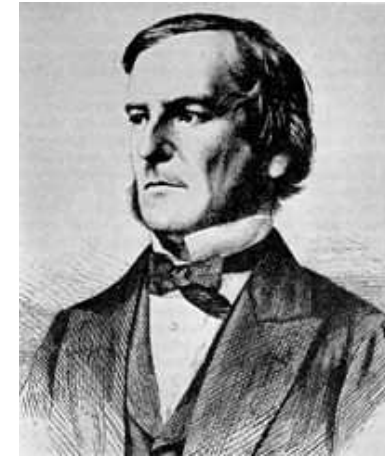
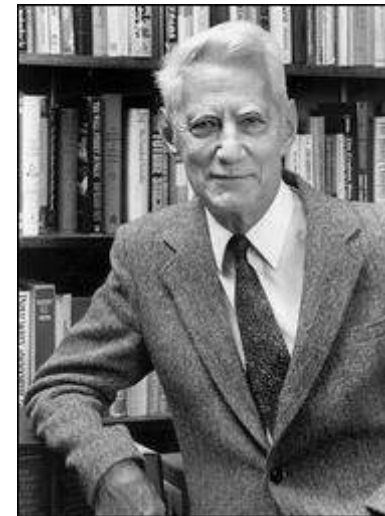


Histórico

- ❑ Em meados do século XIX o matemático inglês George **Boole** desenvolveu um sistema matemático de análise lógica
- ❑ Em meados do século XX, o americano Claude Elwood **Shannon** sugeriu que a Álgebra Booleana poderia ser usada para análise e projeto de circuitos de comutação



George Boole (1815-1864)



Claude Elwood Shannon (1916-2001)

Histórico

- ❑ Nos primórdios da eletrônica, todos os problemas eram solucionados por meio de sistemas analógicos
- ❑ Com o avanço da tecnologia, os problemas passaram a ser solucionados pela eletrônica digital
- ❑ Na eletrônica digital, os sistemas (computadores, processadores de dados, sistemas de controle, codificadores, decodificadores, etc) empregam um pequeno grupo de circuitos lógicos básicos, que são conhecidos como portas **e**, **ou**, **não** e **flip-flop**
- ❑ Com a utilização adequadas dessas portas é possível implementar todas as expressões geradas pela álgebra de Boole

Álgebra Booleana

- ❑ Na álgebra de Boole, há somente dois **estados** (**valores** ou **símbolos**) permitidos
 - Estado **0** (zero)
 - Estado **1** (um)
- ❑ Em geral
 - O estado zero representa **não, falso**, aparelho desligado, ausência de tensão, chave elétrica desligada, etc
 - O estado um representa **sim, verdadeiro**, aparelho ligado, presença de tensão, chave ligada, etc

Álgebra Booleana

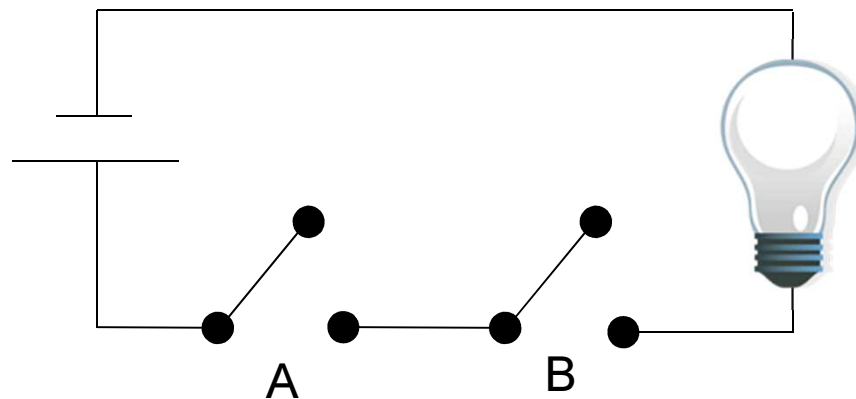
- ❑ Assim, na álgebra booleana, se representarmos por 0 uma situação, a situação contrária é representada por 1
- ❑ Portanto, em qualquer bloco (porta ou função) lógico somente esses dois estados (0 ou 1) são permitidos em suas entradas e saídas
- ❑ Uma variável booleana também só assume um dos dois estados permitidos (0 ou 1)

Álgebra Booleana

- ❑ Nesta apresentação trataremos dos seguintes blocos lógicos
 - E (AND)
 - OU (OR)
 - NÃO (NOT)
 - NÃO E (NAND)
 - NÃO OU (NOR)
 - OU EXCLUSIVO (XOR)
- ❑ Após, veremos a correspondência entre expressões, circuitos e tabelas verdade
- ❑ Por último, veremos a equivalência entre blocos lógicos

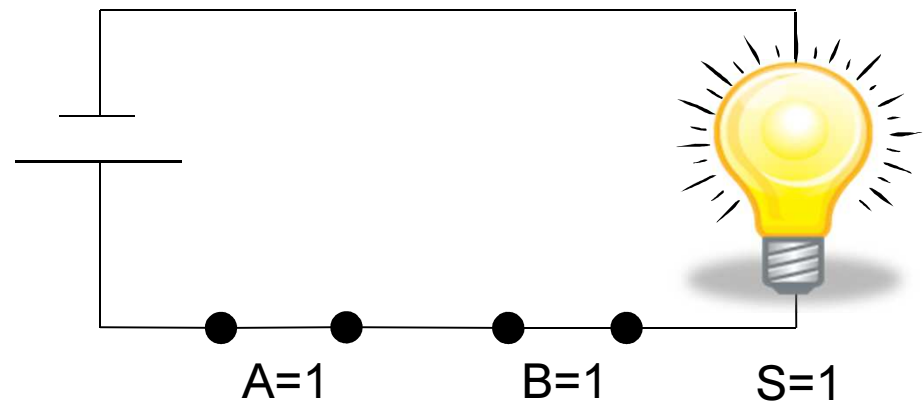
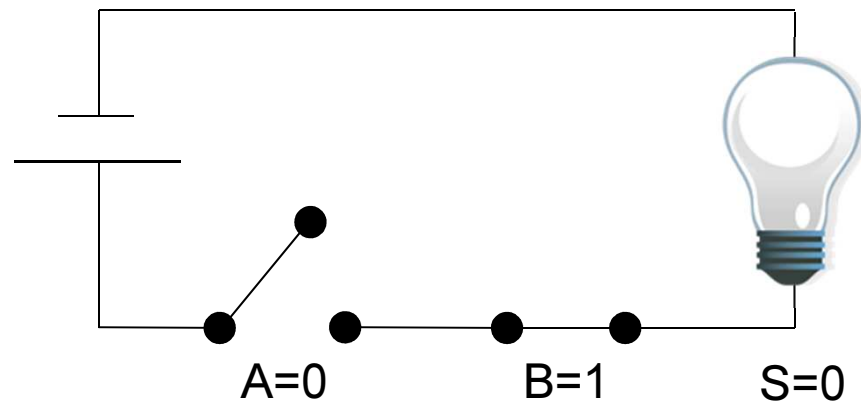
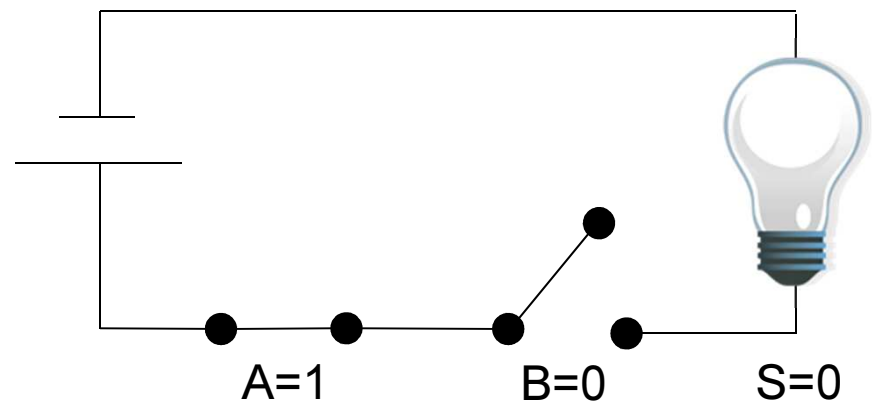
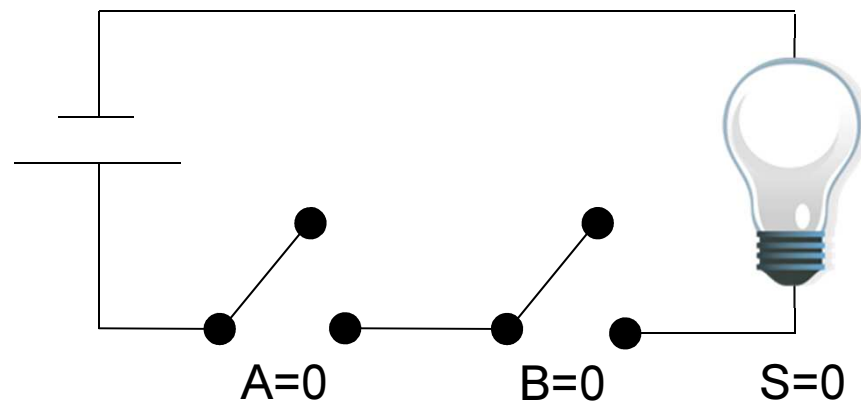
Função **E** (**AND**)

- ❑ Executa a **multiplicação (conjunção)** booleana de duas ou mais variáveis binárias
- ❑ Por exemplo, assumamos a convenção no circuito
 - Chave aberta = 0; Chave fechada = 1
 - Lâmpada apagada = 0; Lâmpada acesa = 1



Função **E** (AND)

□ Situações possíveis:



Função **E** (**AND**)

- ❑ Se a chave A está aberta ($A=0$) e a chave B aberta ($B=0$), não haverá circulação de energia no circuito, logo a lâmpada fica apagada ($S=0$)
- ❑ Se a chave A está fechada ($A=1$) e a chave B aberta ($B=0$), não haverá circulação de energia no circuito, logo a lâmpada fica apagada ($S=0$)
- ❑ Se a chave A está aberta ($A=0$) e a chave B fechada ($B=1$), não haverá circulação de energia no circuito, logo a lâmpada fica apagada ($S=0$)
- ❑ Se a chave A está fechada ($A=1$) e a chave B fechada ($B=1$), haverá circulação de energia no circuito e a lâmpada fica acesa ($S=1$)
- ❑ Observando todas as quatro situações possíveis (interpretações), é possível concluir que a lâmpada fica acesa somente quando as chaves A e B estiverem simultaneamente fechadas ($A=1$ e $B=1$)

Função **E** (**AND**)

- Para representar a expressão
 - $S = A \mathbf{e} B$
- Adotaremos a representação
 - $S = A.B$, onde se lê $S = A \mathbf{e} B$
- Porém, existem notações alternativas
 - $S = A \& B$
 - $S = A, B$
 - $S = A \wedge B$

Tabela Verdade

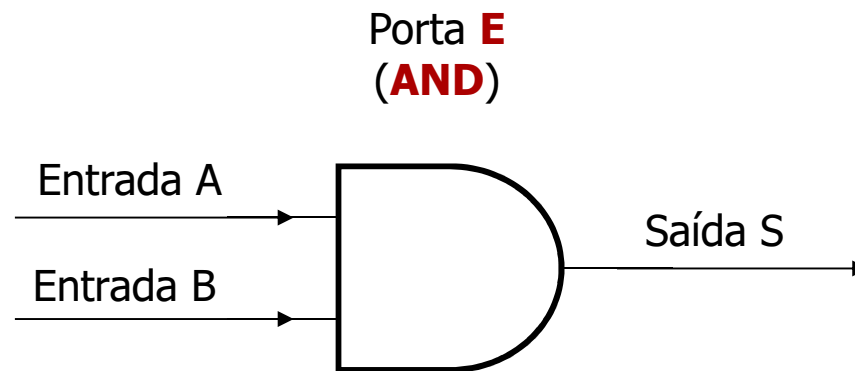
- ❑ A tabela verdade é um mapa onde são colocadas todas as possíveis interpretações (situações), com seus respectivos resultados para uma expressão booleana qualquer
- ❑ Como visto no exemplo anterior, para 2 variáveis booleanas (A e B), há 4 interpretações possíveis
- ❑ Em geral, para N variáveis booleanas de entrada, há 2^N interpretações possíveis

Tabela Verdade da Função **E** (**AND**)

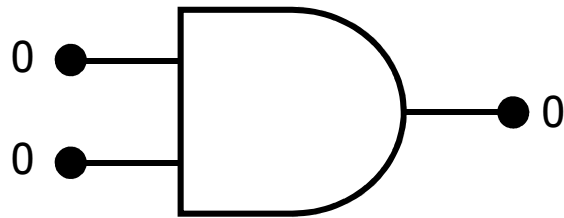
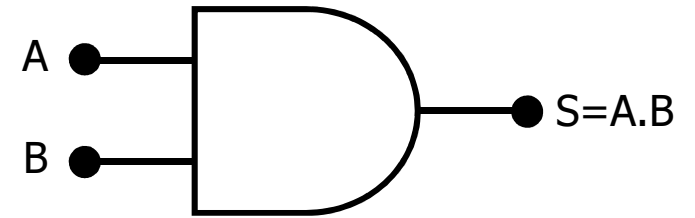
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

