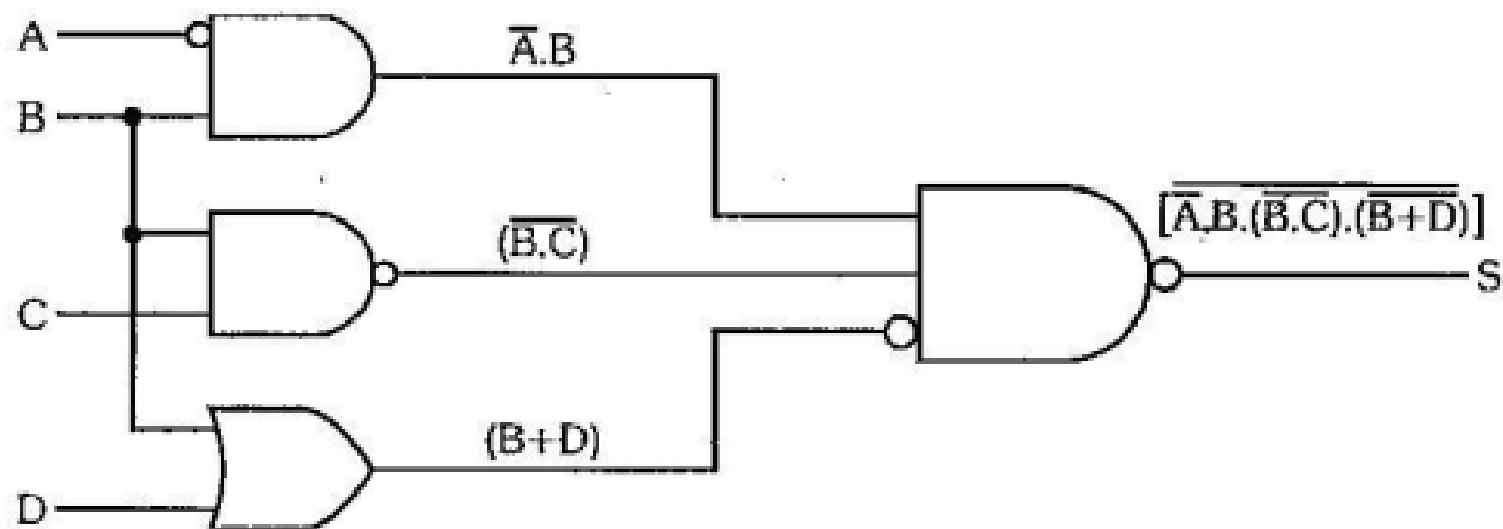
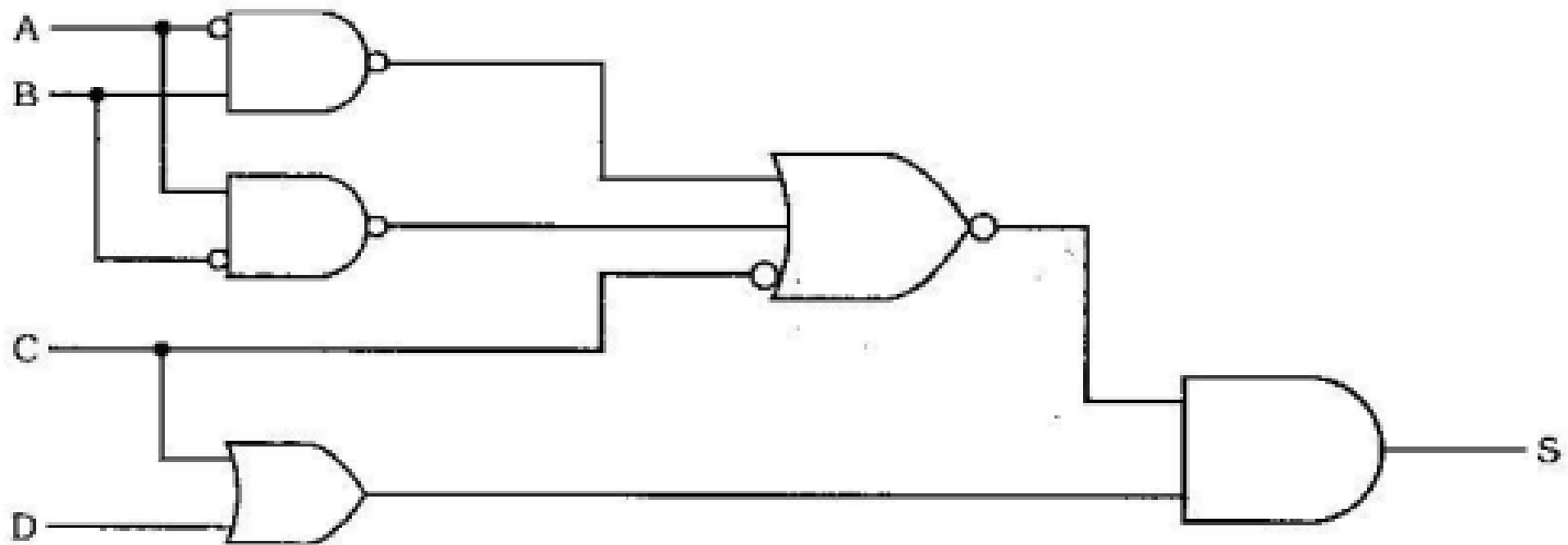


Figura 2.21

$$\therefore S = \overline{[\overline{A} . B . (B . C) . (B + D)]}$$

9. Representando Circuitos por Expressões

- Exemplo: Escreva a expressão do circuito



9. Representando Circuitos por Expressões

- Resposta

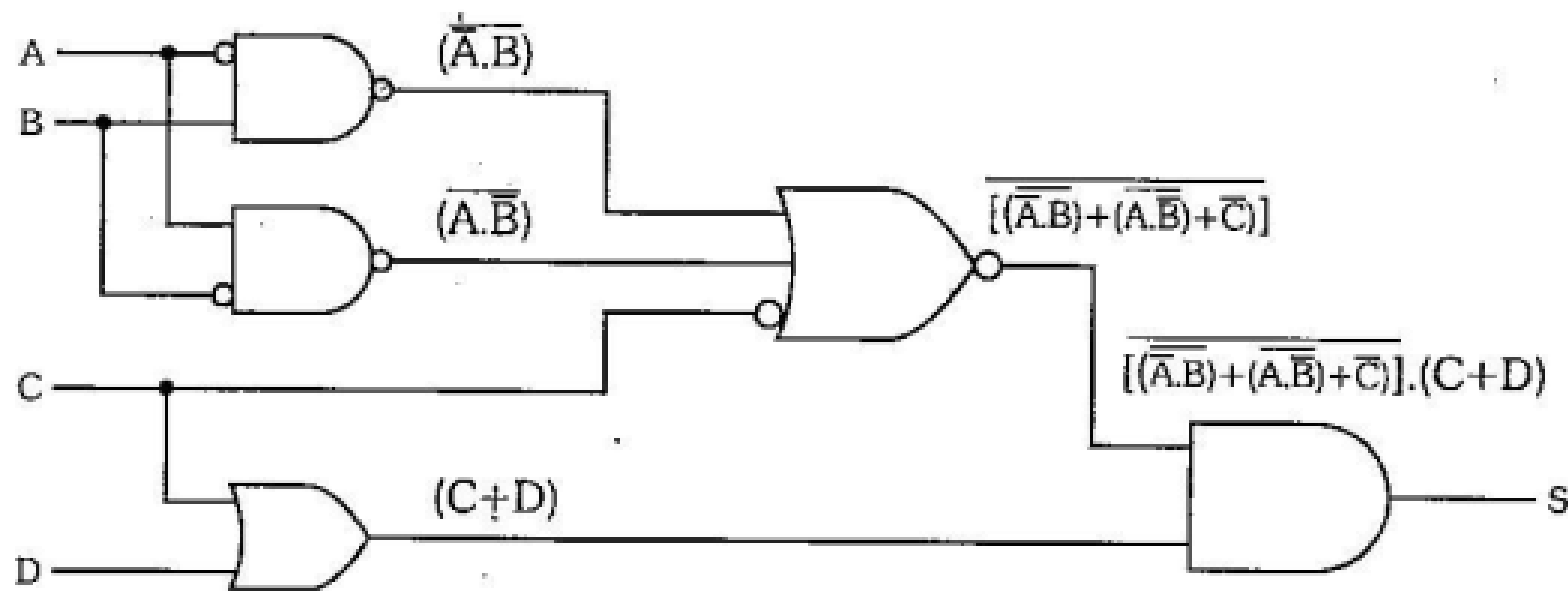


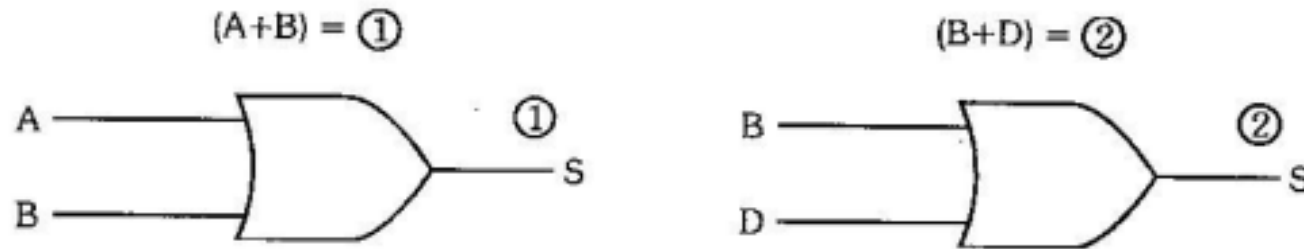
Figura 2.23

$$\therefore S = \left[(\overline{A} . B) + (A . \overline{B}) + \overline{C} \right] . (C + D)$$

