Lógica digital

Prof. Mauro Cesar Cantarino Gil

Descrição

A lógica digital através das operações da álgebra booleana.

Propósito

Compreender a lógica booleana e a importância das aplicações de portas e circuitos lógicos no desenvolvimento de programas e equipamentos eletrônicos.

Objetivos

Módulo 1

Operações básicas da álgebra booleana

Identificar as operações básicas da álgebra booleana.

Módulo 2

Portas e operações lógicas

Compreender portas lógicas, operações lógicas e as suas tabelas-verdade.

Módulo 3

Expressões lógicas e diagramas lógicos

Aplicar as expressões lógicas e diagramas lógicos.

Introdução

As máquinas se comunicam de forma binária (bits 0 ou 1). Fisicamente, os circuitos eletrônicos são construídos para que gerem tensões que representem esses bits 0 e 1 de acordo com as ações que a máquina deve executar.

Dessa forma, o computador possui vários circuitos lógicos que precisam ser orientados em como atuar, a partir de orientações lógicas baseadas em 0 e 1. Para fazer isso, é preciso utilizar a regra booleana, que corresponde a uma linguagem baseada em símbolos, letras e conectores para que as máquinas gerem os resultados pretendidos a partir de entradas de 0 e 1. Interessante, não?

No decorrer do conteúdo, você aprenderá os conceitos básicos das regras booleanas, como elas influenciam no desenvolvimento dos softwares e dos equipamentos eletrônicos.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

1 - Operações básicas da álgebra booleana

Ao final deste módulo, você será capaz de identificar as operações básicas da álgebra booleana.

Portas lógicas e lógica booleana

As operações que são realizadas por um computador digital (binário), vistas como complexas, podem ser compreendidas como simples **combinações de operações aritméticas e lógicas básicas**, como visto a seguir:

- Somar bits
- Complementar bits
- Mover bits
- Comparar bits

Estas operações lógicas são implementadas através de circuitos eletrônicos denominados circuitos lógicos, os quais também são conhecidos como **gates** ou **portas lógicas**.

Na lógica digital, há somente duas condições, 1 e 0, e os circuitos lógicos utilizam faixas de tensões predefinidas para representar esses valores binários. Assim, é possível construir circuitos lógicos que possuem a capacidade de produzir ações que irão permitir tomadas de decisões inteligentes, coerentes e lógicas.

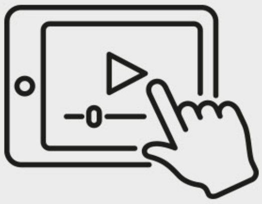
> É importante que tenhamos a capacidade de descrever a operação dos circuitos, pois eles são citados repetidas vezes em textos técnicos.

Boole desenvolveu a sua lógica a partir de símbolos e representou as expressões por letras, efetuando a sua ligação através dos conectivos (símbolos algébricos).

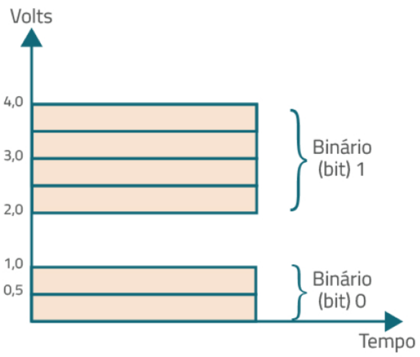
Lógica booleana

Veja, a seguir, como a lógica booleana está presente em nossa vida.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



Para este tipo de aplicação, podemos analisar a conversão de um valor de uma tensão em um determinado circuito, conforme apresentado na gráfico a seguir, em que os valores considerados como baixos serão convertidos em 0 (zeros) e os valores considerados altos serão convertidos em 1 (um).



Mauro Cesar Catarino Gil

Perceba que, com a tensão baixa (bit 0), o equipamento estará desligado, já, com a tensão alta (bit 1), ele estará ligado.

BIT	AÇÃO	CONDIÇÃO	TENSÃO	RESULTADO
1	LIGADO	VERDADEIRO	ALTO	SIM
0	DESLIGADO	FALSO	BAIXO	NÃO

O pesquisador Claude Shannon, do Instituto de Tecnologia de Massachusetts (MIT), em 1938, propôs que a álgebra booleana poderia ser utilizada para resolver problemas com projetos de circuitos com comutadores. A partir das técnicas de Shannon, foi possível a sua aplicação na análise e no desenvolvimento de circuitos digitais eletrônicos. Assim, a álgebra booleana, através das suas propriedades básicas, se mostra eficiente como uma ferramenta para:

Análise

A função de um circuito digital é descrita de acordo com a análise de um modo simplificado.

Projeto

A lógica booleana é utilizada para que seja desenvolvida uma implementação simplificada desta função, ao especificar uma determinada função de um circuito.

Iniciando o nosso estudo, representaremos os operadores lógicos e, a partir desses, perceberemos a representação das suas respectivas portas lógicas. Neste caso, para que possamos compreender os valores resultantes de cada operador lógico, é necessário conhecer as Tabelas Verdade, que são tabelas que representam todas as possíveis combinações dos valores das variáveis de entrada com os seus respectivos valores de saída.

Tabela-verdade

É uma técnica utilizada para descrever como a saída de um circuito lógico é dependente dos níveis lógicos de entrada, isto é, são tabelas que conterão **todas as possíveis combinações das variáveis de entrada de uma determinada função e, como resultado, os valores de saída**.

Neste caso, a tabela-verdade conterá o número necessário de linhas para representar todas as combinações possíveis das suas variáveis de entrada.

Os valores 0 e 1 são considerados como 0 = FALSO e 1 = VERDADEIRO. Como exemplo, observe a imagem a seguir:

Representação de um circuito com duas entradas e uma saída

Agora, veja como fica a tabela-verdade do circuito com duas entradas e uma saída:

Operadores e portas lógicas básicas

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

Portas lógicas

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

Uma porta lógica é um componente de hardware que terá um ou muitos sinais de entrada e, como consequência, produzirá um sinal de saída de acordo com a lógica estabelecida na construção do circuito em questão.

Os operadores booleanos básicos também denominados como **funções lógicas básicas** são:

OR

AND

NOT

O operador e a porta OR (OU)

O operador **OR** (OU) é a primeira das três operações básicas que vamos estudar e, para ilustrar a sua aplicação, vamos utilizar o seguinte cenário:

Ao abrir a porta de um automóvel, a lâmpada de iluminação da cabine do veículo deverá acender?

A resposta é **sim**.

