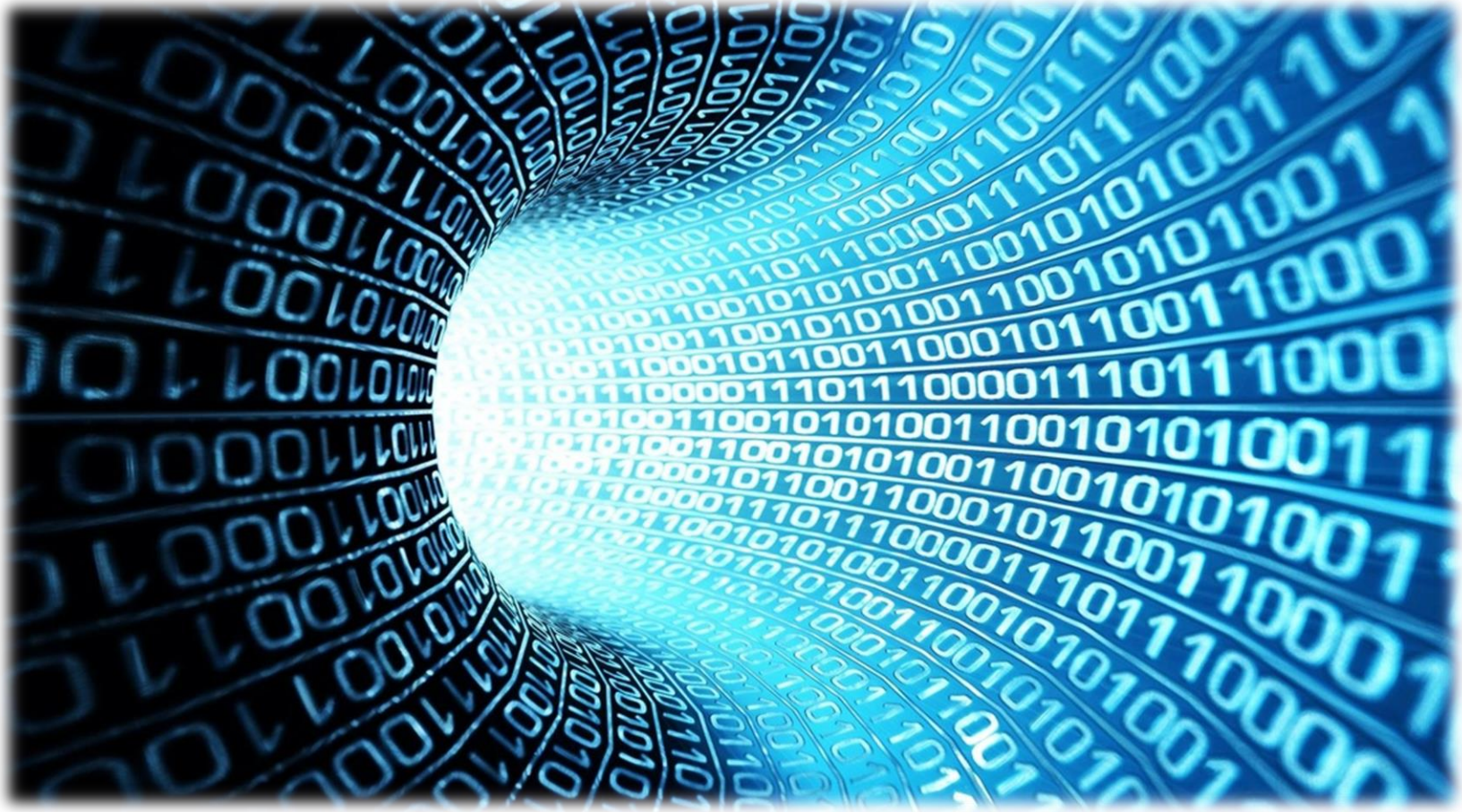


Organização e Arquitetura de Computadores



Fonte da imagem: <https://cutt.ly/D4jVvQY>

PORTAS LÓGICAS

CONCEITOS

Conceito da Lógica Digital

A leitura da lógica digital acontece em baixo nível (*hardware*) ou “**nível lógico digital**”, estando entre a Ciência da Computação e a Engenharia Elétrica como base para os circuitos digitais.

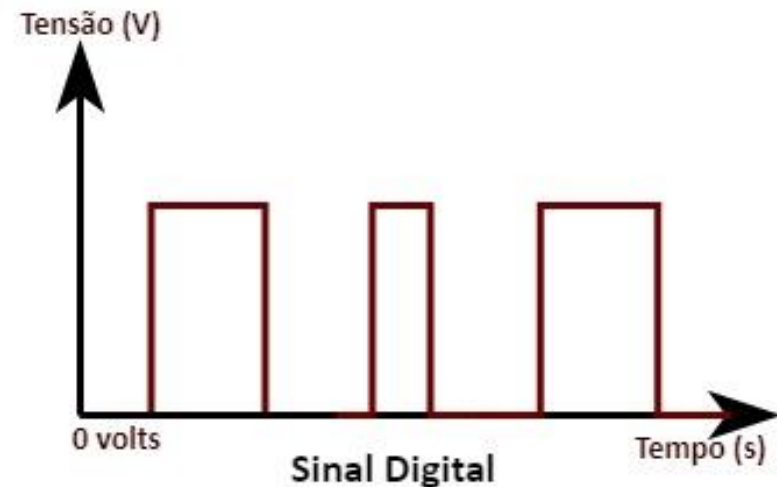
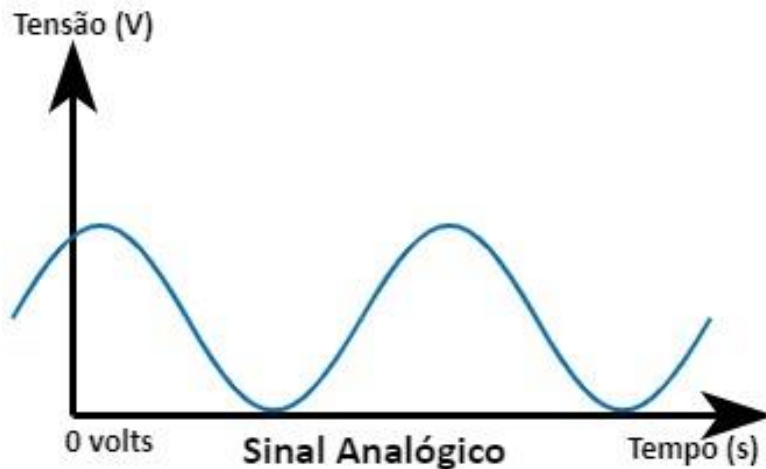
O QUE VEREMOS!

