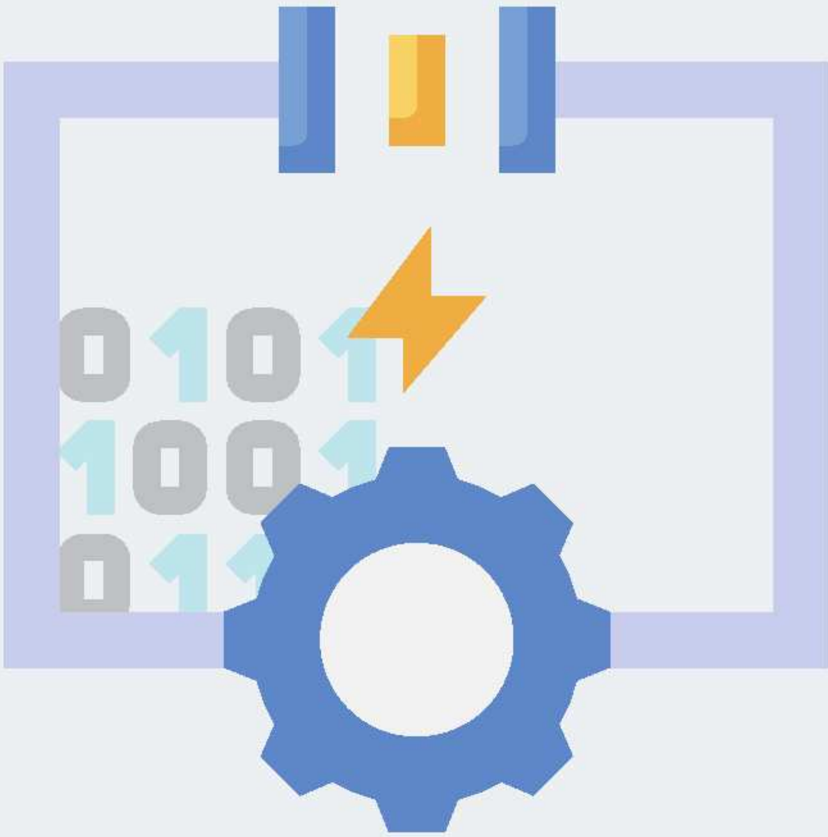
ÁLGEBRA BOOLEANA



UniCesumar

George Boole nasceu em Lincoln - Inglaterra em 2 de Novembro de 1815, filho de um sapateiro pobre. A sua formação base na escola primária da National Society foi muito rudimentar. Autodidata, fundou aos 20 anos de idade a sua própria escola e dedicou-se ao estudo da Matemática.

Em 1840 publicou o seu primeiro trabalho original e em 1844 foi condecorado com a medalha de ouro da Royal Society pelo seu trabalho sobre cálculo de operadores.

Em 1847 publica um volume sob o título The Mathematical Analysis of Logic em que introduz os conceitos de lógica simbólica demonstrando que a lógica podia ser representada por equações algébricas.

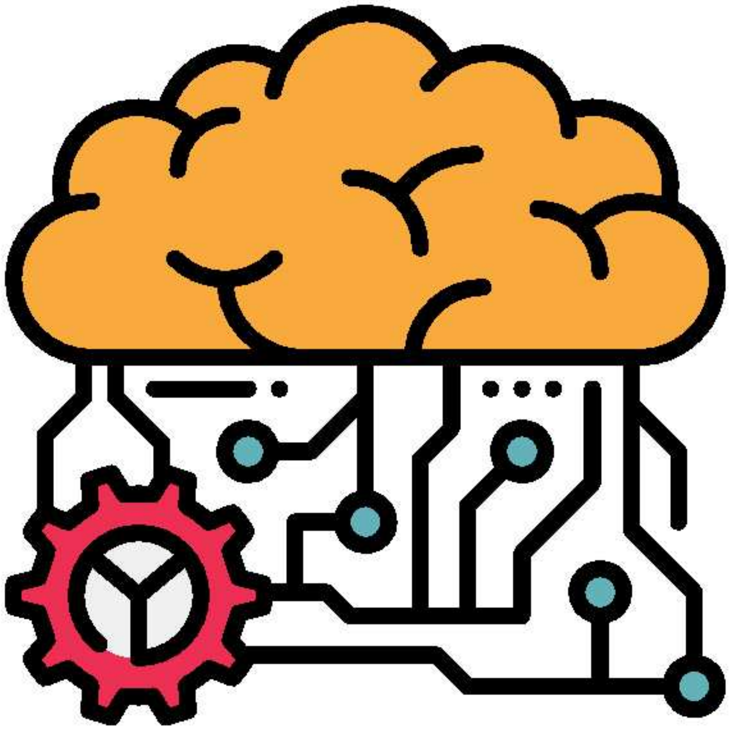
Este trabalho é fundamental para a construção e programação dos computadores eletrônicos iniciada cerca de 100 anos mais tarde. Na Álgebra de Boole existem apenas três operadores E, OU e NÃO (AND, OR, NOT). Estas três funções são as únicas operações necessárias para efetuar comparações ou as quatro operações aritméticas base.