Porta Lógica **E** (**AND**)

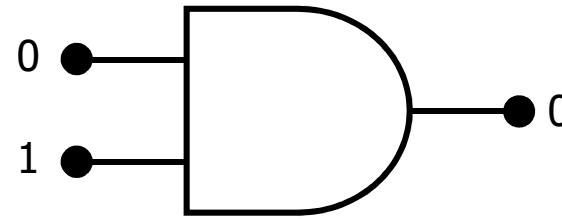
- ❑ A porta **E** é um circuito que executa a função **E**
- ❑ A porta **E** executa a tabela verdade da função **E**
 - Portanto, a saída será 1 somente se ambas as entradas forem iguais a 1; nos demais casos, a saída será 0
- ❑ Representação



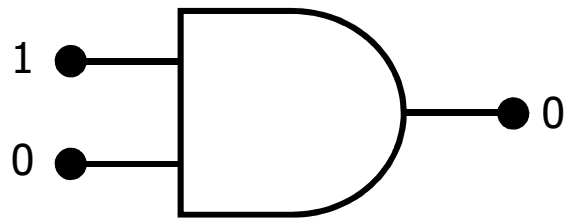
Porta Lógica **E** (AND)



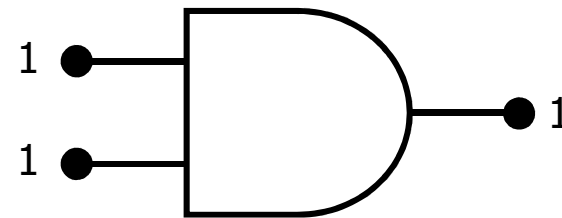
A	B	S=A.B
0	0	0
0	1	0
1	0	0
1	1	1



A	B	S=A.B
0	0	0
0	1	0
1	0	0
1	1	1



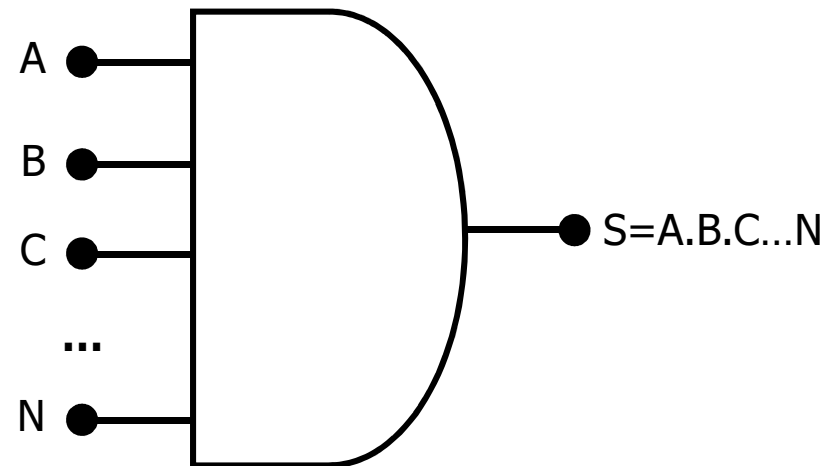
A	B	S=A.B
0	0	0
0	1	0
1	0	0
1	1	1



A	B	S=A.B
0	0	0
0	1	0
1	0	0
1	1	1

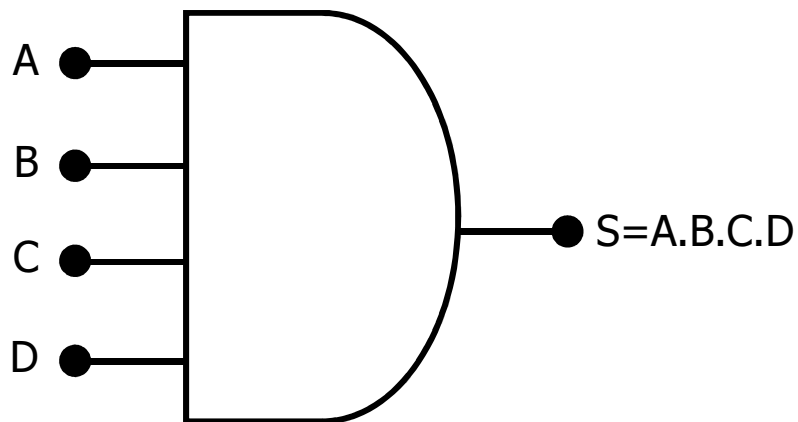
Porta Lógica **E** (**AND**)

- ❑ É possível estender o conceito de uma porta **E** para um número qualquer de variáveis de entrada
- ❑ Nesse caso, temos uma porta **E** com N entradas e somente uma saída
- ❑ A saída será 1 se e somente se as N entradas forem iguais a 1; nos demais casos, a saída será 0



Porta Lógica **E** (**AND**)

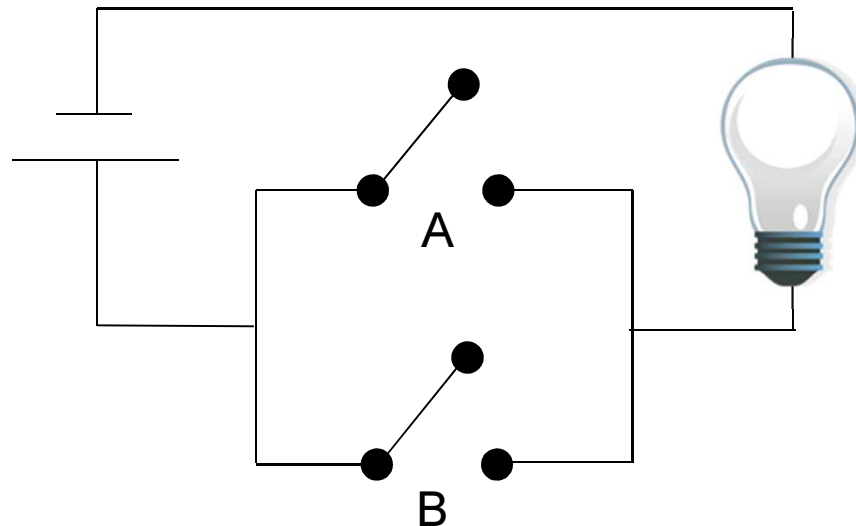
- Por exemplo,
 $S=A.B.C.D$



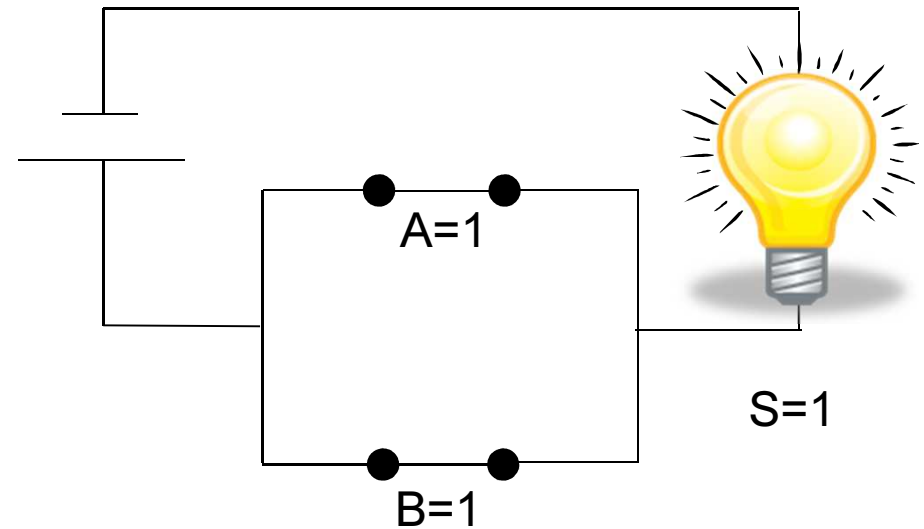
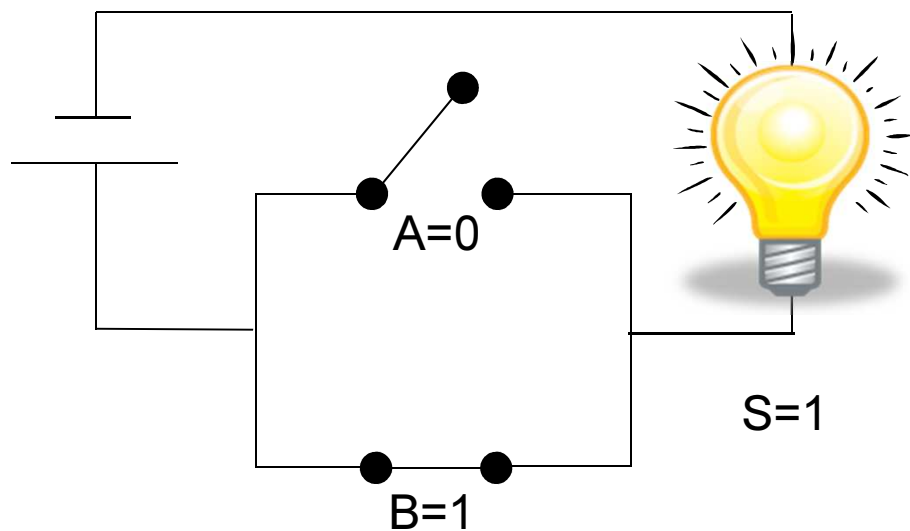
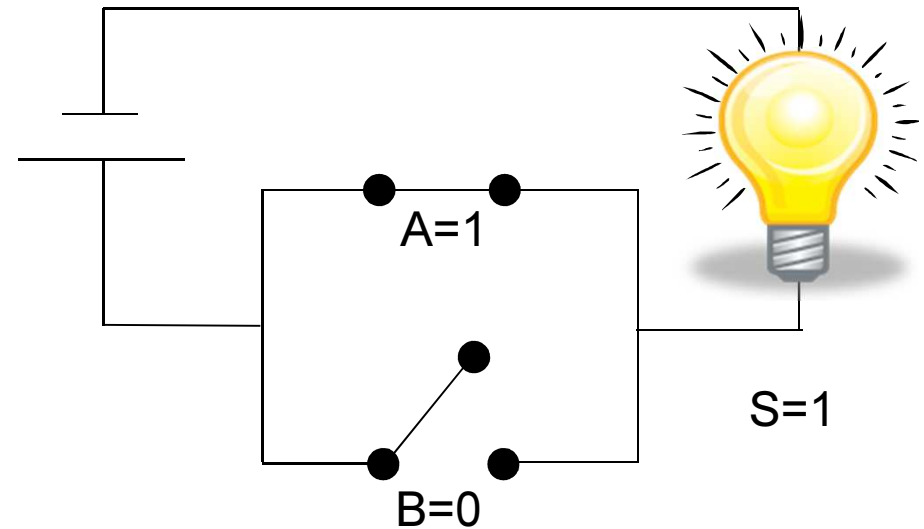
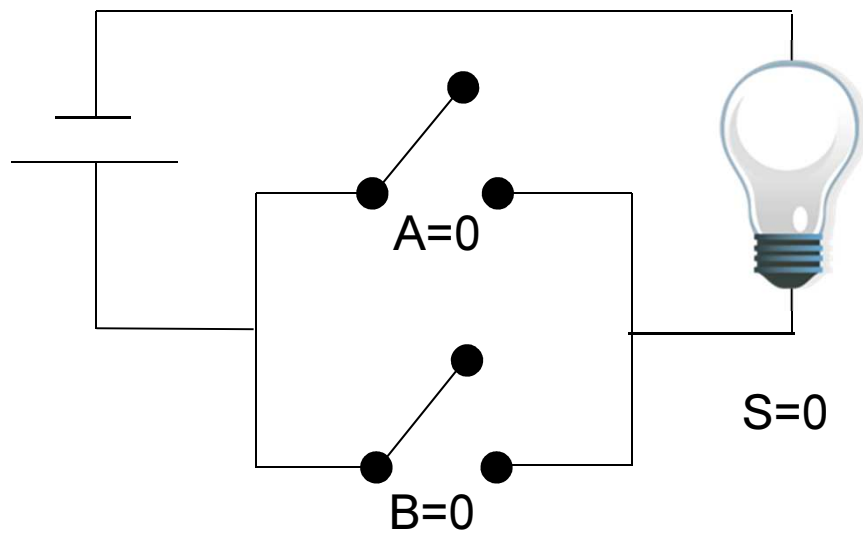
A	B	C	D	S
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Função **OU** (**OR**)

- ❑ Executa a **soma (disjunção)** booleana de duas ou mais variáveis binárias
- ❑ Por exemplo, assumamos a convenção no circuito
 - Chave aberta = 0; Chave fechada = 1
 - Lâmpada apagada = 0; Lâmpada acesa = 1