E, ao fechar a porta, a lâmpada deverá ser desligada.

Dessa forma, a lâmpada estará acesa em duas situações distintas, se a porta do veículo estiver aberta **OU (OR)** o interruptor da lâmpada for acionado, mesmo com a porta fechada.

Neste cenário, vamos representar cada uma dessas seguintes possibilidades:

- A variável **A** representará a abertura da porta;
- A variável **B** representará o interruptor;
- A variável **X** representará o estado da lâmpada, se está acesa ou apagada.

Assim, a expressão booleana para a operação OR é definida como:

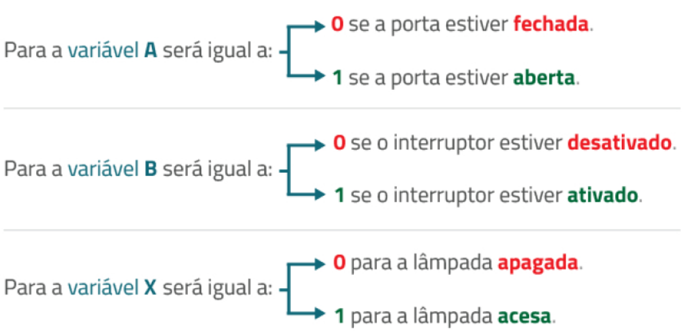
$$X = A + B$$

Rotacione a tela.

Em que o sinal (+) **não** representa uma soma, e sim a operação OR, cuja expressão é lida como:

> X é igual a A OR B.

Ao analisar as combinações possíveis, levando em consideração os valores, temos o seguinte:



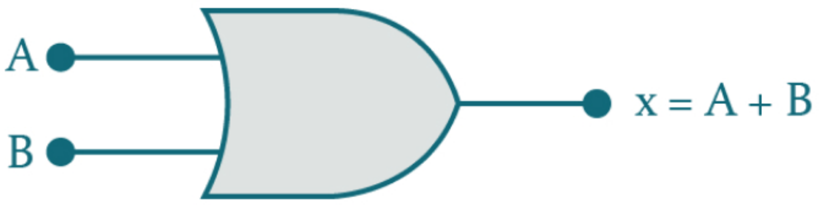
Resultado das análises das combinações dos valores

Em síntese, na tabela a seguir, estão representadas as combinações dos valores possíveis com a construção da tabela-verdade para o operador OR com duas entradas:

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

Ao analisar a tabela-verdade, chegaremos à conclusão de que a lâmpada estará **apagada** (valor igual a 0, FALSO) se — e somente se — tanto o interruptor quanto a porta possuírem o valor de entrada igual a FALSO (igual a 0) e, para as demais combinações, a lâmpada estará **acesa** (igual a 1, VERDADEIRO).

Nos circuitos digitais, uma porta **OR** é um circuito que tem duas ou mais entradas e a sua saída é igual à combinação das entradas através da operação OR, como ilustrado na seguinte imagem:



Símbolo de uma porta OR com duas entradas e uma saída

A seguir, veja outro caso e a sua correspondente tabela-verdade com três entradas e uma saída:

Símbolo de uma porta OR

Vamos praticar!

Seja **A = 1100, B = 1111 e C = 0001**, para calcular **L = A + B + C (A or B or C)**, o cálculo deve ser realizado em duas etapas, utilizando a seguinte Tabela Verdade da porta **OR**:

Na primeira etapa, vamos calcular **M = A + B (A or B)** e, em seguida, o resultado parcial será obtido **(M)**, combinado com C em outra operação lógica **OR (M or C)**, sempre utilizando as combinações de entrada e os resultados definidos nas seguintes tabelas-verdade da porta **OR**, uma de resultado parcial **M = 1111** e outra de resultado: **L = 1111**:

O operador e a porta AND (E)

O operador **AND (E)** é a segunda das três operações básicas que vamos estudar e, apenas para ilustrar a sua aplicação, vamos utilizar o seguinte cenário:

Ao acionar a botoeira da cabine de um elevador, será acionado o motor do elevador imediatamente?

Esta resposta dependerá de que outras condições estejam atreladas ao seu acionamento.

Vamos analisar o acionamento deste botão em conjunto com um sensor que identificará se a porta do elevador está fechada. Logo, a resposta será **sim**.

Se a porta estiver fechada, então o motor será ligado. O motor será acionado (valor igual a 1, verdadeiro) em uma única situação, se a porta do elevador estiver fechada (valor igual a 1, verdadeiro) E (AND) o botão do elevador for acionado (valor igual a 1, verdadeiro).

Neste cenário, vamos representar cada uma dessas possibilidades:

- A variável **A** representará o sensor da porta;
- A variável **B** representará o botão;
- A variável **X** representará o acionamento do motor.

Deste modo, a expressão booleana para a operação AND é definida como:

$$\mathrm{X} = \mathrm{A} \cdot \mathrm{B}$$

Rotacione a tela.

Em que o sinal (•) **não** representa uma multiplicação, e sim a operação AND e a expressão é lida como:

> X é igual a A AND B.

Agora, ao analisar as combinações possíveis, levando em consideração os valores, temos o seguinte:

Resultado das análises das combinações dos valores.

Em síntese, na tabela a seguir estão representadas as combinações com dois valores de entrada e uma saída para a da Tabela Verdade com o operador **AND**:

Ao analisar a tabela-verdade, chegaremos à conclusão de que o motor será **acionado** (valor igual a 1, **verdadeiro**) se — e somente se — tanto o botão quanto o sensor da porta possuírem o valor igual a **verdadeiro** (igual a 1) e, para as demais combinações, o motor estará **desligado** (igual a 0).

Nos circuitos digitais, uma porta **AND** é um circuito que tem duas ou mais entradas e a sua saída é igual à combinação das entradas através da operação AND, conforme ilustrado na imagem a seguir:

Símbolo de uma porta AND com duas entradas e uma saída

A seguir, veja outro caso e a sua correspondente tabela-verdade com três entradas e uma saída:

Símbolo de uma porta AND.

Vamos praticar!

Prática 1

Seja A = 1 e B = 0, calcule o valor de X, quando X = A • B (A and B). Analisando a seguinte tabela-verdade da porta AND:

Podemos verificar que o valor de X = 0, pois 1 and 0 = 0.

Prática 2

Agora, seja A = 0110 e B = 1101, calcule o valor de X, quando X = A • B (A and B). Analisando a seguinte tabela-verdade da porta AND:

Temos o seguinte:

Operando bit a bit, encontramos o valor de X = 0100.