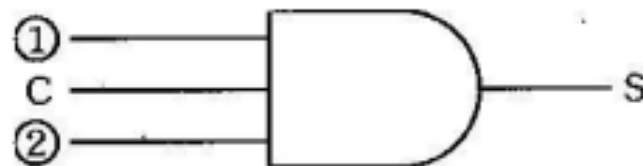
10. Circuitos Obtidos por Expressões

- Também é possível fazer o processo contrário, onde obtemos o circuito a partir da expressão booleana. Por exemplo, vamos montar o circuito da expressão $S = (A+B).C.(B+D)$

Para o primeiro parêntese, temos a soma booleana $A + B$, logo, o circuito que o executa será uma porta OU. Para o segundo, temos a soma booleana $B + D$, logo, o circuito será uma porta OU. Até aí, temos então:

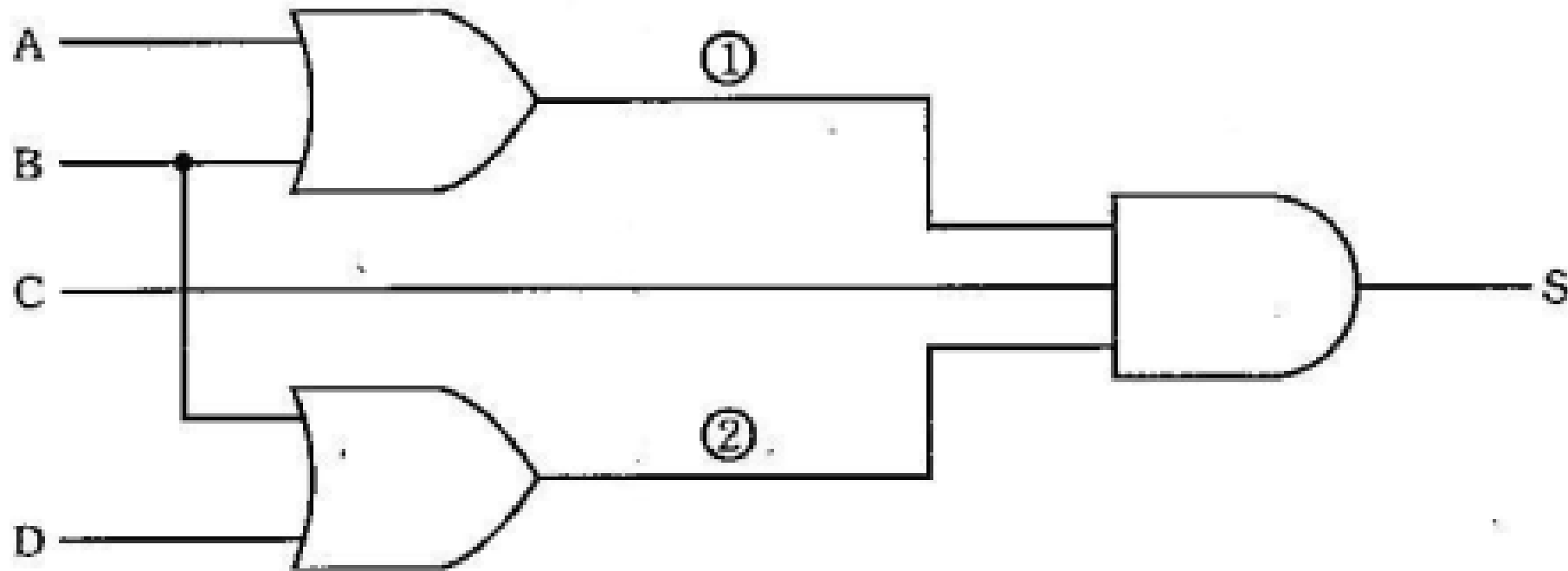


A seguir, temos uma multiplicação booleana dos dois parênteses, juntamente com a variável C , sendo o circuito que executa esta multiplicação, uma porta E:



10. Circuitos Obtidos por Expressões

- Juntando as partes lógicas, temos:



10. Circuitos Obtidos por Expressões

- Exercício:

Desenhe o circuito que executa a expressão booleana
 $S = A.B.C + (A+B) .C.$

10. Circuitos Obtidos por Expressões

- Solução:

Primeiramente, para identificar as portas lógicas, vamos numerar cada termo da expressão:

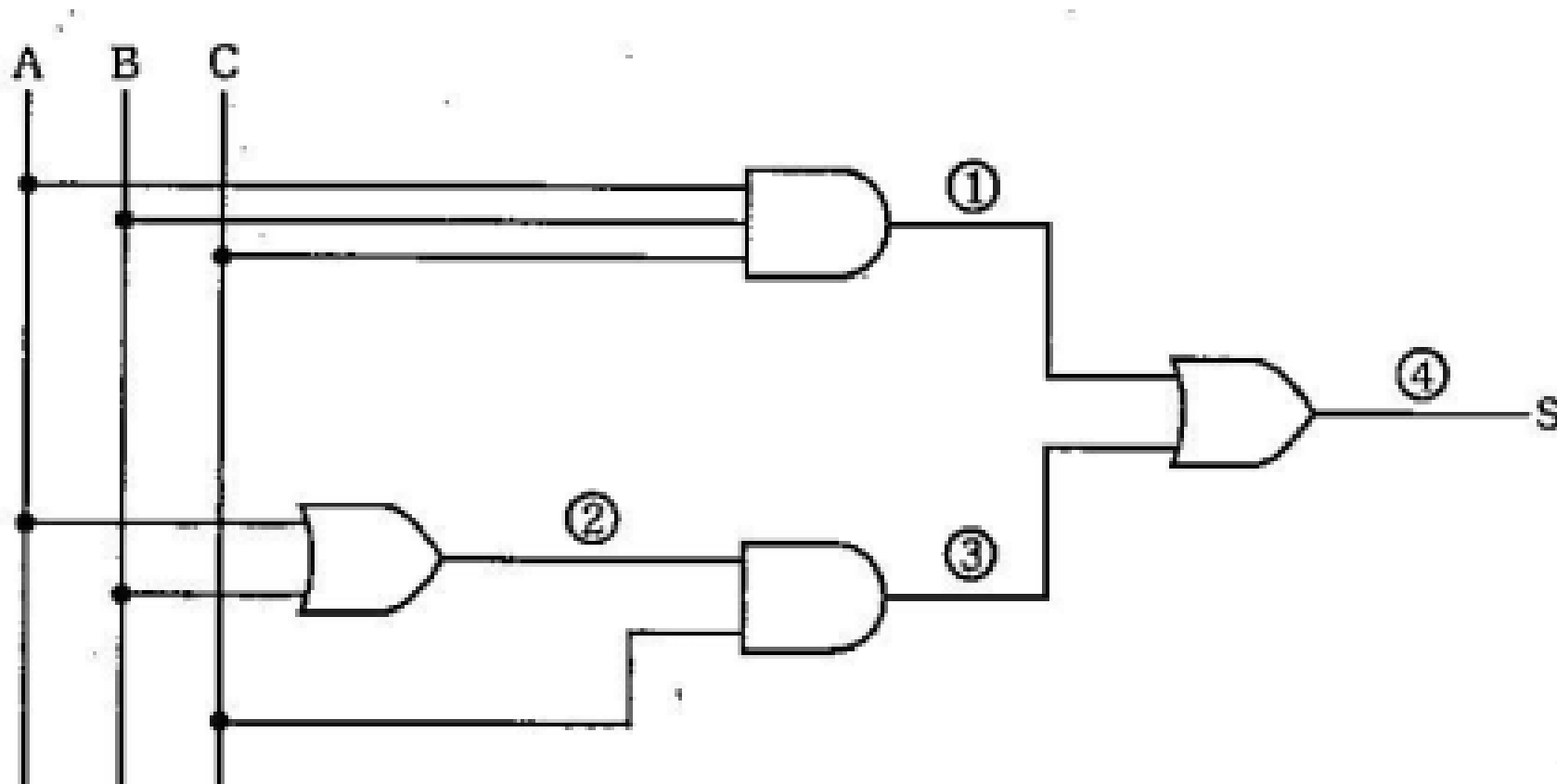
$$S = \underbrace{A.B.C}_{(1)} + \underbrace{(A+B)}_{(2)} . C$$
$$\underbrace{\hspace{10em}}_{(3)}$$
$$\underbrace{\hspace{15em}}_{(4)}$$

Assim sendo, temos:

- ① porta E com A, B e C.
- ② porta OU com A e B.
- ③ porta E com ② e C.
- ④ porta OU com ① e ③.