Função **OU** (OR)



Função **OU** (**OR**)

- ❑ Se a chave A está aberta ($A=0$) e a chave B aberta ($B=0$), não haverá circulação de energia no circuito, logo a lâmpada fica apagada ($S=0$)
- ❑ Se a chave A está fechada ($A=1$) e a chave B aberta ($B=0$), haverá circulação de energia no circuito e a lâmpada fica acesa ($S=1$)
- ❑ Se a chave A está aberta ($A=0$) e a chave B fechada ($B=1$), haverá circulação de energia no circuito e a lâmpada fica acesa ($S=1$)
- ❑ Se a chave A está fechada ($A=1$) e a chave B fechada ($B=1$), haverá circulação de energia no circuito e a lâmpada fica acesa ($S=1$)
- ❑ Observando todas as quatro situações possíveis, é possível concluir que a lâmpada fica acesa somente quando a chave A ou a chave B ou ambas estiverem fechadas

Função **OU** (**OR**)

- Para representar a expressão
 - $S = A \text{ ou } B$
- Adotaremos a representação
 - $S = A+B$, onde se lê $S = A \text{ ou } B$
- Porém, existem notações alternativas
 - $S = A | B$
 - $S = A; B$
 - $S = A \vee B$

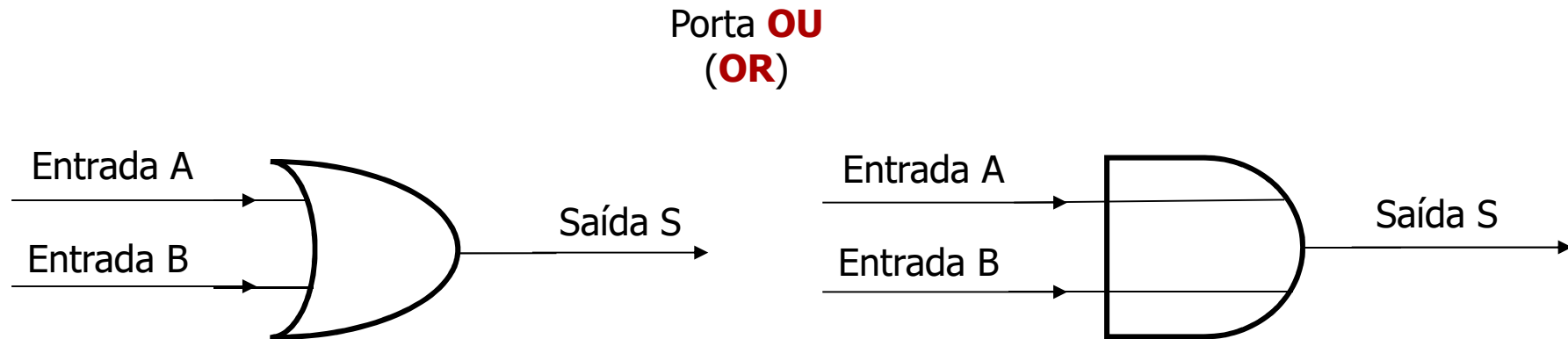
Tabela Verdade da Função **OU** (**OR**)

- ❑ Observe que, no sistema de numeração binário, a soma $1+1=10$
- ❑ Na álgebra booleana, $1+1=1$, já que somente dois valores são permitidos (0 e 1)

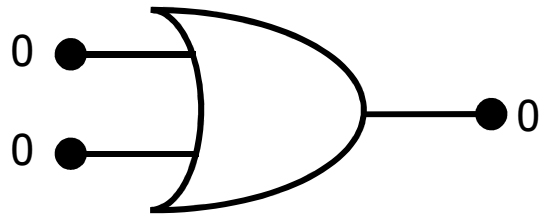
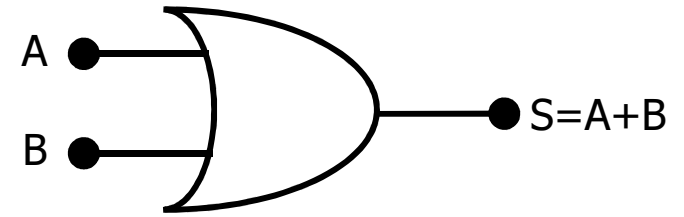
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

Porta Lógica **OU** (**OR**)

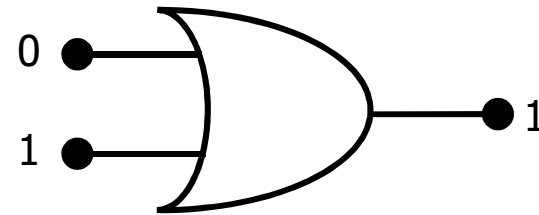
- ❑ A porta **OU** é um circuito que executa a função **OU**
- ❑ A porta **OU** executa a tabela verdade da função **OU**
 - Portanto, a saída será 0 somente se ambas as entradas forem iguais a 0; nos demais casos, a saída será 1
- ❑ Representação



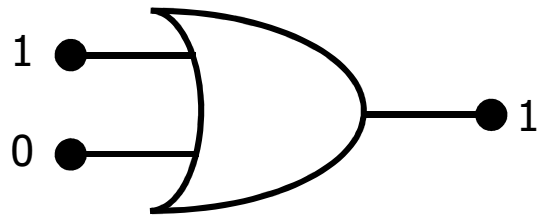
Porta Lógica **OU (OR)**



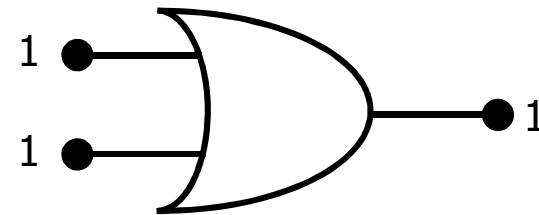
A	B	S=A+B
0	0	0
0	1	1
1	0	1
1	1	1



A	B	S=A+B
0	0	0
0	1	1
1	0	1
1	1	1



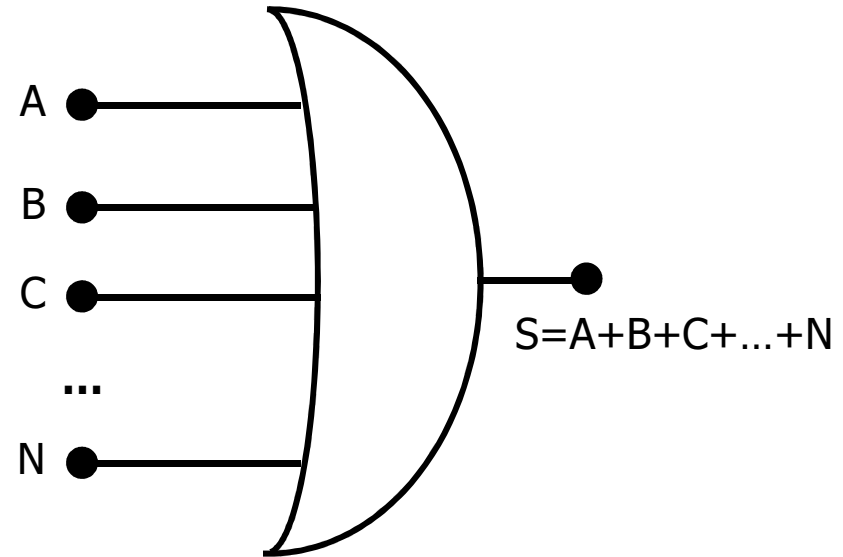
A	B	S=A+B
0	0	0
0	1	1
1	0	1
1	1	1



A	B	S=A+B
0	0	0
0	1	1
1	0	1
1	1	1

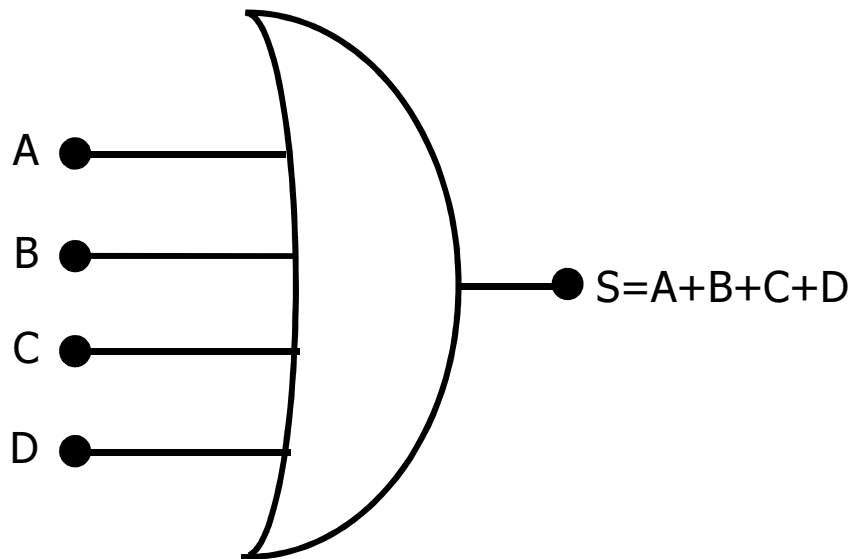
Porta Lógica **OU** (**OR**)

- ❑ É possível estender o conceito de uma porta **OU** para um número qualquer de variáveis de entrada
- ❑ Nesse caso, temos uma porta **OU** com N entradas e somente uma saída
- ❑ A saída será 0 se e somente se as N entradas forem iguais a 0; nos demais casos, a saída será 1



Porta Lógica **OU (OR)**

□ Por exemplo,
 $S=A+B+C+D$



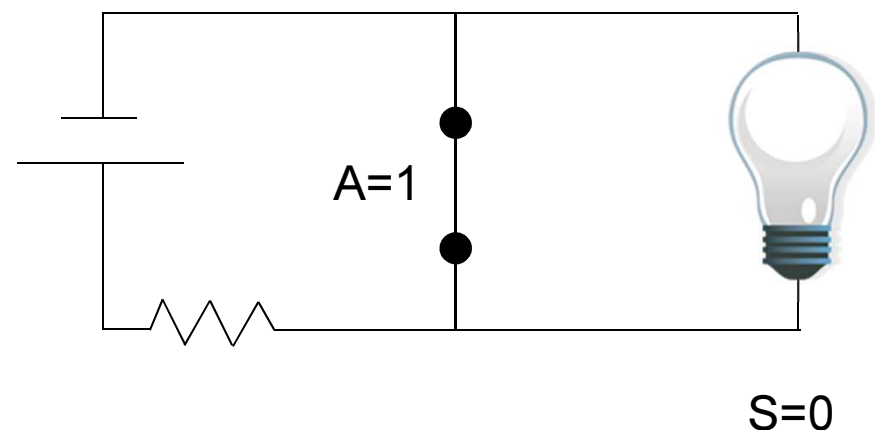
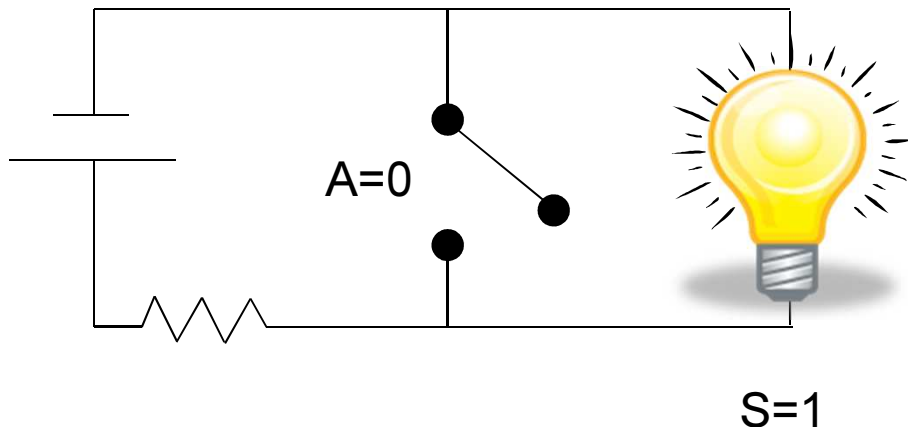
A	B	C	D	S
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Função **NÃO** (**NOT**)

- ❑ Executa o **complemento** (**negação**) de uma variável binária
 - Se a variável estiver em 0, o resultado da função é 1
 - Se a variável estiver em 1, o resultado da função é 0
- ❑ Essa função também é chamada de **inversora**

Função **NÃO** (**NOT**)

- ❑ Usando as mesmas convenções dos circuitos anteriores, tem-se que:
 - Quando a chave A está aberta ($A=0$), passará corrente pela lâmpada e ela acenderá ($S=1$)
 - Quando a chave A está fechada ($A=1$), a lâmpada estará em curto-circuito e não passará corrente por ela, ficando apagada ($S=0$)



Função **NÃO** (**NOT**)

