

AULA 5

CONCEITOS DE LÓGICA DIGITAL

NESTA AULA

- » Histórico da Álgebra de Boole
- » Funções Booleanas Básicas
- » Correspondência e conversões de funções lógicas
- » Correspondência entre expressões booleanas e circuitos com tabelas verdade
- » Obtenção de tabela verdade a partir de um circuito lógico
- » Equivalência entre blocos e funções lógicas

METAS DE COMPREENSÃO

- » Conhecer o histórico da Álgebra de Boole.
- » Conhecer as funções booleanas básicas, as expressões representativas, os circuitos lógicos e as tabelas verdade correspondentes.
- » Conhecer as correspondências, conversões das funções lógicas.
- » Conhecer as equivalências entre circuitos lógico e expressões booleanas.

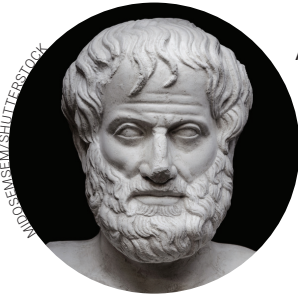
APRESENTAÇÃO

Nesta aula vamos apresentar os conceitos básicos de lógica digital. Inicialmente, o assunto será apresentado sob a perspectiva histórica, desde a origem da introdução da lógica formal pelo filósofo grego da antiguidade Aristóteles, passando pelo matemático alemão Leibniz que integrou a lógica com a álgebra clássica, até chegar ao matemático inglês George Boole que desenvolveu a álgebra que leva seu nome, a álgebra de Boole.

Em seguida, apresentaremos as lógicas básicas de Boole e suas representações através de expressões booleanas, circuitos lógicos e tabelas verdade.

Posteriormente, mostraremos e exercitaremos as conversões entre as diversas representações das funções booleanas. No final, apresentaremos as equivalências entre circuitos lógicos e expressões booleanas.

HISTÓRICO DA ÁLGEBRA DE BOOLE



Aristóteles

A álgebra de Boole é baseada em princípios de lógica formal, que é uma área de estudo da filosofia. A lógica formal, também denominada de lógica simbólica, trata da estrutura do raciocínio humano. Ela foi desenvolvida para fornecer um método para descrever proposições e verificar a sua validade.

Um dos pioneiros do estudo da lógica formal foi Aristóteles, filósofo grego, aluno de Platão e tutor de Alexandre o Grande, da Macedônia, e que viveu no período de 384 a 322 A. C.



Gottfried Leibniz

Ele publicou um tratado sobre o tema, denominado "*Organon*", composto de seis partes, no qual procurava instrumentalizar a lógica para o pensamento filosófico.

Posteriormente, o filósofo e matemático alemão Gottfried Wilhelm Leibniz, que viveu no período de 1646 a 1716, desenvolveu uma lógica matemática, mostrando que a lógica está profundamente ligada e correlacionada à matemática.

Ele compreendeu que a lógica, com os seus termos, proposições e silogismos, guarda uma certa semelhança com a álgebra.



George Boole

Dois séculos depois, em 1847, o matemático inglês George Boole publica um trabalho inovador, "*A Análise Matemática da Lógica*", no qual estabelece uma formulação mais precisa da lógica matemática, como sendo uma espécie de cálculo lógico de classes. É a primeira contribuição para a lógica simbólica. Nesta obra, ele introduz a Lógica ou Álgebra Booleana.

Ele é considerado um dos pais da Ciência da Computação e do desenvolvimento da lógica simbólica.



Claude Shannon

Claude Elwood Shannon, engenheiro eletrônico e matemático norte-americano, que viveu de 1916 a 2001, é considerado o pai da teoria da informação. Ele sugeriu que a álgebra booleana poderia ser utilizada para análise e projeto de circuito de comutação telefônica. Em sua tese de mestrado no MIT (Massachusetts Institute of Technology), em 1937, demonstrou que uma aplicação elétrica utilizando álgebra booleana poderia resolver qualquer problema de lógica.

De acordo com a Wikipédia, *"Neste trabalho, Shannon provou que a álgebra booleana e a aritmética binária poderiam ser utilizadas para simplificar o arranjo dos relés eletromecânicos e, então, utilizados em comutadores para roteamento telefônico. Expandindo o conceito, ele também mostrou que deveria ser possível a utilização de arranjos de relés para resolver problemas de álgebra booleana. A exploração dessa propriedade de interruptores elétricos criou a lógica e os conceitos mais básicos dos computadores digitais. O trabalho de Shannon tornou-se o principal na área de circuitos digitais, quando se tornou amplamente conhecido entre a comunidade de engenharia elétrica durante e após a segunda guerra mundial. O trabalho teórico rigoroso de Shannon substituiu completamente os métodos ad hoc que haviam prevalecido anteriormente."*

Suas observações foram complementadas em 1938, em seu trabalho *"Uma Análise Simbólica de Relés e Circuitos de Comutação"*.



você sabia?

Nos primórdios da Eletrônica, todos os problemas eram solucionados por meio da utilização de circuitos analógicos. Posteriormente, com o avanço da tecnologia, passou-se a utilizar a eletrônica digital.

A álgebra booleana ou de Boole foi, portanto, desenvolvida por Boole e complementada por Shannon.

Uma álgebra Booleana pode ser definida com um conjunto de operadores e um conjunto de axiomas, que são assumidos verdadeiros sem necessidade de prova. As variáveis da álgebra booleana, diferentemente da álgebra tradicional, só podem assumir dois valores possíveis, que são representados por F (Falso) ou V (Verdadeiro, ou 0 (zero) e 1 (um)).

Os computadores atuais utilizam circuitos digitais que empregam um pequeno grupo de circuitos lógicos básicos, que são conhecidos como portas lógicas e que veremos no tópico seguinte.

Com a utilização adequada destas portas, é possível implementar todas as expressões geradas pela álgebra de Boole.

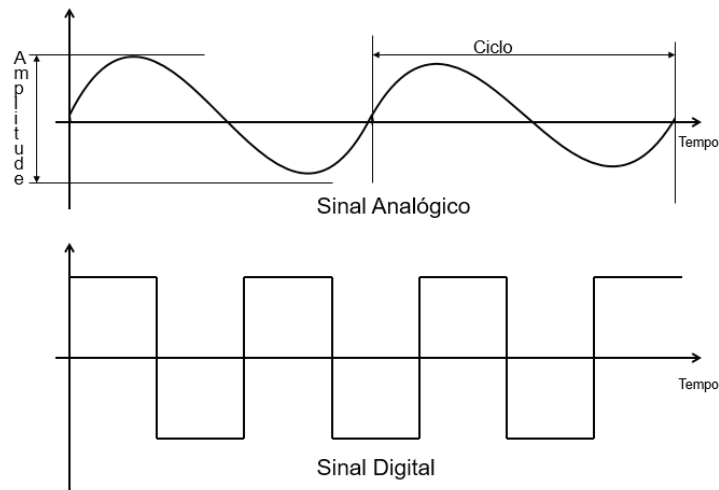


Saiba a diferença entre circuitos analógicos e digitais:

Os circuitos analógicos utilizam sinais analógicos que flutuam em ondas contínuas através do tempo entre as voltagens altas e as voltagens baixas.