- Compreender os elementos básicos;
- Álgebra Booleana (teoria):
 - Portas lógicas (NOT, AND, OR, NAND, NOR, XOR, XNOR);
 - Expressões lógicas
 - Tabelas verdade;
 - Teoria dos Circuitos Lógicos e as Leis da álgebra booleana;
 - Circuitos Integrados e Lógicos Digitais Básicos;
 - Mapa de Karnaugh.

- ☺ Foi estudado que no sistema computacional o **armazenamento** e a **manipulação** das informações de um **sistema digital** possui dois valores, **0 ou 1**.
- ☺ Nas pesquisas do “**fórum**” os grupos analisaram a leitura quântica, **0 e 1**, como leitura conhecida como “**qubits ou bits quânticos**”, como unidade básica de informação quântica, mas que **mescla com** a leitura binária **0 ou 1** dos sistemas computacionais convencionais.
- ☺ Essa combinação de valores aritméticos permite a realização de cálculos complexos, criptografia avançada, e com uma velocidade de leitura e interpretação muito acima dos sistemas casuais.

➡ Assim, o estado de espaço quântico seria “qualquer estado possível, onde um sistema mecânico quântico, como um sistema isolado, fornece uma distribuição probabilística aos possíveis resultados, quânticos e casuais, sobre o processamento dos sistemas computacionais”.

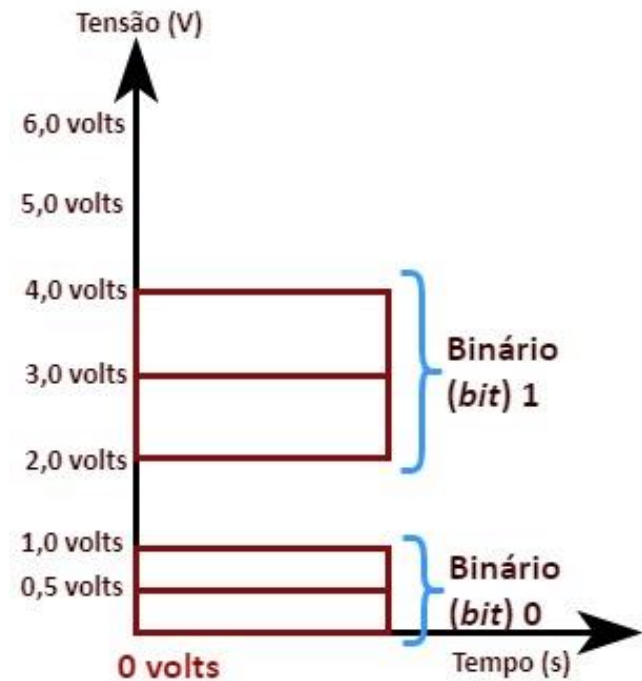
- Quando se analisa a forma de trabalho dos circuitos digitais ou lógicos, vê-se a utilização de **sinais elétricos** em dois níveis de tensão, chamados de **sinais digitais** ou **sinais binários**.
- **Ao contrário**, os sistemas analógicos são formados por **sinais em onda** com valores **infinitos**, oscilando conforme a tensão entrada.



Fonte: Adaptado pelo autor (2023)

→ Então, esses **sinais elétricos** são informações binárias que compõe um sistema digital **gerados internamente ou recebidos de elementos externos**, com dois níveis de intensidade, +3 volts e +0,5 volts, onde:

- Um sinal de **+3 volts** representa o valor de 1 *bit*;
- Um sinal de **+0,5 volts** representa o valor de 0 *bit*;
- **Ambos** são medidos com base em uma faixa de tolerância de **0,0 à 5,0 volts** devido as oscilações da tensão de entrada e saída.



Fonte: Adaptado pelo autor (2023)

Foi visto e estudado que o computador possui diversos componentes: resistores, capacitores e **transistores**, considerados:

- Componentes que **armazenam sinais binários** para algum modelo de operação, “**circuitos digitais**”.
- Desenvolvidos para leituras binárias através de suas **portas lógicas** permitindo ou não a passagem dos sinais digitais, **onde**:
 - ✎ A **porta lógica** é um elemento de *hardware* que recebe um ou mais sinais de entrada, produzindo um sinal de saída;
 - ✎ Seu valor do sinal depende do tipo da **regra lógica** estabelecida em na construção para o circuito.



Transistores

Por definição, um computador digital seria:

“Uma infinidade de **“circuitos lógicos”** ou **“portas lógicas”**, distribuídos e organizados para armazenar valores binários, permitir e controlar os fluxos de sinais elétricos entre os componentes e realizar suas operações matemáticas”.

Os circuitos digitais dizem ao computador como se comportar perante um processo.

Para a análise desses comportamentos são realizados testes através de **conceitos** e **regras** estabelecidas pela **Álgebra de Chaveamentos (Switching Álgebra)**, um dos conceitos da Álgebra Booleana.