Em 1937, cerca de 75 anos após a morte de Boole, Claude Shannon, então estudante no MIT - Boston, USA - estabeleceu a relação entre a Álgebra de Boole e os circuitos eletrônicos transferindo os dois estados lógicos (SIM e NÃO) para diferentes diferenças de potencial no circuito.

Atualmente todos os computadores usam a Álgebra de Boole materializada em microchips que contêm milhares de interruptores miniaturizados combinados em portas (gates) lógicos que produzem os resultados das operações utilizando uma linguagem binária.

COSTA, Keilla Renata. "**George Boole**"; Brasil Escola. Disponível em: https://brasilescola.uol.com.br/biografia/george-boole.htm. Acesso em 26 de janeiro de 2022.

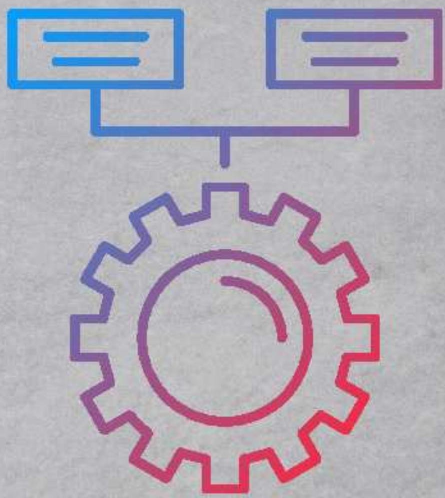
UniCesumar



"Até na filosofia propriamente dita, deve-se ter um raciocínio lógico de fatos e conceituações para a conclusão ser elucidativa e realista".

Jean Carlos Sestrem

LÓGICA DIGITAL



PARA QUE SERVE?

A LÓGICA DIGITAL É A BASE DOS SISTEMAS ELETRÔNICOS. É BASEADA EM CÓDIGO BINÁRIO, OU SEJA, UMA SÉRIE DE ZEROS E UNS.

1

A lógica digital também é a base de sistemas eletrônicos, como computadores e celulares.

2

Vários circuitos digitais podem ser produzidos em chips integrados. Isso nos ajuda a manter sistemas complexos de tamanho menor.

3

Circuitos digitais são mais flexíveis.

Podemos alterar a funcionalidade dos circuitos digitais mudando o software em vez de mudar os circuitos reais.

A lógica digital, ou booleana, é o conceito fundamental que sustenta todos os sistemas de computadores modernos.

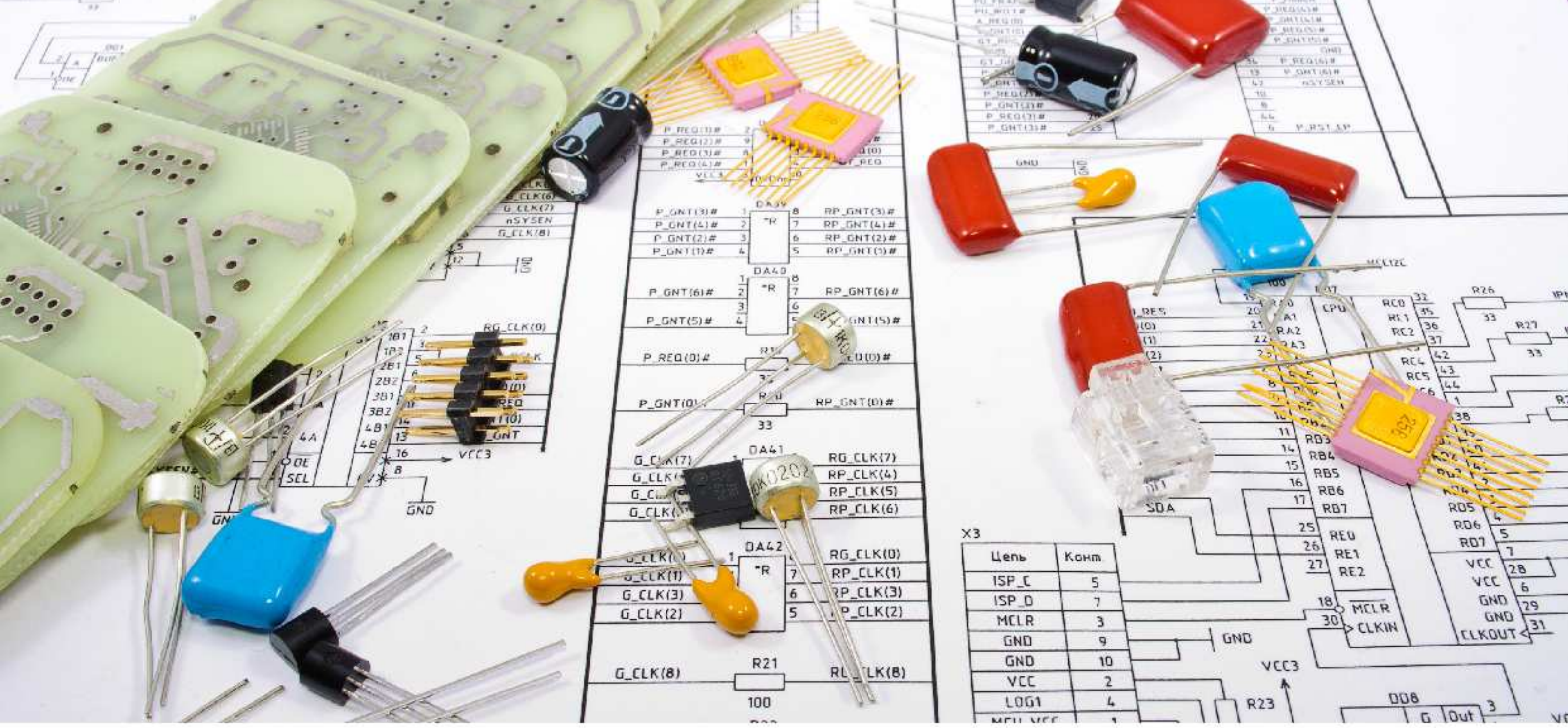
Eletricidade?