10. Circuitos Obtidos por Expressões

- Para facilitar, é possível criar uma malha com as entradas A, B e C:



10. Circuitos Obtidos por Expressões

- Exercício:

Idem para a expressão: $S = \overline{(\overline{A} + B) + (\overline{C} \cdot D)} \cdot \overline{D}$.

10. Circuitos Obtidos por Expressões

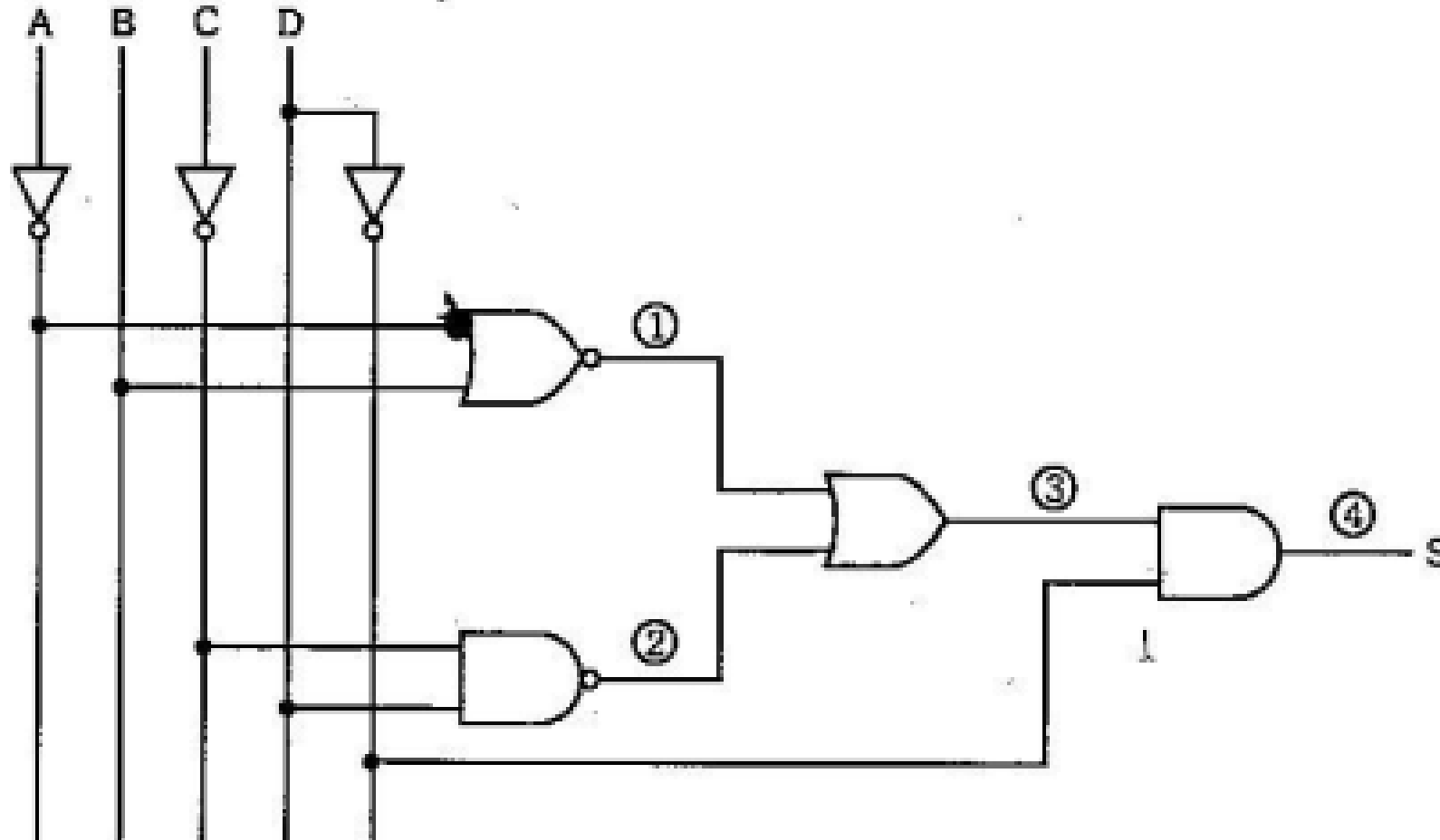
- Solução:

$$S = \underbrace{\underbrace{(\overline{\overline{A} + B})}_{(1)} + \underbrace{(\overline{\overline{C} \cdot D})}_{(2)}}_{(3)} \cdot \overline{D}$$

(4)

10. Circuitos Obtidos por Expressões

- Solução:



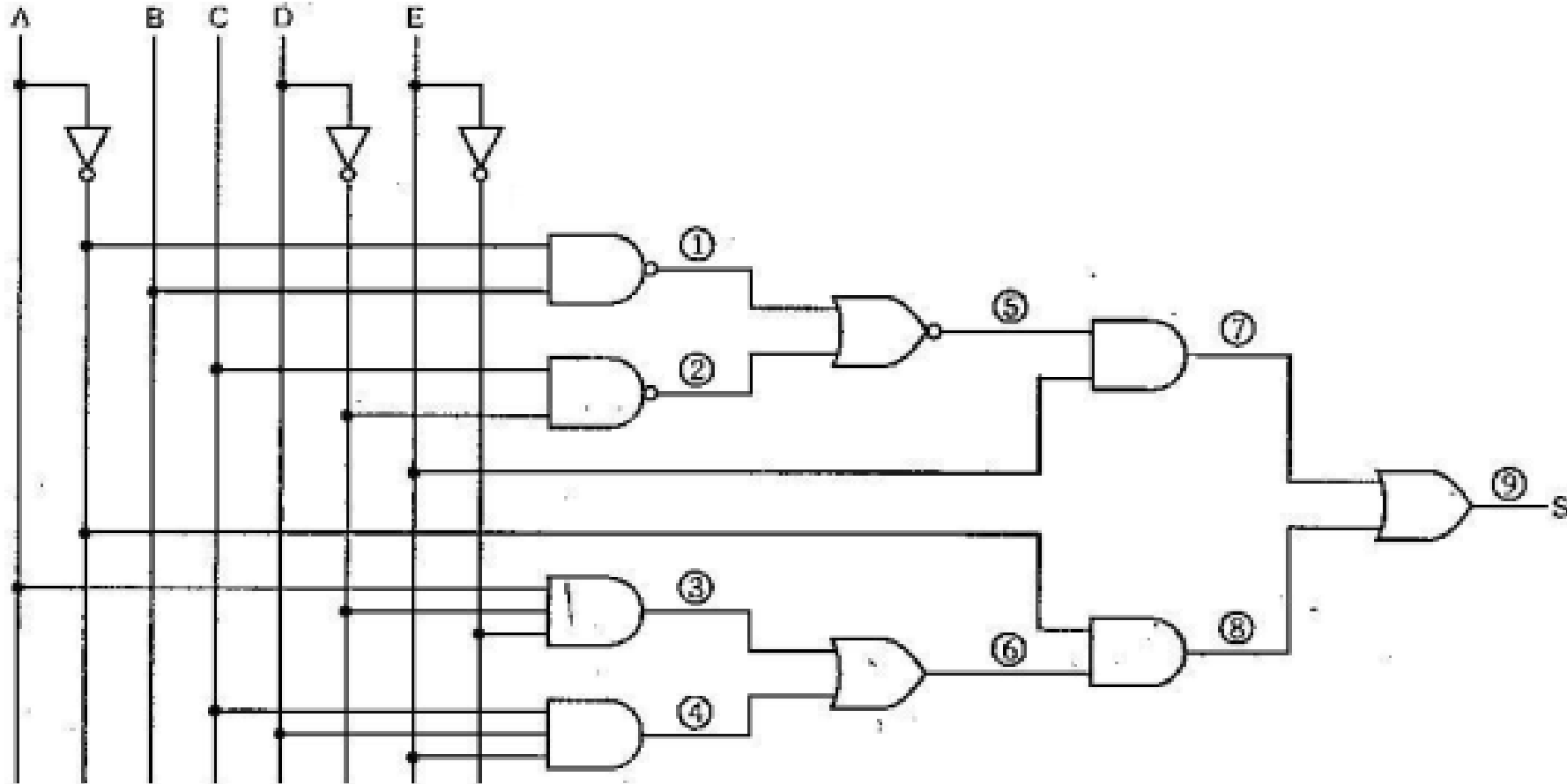
10. Circuitos Obtidos por Expressões

- Exercício:

Idem para a expressão: $S = \overline{(\overline{A} \cdot B) + (C \cdot \overline{D})} \cdot E + \overline{A} \cdot (A \cdot \overline{D} \cdot \overline{E} + C \cdot D \cdot E)$.