O operador e a porta NOT (NÃO)

O operador **NOT** (NÃO) ou **inversor** é a terceira das três operações básicas que estudaremos, sendo este operador totalmente diferente dos outros já estudados, porque pode ser realizado através de uma única variável.

Como exemplo, se uma variável A for submetida à operação de inversão, o resultado X pode ser expresso como:

$$\mathrm{X} = \bar{A}$$

Rotacione a tela.

Em que a barra sobre o nome da variável representa a operação de inversão e a expressão é lida como:

> X é igual a NOT A ou X é igual a A negado ou X é igual ao inverso de A ou X é igual ao complemento de A.

Como utilizaremos a barra para identificar a negação, outra representação também é utilizada para a inversão por outros autores, que é a seguinte:

$$\mathrm{A}' = \overline{\mathrm{A}}$$

Rotacione a tela.

Em síntese, a representação da tabela-verdade para o operador NOT com uma entrada e uma saída é a seguir:

Nos circuitos digitais, uma porta NOT é um circuito que tem uma entrada, e a sua saída, a negação, é indicada por um pequeno círculo, como mostrado a seguir:

Símbolo de uma porta NOT com uma entrada e uma saída.

Outras portas lógicas fundamentais

A porta NOR (Não OU)

Para ampliar o nosso estudo sobre as portas lógicas, é importante perceber que o inversor, ou a função **NOT (NÃO)**, pode ser aplicada tanto em variáveis como em portas lógicas inteiras, assim invertendo toda sua saída.

Neste contexto, seria possível concatenar a saída de uma porta lógica com a entrada do inversor conforme apresentado na figura a seguir, produzindo, assim, a inversão de todos os seus valores de saída.

Mas é possível construir a sua representação de uma forma diferente, permitindo criar a inversão em toda porta lógica de modo bastante peculiar — com a identificação de um pequeno círculo na sua saída, indicando a inversão. Veja a imagem a seguir:

A porta NOR (Não OU).

Você pode perceber que o símbolo da porta NOR de duas entradas representado na imagem a seguir é o mesmo símbolo utilizado para representar a porta OR, com apenas uma diferença — a inclusão de um pequeno círculo na sua saída, que representa a inversão da operação OR.

A expressão que representa a porta NOR é:

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

Rotacione a tela.

Note que a barra que indica a negação/inversão será estendida a todas as variáveis de entrada, neste exemplo com duas variáveis:

Símbolo de uma Porta NOR.

A porta NOR ('NOT OR' 'NÃO-OU').

A tabela-verdade a seguir mostra que a saída da porta NOR é exatamente o inverso da saída da porta OR:

Ao analisar as combinações possíveis com duas entradas, levando em consideração os valores para as variáveis de entrada A e B, somente produzirá uma saída 1 (VERDADEIRO) se — e somente se — todas as entradas sejam 0 (FALSO) e, para as demais condições, produzirá como resultado na saída igual a 0 (FALSO).

Utilizando uma adaptação do cenário descrito na porta OR, podemos ter o seguinte:

Variável A, B e X

Nós podemos representar a aplicação desta função como: a lâmpada poderá estar apagada em duas situações distintas, se a porta do veículo estiver aberta e o interruptor da lâmpada for acionado. Neste cenário, vamos representar cada uma dessas possibilidades: a variável A representará a abertura da porta, a variável B representará o interruptor e a variável X representará o estado da lâmpada; se está acesa ou apagada.

Igual a 0 e igual a 1

Ao analisarmos as combinações possíveis, levando em consideração os valores para a variável A, será igual a 0 se a porta estiver fechada e será igual a 1 se a porta estiver aberta. Em relação à variável B, temos o valor igual a 0 para o interruptor ativado e 1 para o interruptor desativado e, por fim, a variável X possuirá o valor 0 para a lâmpada apagada e 1 para a lâmpada acesa.

A porta NAND (Não E)

Você pode perceber que o símbolo da porta **NAND** de duas entradas, representado na imagem a seguir, é o mesmo símbolo utilizado para representá-la, com apenas uma diferença – a inclusão de um pequeno círculo na sua saída, que representa a inversão da operação **AND**.

A expressão que representa a porta **NAND** é:

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

Rotacione a tela.

Em que a barra sobre o nome da variável representa a operação de inversão e a expressão é lida como:

> X é igual a NOT A ou X é igual a A negado ou X é igual ao inverso de A ou X é igual ao complemento de A.

Veja o símbolo e o circuito a seguir:

Símbolo de uma porta NAND.

A porta NAND (‘NOT AND’ ‘NÃO-E’).

A seguinte A tabela-verdade mostra que a saída da porta NAND é exatamente o inverso da saída da porta AND:

Ao analisar as combinações possíveis, levando em consideração os valores para as variáveis de entrada A e B, somente produzirá uma saída 0 (FALSO) se — e somente se — todas as entradas forem 1 (VERDADEIRO) e, para as demais condições, produzirá como resultado na saída igual a 1 (VERDADEIRO).

Conforme apresentado anteriormente, vamos exibir um cenário:

Um semáforo para bicicletas e um sensor de movimento para a passagem de pedestres.

Assim, o sinal verde (liberando a passagem para ciclistas) estará ligado (0, FALSO) se duas condições forem atendidas: um botão seja acionado (1, VERDADEIRO) e o sensor de movimento não detecte a existência de um pedestre naquele momento (1, VERDADEIRO). Para as demais possibilidades, o semáforo estará com o farol vermelho acionado (1, VERDADEIRO), bloqueando o tráfego de ciclistas.

Vamos praticar!

Seja $A = 10010$ e $B = 11110$, calcule $\mathbf{X} = \overline{\mathrm{A} \cdot \mathrm{B}}$. Uma resposta interessante para este caso é a realização de duas operações lógicas em sequência. Primeiro, realiza-se a operação AND e, em seguida, obtém-se o inverso do resultado, produzindo o valor final para uma operação NAND.

Pela tabela-verdade da porta AND, temos:

O resultado parcial: L = 10010. Invertendo os bits de L, usando a Tabela Verdade da porta NOT:

$$10010 = \mathbf{T} = \mathbf{A} \cdot \mathrm{B}$$

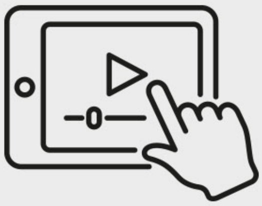
$$01101 = \overline{\mathrm{T}} = \overline{\mathrm{A} \cdot \mathrm{B}}$$

Resultado: $X = \overline{T} = \overline{\mathbf{A} \cdot \mathbf{B}} = 01101$.