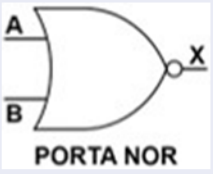
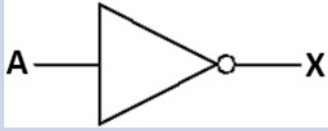
Portas Lógicas

- Os sinais digitais, binários, trabalham com uma corrente elétrica entre +0,5 e +3,0 volts, **não sendo permitidas** tensões fora dessas duas faixas.
- **Já**, a função das portas lógicas é de calcular as várias aritméticas dos sinais binários e, considerado a base do **hardware** nos computadores digitais.

Portas Lógicas





- Em uma **operação lógica** temos apenas dois valores, **0 ou 1**, onde o:
 - ▶ **Bit 0** será considerado como **FALSO** ou **F**.
 - ▶ **Bit 1** será considerado como **VERDADEIRO** ou **V**.
- **Nesse sentido**, os valores dos resultados nas operações lógicas **costumam serem definidos previamente**, mas que dependem das possíveis combinações dos valores de entrada, representados por uma **“Tabela Verdade”**, para analisar e representar essas diversas combinações **lógicas** através de **regras** para definirem seus valores.

Portas Lógicas – Padrões mais usados

Simbologia V ou F	Representação da Porta	Símbolo Matemático	Tabela Verdade	Simbologia da Porta															
Portas = 1 bit são verdadeiras ou “V”. Portas = 0 bit são falsas ou “F”.	AND	$X = A \cdot B$	<table><tr><td>A</td><td>B</td><td>X</td></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	X	0	0	0	0	1	0	1	0	0	1	1	1	 PORTA AND
	A	B	X																
	0	0	0																
0	1	0																	
1	0	0																	
1	1	1																	
OR	$X = A + B$	<table><tr><td>A</td><td>B</td><td>X</td></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	X	0	0	1	0	1	0	1	0	0	1	1	0	 PORTA NOR	
A	B	X																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	0																	
NOT (Porta Inversora)	$X = \overline{A}$	<table><tr><td>A</td><td>A</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	A	0	1	1	0											
A	A																		
0	1																		
1	0																		

Fonte da imagem: Autor.

Portas Lógicas – Derivadas dos padrões mais usados

Simbologia V ou F	Representação da Porta	Símbolo Matemático	Tabela Verdade	Simbologia da Porta																				
Portas = 1 bit são verdadeiras ou “V”.	NAND (Negação de AND)	$X = \overline{A \cdot B}$	<table><tr><td>A</td><td>B</td><td>C</td><td>X</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	A	B	C	X	0	0	0	1	0	1	0	1	1	0	0	1	1	1	1	0	 PORTA NAND
	A	B	C	X																				
	0	0	0	1																				
	0	1	0	1																				
1	0	0	1																					
1	1	1	0																					
NOR (Negação de OR)	$X = \overline{A + B}$	<table><tr><td>A</td><td>B</td><td>C</td><td>X</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	A	B	C	X	0	0	0	1	0	1	1	0	1	0	1	0	1	1	1	0	 PORTA NOR	
A	B	C	X																					
0	0	0	1																					
0	1	1	0																					
1	0	1	0																					
1	1	1	0																					
Portas = 0 bit são falsas ou “F”.	XOR (OR Exclusivo)	$X = A \oplus B$	<table><tr><td>A</td><td>B</td><td>X</td></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	X	0	0	0	0	1	1	1	0	1	1	1	0	 PORTA XOR					
	A	B	X																					
0	0	0																						
0	1	1																						
1	0	1																						
1	1	0																						
	XNOR (NOR Exclusivo)	$X = \overline{A \oplus B}$	<table><tr><td>A</td><td>B</td><td>X</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	1	0	1	0	1	0	0	1	1	1	 PORTA XNOR					
A	B	X																						
0	0	1																						
0	1	0																						
1	0	0																						
1	1	1																						

Fonte da imagem: Autor.

Principais Operadores Lógicos - Tabela Verdade AND

Tabela Verdade AND $X = A \cdot B$ ou $X = AB$				
A	B		$x = A \cdot B$	
0	0	=	0	F
0	1	=	0	F
1	0	=	0	F
1	1	=	1	V

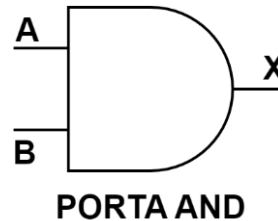


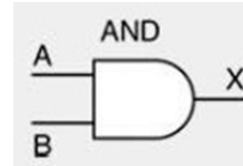
Tabela Verdade AND					
A	B	C		$x = A \cdot B \cdot C$	
0	0	0	=	0	F
0	0	1	=	0	F
0	1	0	=	0	F
0	1	1	=	0	F
1	0	0	=	0	F
1	0	1	=	0	F
1	1	0	=	0	F
1	1	1	=	1	V

Operador	Tipo	Significado de “AND”
AND	Composto	<ul style="list-style-type: none"> » Utiliza o ponto (.) como operador e representação do produto lógico. » A saída é igual a 1 se todas as entradas for 1. » A saída é igual a 0 se ao menos uma entrada for 0.

Principais Operadores Lógicos - Tabela Verdade

→ **Então**, a operação lógica **“AND”** começa com a combinação de dois valores de **entrada** que resultam em uma **saída verdadeira ou bit 1**.