Por sua vez, os circuitos digitais utilizam sinais digitais que possuem apenas dois estados discretos de voltagem, a alta ou a baixa.

A figura seguinte ilustra os dois tipos de sinais:



Portanto, observamos que o sinal analógico é contínuo. Isto significa que se varia entre dois valores, ele assume também todos os possíveis valores intermediários. Por sua vez, o sinal digital é discreto e só assume dois valores. É, portanto, um sinal binário.

Na álgebra de Boole, há somente dois estados (valores ou símbolos) permitidos que são o estado 0 (zero) e o 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.

Assim, na álgebra booleana, se representarmos por 0 (zero) uma situação, a situação contrária é representada por 1 (um).

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).



palavra de autor

“Um sistema digital é uma combinação de dispositivos projetados para manipular informação lógica ou quantidades físicas representadas no formato digital, ou seja, as quantidades podem assumir valores discretos. Esses dispositivos são, na maioria das vezes eletrônicos, mas podem ser mecânicos, magnéticos ou pneumáticos. Alguns dos sistemas digitais mais conhecidos são os computadores, as calculadoras, os equipamentos de áudio e vídeo e o sistema de telecomunicações.

Um sistema analógico contém dispositivos que manipulam quantidades físicas representadas na forma analógica. Em um sistema analógico, as quantidades físicas podem variar ao longo de uma faixa contínua de valores. Por exemplo, a amplitude do sinal de saída de um alto-falante em um receptor de rádio pode apresentar qualquer valor entre zero e o seu limite máximo.”

TOCCI, R. J. *Sistemas Digitais: Princípios e Aplicações*. 11. ed. São Paulo: Pearson Prentice Hall, 2011, p. 6.

■ FUNÇÕES BOOLEANAS BÁSICAS

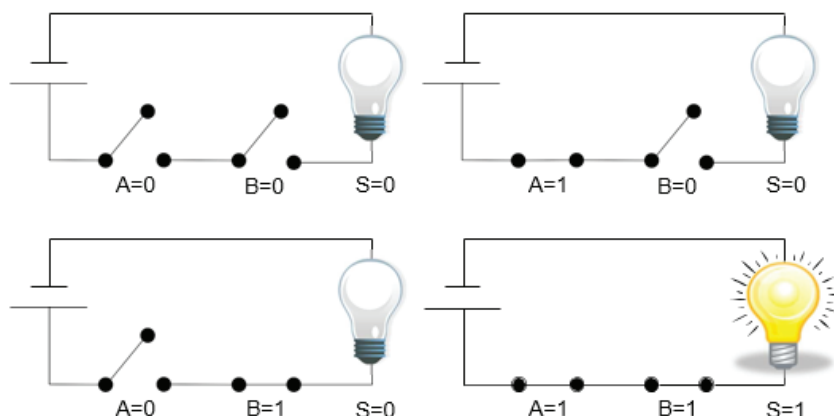
A álgebra booleana trabalha com os seguintes blocos ou funções lógicas:

- ♦ E (AND)
- ♦ OU (OR)
- ♦ NÃO (NOT)
- ♦ NÃO E (NAND)
- ♦ NÃO OU (NOR)
- ♦ OU EXCLUSIVO (XOR)

FUNÇÃO E (AND, EM INGLÊS)

Esta função executa a conjunção ou multiplicação booleana de duas ou mais variáveis binárias e pode ser representada por um circuito, em

que a chave aberta representa 0 (zero) e a chave fechada representa 1 (um), conforme figura seguinte.



Considerando a figura, concluímos que:

- 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 $B=1$)

Para representar a função E (AND), utilizamos a seguinte expressão:

$S = A.B$, onde se lê $S = A$ e B

Portanto, colocamos um ponto entre as variáveis A e B, que significa A e B.

Toda função ou circuito booleano pode ser representado também pela TABELA VERDADE correspondente.

A tabela verdade é um mapa em que 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.

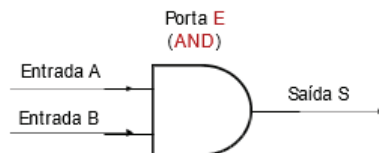
Em geral, para **N** variáveis booleanas de entrada, há **2N** interpretações possíveis

Portanto a tabela verdade correspondente à função **S = A.B** é a seguinte:

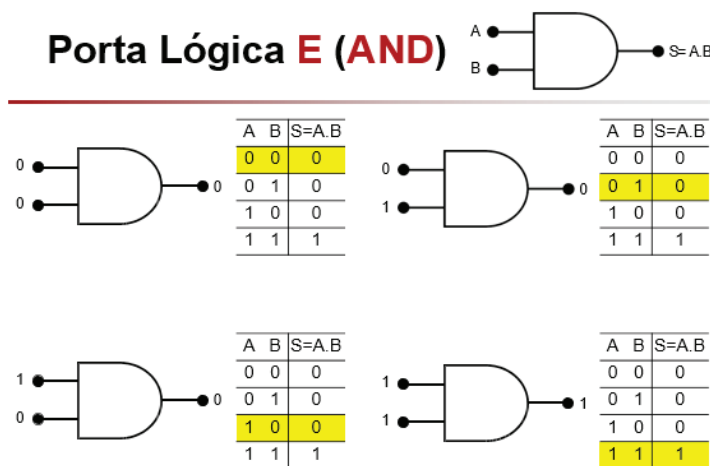
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

Pela tabela verdade que apresenta todas as possíveis combinações de 0 e 1 das duas variáveis A e B, considerando que a saída 0 corresponde a “não passa corrente” e a saída 1 corresponde a “passa corrente”, percebemos que só passa corrente (1) quando as duas variáveis A e B estiverem na posição 1 (chave fechada no circuito).

Vamos agora apresentar a representação da porta lógica (circuito lógico) que demonstra a função E (AND).



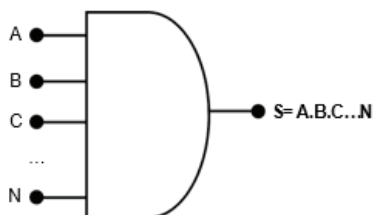
As possíveis situações serão, portanto, representadas pela figura seguinte.



É possível estender o conceito de uma porta E para um número qualquer de variáveis de entrada.

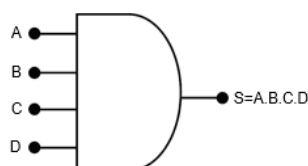
Nesse caso, temos uma porta E com **N** entradas e somente uma saída.

A saída será 1 se e somente se as **N** entradas forem iguais a 1(um); nos demais casos, a saída será 0 (zero).