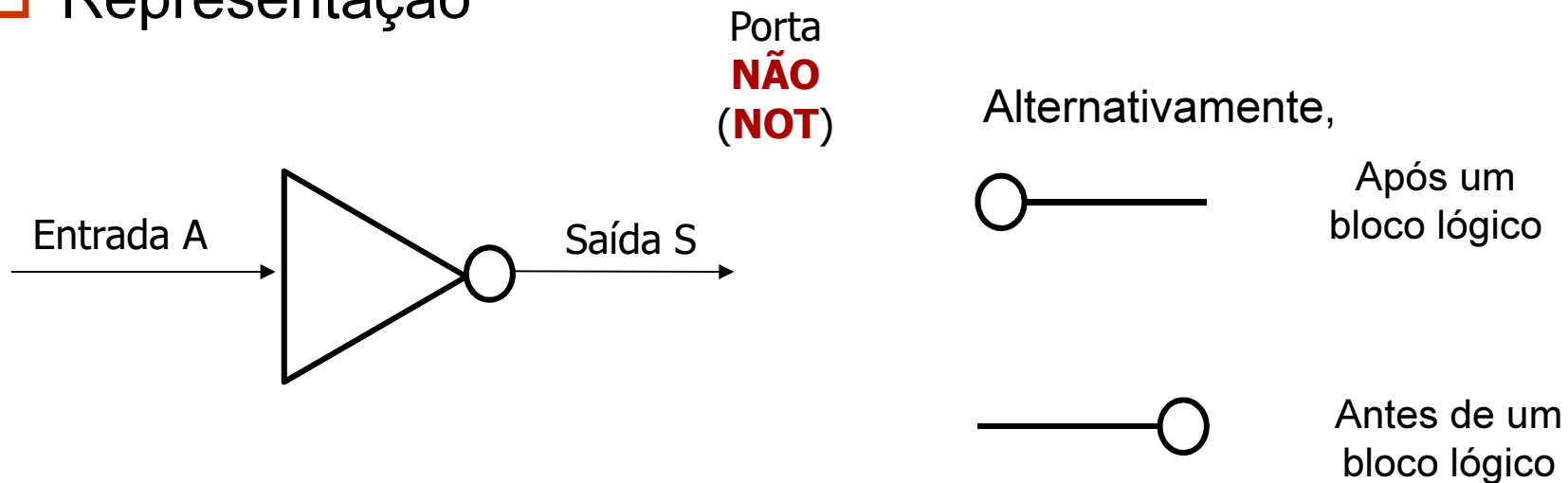
- ❑ Para representar a expressão
 - $S = \text{não } A$
- ❑ Adotaremos a representação
 - $S = \bar{A}$, onde se lê $S = \text{não } A$
- ❑ Notações alternativas
 - $S = A'$
 - $S = \neg A$
 - $S = \bar{A}$

- ❑ Tabela verdade da função **NÃO** (**NOT**)

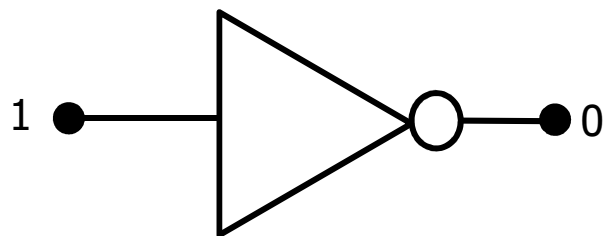
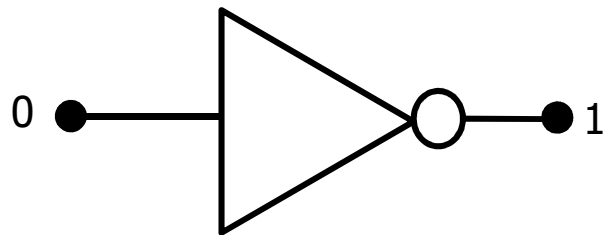
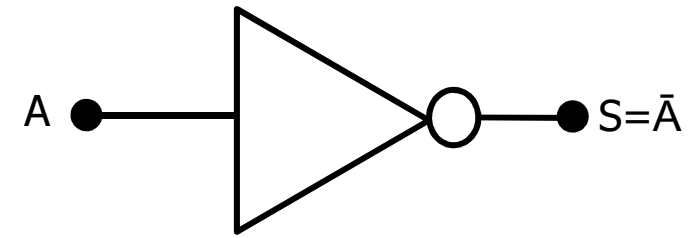
A	\bar{A}
0	1
1	0

Porta Lógica **NÃO** (**NOT**)

- ❑ A porta lógica **NÃO**, ou **inversor**, é o circuito que executa a função **NÃO**
- ❑ O inversor executa a tabela verdade da função **NÃO**
 - Se a entrada for 0, a saída será 1; se a entrada for 1, a saída será 0
- ❑ Representação



Porta Lógica **NÃO** (NOT)



A	S=Ā
0	1
1	0

A	S=Ā
0	1
1	0

Função **NÃO E** (NAND)

□ Composição da função **E** com a função **NÃO**, ou seja, a saída da função **E** é invertida

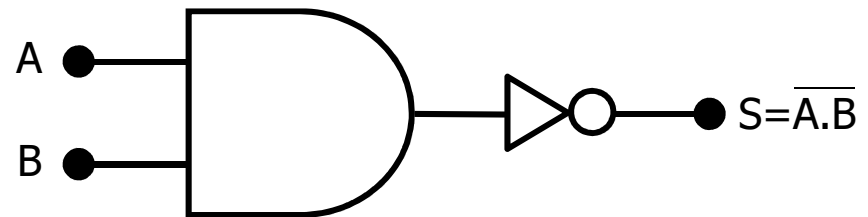
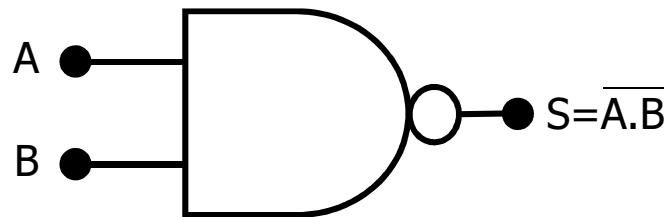
$$\begin{aligned} \square S &= \overline{(A.B)} = \overline{A.B} \\ &= (A.B)' \\ &= \neg(A.B) \end{aligned}$$

□ Tabela verdade

A	B	$S = \overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

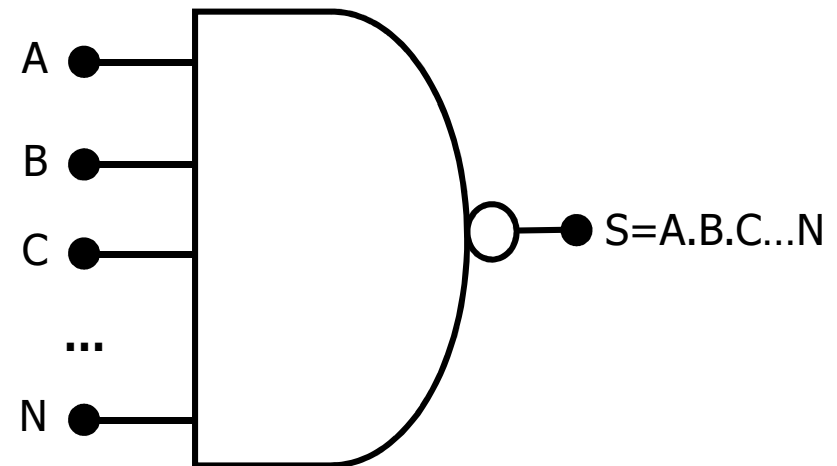
Porta **NÃO E** (NAND)

- ❑ A porta **NÃO E** (NE) é o bloco lógico que executa a função **NÃO E**, ou seja, sua tabela verdade
- ❑ Representação



Porta **NÃO E** (NAND)

- ❑ Como a porta **E**, a porta **NÃO E** pode ter duas ou mais entradas
- ❑ Nesse caso, temos uma porta **NÃO E** com N entradas e somente uma saída
- ❑ A saída será 0 se e somente se as N entradas forem iguais a 1; nos demais casos, a saída será 1



Função **NÃO OU (NOR)**

❑ Composição da função **OU** com a função **NÃO**, ou seja, a saída da função **OU** é invertida

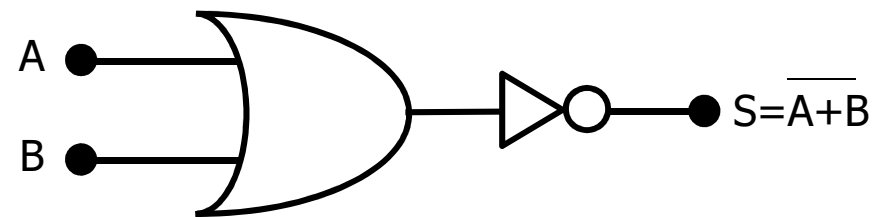
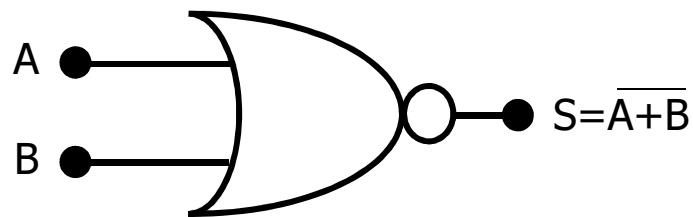
❑
$$S = \overline{(A+B)} = \overline{A+B}$$
$$= (A+B)'$$
$$= \neg(A+B)$$

❑ Tabela verdade

A	B	$S = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

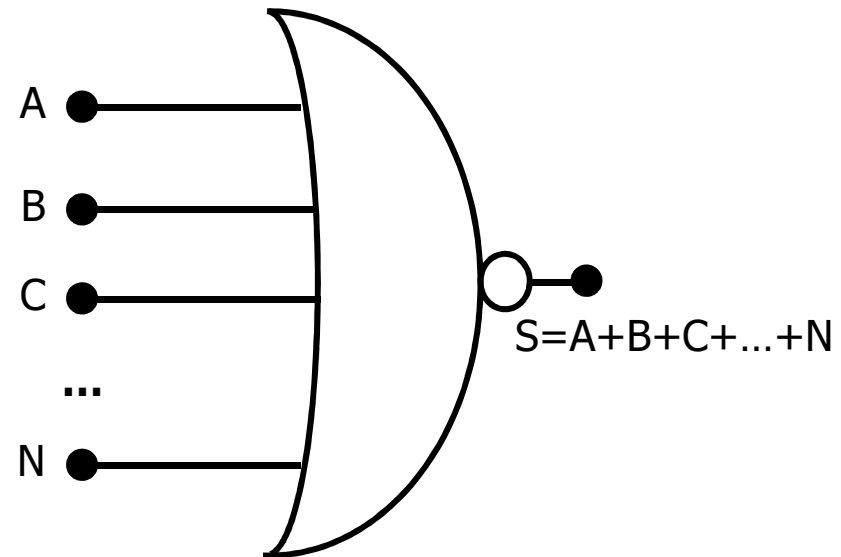
Porta **NÃO OU (NOR)**

- ❑ A porta **NÃO OU (NOR)** é o bloco lógico que executa a função **NÃO OU**, ou seja, sua tabela verdade
- ❑ Representação



Porta **NÃO OU** (**NOR**)

- ❑ Como a porta **OU**, a porta **NÃO OU** pode ter duas ou mais entradas
- ❑ Nesse caso, temos uma porta **NÃO OU** com N entradas e somente uma saída
- ❑ A saída será 1 se e somente se as N entradas forem iguais a 0; nos demais casos, a saída será 0



Função **OU Exclusivo (XOR)**

- A função **OU Exclusivo** fornece
 - 1 na saída quando as entradas forem diferentes entre si e
 - 0 caso contrário

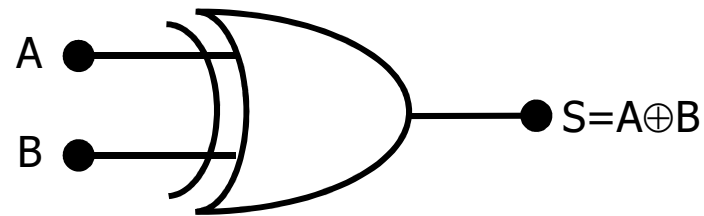
- $S = A \oplus B$
 $= \bar{A}.B + A.\bar{B}$

- Tabela verdade

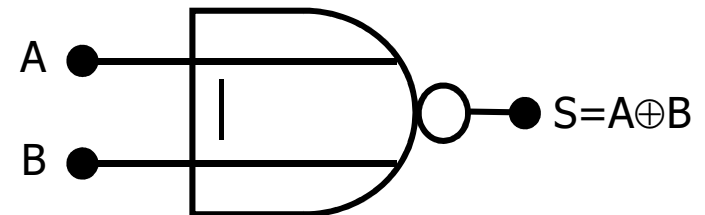
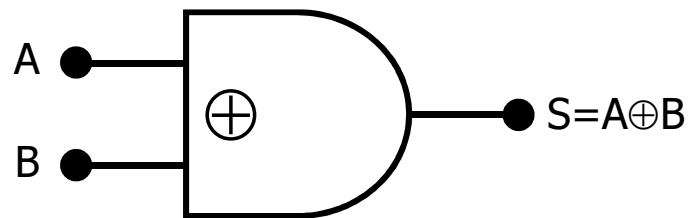
A	B	$S=A\oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Porta **OU Exclusivo** (**XOR**) como Bloco Básico