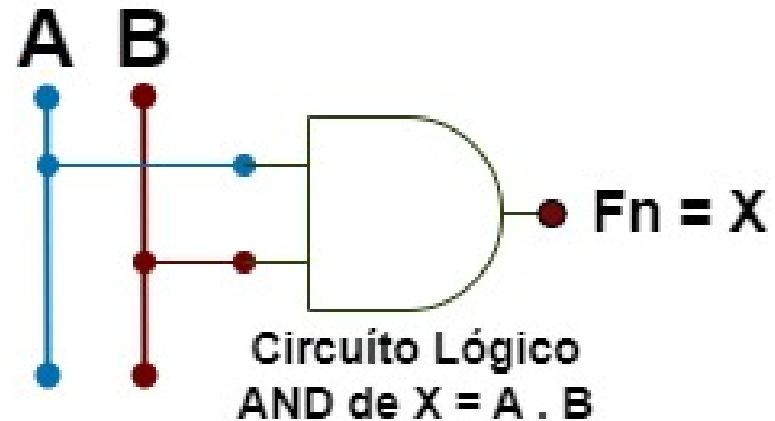
Tabela Verdade AND				
A	B	=	$x = A \cdot B$	
0	0	=	0	F
0	1	=	0	F
1	0	=	0	F
1	1	=	1	V



$$X = A \cdot B \text{ ou } X = AB$$

Poderão ser realizadas em duas situações:

1. Satisfazer **determinado requisito** de **hardware**;
2. Satisfazer a **especificação** do programador.



Lembre-se!

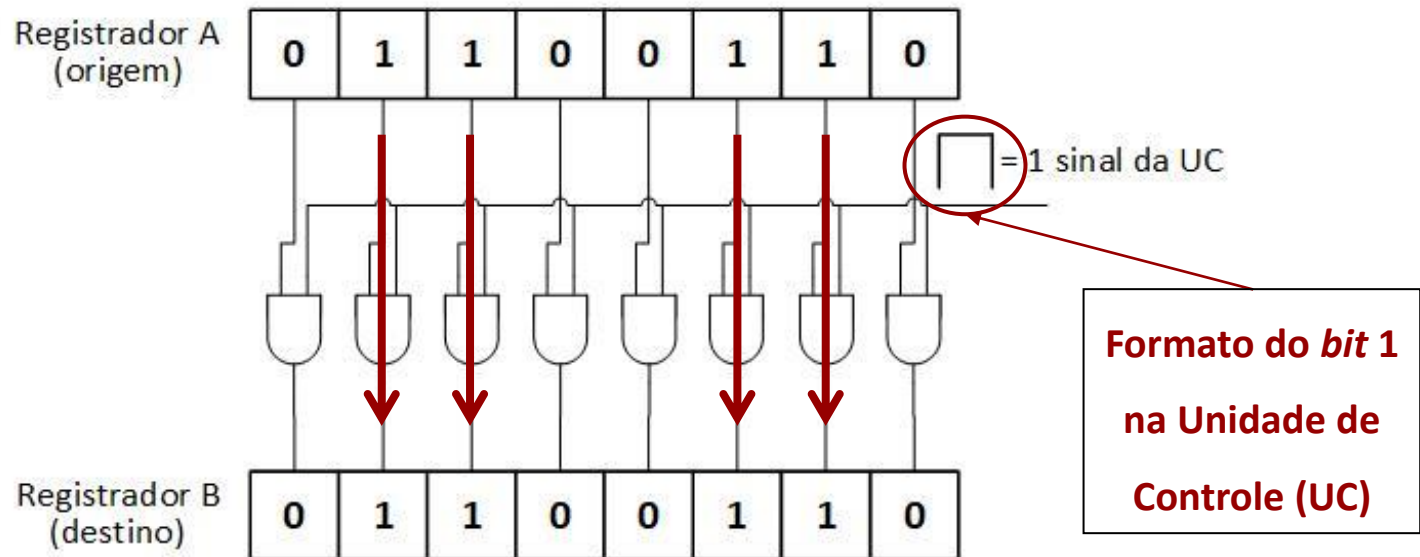
Para tabelas verdades “**AND**” com mais de duas entradas, **por exemplo: A, B, C...**, a opção de soma dos valores não irá funcionar devido a regra da porta lógica, poderá apenas ser usado em portas “AND” de duas entradas, **A e B**.

Mais adiante analisaremos outra forma de elaborar a Tabela Verdade no formato de uma soma binária.

Tabela Verdade AND					
A	B	C		$x = A . B . C$	
0	0	0	=	0	F
0	0	1	=	0	F
0	1	0	=	0	F
0	1	1	=	0	F
1	0	0	=	0	F
1	0	1	=	0	F
1	1	0	=	0	F
1	1	1	=	1	V

Principais Operadores Lógicos - Tabela Verdade – Exemplo de uso “AND”

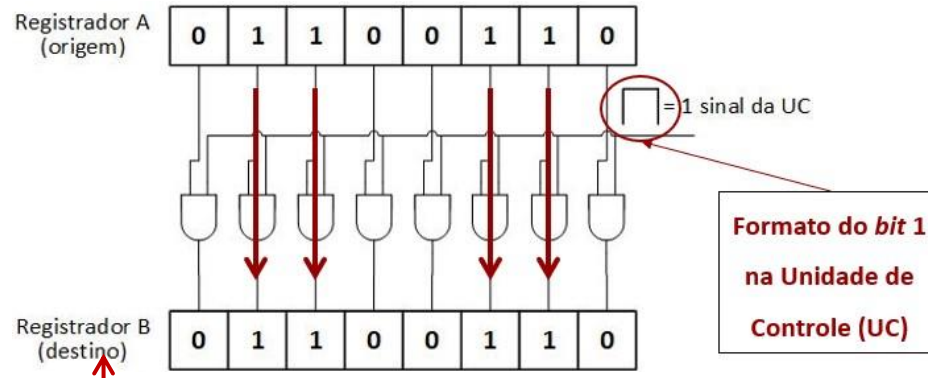
A porta AND é definida como o elemento, operação lógica, que produz um resultado verdade de $V = 1$ na saída, “*se e somente se todas as entradas forem verdade*”.