Resolução da expressão booleana

Veja, a seguir, o vídeo para melhor compreensão da lógica apresentada na situação que acabou de ler.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



A porta XOR (Ou exclusivo)

A porta **XOR** que é uma abreviação do termo **exclusive or**, poderá ser considerada como um caso particular da função **OR**.

Neste sentido, a porta XOR produzirá um resultado igual a 1 (VERDADEIRO), se pelo menos um dos valores das entradas for diferente dos demais (exclusividade de valor da variável), isto é, a porta produzirá o resultado 0 (FALSO) se — e somente se — todos os valores das entradas forem iguais, conforme é apresentado na tabela-verdade a seguir:

A expressão que representa a porta XOR é:

$$X = A \oplus B$$

Rotacione a tela.

E seu símbolo fica da seguinte forma:

Símbolo de uma Porta XOR.

Para ilustrar a sua aplicação, vamos utilizar o seguinte cenário: ao se acionar um motor elétrico de um equipamento por dois botões distintos em dois locais diferentes. O motor somente será acionado se — e somente se — um dos botões for acionado (valor igual a 1, VERDADEIRO). Para os demais casos, o botão não fará a atuação do motor (valor igual a 0, FALSO), isto é, se ambos os botões não forem acionados ou ambos forem acionados ao mesmo instante, o equipamento não será ligado. Assim, podemos, neste exemplo, utilizar uma função XOR.

Vamos praticar!

Seja A = 1 e B = 0, calcule o valor de X, quando $X = A \oplus B (A \text{ xor } B)$. Analisando a tabela-verdade da porta XOR, temos:

Podemos verificar que:

$$A = 1 \oplus B = 0 \text{ ou } 0 \oplus 1 = 1$$

Rotacione a tela.

O valor de X = 1, pois 0 xor 1 = 1.

A porta XNOR (coincidência)

A expressão que representa a porta XNOR é:

$$\mathbf{X} = \mathrm{A} \odot \mathrm{B}$$

Rotacione a tela.

A tabela-verdade da porta XNOR fica da seguinte forma:

E o símbolo a seguir:

Símbolo de uma Porta XNOR.

Poderíamos citar um exemplo que negue a condição definida pela função XOR. Para ilustrar a sua aplicação, vamos utilizar o seguinte cenário:

Ao se acionar uma porta rotatória em um banco através de dois botões distintos, localizados nos dois lados da porta. A porta será liberada (0, FALSO) se — e somente se — um dos botões for acionado (valor igual a 1, VERDADEIRO).

Para os demais casos, ele manterá a porta bloqueada (valor igual a 0, FALSO), isto é, se ambos os botões não forem acionados ou ambos forem acionados ao mesmo instante. Assim, podemos, neste exemplo, utilizar uma função XNOR.

Falta pouco para atingir seus objetivos.

Vamos praticar alguns conceitos?

Questão 1

Sendo os valores para as variáveis de entrada com 4 bits A = 0110 e B = 1101, qual é o resultado da função Z = A • B?

A $Z = 0100$

B $Z = 1011$

C $Z = 1111$

D $Z = 1101$

E $Z = 0000$

Parabéns! A alternativa A está correta.

Como existem duas variáveis de entrada com 4 bits, é necessário efetuar o cálculo da função AND bit a bit entre o par de variáveis, da seguinte forma:

Questão 2

Qual seria a função lógica que representaria o seguinte cenário: Em um ambiente monitorado, existem sensores e uma central de alarme. Neste caso, o alarme sonoro Y será disparado (VERDADEIRO), se pelo menos um dos três sensores (A, B e C) estiver ativado (VERDADEIRO).

A $Y = A \cdot B \cdot C$

B $Y = A \cdot B + C$

C $Y = A \cdot (B + C)$

D $Y = A + B + C$

E $Y = A \cdot C + B$

Parabéns! A alternativa D está correta.

Para produzir essa solução, vamos construir a Tabela Verdade com 3 variáveis de entrada (A,B,C) e uma de saída Y. O valor de saída deverá ser restrito ao cenário, isto é, o alarme somente não irá disparar se nenhum dos sensores estiver ativo.

Neste caso, a representação será uma função OR com três entradas e uma saída, isto é, a saída será VERDADEIRA sempre que existir ao menos uma entrada verdadeira.

$Y = A + B + C$

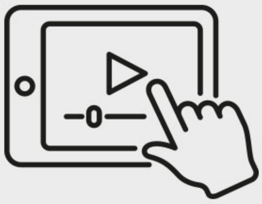
Apenas para subsidiar a solução, considere a função OR para as duas primeiras variáveis A + B. O resultado somente será falso se ambas as entradas forem FALSAS. Agora, combinando este resultado (FALSO) com a função OR e a variável C. Novamente, somente será FALSO quando ambas as entradas forem FALSAS.

2 - Portas e operações lógicas

Ao final deste módulo, você será capaz de compreender portas lógicas, operações lógicas e suas tabelas-verdade.

Expressões lógicas

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



Em muitos casos, a interpretação de um circuito digital exige uma análise aprofundada e meticulosa sobre o circuito com o objetivo de conhecer o seu funcionamento, ou mesmo para analisar situações de criação, falha, expansão e alterações, entre outras possibilidades.

Neste contexto, é interessante que se utilize de uma forma para a representação de um circuito que esteja em formato de uma **expressão algébrica**.

> As expressões lógicas, também denominadas como funções lógicas, podem ser definidas da mesma forma que uma expressão algébrica, isto é, com o uso de sinais de entrada (variáveis lógicas – binárias), ligados por conectivos lógicos (símbolos que representarão uma operação lógica, com parênteses, opcionalmente) e também com o sinal de igualdade (=), produzindo, como resultado, um único sinal de saída.

Desse modo, podemos afirmar que todo circuito lógico executará uma expressão booleana. Por exemplo:

$$\mathrm{X} = \mathrm{A} + \overline{\mathrm{B}} \cdot \mathrm{C}$$

Rotacione a tela.

O exemplo **X** é uma expressão lógica e, como uma função lógica, somente poderá possuir como valor 0 ou 1. O seu resultado dependerá dos valores das variáveis A, B e C e das operações lógicas OR, NOT e AND, nesta expressão apresentada.

Vamos praticar!

Vamos supor que, a partir de um circuito lógico, devemos construir a sua respectiva expressão lógica. Seja o circuito:

Circuito lógico.