10. Circuitos Obtidos por Expressões

- Solução:



11. Tabelas Verdade Obtidas de Expressões

- Uma das formas de estudar o comportamento de uma função booleana é utilizando sua tabela verdade. Para extrair a tabela verdade de um expressão, fazemos o seguinte procedimento:
 1. Monta o quadro com todas as possibilidades de entrada
 2. Montamos colunas para as várias partes da expressão
 3. Preenchemos estas colunas com resultados
 4. A partir destas colunas determinamos o resultado final

Para esclarecer este processo, vamos utilizar a expressão

$$S = A \cdot \bar{B} \cdot C + A \cdot \bar{D} + \bar{A} \cdot B \cdot D.$$

11. Tabelas Verdade Obtidas de Expressões

- Vejamos um exemplo:

$$S = A \cdot \bar{B} \cdot C + A \cdot \bar{D} + \bar{A} \cdot B \cdot D$$

				1º membro	2º membro	3º membro	Resultado final S
A	B	C	D	$A \cdot \bar{B} \cdot C$	$A \cdot \bar{D}$	$\bar{A} \cdot B \cdot D$	
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	1	0	0	1	1
0	1	1	0	0	0	0	0
0	1	1	1	0	0	1	1
1	0	0	0	0	1	0	1
1	0	0	1	0	0	0	0
1	0	1	0	1	1	0	1
1	0	1	1	1	0	0	1
1	1	0	0	0	1	0	1
1	1	0	1	0	0	0	0
1	1	1	0	0	1	0	1
1	1	1	1	0	0	0	0

11. Tabelas Verdade Obtidas de Expressões

- Exercícios:

Prove as identidades abaixo relacionadas:

a) $\overline{A} \cdot \overline{B} \neq \overline{A \cdot B}$

b) $\overline{A} + \overline{B} \neq \overline{A + B}$

c) $\overline{A} \cdot \overline{B} = \overline{A + B}$

d) $\overline{A} + \overline{B} = \overline{A \cdot B}$

11. Tabelas Verdade Obtidas de Expressões

- Montando a tabela, conseguimos provar as equivalências

A	B	$\bar{A} \cdot \bar{B}$	$\overline{A \cdot B}$	$\bar{A} + \bar{B}$	$\overline{A + B}$
0	0	1	1	1	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	0	0	0	0

11. Tabelas Verdade Obtidas de Expressões

- Exercício:

Levante a tabela da verdade da expressão:

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

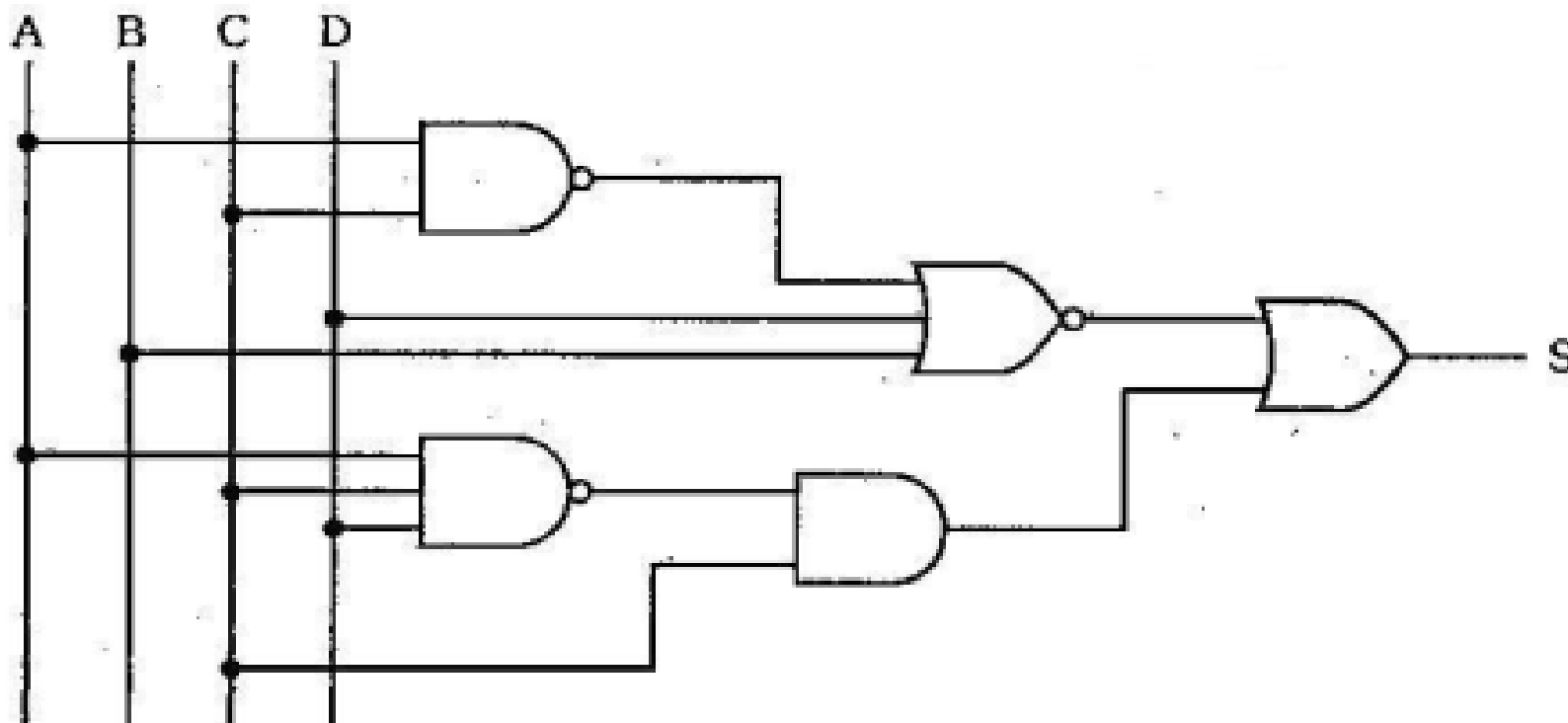
11. Tabelas Verdade Obtidas de Expressões

- Resposta:

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

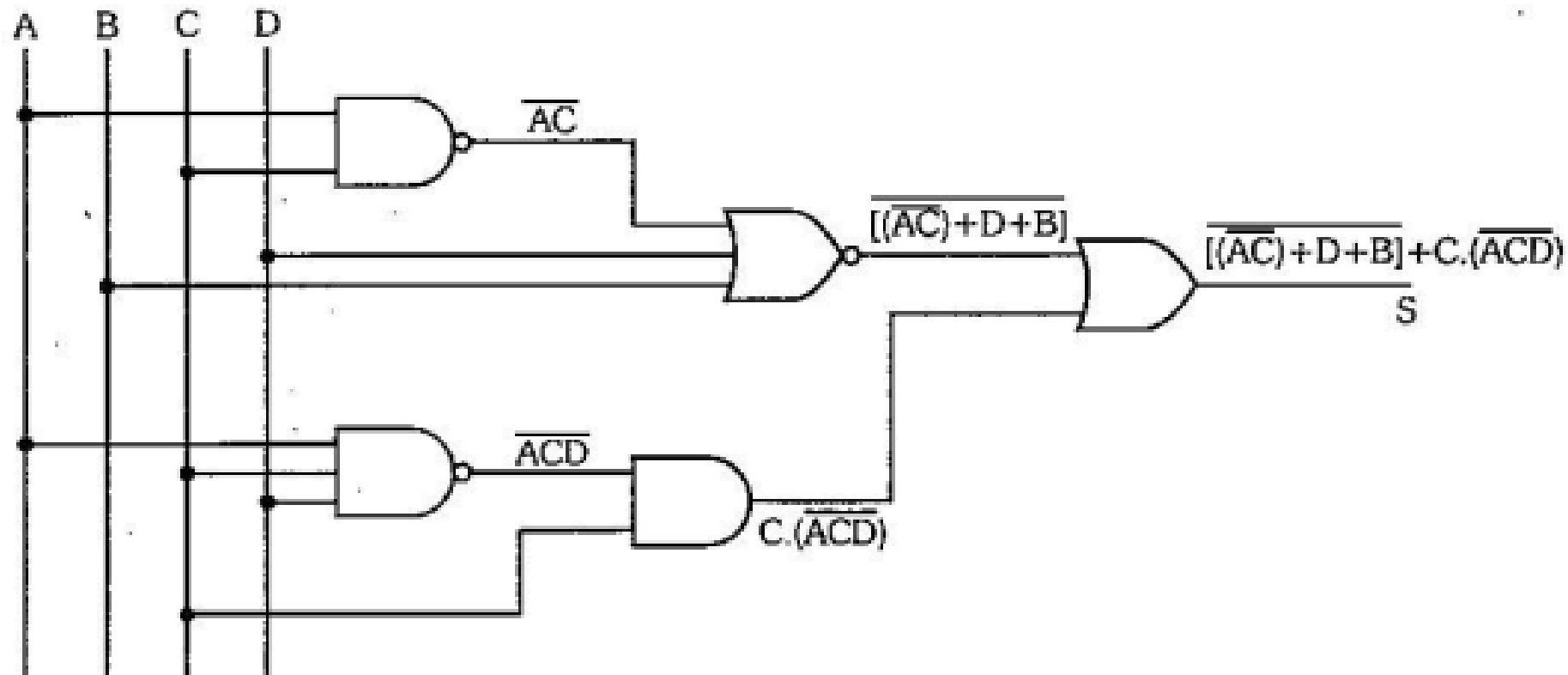
11. Tabelas Verdade Obtidas de Expressões