Os computadores eletrônicos foram projetados para armazenar, em capacitores, a informação, armazenando energia ou não.

Os circuitos eletrônicos digitais trabalham, basicamente, com dois níveis de tensão. Normalmente, um sinal entre 0 e 0,5 volt representa um valor 0 e um sinal entre 1 e 1,5 volt representa o valor 1 (TANENBAUM, 2013).

Na prática, o valor real da tensão pode variar, mas é convencionado usar baixa tensão representando o valor 0 e uma tensão mais alta representando o valor 1.

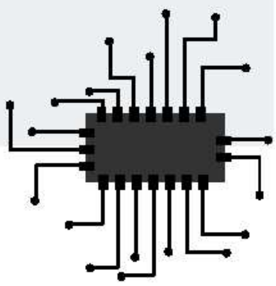
UniCesumar



Circuitos Digitais

A partir dessa noção de que um computador é uma máquina digital, podemos entender que ele é formado por diversos circuitos eletrônicos que trabalham com sinais binários.

Esses circuitos são conhecidos como circuitos digitais ou circuitos lógicos, que possuem pequenos elementos chamados portas (gates) que permitem, ou não, a passagem de um sinal binário (MONTEIRO, 2002).

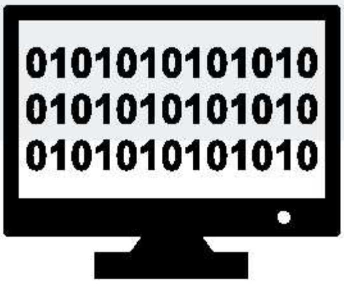


UniCesumar

Sistema Binário

De acordo com a eletrônica digital e a matemática, um número binário é definido como um número expresso no sistema binário ou sistema numeral base 2. Existem quatro tipos diferentes de conversões entre bases numéricas:

- Binário
- Decimal
- Octal
- Hexadecimal



Conversões entre bases numéricas

CONVERSÃO DECIMAL-BINÁRIO

Para essa conversão, divide-se sucessivamente o número decimal por 2 e os quocientes que vão sendo obtidos, até que o quociente numa das divisões seja 0. A sequência de todos os restos obtidos dispostos na ordem inversa representa o número binário.

Operação	Resultado	Resto
24 / 2	12	0
12 / 2	6	0
6 / 2	3	0
3 / 2	1	1
1 / 2	0	1

Número convertido: 11000

CONVERSÃ BINÁRIO-OCTAL

O sistema octal possui apenas 8 algarismos disponíveis para formar todos os números: 0, 1, 2, 3, 4, 5, 6 e 7. Para essa conversão, agrupam-se os dígitos binários de 3 em 3 do ponto decimal para a esquerda, substituindo-se cada trio de dígitos binários pelo equivalente dígito octal.

Grupo de 3 dígitos binários	Octal equivalente
000	0
011	3

Número convertido: 30

UniCesumar

Conversões entre bases numéricas

CONVERSÃO BINÁRIO-HEXADECIMAL

O sistema hexadecimal possui 16 algarismos. Como não temos números suficientes para representar, emprestamos letras. Os algarismos são: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E e F. Para essa conversão, agrupam-se os dígitos binários de 4 em 4 do ponto decimal para a esquerda, substituindo-se cada grupo de quatro dígitos binários pelo equivalente dígito hexadecimal.

Grupo de 4 dígitos binários	Hexadecimal equivalente
1000	8
0001	1

Número convertido: 18

CONVERSÃO BINÁRIO-DECIMAL

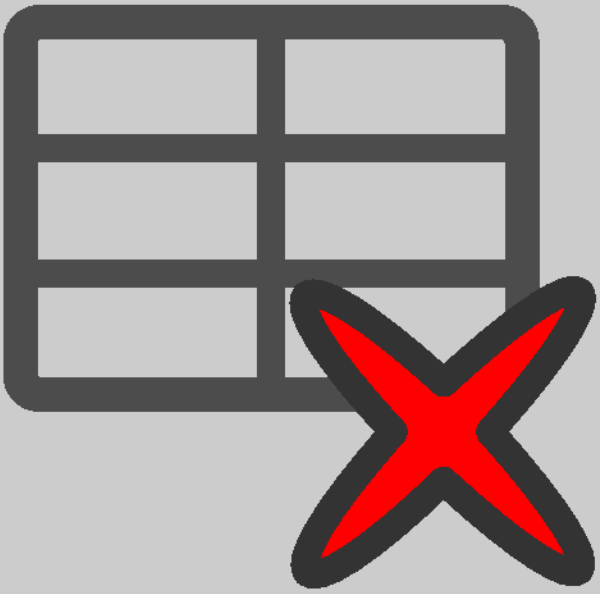
O método consiste em reescrever o número binário na vertical de tal forma que a parte esquerda do número fique acima da parte direita. Posteriormente o seguinte processo deve ser repetido para cada um dos dígitos começando pelo que se encontra mais acima: soma-se esse dígito aos produto igual a 2 vezes o resultado da operação anterior.