Exemplo de utilização de porta AND na movimentação de dados de um registrador para outro.

Fonte: Adaptado pelo autor.

Explicação sobre o exemplo da Porta Lógica “AND”!



- A aplicação do circuito lógico **AND** controla as transferências de dados **para** cada *bit* do registrador criando um sinal na Unidade de Controle (UC) como **entrada** e correspondente ao *bit* do **registrador A (origem) ou RA**;
- Quando o sinal da unidade de controle for igual a **“1”** o pulso elétrico de intensidade **igual** ao *bit* 1 será **“verdade”** ao contrário será **“falso”**;
- A **combinação desses sinais de entrada produz na saída RA um valor igual** no do *bit* do **registrador B (destino) ou RB** durante o período em que a linha da UC (Unidade de Controle) estiver com o *bit* 1 ativo.

Para conhecimento!

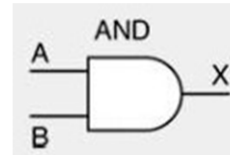
A maioria dos processadores possuem uma instrução de máquina “**AND**” em seu conjunto de instruções que trabalham juntamente com outras linguagens de programação de **alto nível** para atender as condições dos programas.

Principais Operadores Lógicos - Tabela Verdade - Algoritmo

Exemplo de uma instrução de programa simples:

- Leia X, Y e Z
- $T = X + Y$
- $R = Z + X$
- **Se** ($T > 6$ **and** $R < 10$)
 - **Então** imprimir T
 - **Senão** Imprimir R
- Fim

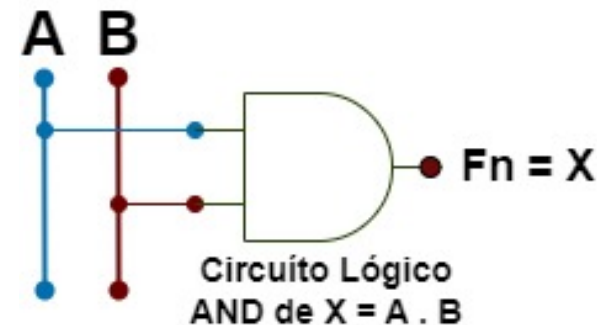
Tabela Verdade AND				
A	B		$x = A \cdot B$	
0	0	=	0	F
0	1	=	0	F
1	0	=	0	F
1	1	=	1	V



$$X = A \cdot B \text{ ou } X = AB$$

→ **Se** a condição da porta lógica **AND** for “**VERDADE**”, **então** o valor da variável **T** poderá ser impresso, **mas** somente se ambas as condições forem verdadeiras: “**T > 6 e R < 10**”.

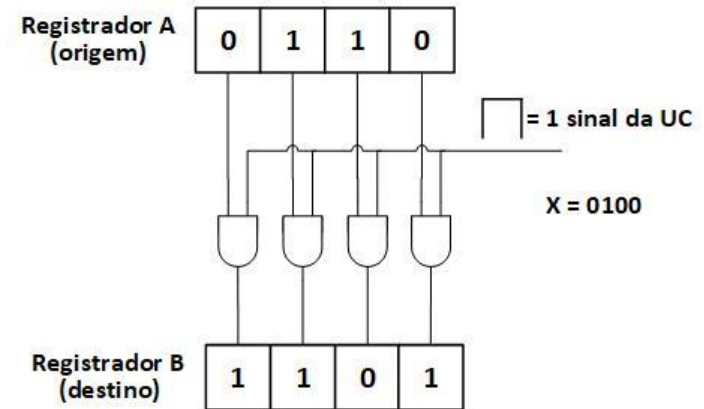
Nesse sentido, as operações lógicas **AND** são usadas com valores binários constituídos de vários *bits* e usados pela álgebra da ULA (Unidade Lógica Aritmética).



Principais Operadores Lógicos - Tabela Verdade - Exemplo Prático 1 – Soma AND

→ Seja $A = 0110$ e $B = 1101$.

→ Vamos calcular $X = A \cdot B$ (**A and B**).

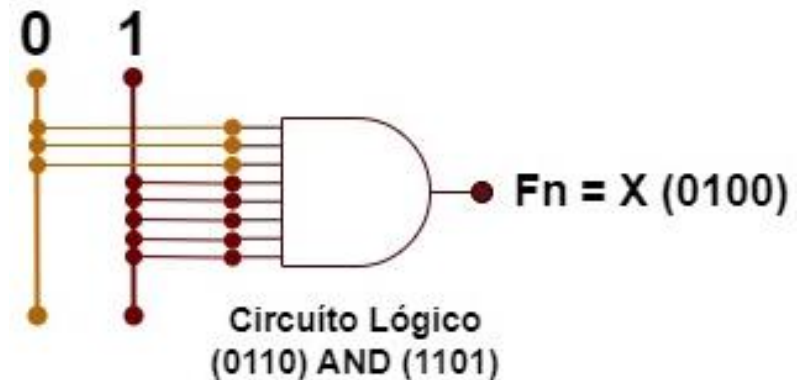


Porta AND

Fonte: Adaptado pelo autor.

Tabela Verdade AND (solução)

Tabela Verdade AND (solução)									
						A	B	$X = A . B$	
0 1 1 0 ← A						0	1	0	F
and	1	1	0	1	← B	1	1	1	V
0 1 0 0 ← X						1	0	0	F
Resultado: X = 0100						0	1	0	F



Principais Operadores Lógicos - Tabela Verdade - Exemplo Prático 2 – Soma AND

Seja $A = 0101$, $B = 0011$ e $C = 1111$. Calcular $X = A . B . C$ (A and B and C).