- Exercícios: Analise o comportamento do circuito montando sua tabela verdade:



11. Tabelas Verdade Obtidas de Expressões

- Resposta: Montando a expressão



11. Tabelas Verdade Obtidas de Expressões

- Resposta: $S = \overline{[(A \cdot C) + D + B]} + C \cdot \overline{(A \cdot C \cdot D)}$

A	B	C	D	$\overline{[(A \cdot C) + D + B]}$	$C \cdot \overline{(A \cdot C \cdot D)}$	S
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	0	1	1
0	0	1	1	0	1	1
0	1	0	0	0	0	0
0	1	0	1	0	0	0
0	1	1	0	0	1	1
0	1	1	1	0	1	1
1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	1	1
1	0	1	1	0	0	0
1	1	0	0	0	0	0
1	1	0	1	0	0	0
1	1	1	0	0	1	1
1	1	1	1	0	0	0

12. Expressões a partir da Tabela Verdade

- Para demonstrar o procedimento, vejamos o exemplo:

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

Observando a tabela, notamos que expressão é verdadeira ($S = 1$) nos casos onde $A = 0$ e $B = 0$ ou $A = 1$ e $B = 0$ ou $A = 1$ e $B = 1$. Para obter a expressão, basta montar os termos relativos aos casos onde a expressão for verdadeira e somá-los:

Caso 00: $S = 1$ quando, $A = 0$ e $B = 0$ ($\bar{A} = 1$ e $\bar{B} = 1$) $\Rightarrow \bar{A} \cdot \bar{B}$

Caso 10: $S = 1$ quando, $A = 1$ e $B = 0$ ($A = 1$ e $\bar{B} = 1$) $\Rightarrow A \cdot \bar{B}$

Caso 11: $S = 1$ quando, $A = 1$ e $B = 1 \Rightarrow A \cdot B$

$$\therefore S = \bar{A} \cdot \bar{B} + A \cdot \bar{B} + A \cdot B$$

12. Expressões a partir da Tabela Verdade

- Exercício: Encontre a expressão da tabela verdade abaixo:

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

12. Expressões a partir da Tabela Verdade

- Resposta:

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

Para solucionar, extraímos os casos onde a expressão é verdadeira ($S = 1$):
000 ou 010 ou 110 ou 111.

$$\therefore S = \bar{A}.\bar{B}.\bar{C} + \bar{A}.B.\bar{C} + A.B.\bar{C} + A.B.C$$

13. Porta Ou Exclusivo

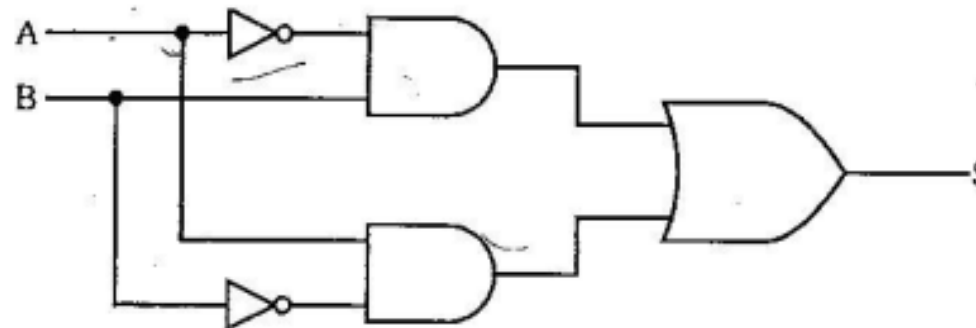
- O "**OU Exclusivo**" possui sua saída igual a 1 quando as duas entradas forem diferentes. Apesar de ser considerado uma porta lógica básica, na prática esta porta é formada pela combinação de outras portas.
- A partir da tabela verdade, podemos deduzir a fórmula do "OU Exclusivo" bem como seu verdadeiro circuito (formada por AND e OR):

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

Da tabela obtemos sua expressão característica:

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

Da expressão esquematizamos o circuito representativo da função OU Exclusivo, visto na figura 2.33:



13. Porta Ou Exclusivo

- A imagem abaixo mostra como podemos representar em um circuito o "OU Exclusivo":

A notação algébrica que representa a função OU Exclusivo é $S = A \oplus B$.



Uma importante observação é que, ao contrário de outros blocos lógicos básicos, o circuito OU Exclusivo só pode ter 2 variáveis de entrada, fato este devido à sua definição básica. O circuito OU Exclusivo também é conhecido como Exclusive OR (EXOR), termo derivado do inglês.

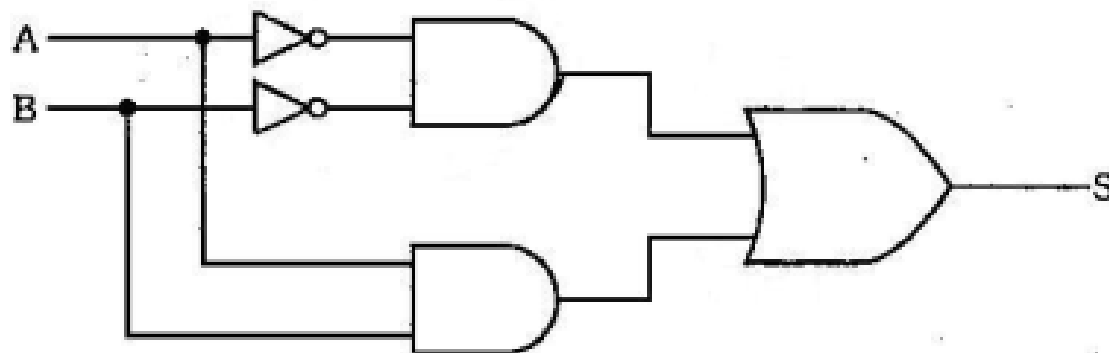
14. Bloco Coincidência (NOR Exclusive)

- Essa porta lógica é contrária ao "Ou Exclusivo", tendo valor de saída 1 quando as duas entradas possuem valores iguais:

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

A tabela gera a expressão $S = \bar{A} \cdot \bar{B} + A \cdot B$.

A partir desta expressão, vamos esquematizar o circuito:



14. Bloco Coincidência (NOR Exclusive)

- Para representar no circuito, temos o seguinte símbolo:

A notação algébrica que representa a função Coinidência é $S = A \odot B$



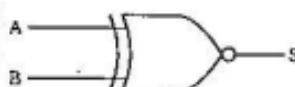
Se compararmos as tabelas da verdade dos blocos OU Exclusivo e Coinidência, iremos concluir que estes são complementares, ou seja, teremos a saída de um invertida em relação à saída do outro. Assim sendo, podemos escrever:

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

O bloco Coinidência é também denominado de NOR Exclusivo e do inglês Exclusive NOR.