Dígito	Operação	Resultado
1	1 + 2 x 0	1
1	1 + 2 x 1	3
0	0 + 2 x 3	6
0	0 + 2 x 6	12
0	0 + 2 x 12	24

Número convertido: 24

UniCesumar

TABELA VERDADE



VERDADEIRO (E) (OU) FALSO!

AND OR NOT XOR

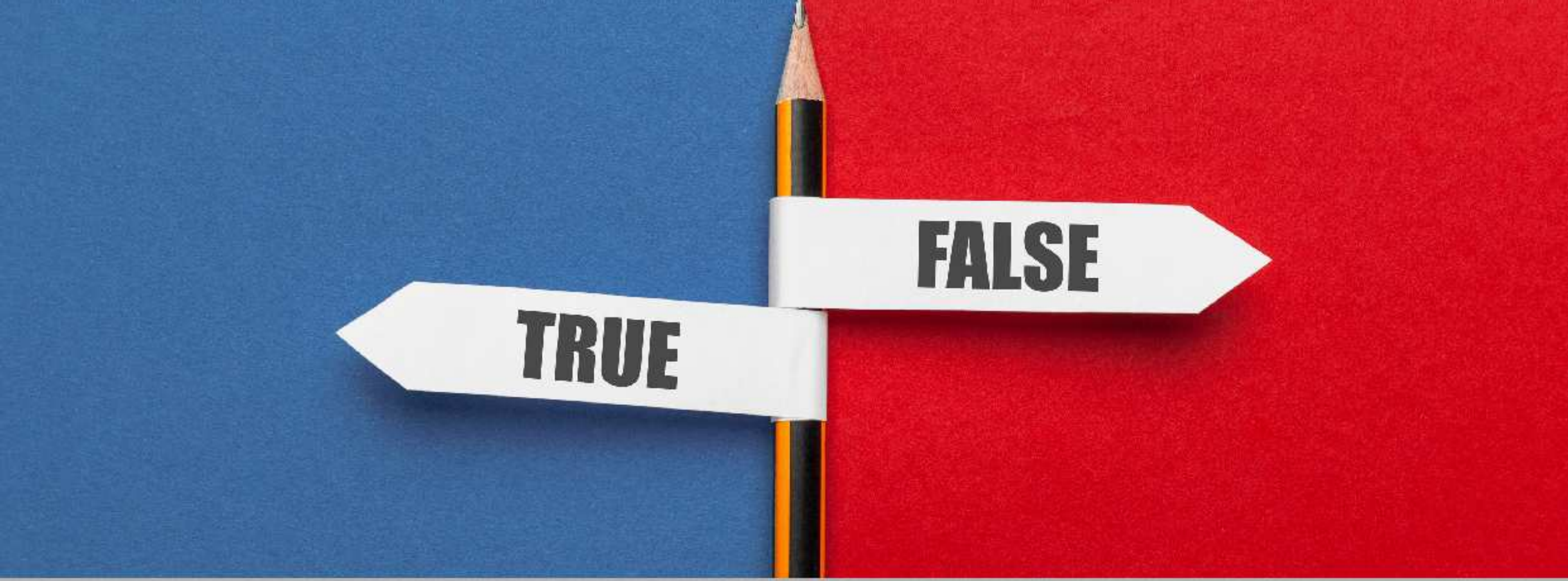


TABELA VERDADE

Tabela verdade é uma ferramenta de natureza matemática muito utilizada no campo do raciocínio lógico. Seu objetivo é verificar a validade lógica de uma proposição composta (argumento formado por duas ou mais proposições simples).

Disponível em: <Tabela Verdade (O que é, Conceito, Regras e Como Funciona) - Significados>

1.

Como montar a tabela verdade?

2.

Quantas linhas e colunas devo colocar na tabela?

UniCesumar

COMO MONTAR A TABELA VERDADE

- A tabela-verdade é uma forma prática de chegarmos aos valores de saída de um circuito lógico. Ela pode ser montada colocando, à esquerda, todas as possibilidades de combinação de valores de entrada para as variáveis envolvidas (no caso anterior, A e B), enquanto à direita colocamos o valor de saída para cada combinação de entradas, ou seja, para cada linha da tabela.

Para duas variáveis, nossa tabela terá 4 linhas de combinações de entradas, para três variáveis, teremos 8 linhas.

Resumindo, para n variáveis de entrada, teremos 2n linhas de combinações dos valores de entrada.