Como sugestão, utilizaremos a decomposição deste circuito em partes (blocos) a partir da sua saída (S).

Decomposição do circuito.

Dessa maneira, analisaremos o circuito nas duas seguintes partes:

Decomposição do circuito.

Agora, para obter a expressão final deste circuito, vamos substituir a expressão S1 em função de A e B. Como temos a seguinte fórmula:

$$\mathrm{S_1 = A \cdot B}$$
$$\mathrm{S = S_1 + C}$$

Rotacione a tela.

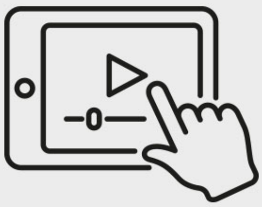
Temos, então:

$$\mathrm{S}_1 = (\mathrm{A} \cdot \mathrm{B}) + \mathrm{C}$$

Rotacione a tela.

Avaliação de uma expressão lógica

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



Na avaliação de uma expressão lógica, uma ordem de precedência deverá ser seguida da mesma forma que é considerada em uma expressão aritmética, de acordo com o definido a seguir:

1. Avalie NOT;
2. Avalie END;
3. Avalie OR.

Lembre-se de que os conteúdos entre parênteses **devem ser executados primeiro**.

Seguindo o exemplo anterior, $\mathrm{X} = \mathrm{A} + \overline{\mathrm{B}} \cdot \mathrm{C}$, lê-se:

$$X = AOR\ (\mathrm{NOT\ B})\ \mathrm{AND\ C}$$

Rotacione a tela.

A seguir, o diagrama da função fica:

Diagrama lógico da função X.

Como sugestão, faça a construção desta tabela-verdade seguindo a ordem de precedência da expressão:

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

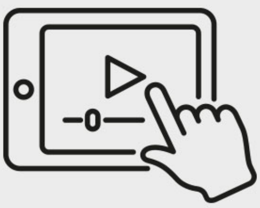
Rotacione a tela.

Ou seja, NOT B, (NOT B) AND C, A OR (NOT B) AND C. Como temos três variáveis de entrada com dois valores possíveis (0 e 1) para cada uma, temos **2^3 = 8** combinações, conforme a tabela-verdade da função X a seguir:

Expressões lógicas

Veja, a seguir, um vídeo sobre expressões lógicas.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



Equivalência de funções lógicas

Conceito

Duas funções lógicas são equivalentes se — e somente se — **para a mesma entrada, produzirem iguais valores de saída**, isto é, quando duas funções lógicas possuírem o mesmo resultado na sua tabela-verdade, esses circuitos serão considerados como equivalentes.

Vamos praticar!

Prática 1

Seja a função $\mathrm{X} = \overline{\mathbf{A} \cdot \mathbf{A}}$, ao construir a tabela-verdade da função X, temos:

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

Como você pode verificar, o resultado da tabela-verdade da função $\mathrm{X} = \overline{\mathrm{A} \cdot \mathrm{A}}$ é idêntico à tabela-verdade da função $\mathrm{Y} = \overline{\mathrm{A}}$.

Assim, podemos afirmar que tanto a **função** X como a **função** Y são **equivalentes**.

Prática 2

Escreva a expressão lógica a partir do diagrama lógico a seguir:

Diagrama lógico.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

Partindo da variável **saída X**, nós podemos representar a expressão deste circuito nas seguintes partes:

A partir da variável X, nós temos a porta **NAND** que receberá os seguintes valores de entrada **not** $\mathbf{B}$ e um resultado intermediário $\mathbf{T}_1$, assim $X = B + \mathbf{T}_1$.

Para resolver o valor de **T1**, iremos identificar que **T1** será o valor de saída da porta **NAND** que possui os seguintes valores de entrada: **not** B e A, assim, $\mathrm{T}_1 = \bar{B} \cdot \mathrm{A}$.

Substituindo a equação que possui T1 como saída na primeira equação, temos:

$X = \overline{\bar{B} \cdot \overline{(\bar{B} \cdot A)}}$

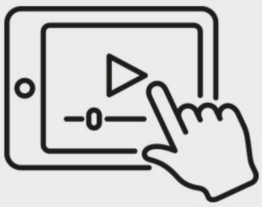
Substituindo a equação que possui T1 como saída na primeira equação, temos:
$\mathrm{F} = \mathrm{X} + \bar{Y} \cdot \mathrm{Z}$.

Prática 3

Escreva a expressão lógica a partir do diagrama lógico a seguir:

Mauro Cesar Catarino Gil

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



Partindo da variável saída **F**, nós podemos representar a expressão deste circuito nas seguintes partes:

A partir da variável **F** nós temos a porta **OR** que receberá os seguintes valores de entrada **X** e um resultado intermediário **T_1**, assim, $\mathrm{F} = \mathrm{X} + \mathrm{T}_1$.

Para resolver o valor de **T1**, iremos identificar que **T1** será o valor de saída da porta **AND** que possui os seguintes valores de entrada: **not Y e Z**, assim, $\mathrm{T}_1 = \overline{\mathrm{Y}} \cdot \mathrm{Z}$.

Substituindo a equação que possui T1 como saída na primeira equação, temos:
$\mathrm{F} = \mathrm{X} + \overline{\mathrm{Y}} \cdot \mathrm{Z}$

Prática 4

Seja A = 1, B = 0, C = 1, D = 1, calcule $\mathrm{X} = \mathrm{A} + \overline{\mathrm{B} \cdot \mathrm{C}} \oplus \mathrm{D}$.

Adotando o esquema de prioridade, o valor de X será obtido com a realização das quatro etapas seguintes:

Prática 4

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

Realizar a operação **AND** (maior prioridade, além de ter uma inversão determinada sobre a operação). Assim, trata-se de calcular $\mathrm{B} \cdot \mathrm{C} = \mathrm{T}_1$.

Inverter o resultado parcial T_1.

Realizar a operação OR (as operações $\mathbf{OR}$ e XOR têm mesma prioridade, optando-se pela que está primeiro à esquerda).

Assim, calcula-se: $T_2 = \mathrm{A} + T_1$.

Realizar a operação XOR, calculando-se $\mathrm{X} = T_2 \oplus \mathrm{D}$.

Assim, vamos efetuar as etapas indicadas:

$$0 \cdot 1 = 0 = \mathrm{T}_1$$

$$\overline{0} = 1$$

$$1 + 1 = 1 = T_2$$

$$1 \oplus 1 = 0 = \mathrm{X}$$

Resultado igua a X = 0.