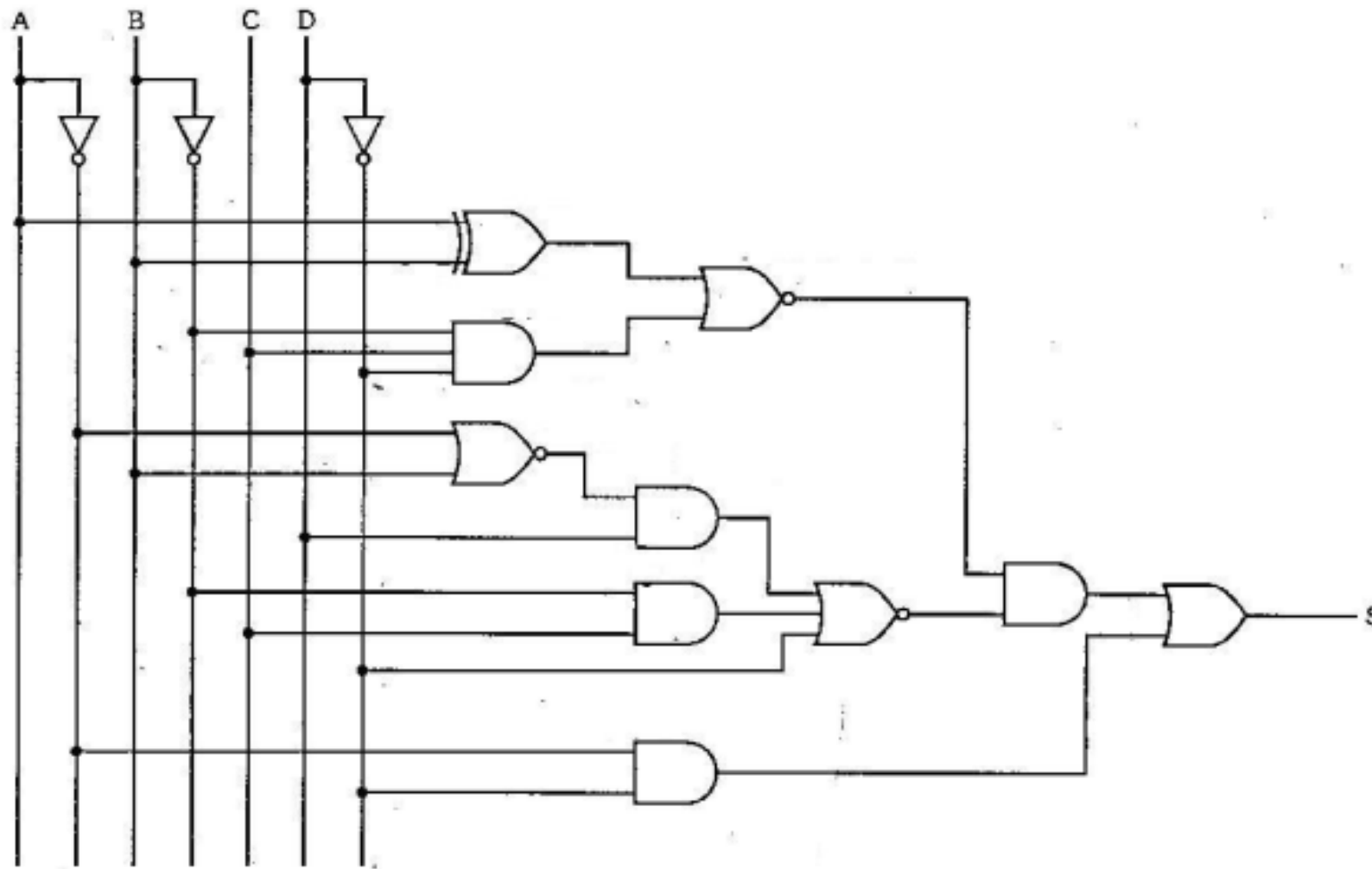
14. Bloco Coincidência (NOR Exclusive)

- A tabela abaixo faz o resumo:

BLOCOS LÓGICOS BÁSICOS																			
Porta	Símbolo Usual	Tabela da Verdade	Função Lógica	Expressão															
OU EXCLUSIVO EXCLUSIVE OR		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	S	0	0	0	0	1	1	1	0	1	1	1	0	Função OU Exclusivo: assume 1 quando as variáveis assumirem valores diferentes entre si.	$S = \overline{A}.B + A.\overline{B}$ $S = A \oplus B$
A	B	S																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	0																	
NOU EXCLUSIVO EXCLUSIVE NOR COINCIDÊNCIA		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	S	0	0	1	0	1	0	1	0	0	1	1	1	Função Coincidência: assume 1 quando houver coincidência entre os valores das variáveis.	$S = \overline{A}.B + A.B$ $S = A \odot B$
A	B	S																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	1																	

14. Bloco Coincidência (NOR Exclusive)

- Exercício: Determine a expressão e a tabela verdade do circuito abaixo:



15. Equivalência de Portas Lógicas

- Quando um circuito eletrônico é projetado, ele deve ser capaz de fazer sua função com o máximo de eficiência possível (menor custo).
- Desta forma, é possível trocar determinadas portas lógicas por outras, com o objetivo de reduzir o tamanho do circuito, ou até mesmo aproveitar componentes que eventualmente estejam sobrando.
- Analisaremos algumas das principais equivalências das portas lógicas. É sempre importante analisar a tabela verdade para verificar se a equivalência é real.

16. Inversor a partir da porta NAND (NE)

- Fazendo um curto na NAND, ela faz a função de inversor:

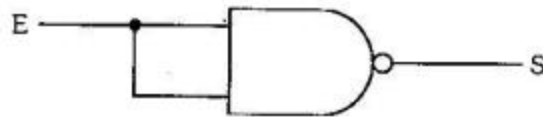
Vamos analisar a tabela da verdade de uma porta NE:

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

Tabela 2.23

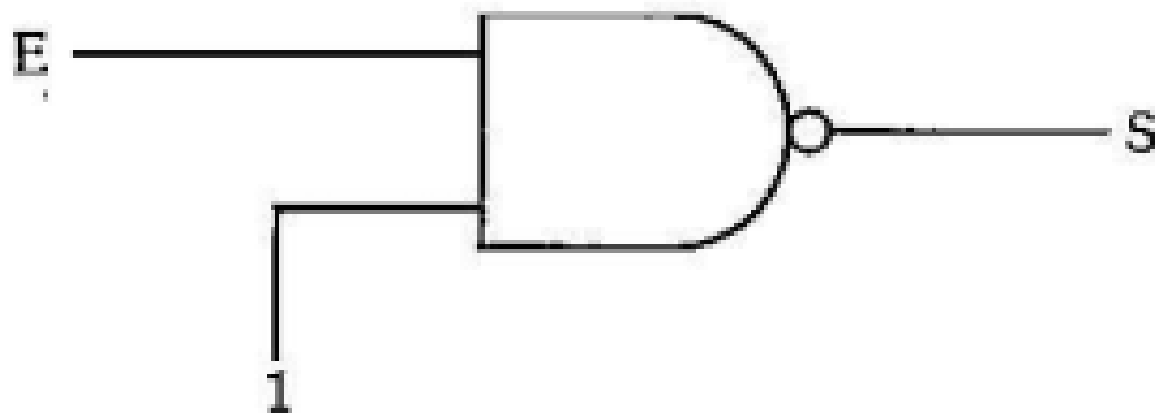
Podemos notar que no caso $A = 0$ e $B = 0$, a saída assume valor 1, e no caso $A = 1$ e $B = 1$, a saída assume valor 0.

Interligando os terminais de entrada da porta, estaremos fornecendo o mesmo nível às 2 entradas ($A = B$). Sendo este nível igual a 0 a saída é igual a 1, e sendo este nível igual a 1, a saída é 0, estando assim, formado um inversor. A figura 2.41 mostra uma porta NE com as entradas curto-circuitadas, formando um inversor, e a tabela 2.24 sua função lógica.



16. Inversor a partir da porta NAND (NE)

- Outra forma de inverter é fixando 1 em umas das portas:



17. Inversor a partir da porta NOR (NOU)

- É similar ao caso anterior:

Analogamente ao caso anterior, vamos analisar a tabela da verdade de uma porta NOU:

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

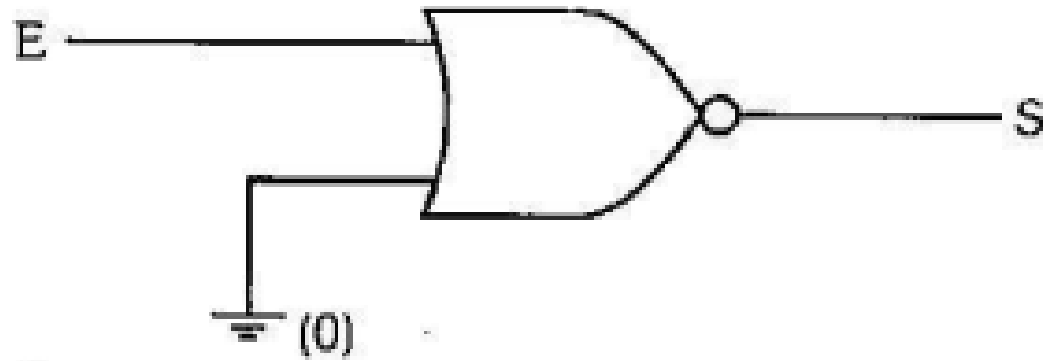
Tabela 2.25

Interligando A e B, cairemos num caso idêntico ao do item anterior, transformando a porta NOU em um inversor. A figura 2.43 e a tabela 2.25 mostram esta situação.