Fórmula:

$$2^n$$

Exemplo:

S = (A + ~B) . C

No exemplo temos 3 variáveis que são: A, B e C. Utilizando a fórmula teremos 2^3 ou seja, 2 * 2 * 2 = 8 (linhas). A quantidade de colunas é o número de variáveis mais as operações que devem ser calculadas.

EXEMPLO DE TABELA VERDADE

Expressão:
S = (A+ ~B) . C

Para montar a tabela verdade:

As primeiras colunas são as entradas, que faremos todas as combinações possíveis de entradas para A, B e C. Depois, podemos fazer colunas com as equações temporárias, ou subequações, para facilitar. Por fim, a coluna com a saída final.

A	B	C	~B	A + ~B	(A + ~B) . C
0	0	0	1	1	0
0	0	1	1	1	1
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	0	1	0
1	1	1	0	1	1

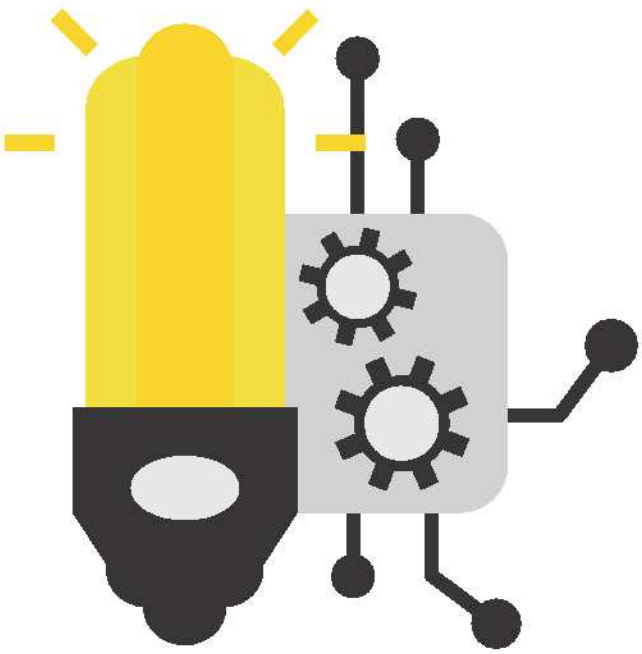
Na 1ª coluna preenchemos a metade com 0 e a outra metade com 1. Como são 8 linhas teremos 4 zeros e 4 uns.

Na 2ª coluna preenchemos com 0 e 1 alternados em grupos de 2, iniciando pelo 0.

Na 3ª coluna preenchemos com 0 e 1 alternando entre sim, iniciando pelo 0.

As outras colunas devem ser preenchidas com as equações e última coluna é o resultado do cálculo.

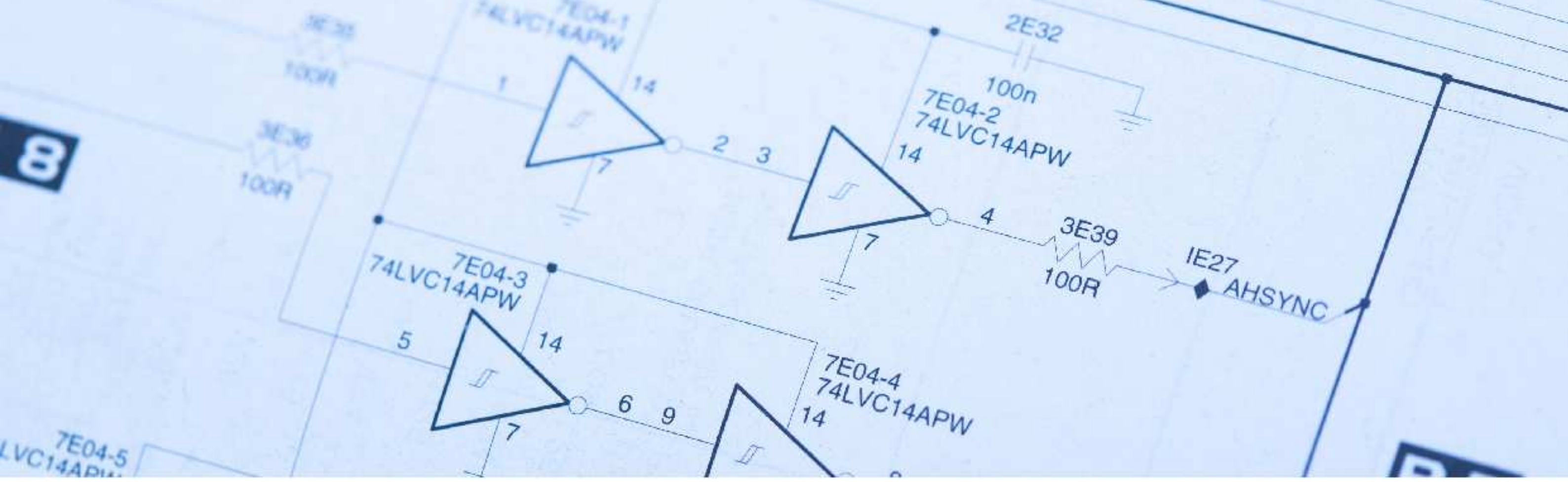
CIRCUITOS DIGITAIS



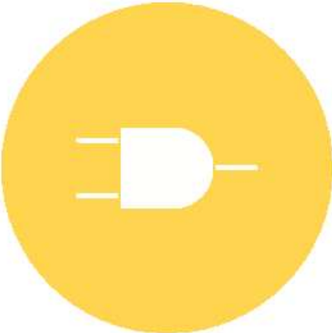
"Os circuitos são uma combinação de diversas portas, essas portas podem ser representadas de forma gráfica, por meio de um esquema de circuitos, ou em forma de uma expressão, a qual a saída depende de um conjunto de operações que efetue sobre as entradas.".