A figura 8 seguinte mostra a situação de uma função E (AND), com quatro variáveis (A, B, C, e D).

○ 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

Percebam que a tabela verdade correspondente possui 16 combinações de 0 e 1 das quatro variáveis de entrada A, B, C e D. Utilizando a regra apresentada para saber a quantidade de combinações, temos o seguinte: $2^4 = 16$.



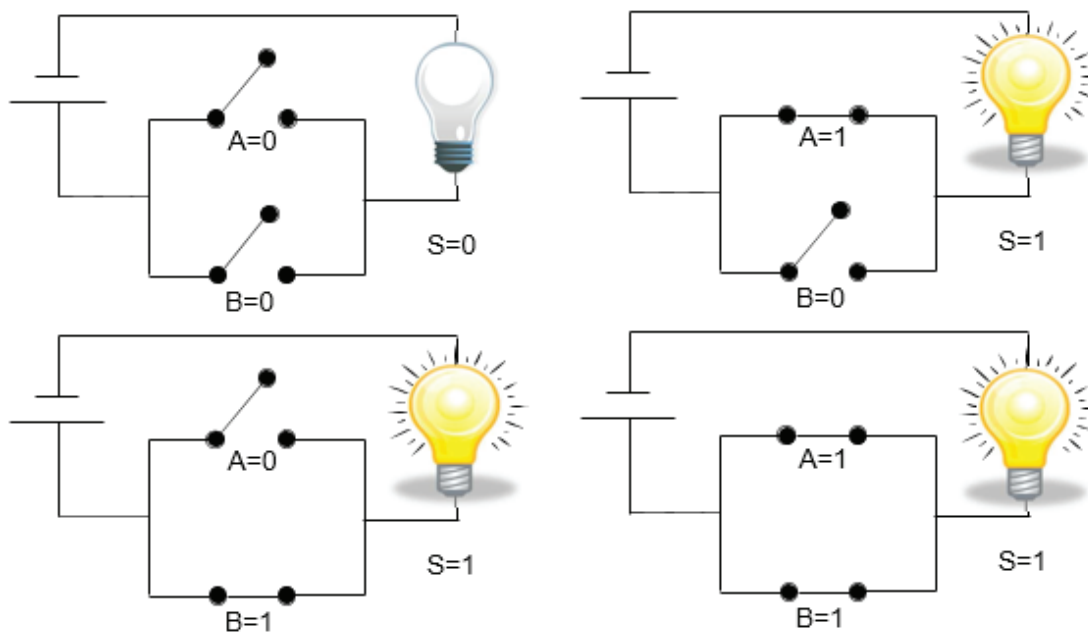
dica

Para construir a tabela verdade do exemplo de porta lógica com 16 combinações, percebam que iniciamos o A com oito 0 (zero) seguidos de oito 1 (um), o B com alternância de 0 e 1 de quatro em quatro, o C com alternância de 0 e 1 de dois em dois, e finalmente o D com alternâncias de 0 e 1 de um em um.

Esta regra se aplica para qualquer quantidade de variáveis. Por exemplo, se entrarmos com 5 variáveis (A, B, C, D e E), teremos 32 combinações ($2^5 = 32$) e começaremos com E com 16 zeros seguidos de 16 um, decrescendo para de 8 em 8, 4 em 4, 2 em 2, e terminando de 1 em 1.

FUNÇÃO OU (OR, EM INGLÊS)

Esta função executa a disjunção ou soma booleana de duas ou mais variáveis binárias e pode ser representada por um circuito, em que a chave aberta representa 0 (zero) e a chave fechada representa 1 (um), conforme figura seguinte.



Percebam na figura:

- 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.

Para representar a expressão $S = A \text{ ou } B$, adotaremos a seguinte representação:

$S = A+B$, onde se lê $S = A \text{ ou } B$

A tabela verdade correspondente é a seguinte:

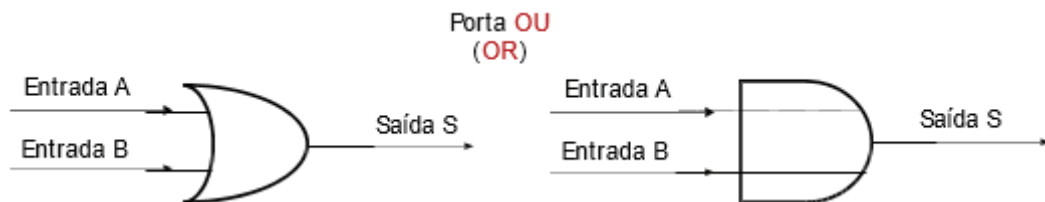
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1



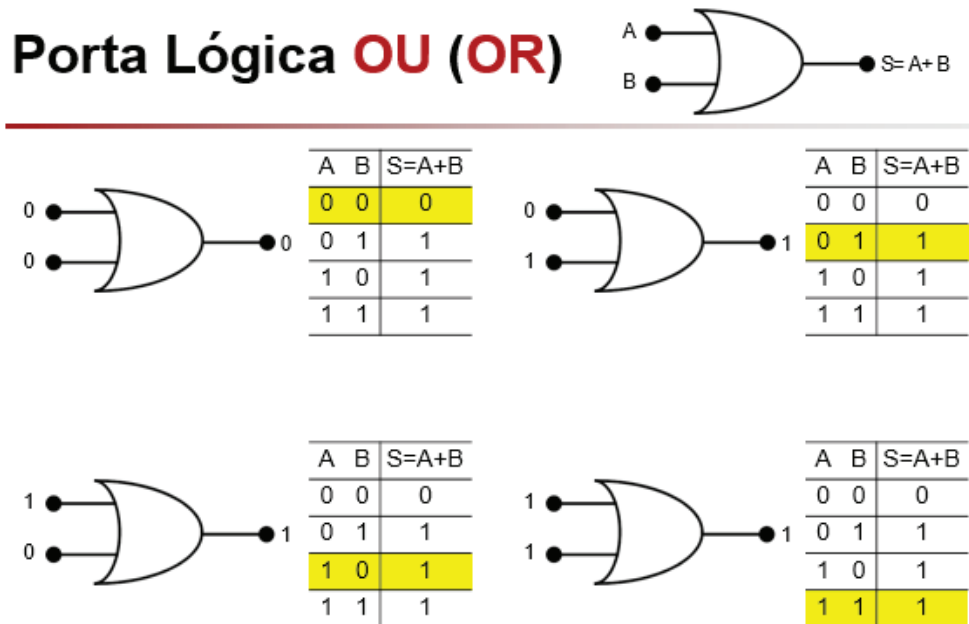
pense nisso

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).

Vamos agora apresentar a representação da porta lógica (circuito lógico), que demonstra a função OU (OR).