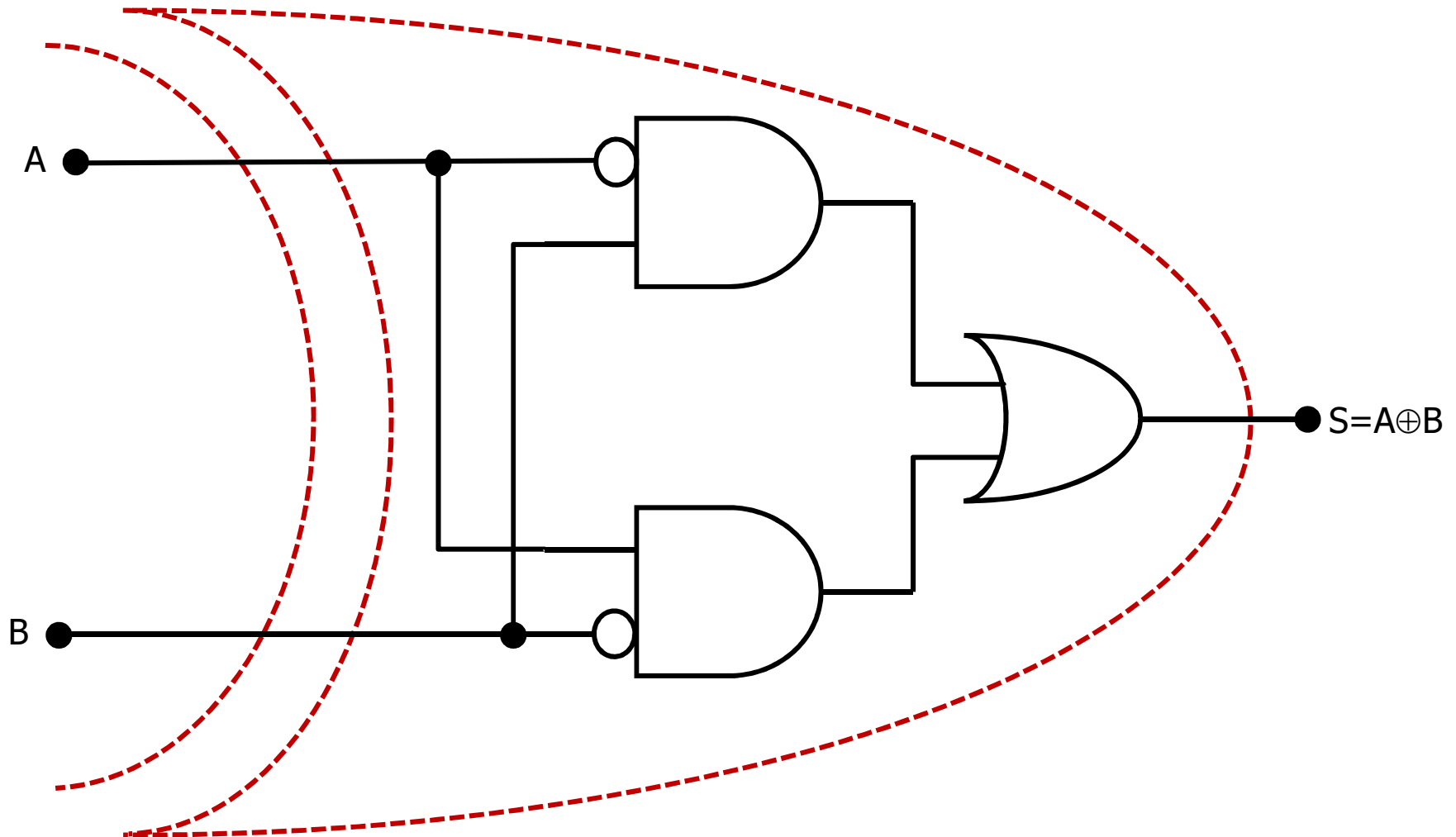
Simbologia adotada



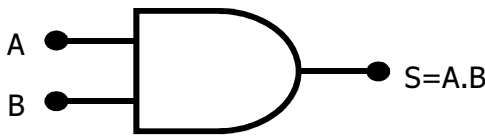
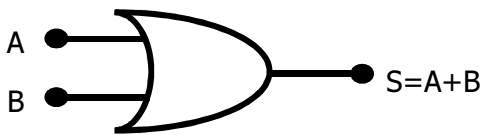
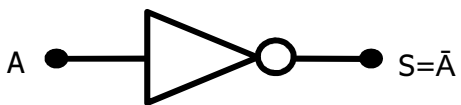
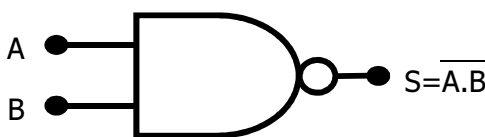
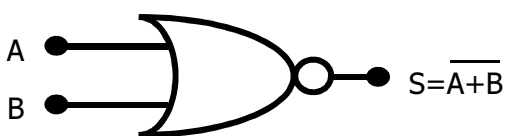
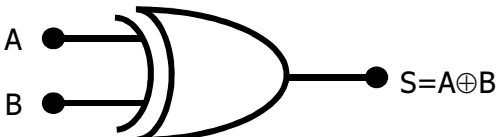
Outros símbolos utilizados



Porta **OU Exclusivo (XOR)** como Circuito Combinacional



Resumo dos Blocos Lógicos Básicos

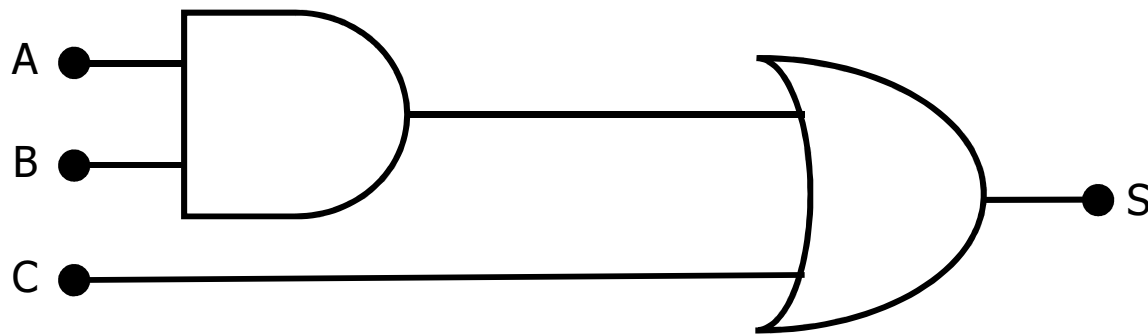
Nome	Símbolo Gráfico	Função Algébrica	Tabela Verdade															
E (AND)		$S=A.B$ $S=AB$	<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>S=A.B</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </tbody> </table>	A	B	S=A.B	0	0	0	0	1	0	1	0	0	1	1	1
A	B	S=A.B																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OU (OR)		$S=A+B$	<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>S=A+B</th> </tr> </thead> <tbody> <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> </tbody> </table>	A	B	S=A+B	0	0	0	0	1	1	1	0	1	1	1	1
A	B	S=A+B																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
NÃO (NOT) Inversor		$S=\bar{A}$ $S=A'$ $S=\neg A$	<table border="1"> <thead> <tr> <th>A</th> <th>S=\bar{A}</th> </tr> </thead> <tbody> <tr><td>0</td><td>1</td></tr> <tr><td>1</td><td>0</td></tr> </tbody> </table>	A	S= \bar{A}	0	1	1	0									
A	S= \bar{A}																	
0	1																	
1	0																	
NE (NAND)		$S=\overline{A.B}$ $S=(A.B)'$ $S=\neg(A.B)$	<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>S=$\overline{A.B}$</th> </tr> </thead> <tbody> <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> </tbody> </table>	A	B	S= $\overline{A.B}$	0	0	1	0	1	1	1	0	1	1	1	0
A	B	S= $\overline{A.B}$																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
NOU (NOR)		$S=\overline{A+B}$ $S=(A+B)'$ $S=\neg(A+B)$	<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>S=$\overline{A+B}$</th> </tr> </thead> <tbody> <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> </tbody> </table>	A	B	S= $\overline{A+B}$	0	0	1	0	1	0	1	0	0	1	1	0
A	B	S= $\overline{A+B}$																
0	0	1																
0	1	0																
1	0	0																
1	1	0																
XOR		$S=A\oplus B$	<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>S=A\oplusB</th> </tr> </thead> <tbody> <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> </tbody> </table>	A	B	S=A \oplus B	0	0	0	0	1	1	1	0	1	1	1	0
A	B	S=A \oplus B																
0	0	0																
0	1	1																
1	0	1																
1	1	0																

Correspondência entre expressões, circuitos e tabelas verdade

- ❑ Todo circuito lógico executa uma expressão booleana
- ❑ Um circuito, por mais complexo que seja, é composto pela interligação dos blocos lógicos básicos
- ❑ Veremos, a seguir, como obter as expressões booleanas geradas por um circuito lógico

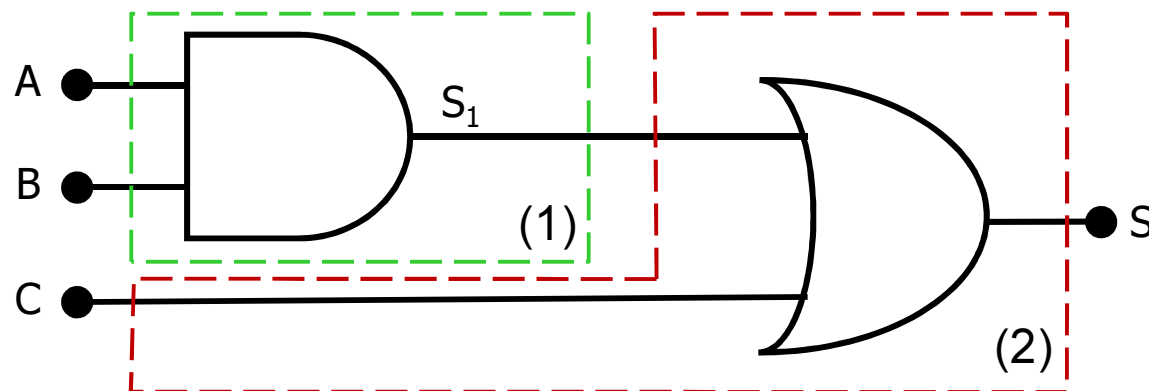
Expressões Booleanas Geradas por Circuitos Lógicos

□ Seja o circuito:



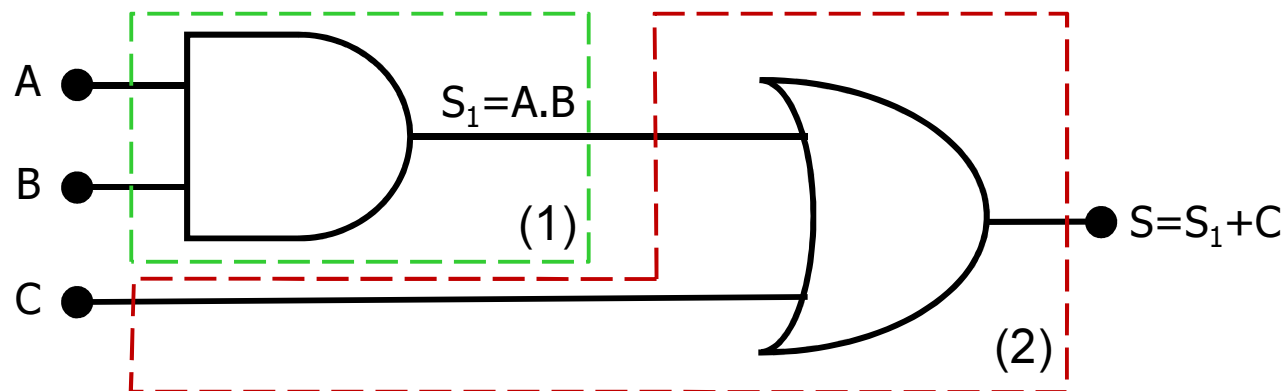
Expressões Booleanas Geradas por Circuitos Lógicos

- Vamos dividi-lo em duas partes (1) e (2)
 - No circuito (1), a saída S_1 contém o produto $A.B$, já que o bloco é uma porta **E**
 - Portanto, $S_1 = A.B$