André Abdala Noel

UniCesumar

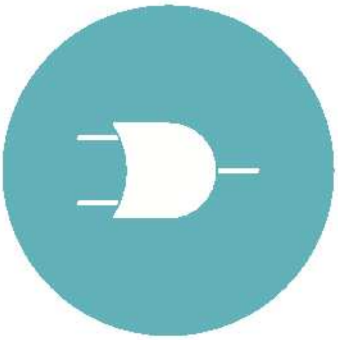


PORTAS LÓGICAS



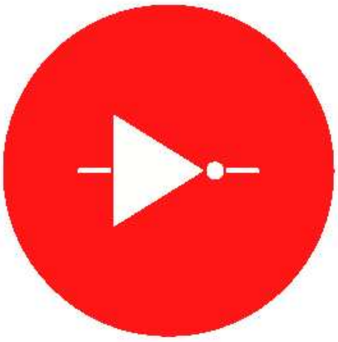
PORTA AND

A operação E (AND) é representada pelo símbolo ".". Ela analisa dois (ou mais) valores de entrada e retorna 1 apenas se todos os valores de entrada forem 1.



PORTA OR

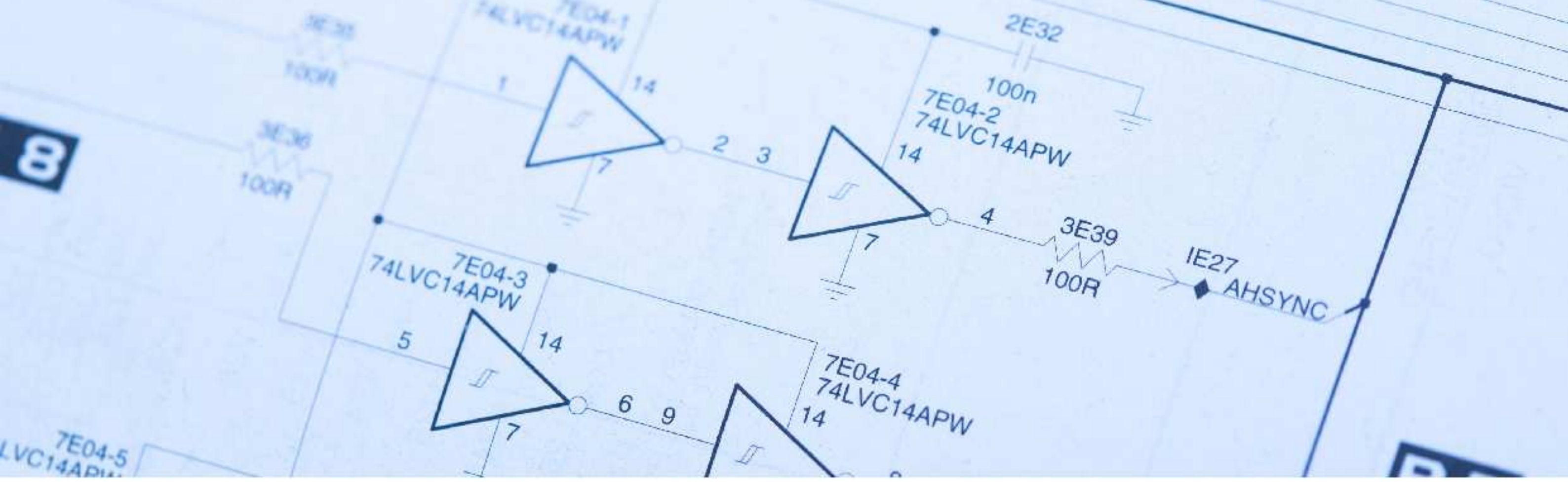
A operação OU (OR) é representada pelo símbolo "+" (sim, o mesmo símbolo de adição). Ela analisa dois (ou mais) valores de entrada e retorna 1, se qualquer um dos valores de entrada for 1, retorna zero apenas quando todos os valores de entrada forem zero.



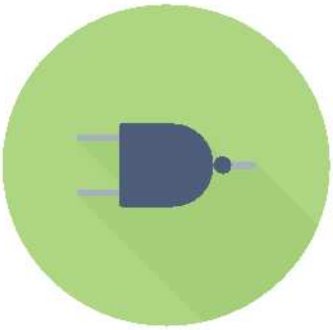
PORTA NOT

A operação NÃO (NOT) é representada pelo símbolo "–" (um traço acima da letra), ou ainda por um til (~) ou por um "¬". É um operador unário, ou seja, atua com apenas um operando. Basicamente, ela inverte o valor de entrada: quando entra 0 retorna 1, quando entra 1 retorna 0.