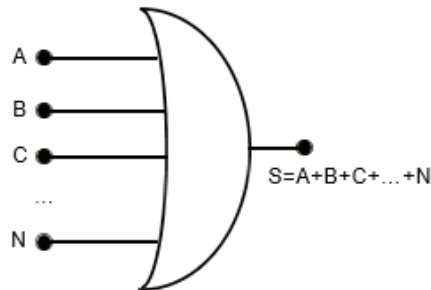
As possíveis situações serão, portanto, representadas pela figura seguinte.



É possível estender o conceito de uma porta **OU** para um número qualquer de variáveis de entrada.

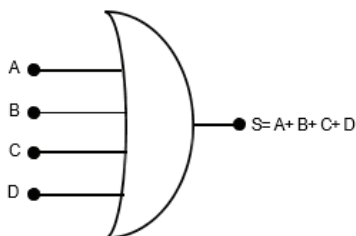
Nesse caso, temos uma porta **OU** com **N** entradas e somente uma saída.

A saída será **0 (zero)** se e somente se as **N** entradas forem iguais a **0 (zero)**; nos demais casos, a saída será **1 (um)**.



A figura seguinte mostra a situação de uma função E (AND), com quatro variáveis (A, B, C, e D).

○ 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

Percebam que a tabela verdade correspondente possui 16 combinações de 0 e 1 das quatro variáveis de entrada A, B, C e D. Utilizando a regra apresentada para saber a quantidade de combinações, temos o seguinte: $2^4 = 16$.

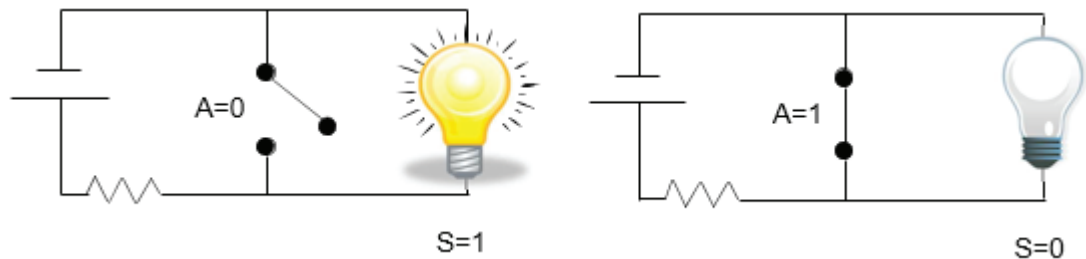
FUNÇÃO NÃO (NOT)

Esta função executa o complemento (negação) de uma variável binária. Se a variável estiver em **0 (zero)**, o resultado da função é **1 (um)** e se a variável estiver em **1 (um)**, o resultado da função é **0 (zero)**. Essa função também é chamada de inversora.

Utilizando 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 ($S=0$)

A visualização desta função está na figura seguinte.



Para representar esta função $S = \text{NÃO } A$, utilizaremos a expressão:

$$S = A \text{ ou } S = A'$$

Há, portanto, para a função NÃO duas notações alternativas que são um traço acima da entrada ou um apóstrofo (') após a entrada.

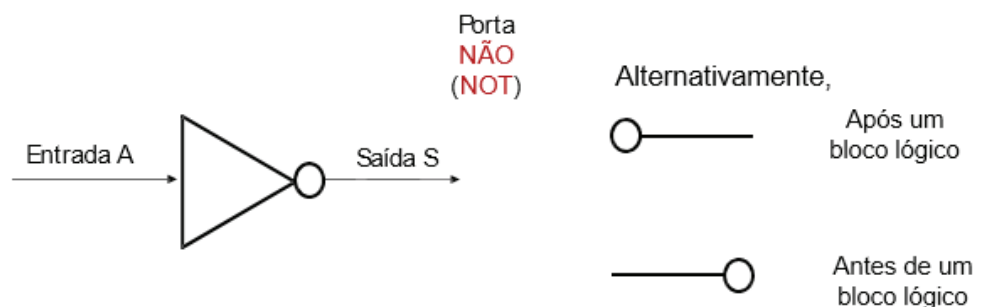
A tabela verdade que corresponde á função **NÃO (NOT)** é a seguinte:

A	A
0	0
0	1

- ▶ A porta lógica **NÃO**, ou inversor, é o circuito que executa a função **NÃO**.
- ▶ O inversor executa a tabela verdade da função **NÃO**.

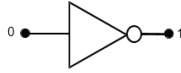
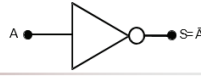
Se a entrada for **0 (zero)**, a saída será **1 (um)**; se a entrada for **1 (um)**, a saída será **0 (zero)**.

A sua representação é a seguinte:

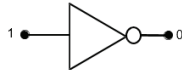


As possíveis situações serão, portanto, representadas pela figura seguinte.

Porta Lógica **NÃO** (NOT)



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



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

FUNÇÃO NÃO E (NAND)

Esta função é a composição da função E (AND) com a função NÃO (NOT), ou seja, a função E é invertida.

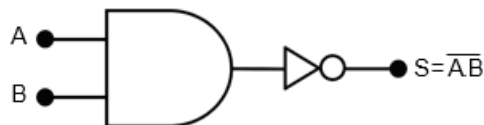
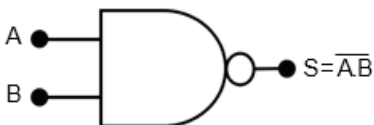
A expressão representativa é a seguinte:

$$S = (A.B) = A.B = (A.B)'$$

A tabela verdade correspondente é a seguinte:

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

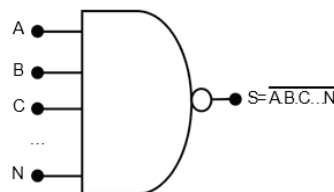
A sua representação é a seguinte:



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 (zero) se e somente se as N entradas forem iguais a 1 (um); nos demais casos, a saída será 1 (um), conforme vemos na representação a seguir:



FUNÇÃO NÃO OU (NOR)

Esta função é a composição da função **OU (OR)** com a função **NÃO (NOT)**, ou seja, a função **OU** é invertida.

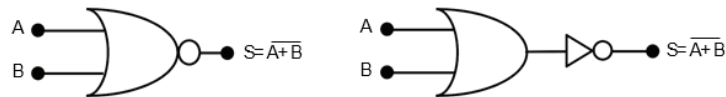
A expressão representativa é a seguinte:

$$S = (A+B) = A+B = (A+B)'$$

A tabela verdade correspondente é a seguinte:

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

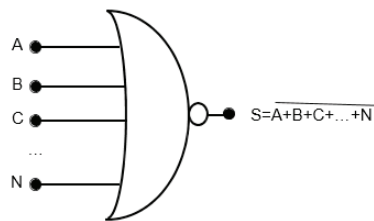
A sua representação é a seguinte:



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 (um)** se e somente se as **N** entradas forem iguais a **0 (zero)**; nos demais casos, a saída será **0 (zero)**, conforme vemos na representação a seguir:



FUNÇÃO OU EXCLUSIVO (XOR)

A função **OU Exclusivo (XOR)** fornece

- ▶ **1 (um)** na saída quando as entradas forem diferentes entre si e
- ▶ **0 (zero)**, caso contrário.