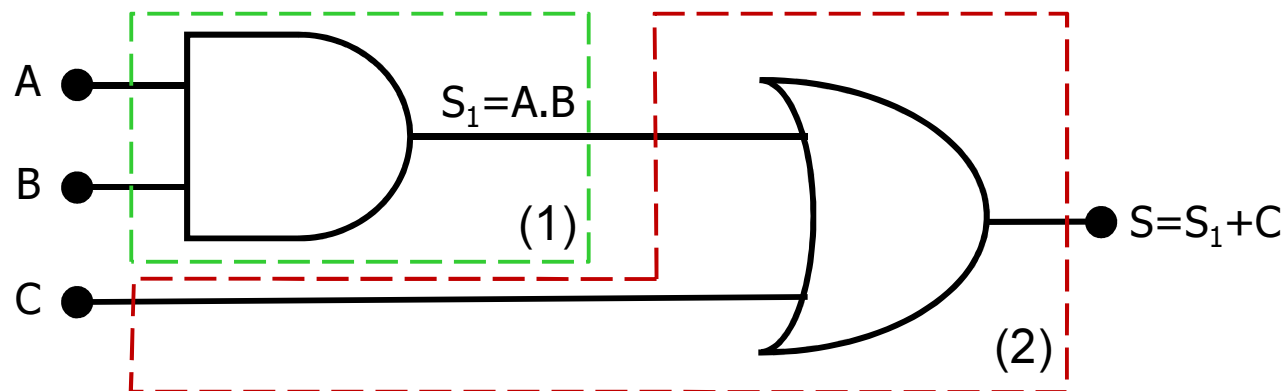
Expressões Booleanas Geradas por Circuitos Lógicos

- ❑ No circuito (2), note que a saída S_1 é utilizada como uma das entradas da porta **OU**
- ❑ A outra entrada da porta **OU** corresponde à variável C , o que nos leva à:
 - $S = S_1 + C$



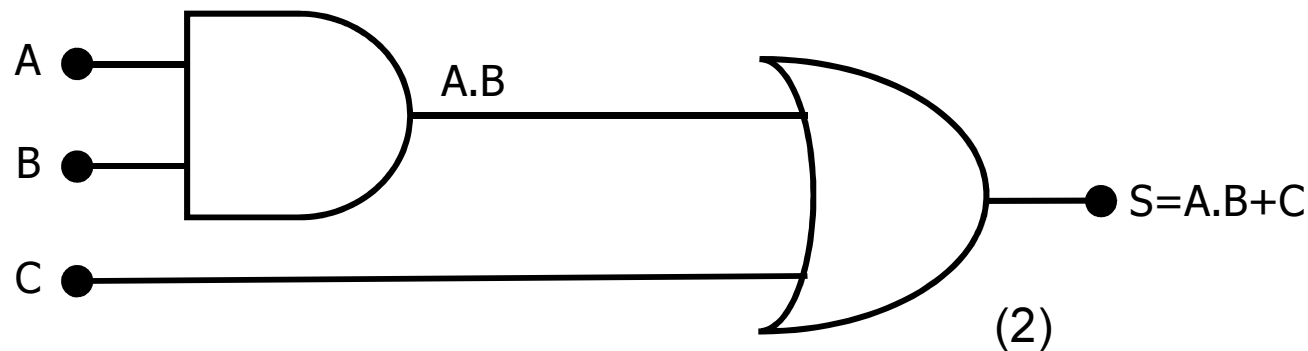
Expressões Booleanas Geradas por Circuitos Lógicos

- Para obter a expressão final em relação às entradas A, B e C basta substituir a expressão S_1 na expressão de S, ou seja:
 - (1) $S_1 = A.B$
 - (2) $S = S_1 + C$
 - Obtém-se $S = S_1 + C = (A.B) + C$



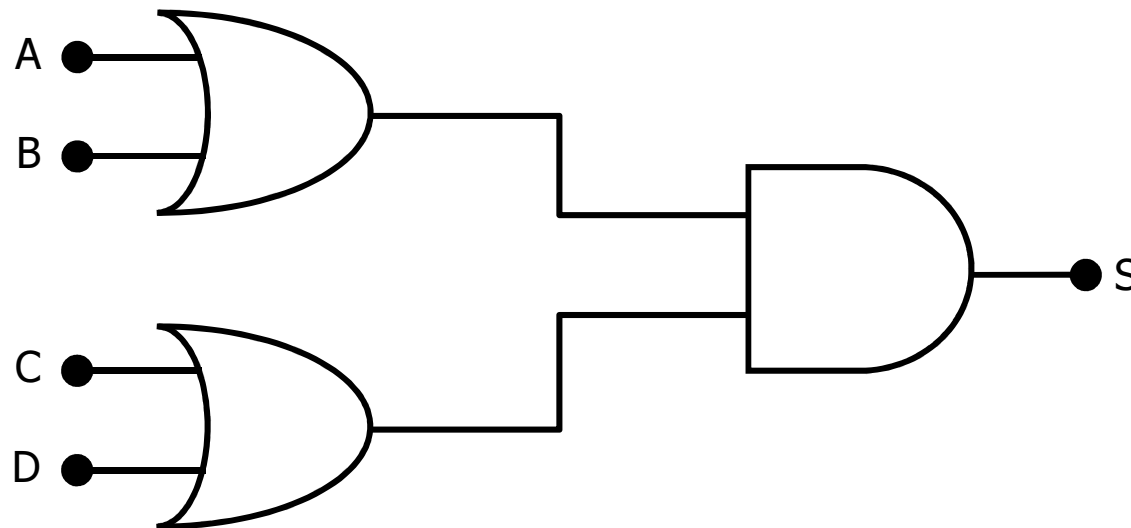
Expressões Booleanas Geradas por Circuitos Lógicos

- Portanto, a expressão que o circuito executa é:
 - $S = (A.B) + C = A.B + C$



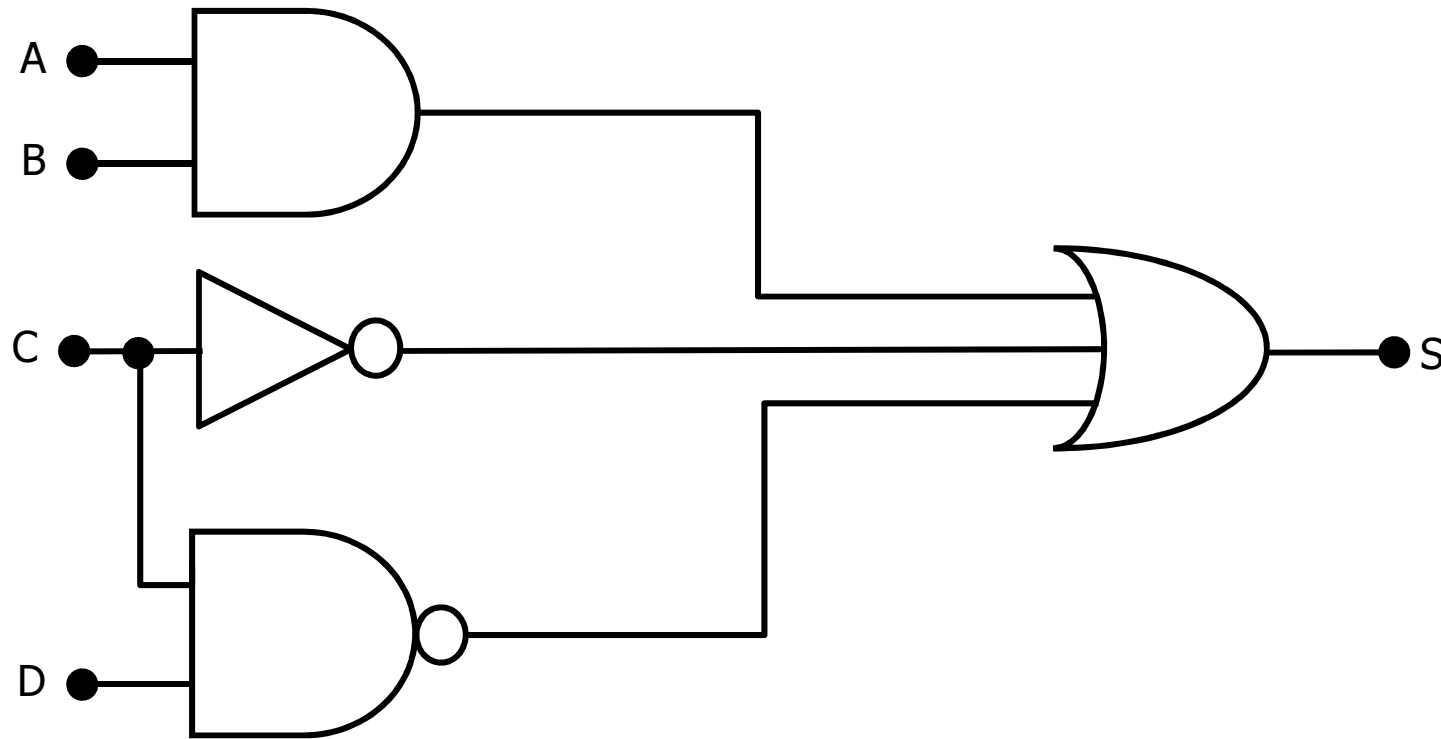
Exercício

- ❑ Escreva a expressão booleana executada pelo circuito



Exercício

- ❑ Determinar a expressão booleana característica do circuito



Circuitos Gerados por Expressões Booleanas

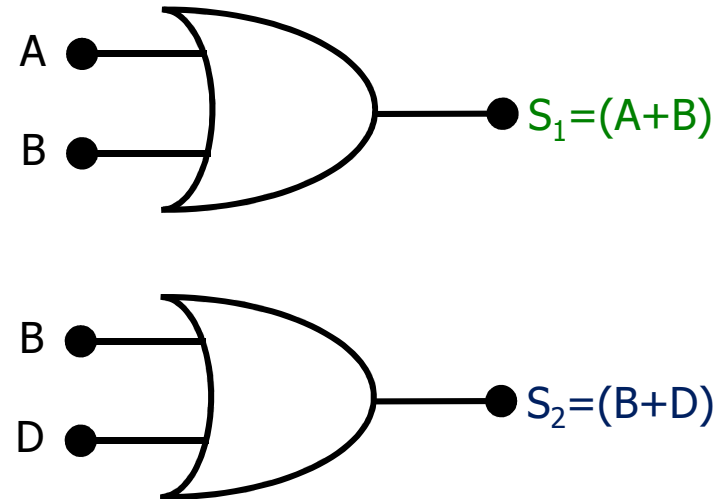
- ❑ Até o momento, vimos como obter uma expressão característica a partir de um circuito
- ❑ Também é possível obter um circuito lógico, dada uma expressão booleana
- ❑ Nesse caso, como na aritmética elementar, parênteses têm maior prioridade, seguidos pela multiplicação (função **E**) e, por último, pela soma (função **OU**)

Circuitos Gerados por Expressões Booleanas

- Seja a expressão
 - $S = (A+B).C.(B+D)$
- Vamos separar as subfórmulas da expressão, ou seja:
 - $S = (A+B) . C . (B+D)$

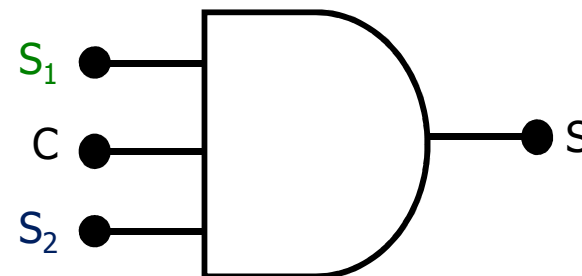
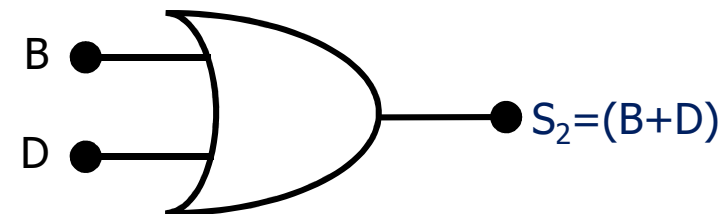
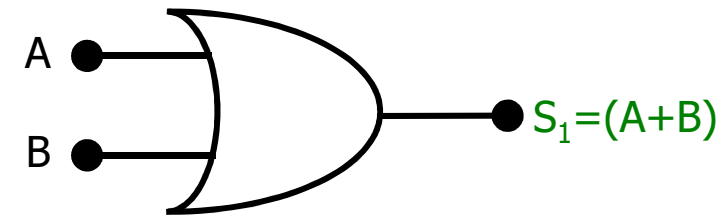
Circuitos Gerados por Expressões Booleanas

- Seja a expressão
 - $S = (A+B).C.(B+D)$
- Vamos separar as subfórmulas da expressão, ou seja:
 - $S = (A+B) . C . (B+D)$
- Dentro do primeiro parêntese temos a soma booleana $S_1=(A+B)$, portanto o circuito que executa esse parêntese será uma porta **OU**
- Dentro do segundo parêntese temos a soma booleana $S_2=(B+D)$. Novamente, o circuito que executa esse parêntese será uma porta **OU**