Prática 5

Seja A = 0, B = 0, C = 1, D = 1, calcule $\mathrm{X} = (A + \overline{\mathrm{B}} \oplus \mathrm{D}) + (\overline{\mathrm{C}} \cdot \mathrm{B}) \oplus \mathrm{A}$.

Adotando uma sequência de etapas, vamos considerar a ordem de precedência de cada operação.

A primeira prioridade é solucionar os parênteses, e dentro ou fora destes, a prioridade é da operação **AND** sobre as demais, exceto se houver inversão (**NOT**).

Assim, temos:

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

Calcular o parêntese mais à esquerda; dentro deste parêntese, efetuar primeiro a inversão do valor de **B: B = 0 e notB = 1**.

Ainda dentro do parêntese, efetuar **A + notB** ou 0 + 1 = 1 = T_1.

Encerra-se o cálculo do interior do parêntese efetuando $\mathrm{T}_1 \oplus \mathrm{D} = \mathrm{T}_2$. Total do parêntese:

$$1 \oplus 1 = 0 = T_2$$

$$T_1 = 1 \text{ e } T_2 = 0$$

Calcula-se o outro parêntese, primeiro invertendo o valor de **C** (**NOT**), depois efetuando a operação **AND** daquele resultado com a variável **B**. O valor final é, temporariamente, T_3. C = 1 e not C = 0

$$0 \cdot 0 = 0 = \mathrm{T}_3$$

Finalmente, calcula-se a operação **OR** do resultado do primeiro parêntese (**T2**) com o do outro parêntese (**T3**), para concluir com a operação **XOR** com **A**.

0 + 0 = 0

$$\mathrm{X} = 0 \oplus 0 = 0$$

Resultado: X = 0

Prática 6

Vamos realizar operações lógicas com palavras de dados? Isto é, com variáveis de múltiplos bits.

Seja A = 1001, B = 0010, C = 11110, D = 1111, calcule o valor de X na seguinte expressão lógica:

$$X = A \oplus (\overline{B \cdot C} + D) + (B \oplus \bar{D})$$

Rotacione a tela.

Para resolver esta expressão, nós utilizaremos o mesmo método, execução por etapas, mas, agora, são 4 algarismos binários em vez de um apenas. Considerando as prioridades já definidas anteriormente, temos:

Etapa a

Executar a operação **AND** de **B** e **C**, obtendo resultado parcial T_1.

Etapa b

Inverter o valor de **T_1 (not T_1)**.

Etapa c

Executar a operação **OR** de not **T_1**, com **D**, atualizando um novo resultado parcial **T_1**, que é a solução do primeiro parêntese.

Etapa d

Inverter o valor de **D** no segundo parêntese.

Etapa e

Executar a operação **OR** de **not T_1**, com **D**, atualizando um novo resultado parcial **T_1**, que é a solução do primeiro parêntese.

Etapa f

Executar a operação **XOR** de **B** com o inverso de **D**, obtendo o resultado parcial T_2, que é a solução do segundo parêntese.

Etapa g

Executar a operação **XOR** de **A** com T_1, obtendo um valor temporário para **X**.

Etapa h

Executar a operação **OR** de **X** com T_2, obtendo o resultado de **X**.

Executando as etapas aqui indicadas, temos etapa (a): $T_1 = B \bullet C$, com resultado parcial: $T_1 = 0010$. Veja a tabela a seguir:

Temos a etapa (b): T_1 = not T_1 em que $T_1 = 0010$ e not $T_1 = 1101$. Como resultado parcial, fica o novo $T_1 = 1101$.

Temos a etapa (c): $T_1 = T_1 + D$, com o resultado parcial: T_1 atualizado: $T_1 = 1111$.

Veja a tabela a seguir:

Temos a etapa (d): not D, em que D = 1111 e not D = 0000.

Temos a etapa (e): $\boldsymbol{T}_2 = \mathbf{B} \oplus \operatorname{not} D$ em que o resultado parcial é T_2 = 0010. Veja a tabela a seguir:

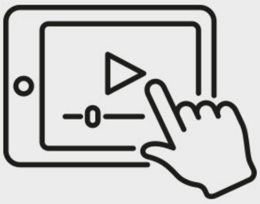
Temos a etapa $(f) : X = A \oplus T_1$, em que o resultado parcial X = 0110. Veja a tabela a seguir:

Temos a etapa (g): X = X + T_2, em que o resultado de **X = 0100**. Veja a tabela a seguir:

Operações lógicas com palavras de dados

Veja, a seguir, o vídeo em que apresentamos a realização de operações lógicas com palavras de dados.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



A etapa g comentada no vídeo será concluída na tabela abaixo. Lembrando que a operação or retornará um valor falso quando todas entradas forem falsas.

Etapa (g):$X = X + T_2$

Assim, sendo
A=1001, B=0010, C=1110, D=1111.
Calcular o valor de X na seguinte expressão lógica:

$$X = A \oplus (\bar{B} \cdot \mathbf{C} + D) + (B \oplus \bar{D})$$

irá retornar o valor final de **X**=0110.

Falta pouco para atingir seus objetivos.

Vamos praticar alguns conceitos?

Questão 1

Qual das expressões indicadas representa o circuito equivalente ao circuito abaixo?

A $X = A + A$

B $X = A \cdot A$

C $X = A$

D $\mathrm{X} = \overline{\mathrm{A}}$

E $X = \bar{A} \cdot A$

Parabéns! A alternativa D está correta.

Para analisar a equivalência, é necessário, inicialmente, produzir a tabela-verdade para o circuito cuja expressão é $\mathrm{X} = \overline{\mathrm{A} = A}$. Levando em consideração o resultado apresentado na tabela à esquerda, temos o seguinte resultado da tabela à direita:

Questão 2

Qual das expressões indicadas representa o circuito equivalente à figura a seguir?

A $X = A + A$

B $X = A \cdot A$

C $X = A \oplus A$

D $X = \overline{\mathrm{A}}$

E	$X = \bar{A} \cdot + A$

Parabéns! A alternativa D está correta.