A expressão representativa desta função é a seguinte:

$$S = A \oplus B$$

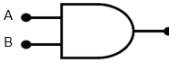


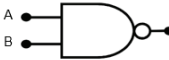


A tabela verdade correspondente é a seguinte:

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

A sua representação é a seguinte:



Resumindo, a figura a seguir apresenta um resumo de todos os blocos lógicos apresentados e suas representações.

Nome	Circuito Lógico	Expressão booleana	Tabela Verdade															
E (AND)		$S=A.B$	<table><tr><th>A</th><th>B</th><th>S=A.B</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=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><tr><th>A</th><th>B</th><th>S=A+B</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=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		$S=\bar{A}$ $S=A'$	<table><tr><th>A</th><th>$S=\bar{A}$</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	$S=\bar{A}$	0	1	1	0									
A	$S=\bar{A}$																	
0	1																	
1	0																	
NE (NAND)		$S=\overline{A.B}$ $S=(A.B)'$	<table><tr><th>A</th><th>B</th><th>$S=\overline{A.B}$</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	$S=\overline{A.B}$	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)		$S=\overline{A+B}$ $S=(A+B)'$	<table><tr><th>A</th><th>B</th><th>$S=\overline{A+B}$</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	$S=\overline{A+B}$	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><tr><th>A</th><th>B</th><th>$S=A\oplus B$</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	$S=A\oplus B$	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																

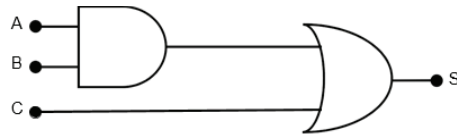
■ CORRESPONDÊNCIA E CONVERSÕES DE FUNÇÕES LÓGICAS

Todo circuito lógico executa uma função booleana, e esta função possui uma tabela verdade correspondente.

A seguir, veremos que dada uma determinada representação de função lógica, podemos obter as suas outras representações, mostrando a correspondência entre as três representações (expressão booleana, circuito lógico e tabela verdade).

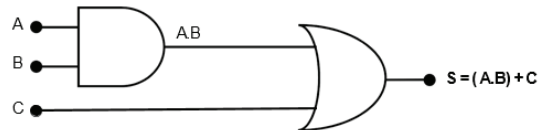
Correspondências entre circuitos lógicos e expressões booleanas

Dado o seguinte circuito lógico:



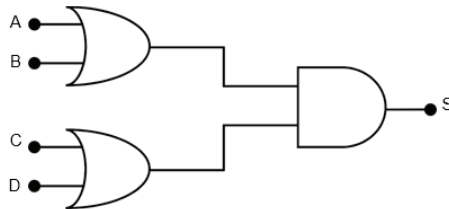
Podemos obter a expressão booleana correspondente, que é:

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



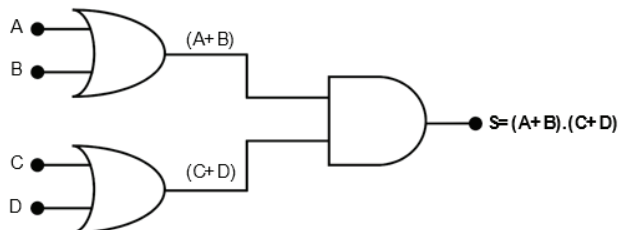
Outro exemplo:

Obter a expressão booleana de circuito lógico seguinte:



A solução é a seguinte:

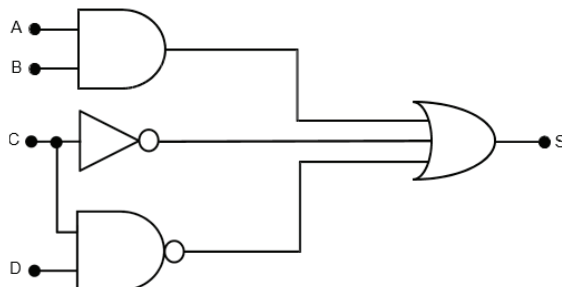
$$S = (A+B).(C+D)$$



Vamos agora fazer um exercício.

Atividade 1

Determinar a expressão booleana do circuito lógico seguinte:



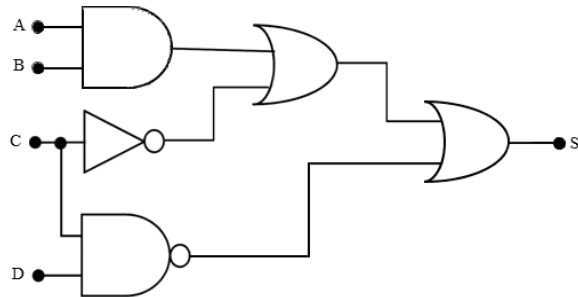
[illegible]

The logic diagram implements the sum $S = (A.B) + C + (C.D)$. It consists of three AND gates and one OR gate. The first AND gate takes inputs A and B to produce the term $(A.B)$. The second AND gate takes inputs C and D to produce the term $(C.D)$. The output of the second AND gate is inverted to produce $\overline{(C.D)}$. A third AND gate takes inputs C and \overline{C} to produce the term C . The OR gate then combines the outputs of the three AND gates: $(A.B)$, C , and $\overline{(C.D)}$ to produce the final sum $S = (A.B) + C + \overline{(C.D)}$.

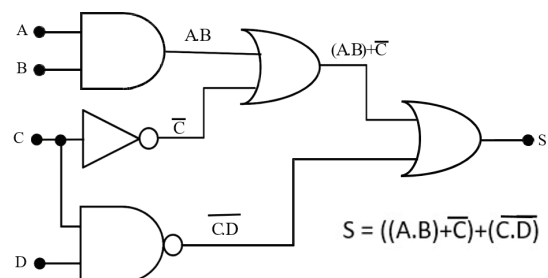
Agora tente fazer a atividade seguinte.

Atividade 2

Determinar a expressão booleana do circuito lógico seguinte:

[illegible]

A resposta é a seguinte:



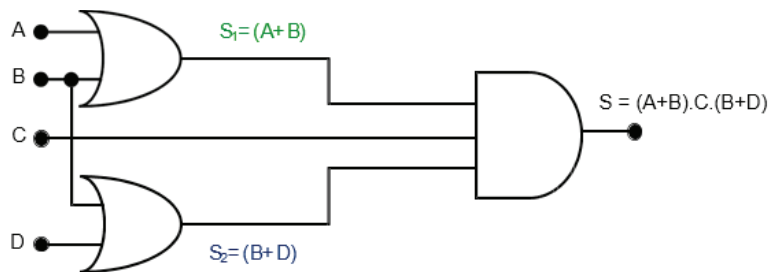
Até o momento, vimos como obter uma expressão booleana a partir de um circuito lógico. Também é possível obter um circuito lógico, dada uma expressão booleana.