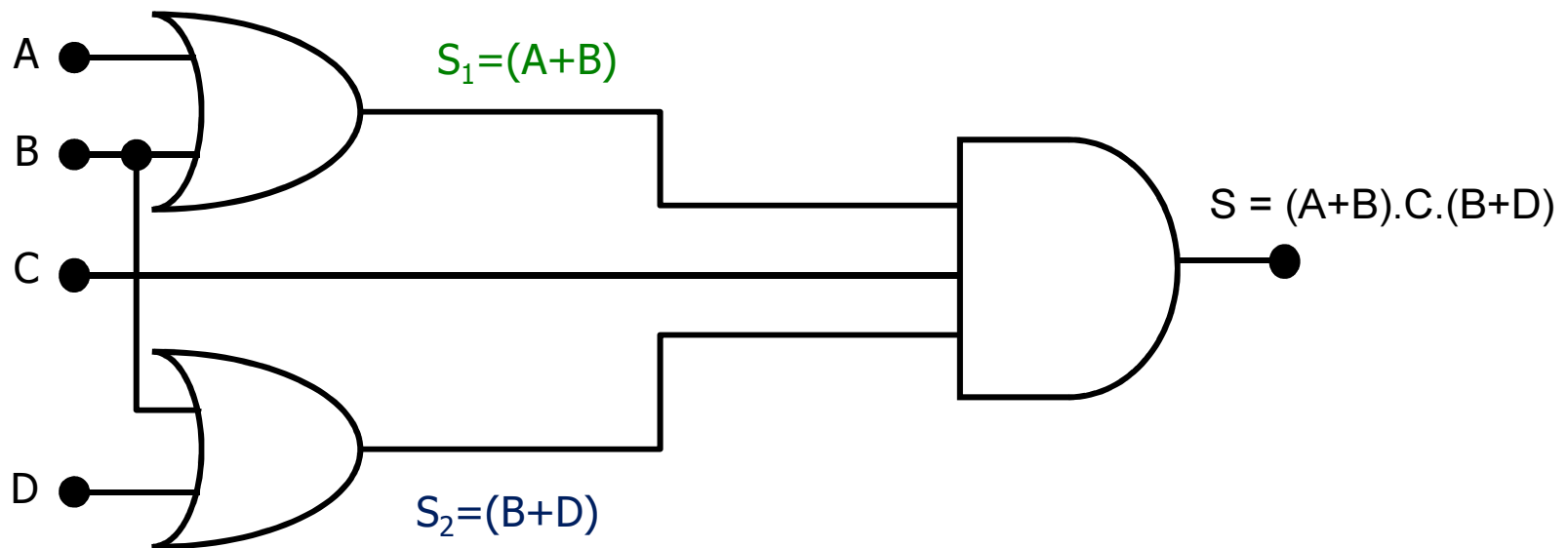
Circuitos Gerados por Expressões Booleanas

- Seja a expressão
 - $S = (A+B).C.(B+D)$
- Vamos separar as subfórmulas da expressão, ou seja:
 - $S = (A+B) . C . (B+D)$
- Dentro do primeiro parêntese temos a soma booleana $S_1=(A+B)$, portanto o circuito que executa esse parêntese será uma porta **OU**
- Dentro do segundo parêntese temos a soma booleana $S_2=(B+D)$. Novamente, o circuito que executa esse parêntese será uma porta **OU**
- Portanto, temos:
 - $S = S_1 . C . S_2$
- Agora temos uma multiplicação booleana e o circuito que a executa é uma porta **E**



Circuitos Gerados por Expressões Booleanas

- O circuito completo é:



Exercício

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

Exercício

- Desenhe o circuito lógico cuja expressão característica é
 - $S = (\overline{A.B} + \overline{C.D})'$

Expressões ou Circuitos representados por Tabelas Verdade

- ❑ Uma forma de estudar uma função booleana consiste em utilizar sua tabela verdade
- ❑ Como visto anteriormente, há uma equivalência entre o circuito lógico e sua expressão característica
 - Podemos obter um circuito a partir de sua expressão
 - Podemos obter expressões a partir dos circuitos
- ❑ Uma tabela verdade representa o comportamento tanto do circuito como de sua expressão característica

Como obter a Tabela Verdade a partir de uma Expressão

- ❑ Colocar todas as possibilidades (interpretações) para as variáveis de entrada
 - Lembrar que para N variáveis, há 2^N possibilidades
- ❑ Adicionar colunas para cada subfórmula da expressão
 - Preencher cada coluna com seus resultados
- ❑ Adicionar uma coluna para o resultado final
 - Preencher essa coluna com o resultado final

Exemplo

- ❑ Considere a expressão
 - $S = A.B.C + A.D + A.B.D$
- ❑ Como há 4 variáveis de entrada (A, B, C, D), há $2^4=16$ interpretações
 - Variação 1 zero, 1 um

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

Exemplo

- ❑ Considere a expressão
 - $S = A.B.C + A.D + A.B.D$
- ❑ Como há 4 variáveis de entrada (A, B, C, D), há $2^4=16$ interpretações
 - Variação 1 zero, 1 um
 - Variação 2 zeros, 2 um

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

Exemplo

- ❑ Considere a expressão
 - $S = A.B.C + A.D + A.B.D$
- ❑ Como há 4 variáveis de entrada (A, B, C, D), há $2^4=16$ interpretações
 - Variação 1 zero, 1 um
 - Variação 2 zeros, 2 um
 - Variação 4 zeros, 4 um

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

Exemplo

- ❑ Considere a expressão
 - $S = A.B.C + A.D + A.B.D$
- ❑ Como há 4 variáveis de entrada (A, B, C, D), há $2^4=16$ interpretações
 - Variação 1 zero, 1 um
 - Variação 2 zeros, 2 um
 - Variação 4 zeros, 4 um
 - Variação 8 zeros, 8 um

A	B	C	D
0	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	0	0	0
1	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

Exemplo

- ❑ $S = A.B.C + A.D + A.B.D$
- ❑ A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S
 - A.B.C
 - A.D
 - A.B.D

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

Exemplo

- ❑ $S = A.B.C + A.D + A.B.D$
- ❑ A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S
 - A.B.C
 - A.D
 - A.B.D
- ❑ Preencher cada coluna com seu respectivo resultado

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

Exemplo

- ❑ $S = A.B.C + A.D + A.B.D$
- ❑ A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S
 - A.B.C
 - A.D
 - A.B.D
- ❑ Preencher cada coluna com seu respectivo resultado

A	B	C	D	A.B.C	A.D	A.B.D	S
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	1			
1	1	1	1	1			

Exemplo

- ❑ $S = A.B.C + A.D + A.B.D$
- ❑ A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S
 - A.B.C
 - A.D
 - A.B.D
- ❑ Preencher cada coluna com seu respectivo resultado

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

Exemplo

- ❑ $S = A.B.C + A.D + A.B.D$
- ❑ A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S
 - A.B.C
 - A.D
 - A.B.D
- ❑ Preencher cada coluna com seu respectivo resultado

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

Exemplo

- ❑ $S = A.B.C + A.D + A.B.D$
- ❑ A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S
 - A.B.C
 - A.D
 - A.B.D
- ❑ Preencher cada coluna com seu respectivo resultado

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

Exemplo

- ❑ $S = A.B.C + A.D + A.B.D$
- ❑ A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S
 - A.B.C
 - A.D
 - A.B.D
- ❑ Preencher cada coluna com seu respectivo resultado

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

Exemplo

- ❑ $S = A.B.C + A.D + A.B.D$
- ❑ A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S
 - A.B.C
 - A.D
 - A.B.D
- ❑ Preencher cada coluna com seu respectivo resultado
- ❑ Por último, preencher a coluna do resultado final

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

Exemplo

- ❑ $S = A.B.C + A.D + A.B.D$
- ❑ A seguir, adicionar uma coluna para cada subfórmula de S, além de uma coluna para o resultado final S
 - A.B.C
 - A.D
 - A.B.D
- ❑ Preencher cada coluna com seu respectivo resultado
- ❑ Por último, preencher a coluna do resultado final

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

Exercício

- ❑ Encontre a tabela verdade da expressão
 - $S = \bar{A} + B + A.B.C'$

Exercício

- Montar a tabela verdade da expressão
 - $S = A.B.C + A.B'.C + A'.B'.C + A'.B'.C'$

Equivalência de Expressões Booleanas por Tabela Verdade

- ❑ Sejam $S1$ e $S2$ duas expressões booleanas
- ❑ $S1$ e $S2$ são **equivalentes** se e somente se para todas as interpretações possíveis (linhas) na tabela verdade ocorre $S1=S2$
- ❑ Se $S1 \neq S2$ em pelo menos uma interpretação, então $S1$ e $S2$ não são equivalentes

Exercício

- Verifique, usando tabela verdade, se as expressões S1 e S2 são equivalentes
 - $S1 = A$
 - $S2 = A.(A+B)$

A	B	A+B	S1	S2
0	0			
0	1			
1	0			
1	1			

Solução

- ❑ Verifique, usando tabela verdade, se as expressões S1 e S2 são equivalentes
 - $S1 = A$
 - $S2 = A.(A+B)$
- ❑ Como $S1=S2$ em todas as interpretações possíveis na tabela verdade, as expressões são equivalentes
 - $A.(A+B) = A$
- ❑ Como veremos mais adiante, esta é uma propriedade, conhecida como **absorção**

A	B	A+B	S1	S2
0	0	0	0	0
0	1	1	0	0
1	0	1	1	1
1	1	1	1	1

Exercício

- Verifique, usando tabela verdade, se as expressões S1, S2, S3 são equivalentes entre si
- $S1 = A$
 - $S2 = A.(1 + B)$
 - $S3 = A + A.B$

A	B	1+B	A.B	S1	S2	S3
0	0					
0	1					
1	0					
1	1					

Exercício

- ❑ Verifique, usando tabela verdade, se as expressões S1 e S2 são equivalentes
 - $S1 = A.(B + C)$
 - $S2 = A.B + A.C$

A	B	C	B+C	A.B	A.C	S1	S2
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

Exercício

- Verifique, usando tabela verdade, se as expressões S1 e S2 são equivalentes
 - $S1 = A + (B \cdot C)$
 - $S2 = (A + B) \cdot (A + C)$

A	B	C	B.C	A+B	A+C	S1	S2
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

Exercício

- ❑ Verifique, usando tabela verdade, se as expressões S1 e S2 são equivalentes
 - $S1 = (\bar{A}.\bar{B})$
 - $S2 = (A.B)'$

A	B	A'	B'	A.B	S1	S2
0	0					
0	1					
1	0					
1	1					

Resumo de Algumas Propriedades provadas por Tabelas Verdade

□ Absorção

- $A + (A.B) = A$
- $A . (A+B) = A$

□ Distributiva

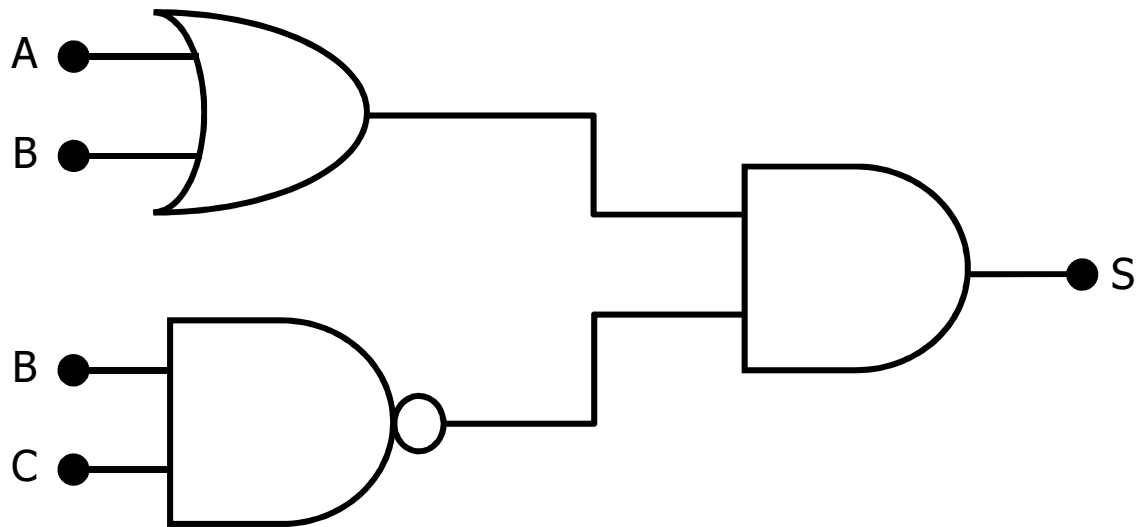
- $A.(B+C) = A.B + A.C$
- $A+(B.C) = (A+B) . (A+C)$

Obtendo a Tabela Verdade a partir de um Circuito

- ❑ De forma análoga, é possível estudar o comportamento de um circuito por meio da sua tabela verdade
- ❑ Dado um circuito, é necessário extrair sua expressão característica; a partir dela é possível montar a tabela verdade correspondente