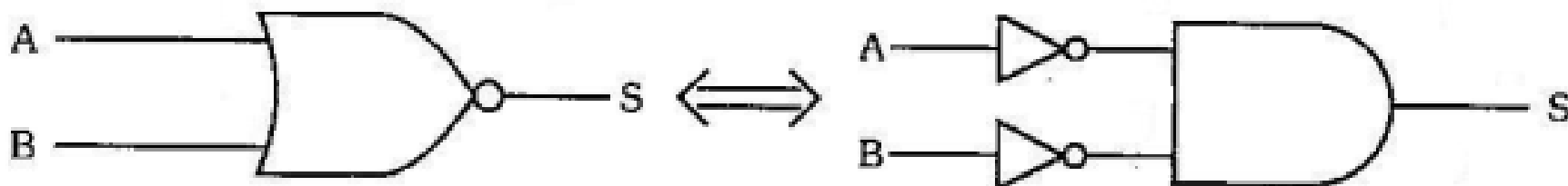
17. Inversor a partir da porta NOR (NOU)

- Também é possível obter um inversor fixando uma das entradas em 0:



17. Porta NOU e OU a partir de E e NE

- Uma porta NOU pode ser formada por um E com suas entradas invertidas:

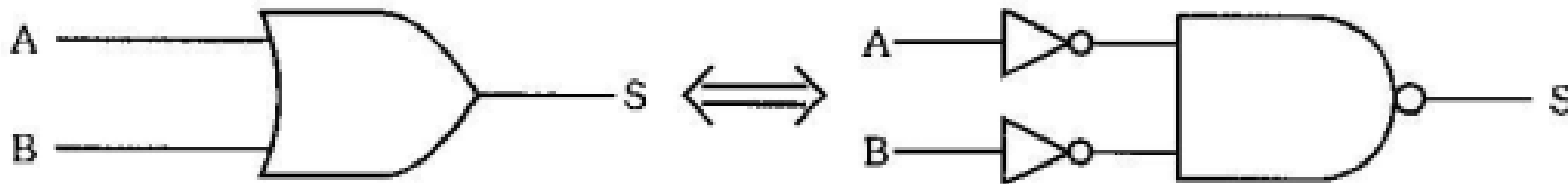


A	B	$\overline{A+B}$	$\overline{A} \cdot \overline{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

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

17. Porta NOU e OU a partir de E e NE

- O circuito abaixo mostra a equivalência entre OU e uma AND com inversores:



18. Porta NE e E a partir de OU e NOU

- Também existem equivalências entre NE e NOU, como mostra a imagem:

A partir da identidade $\overline{A \cdot B} = \overline{A} + \overline{B}$ (outro teorema de De Morgan) obtemos a equivalência entre uma porta NE e a porta OU com suas entradas invertidas. A figura 2.47 mostra esta equivalência e a tabela 2.28 prova a igualdade.

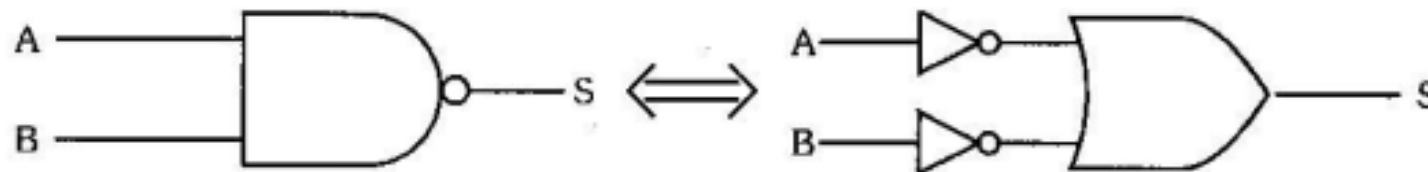
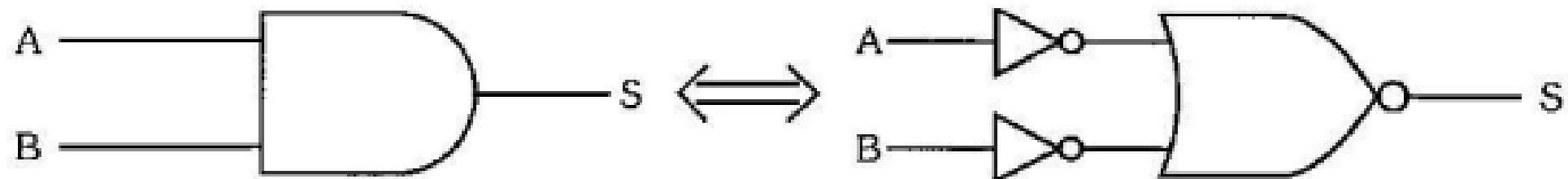


Figura 2.47

A	B	$\overline{A \cdot B}$	$\overline{A} + \overline{B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

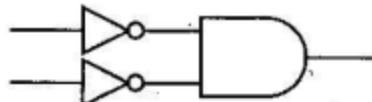
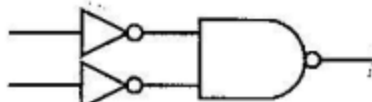
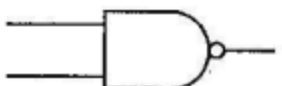
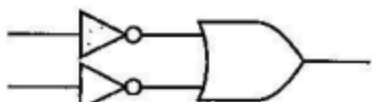
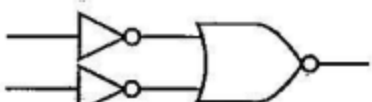
18. Porta NE e E a partir de OU e NOU

- Colocando inversor na saída, obtemos novas equivalências:



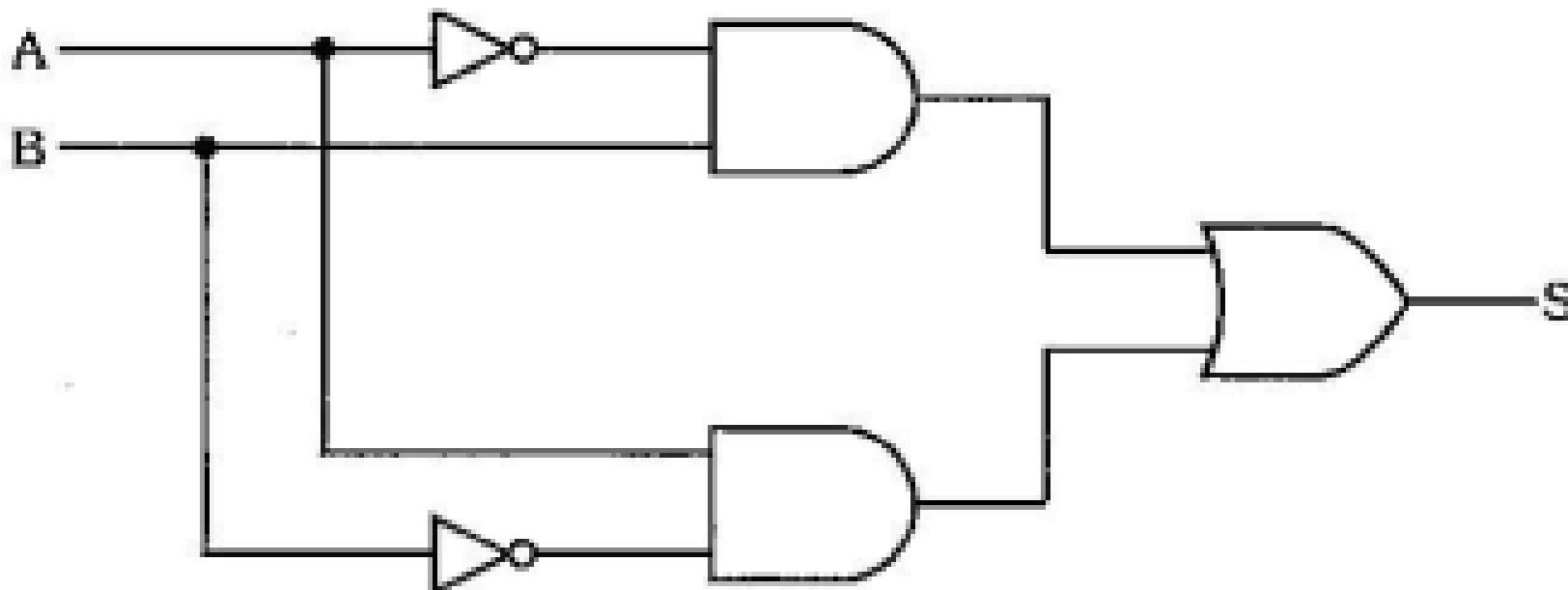
19. Resumo das Equivalências

- Quadro com as principais equivalências:

BLOCO LÓGICO	BLOCO EQUIVALENTE
	   
	
	
	
	

19. Resumo das Equivalências

- Exercício: Desenhe o circuito abaixo utilizando apenas portas NAND(NE):



20. Computação Quântica

- Computação quântica é uma nova forma de processar informações usando as leis da **física quântica**, que são diferentes das leis da física clássica usadas nos computadores de hoje.
- Enquanto os computadores comuns usam **bits**, que só podem estar no estado 0 ou 1, a computação quântica usa **qubits** (bits quânticos), que podem estar em 0, 1 **ou os dois ao mesmo tempo** (isso se chama **superposição**).
- Essas propriedades permitem que os computadores quânticos façam certos tipos de cálculos **muito mais rápido** que os computadores tradicionais.
- As ideias por trás da computação quântica surgiram nos anos **1980**, com cientistas como **Richard Feynman** e **David Deutsch**, que propuseram usar fenômenos da mecânica quântica para simular sistemas físicos — algo que computadores comuns têm dificuldade em fazer.