Nesse caso, como na aritmética elementar, parênteses têm maior prioridade, seguidos pela multiplicação (função E) e, por último, pela soma (função OU).

Seja a expressão

$$S = (A+B).C.(B+D)$$

O circuito correspondente é o seguinte:



Note que, para chegar à solução, adotamos o processo gradual. Primeiro obtivemos o $S_1=(A+B)$, depois o $S_2=(B+D)$ e, finalmente, juntamos os dois para obter a saída final S , que é $S=(A+B).C.(B+D)$

Vamos fazer outro exercício:

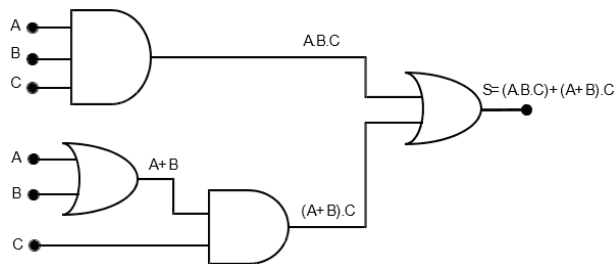
Atividade 3

Dada a expressão booleana

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

Obter o circuito lógico correspondente.

A solução é a seguinte:



É importante lembrar que as entradas que representam a mesma variável estão interligadas. Contudo, o desenho sem interligações facilita a interpretação do circuito.

Vamos agora fazer um outro exercício.

Atividade 4

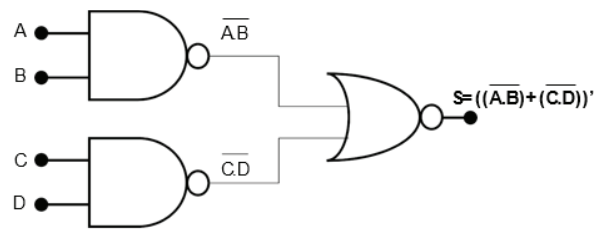
Desenhe o circuito lógico da expressão booleana seguinte:

$$S = (A.B + C.D)'$$

Note que a função NOT pode ser representada tanto pelo apóstrofo como pelo traço.

[illegible]

A solução da atividade 3 é a seguinte:



Note que o pequeno círculo acoplado à saída de uma função representa o **NOT** ou inversão da função.

■ CORRESPONDÊNCIA ENTRE EXPRESSÕES BOOLEANAS E CIRCUITOS COM 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 booleana correspondente:

- ▶ 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 booleana correspondente.

Como obter a tabela verdade a partir de uma expressão booleana?

Precisamos seguir os seguintes passos:

- 01.** Colocar todas as possibilidades (interpretações) para as variáveis de entrada
- 02.** (Lembrar que para N variáveis, há 2^N possibilidades);
- 03.** Adicionar colunas para cada subfórmula da expressão
- 04.** (Preencher cada coluna com seus resultados);
- 05.** Adicionar uma coluna para o resultado final (Preencher essa coluna com o resultado final)

06. 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 (combinações de zero e um).

- Variação 1 zero, 1 um
- Variação 2 zeros, 2 um
- Variação 4 zeros, 4 um
- Variação 8 zeros, 8 um

Conforme a demonstração:

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

A seguir, adicionamos uma coluna para cada subfórmula (componente da expressão) de S, além de uma coluna para o resultado final S. Em seguida, preenchemos cada coluna com seu respectivo resultado, para, por último, preenchermos a coluna do resultado seguinte. O resultado é apresentado na tabela:

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

Vamos fazer um exercício para treinar.

Atividade 5

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

Como temos três variáveis de entrada (A, B e C), teremos $2^3 = 8$ linhas (combinações de 0 e 1) na tabela verdade.

[illegible]

O resultado será o seguinte:

A	B	C	A	C'	A.B.C'	S
0	0	0	1	1	0	1
0	0	1	1	0	0	1
0	1	0	1	1	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	1	0	0	0	0
1	1	0	0	1	1	1
1	1	1	0	0	0	1

Vamos agora fazer outro exercício.

Atividade 6

Montar a tabela verdade da expressão booleana

$$S = A.B.C + A.B'.C + A'.B'.C + A'.B'.C'$$

[illegible]

O resultado é o seguinte:

A	B	C	A'	B'	C'	A.B.C	A.B'.C	A'.B'.C	A'.B'.C'	S
0	0	0	1	1	1	0	0	0	1	1
0	0	0	1	1	0	0	0	1	0	1
0	1	1	1	0	1	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0	0
1	0	0	0	1	0	0	1	0	0	1
1	1	1	0	0	1	0	0	0	0	0
1	1	1	0	0	0	1	0	0	0	1

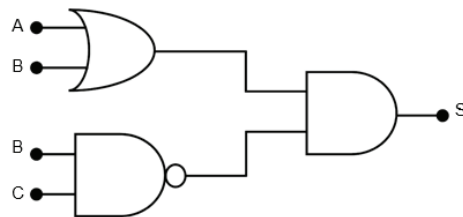
■ OBTENÇÃO DE TABELA VERDADE A PARTIR DE UM CIRCUITO LÓGICO

De forma análoga, é possível estudar o comportamento de um circuito lógico por meio da sua tabela verdade.

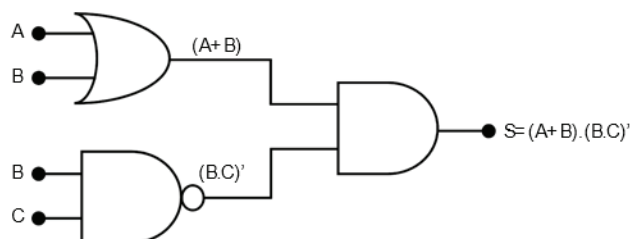
Dado um circuito lógico, é necessário extrair sua expressão booleana; a partir dela é possível montar a tabela verdade correspondente.

Por exemplo:

A partir do circuito lógico



Obtemos sua expressão booleana, que é $S = (A+B).(B.C)'$ conforme vemos a seguir:



A partir da expressão booleana do circuito lógico, obtemos a tabela verdade correspondente, como vimos anteriormente. O resultado é o seguinte:

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

OBTENÇÃO DE EXPRESSÕES BOOLEANAS A PARTIR DE TABELAS VERDADE

Para obter expressões booleanas a partir de tabelas verdade, utilizamos o conceito de **forma normal** ou **forma canônica**.

Este conceito é o seguinte:

Toda expressão booleana pode ser escrita em uma forma padronizada, denominada forma normal ou forma canônica.

Temos duas formas normais, que são:

Forma Normal Conjuntiva (FNC) → Produto de Somas ou Produto de Maxtermos

Forma Normal Disjuntiva (FND) → Soma de Produtos ou Soma de Mintermos

Nos MAXTERMOS ou MAXITERMOS:

- ▶ a variável com valor **0 (zero)** é deixada intacta;
- ▶ a variável com valor **1 (um)** é alterada pela sua negação;
- ▶ variáveis de uma mesma linha na tabela verdade são conectadas por **+** (adição).