UniCesumar

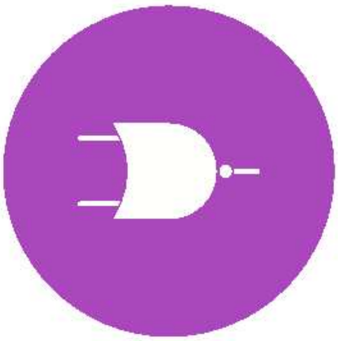


PORTAS LÓGICAS



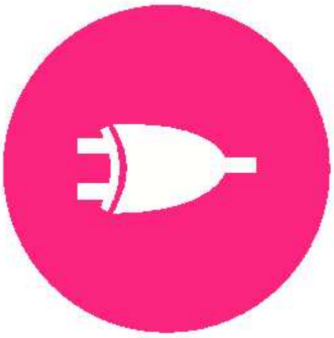
PORTA NAND

A operação NÃO-E (NAND) é representada com um traço sobre toda uma operação E, ou com um til (~) antes da operação E. Ela é a mesma operação AND, mas com o valor invertido ao final da operação.



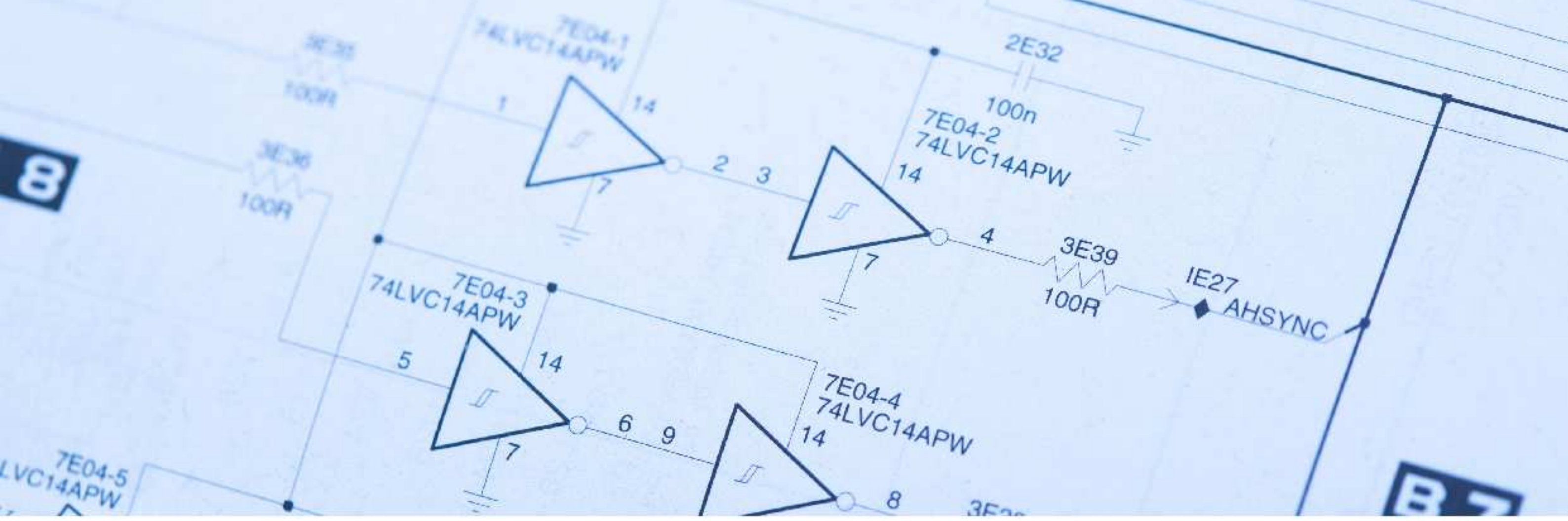
PORTA NOR

A operação NÃO-OU (NOR) é representada com um traço sobre toda uma operação OU, ou com um til (~) antes da operação OU. Ela é a mesma operação OR, mas com o valor invertido ao final da operação

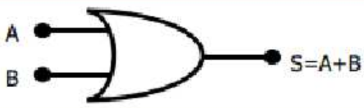
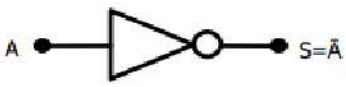
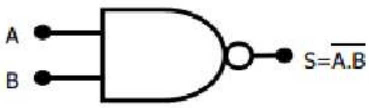