- O resultado será obtido através da realização das operações em **duas** etapas.
- Na primeira parte, calcula-se $A . B$ ($X_{\text{parcial}} = A$ and B) e em seguida o resultado parcial obtido que deverá ser combinado com C **em uma outra operação lógica AND** (X_{parcial} and C).

Tabela Verdade AND (X_{parcial})				
	A	B	$X_p = A . B$	
0 1 0 1 ← A	0	0	0	F
and 0 0 1 1 ← B	1	0	0	F
0 0 0 1 ← X_{parcial}	0	1	0	F
Resultado: $X_{\text{parcial}} = 0001$	1	1	1	V

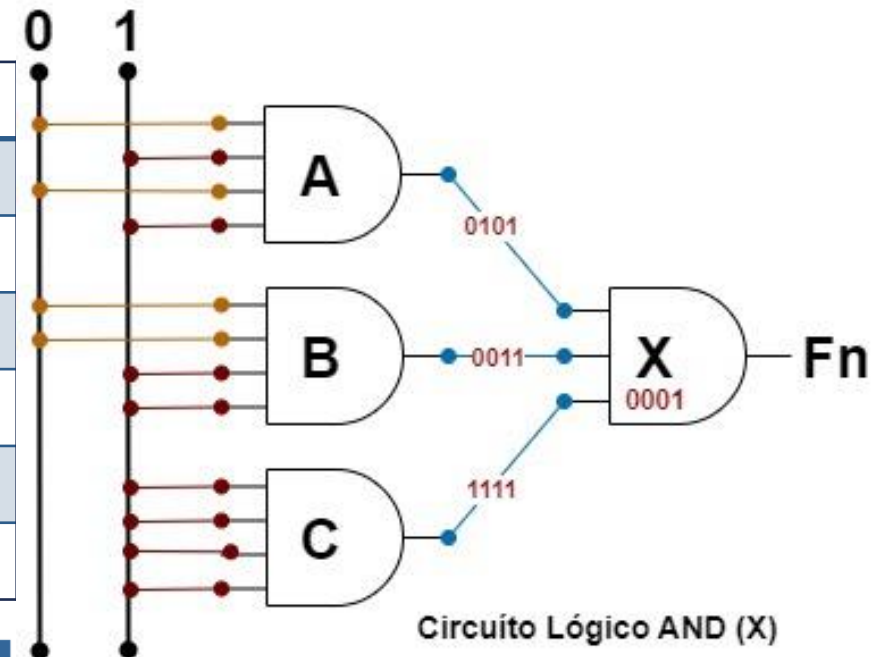
Tabela Verdade AND (X)				
	X_p	C	$X_p = X . C$	
0 0 0 1 ← X_{parcial}	0	1	0	F
and 1 1 1 1 ← C	0	1	0	F
0 0 0 1 ← X	0	1	0	F
Resultado: $X = 0001$	1	1	1	V

Principais Operadores Lógicos - Tabela Verdade - Exemplo Prático 3 – Soma AND

Seja $A = 0101$, $B = 0011$ e $C = 1111$. Calcular $X = A . B . C$ (A and B and C).

→ O resultado será obtido através da realização direta das operações em **uma** etapa.

Tabela Verdade AND (X)					
		A	B	C	$X = A . B . C$
0 1 0 1 ← A		0	0	1	0 F
and 0 0 1 1 ← B		1	0	1	0 F
1 1 1 1 ← C		0	1	1	0 F
0 0 0 1 ← X		1	1	1	1 V
Resultado: X = 0001					



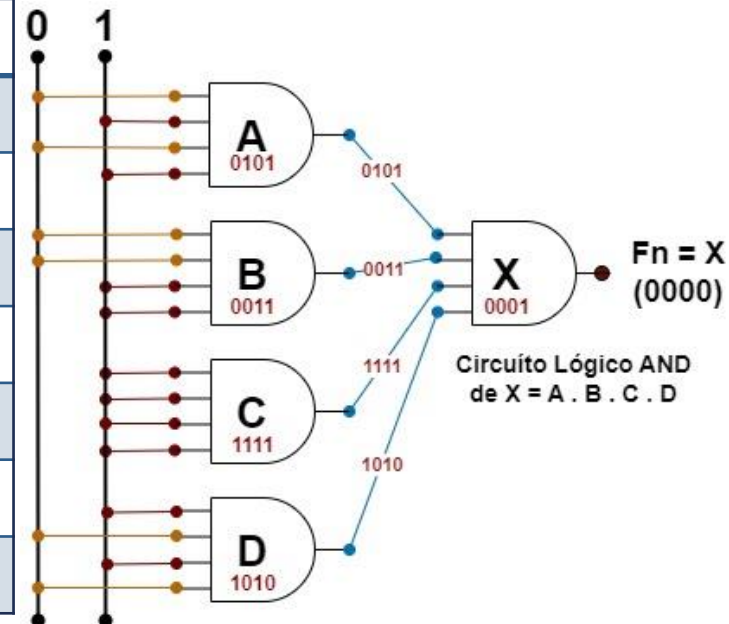
Operador	Tipo	Significado de "AND"
AND	Composto	<ul style="list-style-type: none"> » Utiliza o ponto (.) como operador e representação do produto lógico. » A saída é igual a 1 se todas as entradas for 1. » A saída é igual a 0 se ao menos uma entrada for 0.

Principais Operadores Lógicos - Tabela Verdade - Exemplo Prático 4 – Soma AND

Seja $A = 0101$, $B = 0011$, $C = 1111$ e $D = 1010$. Calcular $X = A . B . C . D$ (A and B and C and D).