Nos MINTERMOS ou MINITERMOS é o contrário:

- ▶ A variável com valor **1(um)** é deixada intacta;
- ▶ A variável com valor **0 (zero)** é alterada pela sua negação;
- ▶ Variáveis de uma mesma linha na tabela verdade são conectadas por **.** (multiplicação).

A tabela seguinte mostra como se aplica este conceito:

A	B	C	Maxtermo	Mintermo
0	0	0	A+B+C	A'.B'.C'
0	0	1	A+B+C'	A'.B'.C

0	1	0	$A+B'+C$	$A'.B.C'$
0	1	1	$A+B'+C'$	$A'.B.C$
1	0	0	$A'+B+C$	$A.B'.C'$
1	0	1	$A'+B+C'$	$A.B'.C$
1	1	0	$A'+B'+C$	$A.B.C'$
1	1	1	$A'+B'+C'$	$A.B.C$

FORMA NORMAL DISJUNTIVA (FND)

Na Forma Normal Disjuntiva (FND):

- ▶ minitermo (ou minitermo) é o termo produto associado a cada linha da tabela verdade, no qual todas as variáveis de entrada estão presentes;
- ▶ dado um dado minitermo, se substituirmos os valores das variáveis associadas, obteremos **1 (um)**;
- ▶ porém, se substituirmos nesse mesmo minitermo quaisquer outras combinações de valores, obteremos **0 (zero)**;
- ▶ dessa forma, se quisermos encontrar a equação para uma função a partir de sua tabela verdade, basta montarmos um **OU (OR)** entre os minitermos associados aos **1s (uns)** da função

Por exemplo:

Seja tabela seguinte:

Situação	A	B	C	S	Mintermo
0	0	0	0	0	
1	0	0	1	0	
2	0	1	0	1	$A'.B.C'$
3	0	1	1	1	$A'.B.C$
4	1	0	0	0	
5	1	0	1	1	$A.B'.C$
6	1	1	0	1	$A.B.C'$
7	1	1	1	0	

S é uma função das variáveis de entrada A, B e C

- ▶ Os valores de (A, B, C) para os quais $S=1$ encontram-se nas situações **2, 3, 5 e 6**
- ▶ Os minitermos associados a essas condições (ou seja, os minitermos 1) são mostrados na tabela anterior.

Logo, a expressão é a soma de produtos (FND) para S será o **OU** entre estes produtos.

$$S = (A'.B.C') + (A'.B.C) + (A.B'.C) + (A.B.C')$$

FORMA NORMAL CONJUNTIVA (FNC)

Outra alternativa é utilizar a Forma Normal Conjuntiva (FNC) na qual:

- ▶ maxtermo (ou maxitermo) é o termo soma associado a cada linha da tabela verdade, no qual todas as variáveis de entrada estão presentes.
- ▶ dado um dado maxtermo, se substituirmos os valores das variáveis associadas, obteremos **0 (zero)**;
- ▶ porém, se substituirmos nesse mesmo maxtermo quaisquer outras combinações de valores, obteremos **1 (um)**;
- ▶ dessa forma, se quisermos encontrar a equação para uma função a partir de sua tabela verdade, basta montarmos um **E (AND)** entre os maxtermos associados aos **0s (zeros)** da função.

Por exemplo:

Na tabela seguinte:

Situação	A	B	C	S	Maxtermo
0	0	0	0	0	$A+B+C$
1	0	0	1	0	$A+B+C'$
2	0	1	0	1	
3	0	1	1	1	
4	1	0	0	0	$A'+B+C$
5	1	0	1	1	
6	1	1	0	1	
7	1	1	1	0	$A'+B'+C'$

S é uma função das variáveis de entrada A, B e C

- ▶ Os valores de (A, B, C) para os quais $S=0$ encontram-se nas situações 0, 1, 4 e 7.
- ▶ Os maxtermos associados a essas condições (ou seja, os maxtermos 0) são mostrados na tabela acima.

Logo, a expressão é o produto de somas (FNC) para S será o E (AND) entre estas somas -

$$S = (A+B+C).(A+B+C').(A'+B+C).(A'+B'+C')$$

ESTUDO DE CASO

Precisamos encontrar a expressão booleana de uma situação, a partir de uma dada tabela verdade.

Dada a tabela verdade seguinte, achar a expressão booleana correspondente, utilizando os mintermos.

A solução é a seguinte:

A	B	C	S	Mintermos
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$A'.B.C$
1	0	0	0	
1	0	1	1	$A.B'.C$
1	1	0	1	$A.B.C'$
1	1	1	1	$A.B.C$

Como vimos, selecionamos na tabela verdade as linhas onde o S é 1 (um) e montamos os mintermos correspondentes, seguindo o conceito. Como resultado, a expressão booleana final corresponde a uma soma (OU) de todos os mintermos obtidos.

Portanto a expressão é a seguinte:

$$S = (A'.B.C) + (A.B'.C) + (A.B.C') + (A.B.C)$$

Com o mesmo exemplo, se quisermos achar a expressão booleana, utilizando os maxtermos, a solução seria a seguinte:

A	B	C	S	Maxtermo
0	0	0	0	$A+B+C$
0	0	1	0	$A+B+C'$
0	1	0	0	$A+B'+C$
0	1	1	1	
1	0	0	0	$A'+B+C$
1	0	1	1	
1	1	0	1	
1	1	1	1	

Como vimos, selecionamos na tabela verdade as linhas onde o S é 0 (zero) e montamos os maxtermos correspondentes, seguindo o conceito. Como resultado, a expressão booleana final corresponde a uma multiplicação (E) de todos os maxtermos obtidos.

Portanto a expressão é a seguinte:

$$S = (A+B+C).(A+B+C').(A+B'+C).(A'+B+C)$$

Observação: As duas expressões obtidas através de mintermos e maxtermos são equivalentes, pois correspondem à mesma tabela verdade.

Vamos agora fazer outra atividade para verificar o aprendizado.

Atividade 7

Dada a tabela verdade seguinte, determinar as suas expressões booleanas, utilizando os mintermos (FND) e os maxtermos (FNC).

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

[illegible]

A solução é a seguinte:

A	B	C	S	Mintermos	Maxtermos
0	0	0	1	$A'.B'.C'$	
0	0	1	0		$A+B+C'$
0	1	0	0		$A+B'+C$
0	1	1	1	$A'.B.C$	
1	0	0	1	$A.B'.C'$	
1	0	1	0		$A'+B+C'$
1	1	0	0		$A'+B'+C$
1	1	1	1	$A.B.C$	

Portanto, utilizando o conceito dos mintermos, a solução é:

$$S = (A'.B'.C') + (A'.B.C) + (A.B'.C') + (A.B.C)$$

Utilizando a solução dos maxtermos, a solução é:

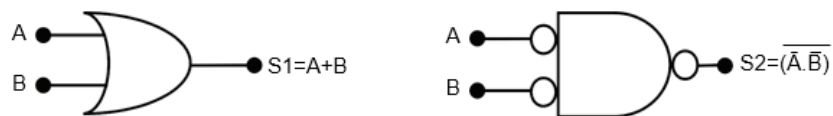
$$S = (A+B+C').(A+B'+C).(A'+B+C').(A'+B'+C)$$

■ EQUIVALÊNCIA ENTRE BLOCOS E FUNÇÕES LÓGICAS

Blocos (circuitos lógicos) e funções booleanas são equivalentes quando suas tabelas verdade são iguais.