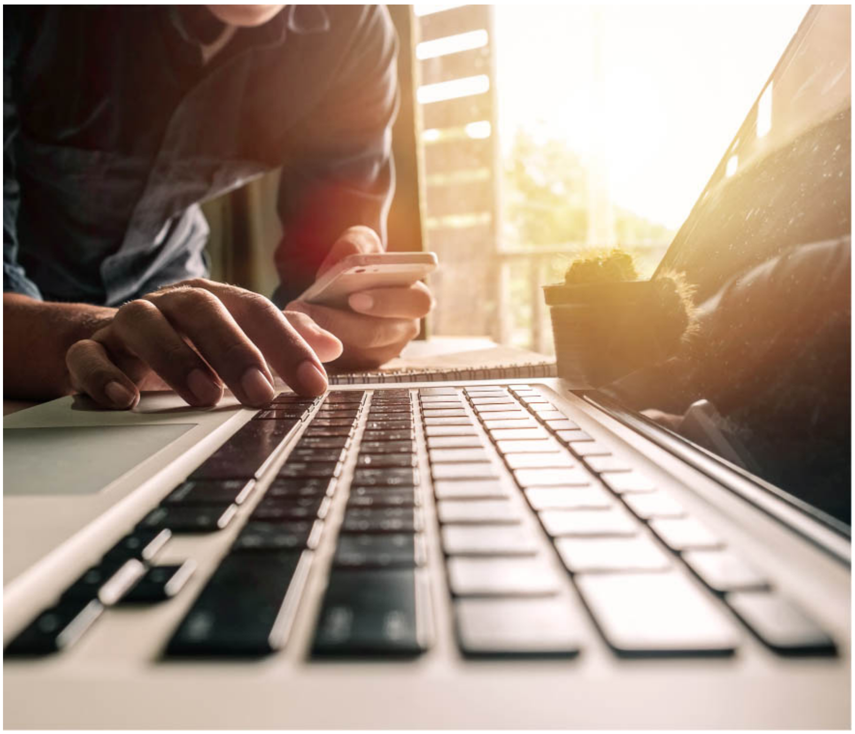
Para analisar a equivalência, é necessário, inicialmente, produzir a tabela-verdade para o circuito cuja expressão é:

$$\mathbf{X} = \overline{\mathbf{A} + \mathbf{A}}$$

Levando em consideração o resultado apresentado na tabela à esquerda, temos o seguinte resultado na tabela à direita:

Novamente a resposta é considerada:

$$\mathbf{X} = \overline{\mathbf{A} + \mathbf{A}} = \overline{\mathbf{A}}$$



3 - Expressões lógicas e diagramas lógicos

Ao final deste módulo, você será capaz de aplicar as expressões lógicas e diagramas lógicos.

Propriedades da álgebra de Boole

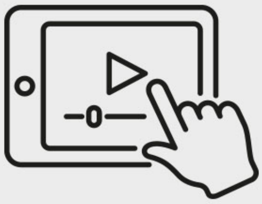
As regras básicas da álgebra de Boole são bastante úteis quando devemos analisar a equivalência e a simplificação das expressões booleanas (lógicas) que definem uma função de um determinado dispositivo digital.

Essas regras também permitem facilitar a compreensão do funcionamento de dispositivos digitais, assim como a redução de custos na fabricação de circuitos digitais com a redução de componentes eletrônicos usados.

Regras básicas da álgebra booleana

Veja, a seguir, a explicação narrada sobre as regras básicas da álgebra booleana.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



Confira, a seguir, a tabela das regras básicas da álgebra boolena:

Stalling, 2017.

Agora observe a tabela das propriedades da função Exclusive or (XOR):

Como sugestão, verifique **as equivalências dessas expressões** através da tabela-verdade.

Vamos praticar!

Prática 1

Vamos analisar a possibilidade de simplificar a seguinte expressão lógica:

$$X = [\overline{(\overline{\mathrm{A}} + \mathrm{B}) \cdot \overline{\mathrm{B}}}]$$

Rotacione a tela.

A seguir, veja o diagrama dessa expressão:

Mauro Cesar Catarino Gil

Como primeiro passo, vamos analisar as regras básicas da álgebra booleana e verificar se é possível aplicar alguma dessas regras na expressão.

Etapa a

Agora, podemos iniciar o processo de simplificação usando a **regra 12** referente ao **Teorema de de Morgan** na versão **AND** de $\overline{X \cdot Y} = \bar{X} + \bar{Y}$:

$$X = [(\overline{\bar{A} + B}) + \bar{B}]$$

Etapa b

Usando novamente o **Teorema de Morgan** na versão **OR** de $\overline{\mathbf{X} + \mathbf{Y}} = \overline{\mathbf{X}} \cdot \overline{\mathbf{Y}}$:

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

Etapa c

Aplicando a **regra 5** da involução $\overline{\bar{X}} = \mathbf{X}$ em $\mathbf{A}$ e $\mathbf{B}$, temos:

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

Etapa d

Continuando com a simplificação pela **regra 6** da comutatividade, temos:

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

Etapa e

Usando a **regra 9** da absorção **1**, temos:

$$X = A + B$$

Por fim, podemos perceber que tanto na expressão inicial quanto na expressão simplificada, ambas produzem o mesmo resultado através da seguinte tabela-verdade e, neste caso, pode ser utilizada uma simplificação de um circuito com uma porta **NAND** e inversores (**NOT**) sendo substituído por uma única porta **OR**.

E a tabela-verdade da expressão X=A+B:

Prática 2

Vamos simplificar a seguinte expressão:

$$X = \bar{A} \cdot B \cdot \bar{C} + \bar{A} \cdot B \cdot C + A \cdot \bar{B} \cdot C + A \cdot B \cdot C$$

Rotacione a tela.

Usando as regras básicas da álgebra booleana, temos:

Usando a **regra 9**, temos:

$$X = \bar{A} \cdot B \cdot (\bar{C} + C) + A \cdot C \cdot (\bar{B} + B)$$

Usando a **regra 4**, temos:

$$X = \bar{A} \cdot B \cdot 1 + A \cdot C \cdot 1$$

Usando a **regra 1**, temos:

$$X = \bar{A} \cdot B + A \cdot C$$

Prática 3

Vamos simplificar a seguinte expressão:

$$X = A \cdot B \cdot C + A \cdot \bar{C} \cdot + A \cdot \bar{B}$$

Rotacione a tela.

Usando as regras básicas da Álgebra booleana, temos:

Usando a **regra 8**, temos:

$$X = A \cdot (B \cdot C + \bar{C} + \bar{B})$$

Ordenando os termos:

$$X = A \cdot (B \cdot C + (\bar{C} + \bar{B}))$$

Usando a **regra 12**, temos:

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

Explicitando o termo **B • C = Y**, temos:

$$X = A \cdot (Y + \bar{Y})$$

Usando a **regra 4**, temos:

X = A • 1

Usando a regra 1, temos:

X = A

Usando a **regra 1**, temos:

$$X = \bar{A} \cdot B + A \cdot C$$

Aplicação das regras da álgebra booleana

Veja, a seguir, a resolução do exercício que aplica as regras da álgebra booleana.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

Prática 4

Vamos simplificar a seguinte expressão:

$$X = A \cdot (A + B)$$

Rotacione a tela.