PORTA XOR

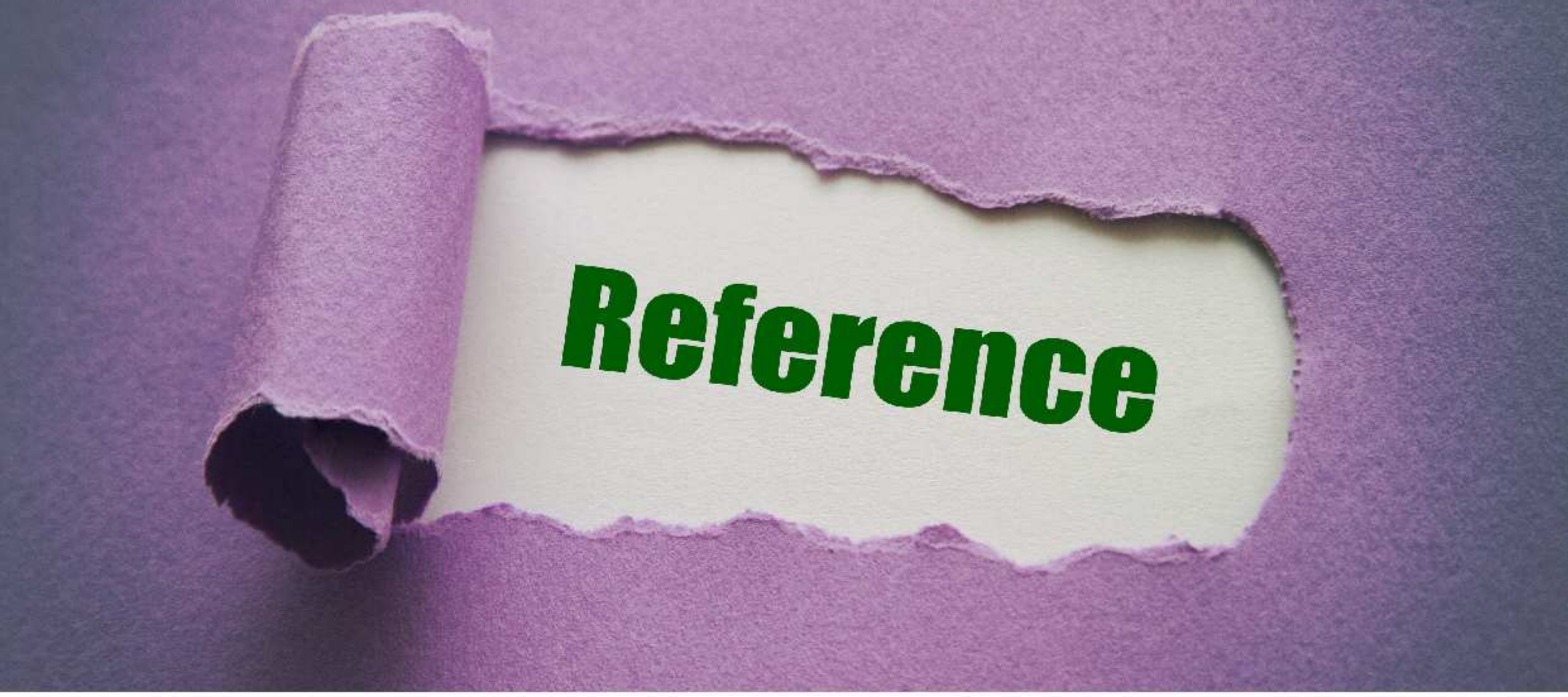
A operação OU-EXCLUSIVO (XOR - eXclusive OR) é representada pelo símbolo "⊕" (um sinal de adição dentro de um círculo). Ela analisa dois valores de entrada e retorna 1 se apenas um dos valores de entrada for 1. Se ambos forem 0 ou se ambos forem 1, a saída será 0.



Resumo dos Blocos Lógicos Básicos

Nome	Símbolo Gráfico	Função Algébrica	Tabela Verdade
E (AND)		S=A.B S=AB	A B | S=A.B 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OU (OR)		S=A+B	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' $S= \neg A$	A | $S=\bar{A}$ 0 | 1 1 | 0
NE (NAND)		$S=\overline{A.B}$ S=(A.B)' $S= \neg(A.B)$	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)' $S= \neg(A+B)$	A B | $S=\overline{A+B}$ 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0
XOR		$S=A\oplus B$	A B | $S=A\oplus B$ 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0

Disponível em: Aula 1. Sistemas Analógicos vs Sistemas Digitais - PDF Download grátis (docplayer.com.br)



CALCULADORA ONLINE. **Calculadora de Conversão Passo a Passo de Bases Numéricas.** Disponível em: https://www.calculadoraonline.com.br/conversao-bases-passo-passo Acesso em 25/01/2022.

COSTA, Keilla Renata. "**George Boole"**; Brasil Escola. Disponível em: https://brasilescola.uol.com.br/biografia/george-boole.htm Acesso em 26 de janeiro de 2022.

DOCPLAYER. Aula 1. **Sistemas analógicos vs Sistemas Digitais.** Disponivel em: https://docplayer.com.br/113902571-Aula-1-sistemas-analogicos-vs-sistemas-digitais.html acesso em 26/01/2022.

MONTEIRO, M. A. **Introdução à organização de computadores.** 4. ed. Rio de Janeiro: LTC, 2001. 498 p.

NOEL, André Abdala. **Fundamentos e Arquitetura de Computadores.** Maringá-Pr.: UniCesumar, 2019. Reimpresso 2020.

SIGNIFICADOS. **Tabela Verdade.** Disponível em: https://www.significados.com.br/tabela-verdade/ acesso em 26/01/2022.

TANENBAUM, A. S. **Organização estruturada de computadores**. 6. ed. São Paulo: Pearson Prentice Hall, 2013.

UniCesumar

Unicesumar