Vamos inicialmente mostrar dois circuitos lógicos e verificar se são equivalentes.

Sejam os circuitos abaixo:



Note que a expressão $S2 = (A'.B')'$ é outra representação da expressão acima de $S2$.

A	B	A'	B'	A'.B'	S1=A+B	S2=(A'.B')'
0	0	1	1	1	0	0
0	1	1	0	0	1	1
1	0	0	1	0	1	1
1	1	0	0	0	1	1

Portanto, verificamos que, na tabela verdade, $S1 = S2$. Portanto, podemos afirmar que os circuitos lógicos **S1** e **S2** são **EQUIVALENTES**.

Do mesmo modo, podemos afirmar que as 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.

Portanto, no exemplo anterior, $A+B = (A'.B')'$

Vamos ver agora se $S1=A$ e $S2=A.(A+B)$ são equivalentes.

A	B	A+B	S1=A	S2=A.(A+B)
0	0	0	0	0
0	1	1	0	0
1	0	1	1	1
1	1	1	1	1

Observando a tabela verdade correspondente, podemos afirmar que S1 e S2 são equivalentes.

Atividade 8

Verifique, usando a tabela verdade se as expressões S1 e S2 são equivalentes.

- ♦ $S1 = A.(B+C)$
- ♦ $S2 = A.B + A.C$

Solução da atividade 8:

São equivalentes, pois a tabela verdade dos dois é igual.

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

Em seguida, apresentamos algumas propriedades provadas por tabelas verdade:

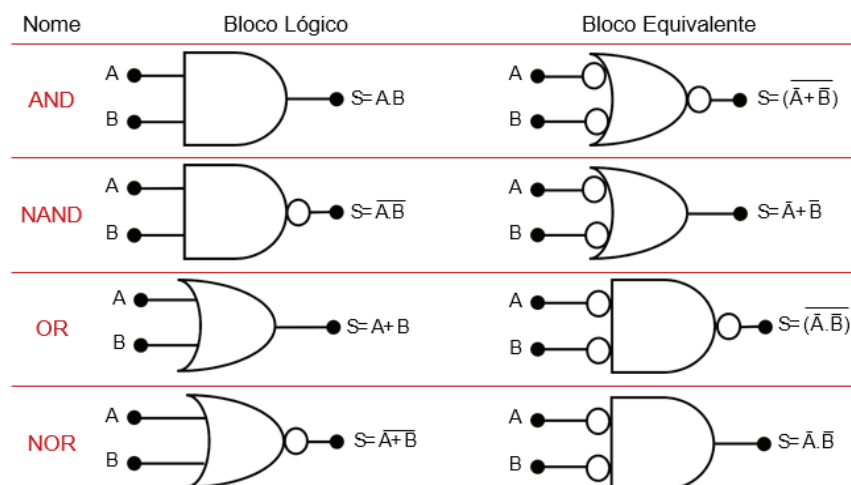
Absorção

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

Distributiva

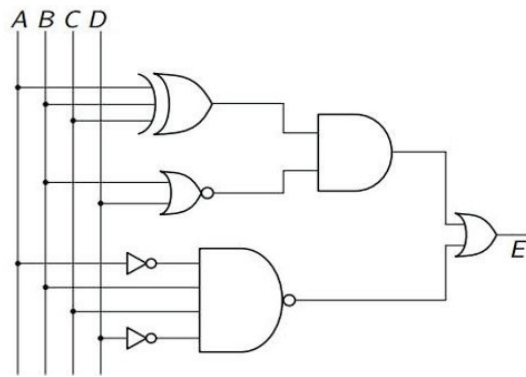
- $A.(B+C) = A.B + A.C$
- $A+(B.C) = (A+B) . (A+C)$

Os seguintes blocos lógicos (circuitos lógicos) da figura 15 são equivalentes e, portanto, esta equivalência pode ser comprovada através das tabelas verdade iguais.



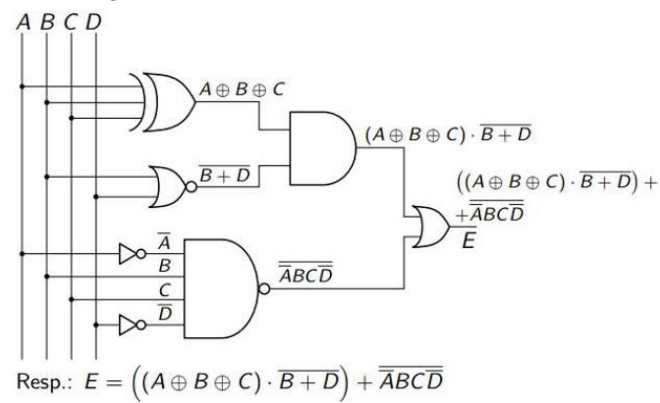
Atividade 9

- a. Dado o circuito a seguir, encontre uma expressão lógica para E em função das entradas A, B, C e D.



Aula 5: Conceitos de Lógica Digital

A solução é a seguinte:

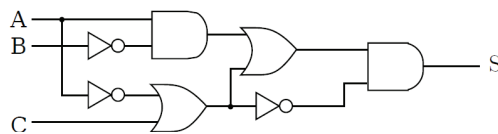


- b.** Dada a expressão booleana seguinte, construa o circuito lógico correspondente:

$$S = [(A.B') + (A'+C)].(A'+C)'$$

[illegible]


A solução é a seguinte:



$$S = [(A.B') + (A'+C)].(A'+C)'$$



leitura indicada

Para saber mais sobre a lógica de Boole e suas propriedades, sugerimos a leitura complementar do artigo **A Álgebra de Boole** disponível em:  <<https://goo.gl/d1fGnX>>.



síntese

Nesta aula de Conceitos de Lógica Digital ,começamos mostrando um histórico da evolução da lógica formal, que se integrou à Álgebra e originou a Álgebra de Boole. Em seguida, apresentamos todas as funções lógicas básicas utilizadas em circuitos lógicos. Mostramos a correspondência entre as lógicas digitais básicas com as expressões booleanas e as tabelas verdade correspondentes. Posteriormente, apresentamos a equivalência entre blocos lógicos e entre expressões booleanas. Estes tópicos, além da apresentação conceitual, foram profundamente praticados através de exercícios e tarefas diversas.

[illegible]