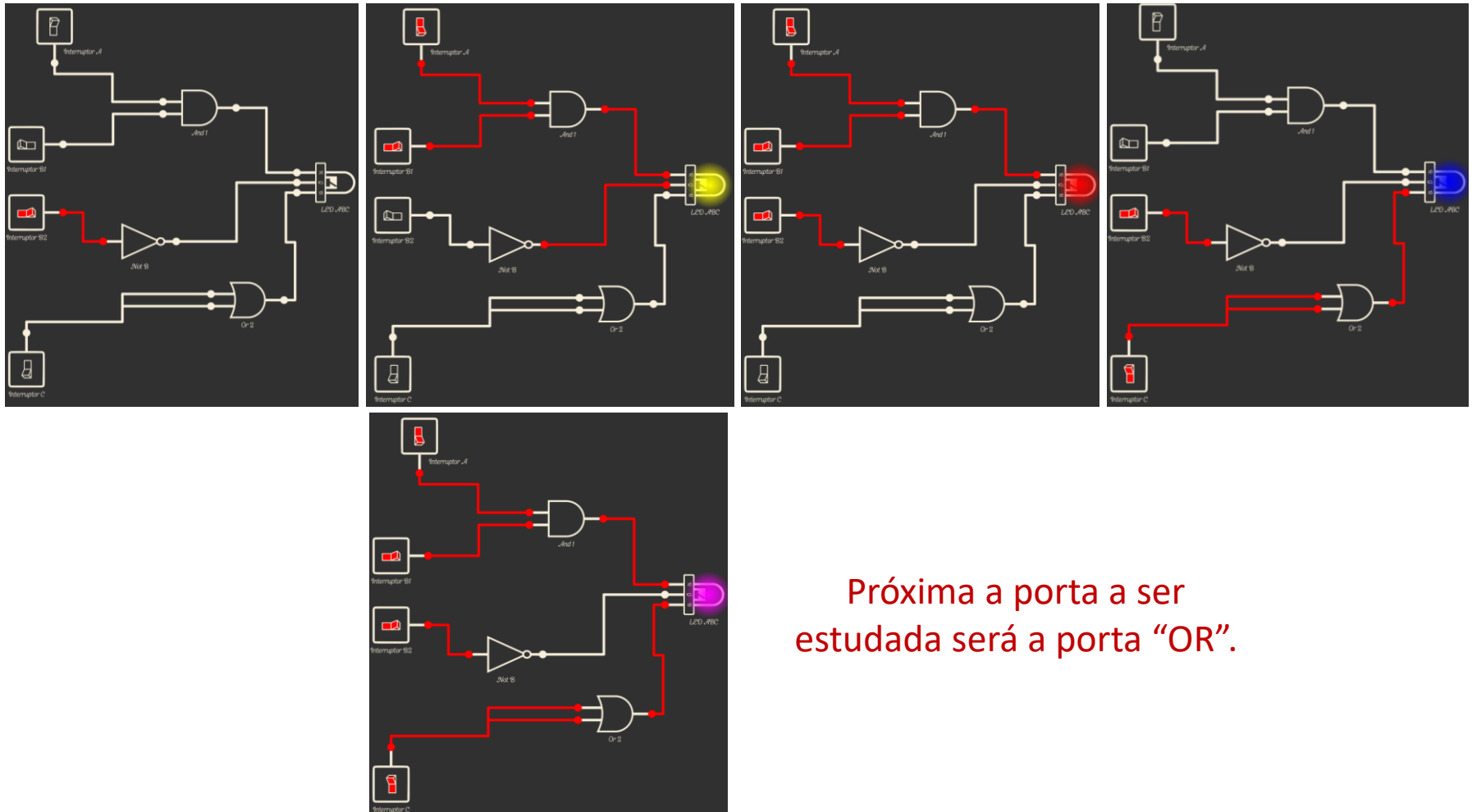
→ O resultado será obtido através da realização direta das operações em **uma** etapa.

Tabela Verdade AND (X)						
		A	B	C	D	$X = A . B . C . D$
	$0101 \leftarrow A$	0	0	1	1	0
and	$0011 \leftarrow B$	1	0	1	0	0
	$1111 \leftarrow C$	0	1	1	1	0
	$1010 \leftarrow D$	1	1	1	0	0
	$0000 \leftarrow X$					
Resultado: $X = 0000$						



Operador	Tipo	Significado de "AND"
AND	Composto	<ul style="list-style-type: none"> » Utiliza o ponto (.) como operador e representação do produto lógico. » A saída é igual a 1 se todas as entradas for 1. » A saída é igual a 0 se ao menos uma entrada for 0.

Segue etapas de um circuito que usa portas lógicas AND, OR e NOT para mostrar várias cores em um Diodo de LED?

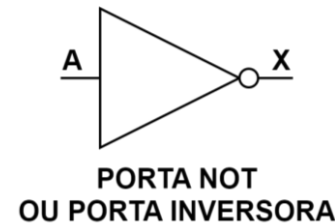


Próxima a porta a ser estudada será a porta “OR”.

Principais Operadores Lógicos - Tabela Verdade NOT

- A operação lógica **NOT** é também conhecida como porta “**inversora**”.
- Ao inverter o valor de um sinal binário colocado em sua entrada, **produz na saída um valor oposto ou seu complemento de 1**.
- Considerado um circuito lógico com apenas um valor na entrada aos símbolos utilizados em sua tabela verdade.