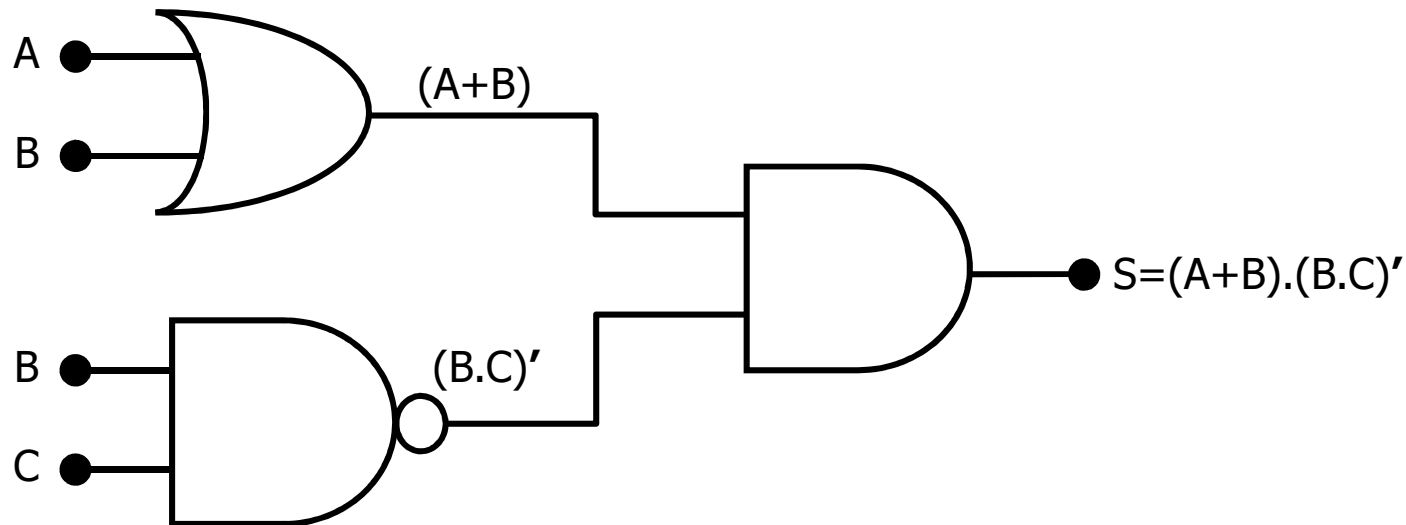
Exemplo

- A partir do circuito:



Exemplo

- A partir do circuito:



- Extraímos sua expressão característica
 - $S = (A+B) . \overline{(B.C)}$

Exemplo

- A partir da expressão

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

- Obtém-se a tabela verdade, como anteriormente explicado

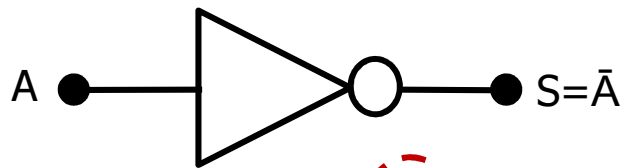
A	B	C	A+B	B.C	(B.C)'	S
0	0	0	0	0	1	0
0	0	1	0	0	1	0
0	1	0	1	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	0	1	1
1	1	0	1	0	1	1
1	1	1	1	1	0	0

Equivalência de Blocos Lógicos

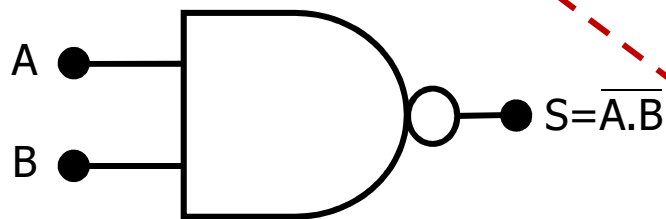
- ❑ Qualquer bloco lógico básico pode ser obtido utilizando outro bloco qualquer e inversores
- ❑ Inversores podem ser obtidos a partir de portas **NAND** e **NOR**
- ❑ Veremos a seguir essas equivalências entre determinados blocos
- ❑ Tais equivalências podem ser provadas pela tabelas verdades correspondentes da seguinte forma
 - Seja $S1$ a expressão característica do primeiro bloco $B1$
 - Seja $S2$ a expressão característica do segundo bloco $B2$
 - Se para todas as interpretações possíveis de $B1$ e $B2$, sempre ocorrer que $S1=S2$, então $B1$ é equivalente a $B2$

Inversor a partir de porta **NAND**

□ Inversor

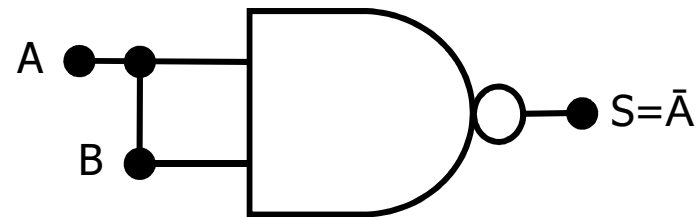


A	S
0	1
1	0



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

□ Ao interligar as entradas de uma porta **NAND**, obtém-se um inversor

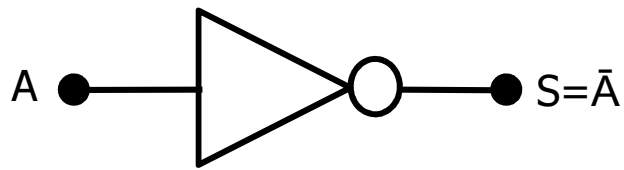


Note que, para cada interpretação possível, os resultados são equivalentes

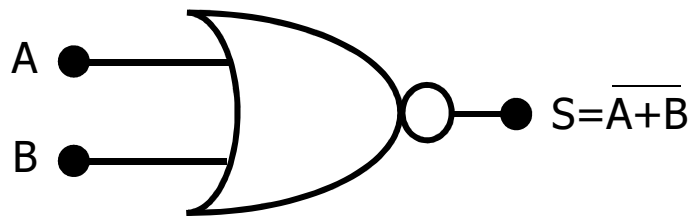
A	B	S
0	0	1
1	1	0

Inversor a partir de porta **NOR**

□ Inversor

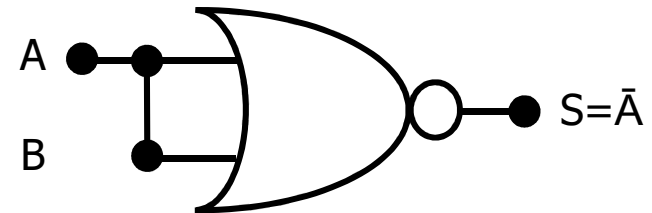


A	S
0	1
1	0



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

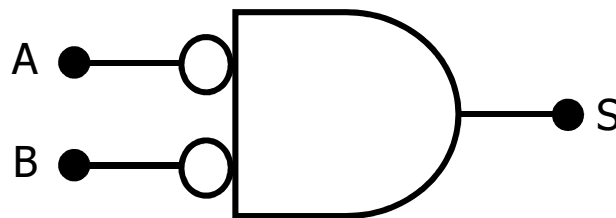
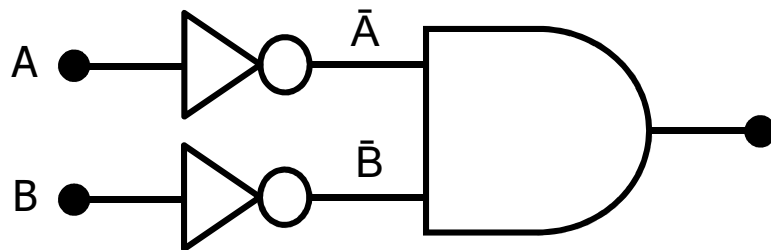
□ Ao interligar as entradas de uma porta **NOR**, obtém-se um inversor



A	B	S
0	0	1
1	1	0

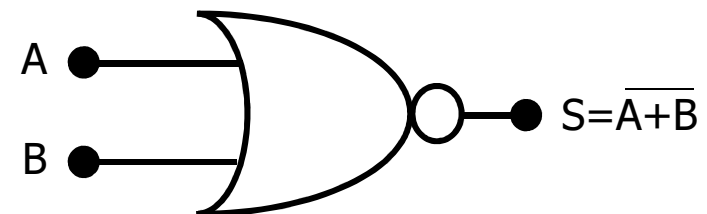
Porta **NOU** a partir de porta **E** e inversores

□ Porta **E** e inversores



A	B	\bar{A}	\bar{B}	S
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

□ Porta **NOU**

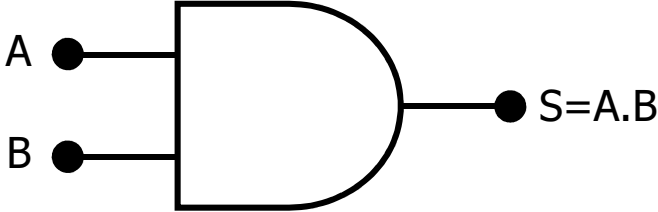
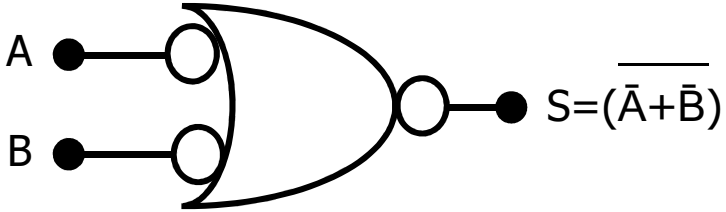
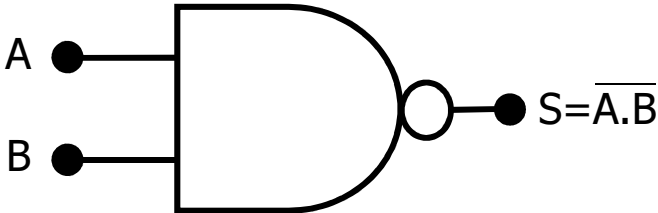
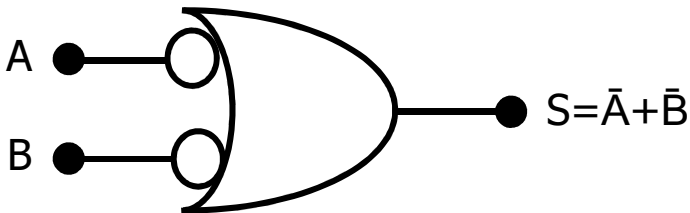
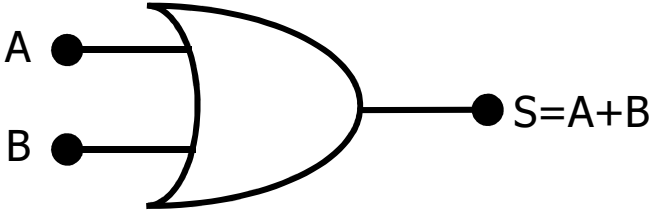
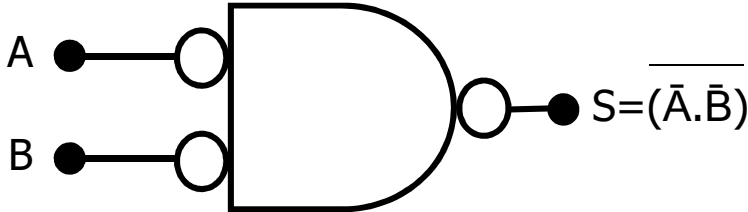
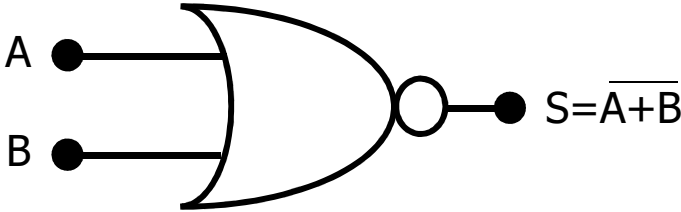
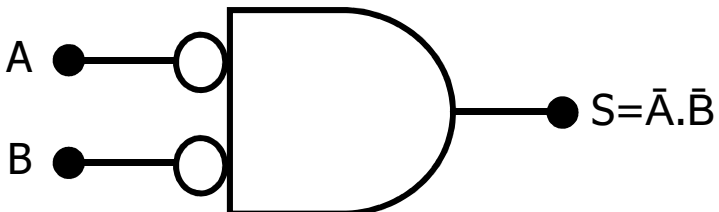


A	B	$S = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

Equivalência de Blocos Lógicos

- ❑ De maneira similar, a equivalência entre os blocos mostrados a seguir pode ser verificada

Blocos Lógicos Equivalentes

Nome	Bloco Lógico	Bloco Equivalente
AND		
NAND		
OR		
NOR		

Exercício

- Prove, usando tabela verdade, que os seguintes blocos lógicos são equivalentes