Usando a **regra 9**, temos: X = A.

Prática 5

Vamos simplificar a seguinte expressão:

$$X = A \cdot A + A \cdot C + B \cdot A + B \cdot C$$

Rotacione a tela.

Usando as regras básicas da álgebra booleana, temos:

Usando a **regra 3**, temos:

X = A + A • C + B • A + B • C

Usando a **regra 9**, temos:

X = A + B • A + B • C

Usando a **regra 9 novamente**, temos:

X = A + B • C

Falta pouco para atingir seus objetivos.

Vamos praticar alguns conceitos?

Questão 1

Qual é a expressão simplificada que representa o circuito abaixo?

A $X = A + B$

B $X = A \cdot B$

C $X = A + \bar{B}$

D $X = A \oplus B$

E $X = \bar{A} + B$

Parabéns! A alternativa D está correta.

Para efetuar a simplificação, é necessário, inicialmente, escrever a expressão que representa este circuito. Como todas as portas utilizadas são **NAND**, será mais fácil de compreender. Apenas, como sugestão, procure interpretar a expressão a partir do valor de saída

$\mathbf{X} = \overline{\mathbf{A} \cdot \overline{\mathbf{A} \cdot \mathbf{B}}} \cdot \overline{\mathbf{B} \cdot \overline{\mathbf{A} \cdot \mathbf{B}}}$, aplicando a regra de Morgan, temos

$\mathbf{X} = \overline{\mathbf{A} \cdot \overline{\mathbf{A} \cdot \mathbf{B}}} + \overline{\mathbf{B} \cdot \overline{\mathbf{A} \cdot \mathbf{B}}}$, aplicando a regra da involução, temos

$\mathbf{X} = \mathbf{A} \cdot \overline{\mathbf{A} \cdot \mathbf{B}} + \mathbf{B} \cdot \overline{\mathbf{A} \cdot \mathbf{B}}$, aplicando a regra distributiva, temos

$\mathbf{X} = \overline{\mathbf{A} \cdot \mathbf{B}} \cdot (\mathbf{A} + \mathbf{B})$, aplicando a regra de Morgan, temos

$$X = (\bar{A} + \bar{B}) \cdot (A + B)$$

$\mathbf{X = \overline{A} \cdot A + \overline{A} \cdot B + \overline{B} \cdot A + \overline{B} \cdot B,}$

aplicando a regra do complemento, em que $\mathbf{\overline{A} \cdot A = 0}$ e $\mathbf{\overline{B} \cdot B = 0}$, temos

$\mathbf{X = \overline{A} \cdot B + \overline{B} \cdot A}$, aplicando a propriedade do XOR, temos

$$X = A \oplus B$$

Assim, seria possível, a partir de um circuito com portas **NAND**, substituí-lo por uma porta XOR.

Questão 2

Dados os valores de entrada, qual é o resultado da tabela-verdade para o circuito abaixo?

A K=0; L=0; M=0; N=1

B K=0; L=1; M=1; N=1

C K=1; L=0; M=1; N=0

D K=0; L=1; M=1; N=0

E	K=1, L=1, M=1, N=1

Parabéns! A alternativa D está correta.

Como o circuito apresenta duas variáveis binárias de entrada, existem somente quatro combinações distintas, sendo assim, a construção da Tabela Verdade dependerá da expressão que representa o circuito, como apresentado na questão anterior a

$$\mathbf{X} = \overline{\mathbf{A} \cdot \overline{\mathbf{A} \cdot \mathbf{B}}} \cdot \overline{\mathrm{B} \cdot \overline{\mathbf{A} \cdot \mathbf{B}}}$$

A Tabela Verdade será:

Conforme apresentado, o resultado da Tabela Verdade da expressão $\mathbf{X} = \overline{\mathbf{A} \cdot \overline{\mathbf{A}} \cdot \mathbf{B}} \cdot \mathbf{B} \cdot \overline{\mathbf{A} \cdot \mathbf{B}}$ é a mesma da expressão $\mathbf{X} = \mathbf{A} \oplus \mathbf{B}$.

Considerações finais

Neste estudo, vimos o desenvolvimento do sistema de análise lógica conhecido, atualmente, como álgebra de Boole. Esse sistema permite expressar a operação de um circuito na forma de uma operação algébrica em que as constantes e variáveis podem assumir apenas dois valores.

Identificamos os elementos básicos para o projeto de sistemas digitais, conhecidos como portas e funções lógicas, bem como a combinação das portas lógicas em circuitos digitais que, muitas vezes, podem produzir uma redução do número de portas lógicas utilizadas no circuito.

O estudo desta redução ou simplificação de circuitos lógicos requer o conhecimento da álgebra de Boole, na qual encontram-se os fundamentos da eletrônica digital de circuitos, que poderá diminuir o grau de dificuldade na montagem e no custo do sistema digital.

Ouça um resumo sobre os principais assuntos abordados no tema.

Para ouvir o *áudio*, acesse a versão online deste conteúdo.

Referências

MONTEIRO, Mário. **Introdução à Organização de Computadores**. 5. ed. Rio de Janeiro: LTC, 2007.

STALLINGS, William. **Arquitetura e organização de computadores**. 10. ed. São Paulo: Pearson Education do Brasil, 2017.

TANENAUM, Andrew S. **Organização Estruturada de Computadores**. 5. ed. São Paulo: Pearson Prentice Hall, 2007.

TOCCI, Ronald J. **Sistemas digitais e aplicações**. 10. ed. São Paulo: Pearson Prentice Hall, 2007.

Explore +

Para saber mais sobre os assuntos explorados neste tema, leia:

Conceitos da Lógica Digital (anexo B), MONTEIRO, Mário. **Conceitos da Lógica Digital**. In: Introdução à Organização de Computadores. 5. ed. Rio de Janeiro: LTC, 2007.

Lógica Digital (capítulo 11), STALLINGS, William. **Lógica Digital**. In: Arquitetura e organização de computadores. 10. ed. São Paulo: Pearson Education do Brasil, 2017.

Portas lógicas, MARTINS, Elaine. **Lógica Booleana? Saiba um pouco mais sobre esta lógica e como ela funciona**. Portas Lógicas. In: Tecmundo. Publicado em: 9 fev. 2009.