20. Computação Quântica

- Por que os computadores quânticos são mais rápidos?
- **Superposição: faz várias contas ao mesmo tempo**
- Nos computadores comuns, os bits são 0 **ou** 1 (cada combinação deve ser testada uma por uma)
- Já os qubits podem estar em **superposição**, ou seja, em 0 **e** 1 ao mesmo tempo. Isso permite que um computador quântico **teste muitas combinações ao mesmo tempo**.

Exemplo:

- Imagine que você está tentando abrir um cofre com uma senha de 3 dígitos (de 0 a 9).
- Um computador clássico testa uma senha por vez: 000, 001, 002, etc.
- Um computador quântico pode "testar todas as senhas de uma vez" usando superposição.

20. Computação Quântica

- Entretanto, os computadores Quânticos **não são mais rápidos para tudo**, só para **alguns problemas específicos**, como:
 - Fatoração de números grandes (importante para segurança digital)
 - Otimização (melhor caminho, menor custo)
 - Simulação de moléculas e materiais complexos
 - Para tarefas comuns (como abrir um navegador ou escrever um texto), os computadores tradicionais ainda são melhores e mais práticos.

20. Computação Quântica

- Os países que estão na liderança do desenvolvimento de computadores quânticos são:

Estados Unidos (empresas como IBM, Google, Intel, Microsoft)

China (universidades e centros como a USTC)

Canadá (empresa D-Wave)

Alemanha, Reino Unido e França

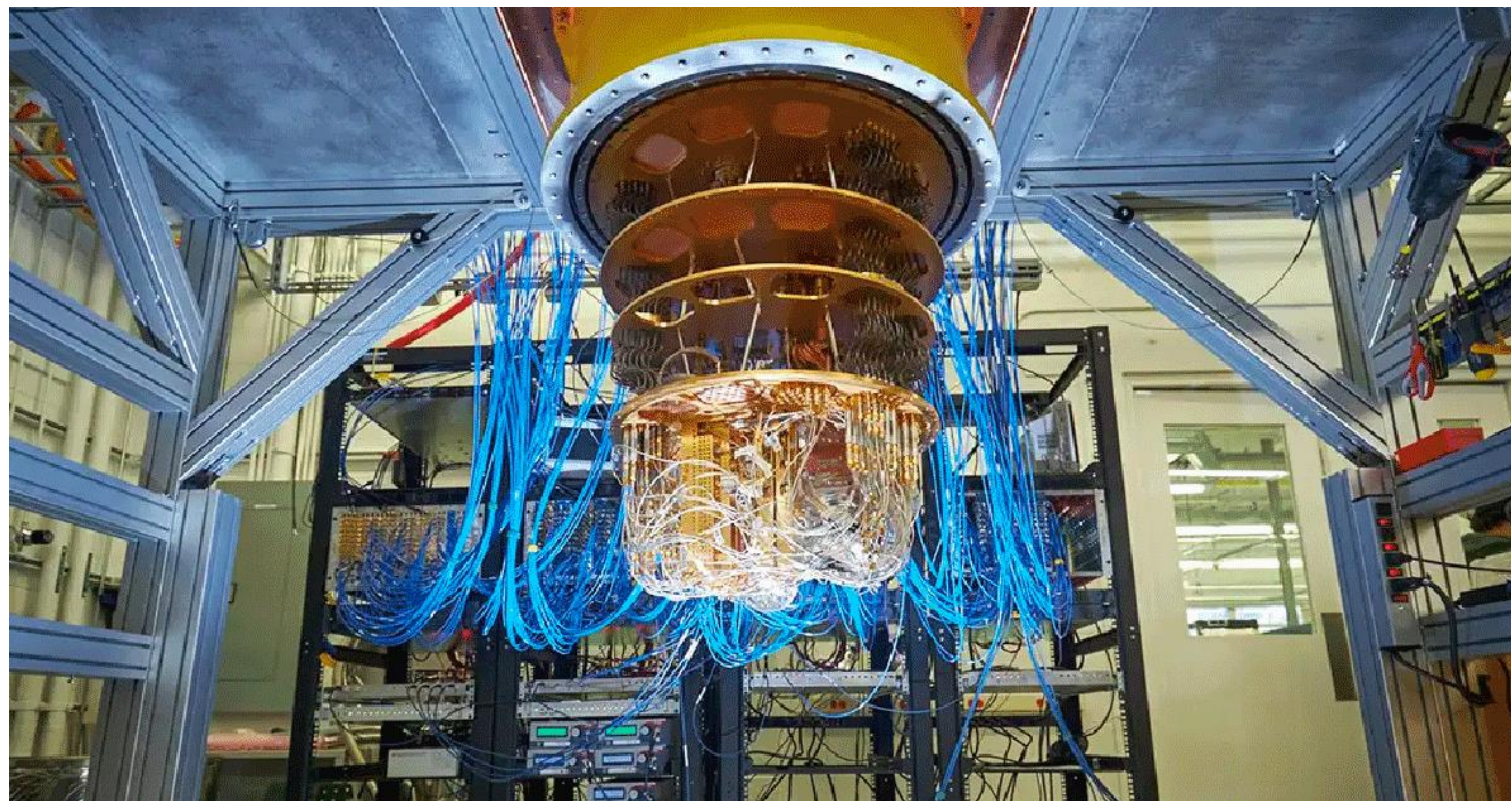
Japão

Holanda

Austrália

20. Computação Quântica

- A imagem abaixo mostra um computador quântico:



21. Benefícios da Computação Quântica

1. Velocidade para certos problemas

- Resolve problemas extremamente complexos **muito mais rápido** que supercomputadores atuais (ex: simulações químicas, otimização, criptografia).
- Pode revolucionar áreas como **medicina, finanças, logística e inteligência artificial**.

2. Simulação de sistemas quânticos

- Pode simular moléculas e materiais em nível atômico com alta precisão.
- Isso ajuda no **desenvolvimento de novos medicamentos**, baterias e materiais.

21. Benefícios da Computação Quântica

3. Otimização avançada

- Útil para resolver problemas de logística, como rotas de entrega, planejamento de produção e redes de energia.
- Pode trazer ganhos enormes de **eficiência**.

4. Criação de algoritmos inovadores

- Algoritmos como **Shor** (fatoração de números grandes) e **Grover** (busca em banco de dados) mostram que o paradigma quântico oferece **formas totalmente novas** de resolver problemas.

22. Problemas da Computação Quântica

1. Ameaça à segurança digital atual

- Algoritmos quânticos podem **quebrar sistemas de criptografia** usados hoje (como RSA e ECC), colocando em risco **dados bancários, militares e pessoais**.
- Isso levou à criação da **criptografia pós-quântica**.

2. Custo e complexidade

- Computadores quânticos são **extremamente caros** e frágeis.
- Operam em condições extremas (temperaturas próximas do zero absoluto).

22. Problemas da Computação Quântica

3. Uso indevido

- Como toda tecnologia poderosa, pode ser usada para fins **maliciosos**, como espionagem digital em larga escala ou ataques a infraestruturas críticas.

4. Desigualdade tecnológica

- Países ou empresas com acesso exclusivo à computação quântica poderiam **concentrar poder**, gerando desequilíbrios geopolíticos ou econômicos.

22. Problemas da Computação Quântica

5. Erro e instabilidade dos qubits

- Qubits são **extremamente sensíveis** a ruídos externos (como luz, vibração ou campos magnéticos).
- Isso gera **erros de cálculo** com muita facilidade.

6. Infraestrutura extrema

- A maioria dos computadores quânticos precisa funcionar em **temperaturas criogênicas** (quase -273°C).
- Isso torna os sistemas difíceis de manter, escalar e transportar.

22. Problemas da Computação Quântica

7. Falta de aplicações práticas no curto prazo

- A computação quântica ainda está **longe de substituir ou complementar computadores clássicos em larga escala.**
- A maioria dos algoritmos práticos ainda está sendo **teorizada ou testada em simulações.**
- **Poucos problemas reais** hoje tiram proveito de um computador quântico com vantagem clara.

8. Escassez de profissionais qualificados

- É uma área **altamente interdisciplinar**, envolvendo física, matemática, ciência da computação e engenharia.

22. Problemas da Computação Quântica

9. Risco de "hype" e decepção

- A computação quântica sofre com o excesso de promessas exageradas (o famoso *hype*).
- Investimentos podem diminuir se o mercado achar que é só uma “moda”.

10. Privacidade e espionagem

- Estados que dominarem essa tecnologia poderão **decifrar comunicações criptografadas**, o que pode ser usado para **espionagem governamental ou corporativa**.

11. Desigualdade digital e tecnológica

- Países em desenvolvimento podem ficar ainda mais atrás no acesso a tecnologias de ponta.
- Pode aumentar a **dependência de grandes potências tecnológicas**.