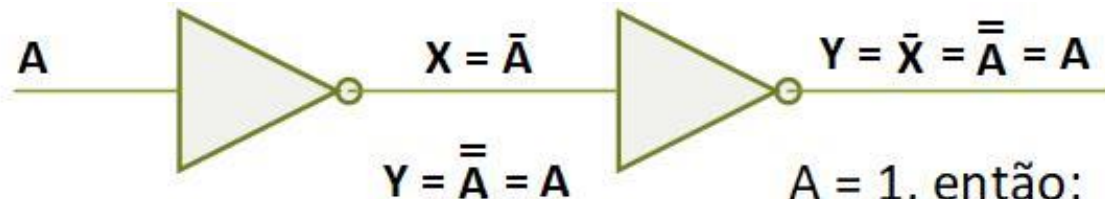
Tabela Verdade NOT		
A	$x = \bar{A}$	
0	1	V
1	0	F



Operador	Tipo	Significado NOT
“NOT ou Inversora”	Unitário	<ul style="list-style-type: none"> » O valor de entrada produz um valor contrário ao na saída. » Se a entrada for 1, a saída será 0. » Se a entrada for 0, a saída será 1.

Principais Operadores Lógicos - Tabela Verdade

→ Uma das aplicações mais comuns do circuito inversor ou “**NOT**” seria em operações aritméticas em ponto fixo, como no uso da aritmética de complemento de 1.



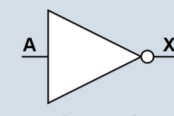
**Exemplo de um duplo inversor
que restaura o valor de entrada.**

$A = 1$, então:
 $X = \bar{A} = 0$, logo:
 $Y = \bar{X} = 1 = A$

Tabela Verdade – Exemplo 01:

- Seja $A = 0$.
- Calcular $X = \bar{A}$.
- Utilizando a tabela verdade já apresentada, temos $X = 1$ porque “**0 = 1**”.

Tabela Verdade NOT		
A	$x = \bar{A}$	
0	1	V
1	0	F



PORTA NOT
OU PORTA INVERSORA

Exemplo 02:

- Seja $A = 10010$ e $B = 11110$.
- Calcular $X_{\text{parcial}} = \overline{A \cdot B}$, seria uma operação **AND**.
- Para este exemplo teremos a realização de duas operações lógicas em sequência. Primeiro, realiza-se a operação lógica AND e, em seguida, obtém-se o inverso do resultado.
- Comparar com o resultado de “**NOT**” com “**complemento de 1**”.

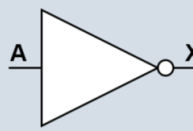
Tabela Verdade – Exemplo 02 (cont.)

Operação Lógica **AND** para o exemplo 02:

Tabela Verdade AND									
						A	B	$X_{parcial} = A . B$	
1 0 0 1 0 ← A						1	1	1	V
and	1	1	1	1	0 ← B	0	1	0	F
1 0 0 1 0 ← X						0	1	0	F
Resultado parcial de “ $X_{parcial} = 10010$ ”						1	1	1	V
						0	0	0	F

Tabela Verdade – Exemplo 02 (cont.)

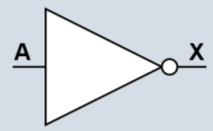
→ **Próximo passo:** Inverter usando a tabela verdade da operação lógica **OR** para obter “.não. A ou \bar{A} ” da operação AND realizado com resultado parcial de “ $X_{\text{parcial}} = 10010$ ”:

Tabela Verdade NOT		
X_{parcial}	$X = \bar{A}$	
1	0	F
0	1	V
0	1	V
1	0	F
0	1	V
 PORTA NOT OU PORTA INVERSORA		
Resultado: $X = \bar{T} = \overline{\bar{A} \cdot B} = 01101$		

Complemento de 1 = Tabela Verdade NOT	
$(1\ 0\ 0\ 1\ 0)_2$	=> Número binário original
↓ ↓ ↓ ↓ ↓	=> Onde for 0 (zero) no original passa a ser 1.
$(0\ 1\ 1\ 0\ 1)_2$	=> Complementando cada bit para obter o complemento de 1 .
+1	Calculando complemento de 2
$(0\ 1\ 1\ 1\ 0)_2$	Com a tabela verdade NOT também é possível encontrar o complemento de 2 do binário.

Tabela Verdade – Exemplo 03

→ Calcular o complemento de 1 do binário $(1110010)_2$ usando a “**tabela verdade NOT**” e depois encontrar o seu **complemento de 2**:

Tabela Verdade NOT		
A	$X = \bar{A}$	
1	0	F
1	0	F
1	0	F
0	1	V
0	1	V
1	0	F
0	1	V
 PORTA NOT OU PORTA INVERSORA		
Resultado: $X = \bar{A} = 0001101$		

Complemento de 1 = Tabela Verdade NOT	
$(1\ 1\ 1\ 0\ 0\ 1\ 0)_2$	=> Número binário original
↓ ↓ ↓ ↓ ↓	=> Onde for 0 (zero) no original passa a ser 1.
$(0\ 0\ 0\ 1\ 1\ 0\ 1)_2$	=> Complementando cada bit para obter o complemento de 1 .
+ 1	Calculando complemento de 2
$(0\ 0\ 0\ 1\ 1\ 1\ 0)_2$	Com a tabela verdade NOT também é possível encontrar o complemento de 2 do binário.

Tabela Verdade – Exemplo 04

- Calcular o “**AND**” do binário A = 100111 e B = 110001 ou $X = A . B$.
- Após calcular o “AND”, calcular o “NOT” de seu resultado.
- O sinal de “+” representa uma operação booleana “**OR**”, que veremos mais adiante.

Tabela Verdade de (A . B) + NOT					
A	B	X = A . B (AND)	.não. X = \overline{X} (NOT)		<div><p>PORTA AND</p> <p>PORTA NOT OU PORTA INVERSORA</p></div>
1	1	1	0	F	
0	1	0	1	V	
0	0	0	1	V	
1	0	0	1	V	
1	0	0	1	V	
1	1	1	0	F	
Resultado: .não. X = 011110					

Principais Operadores Lógicos - Tabela Verdade OR

Tabela Verdade OR				
A	B		$x = A + B$	
0	0	=	0	F
0	1	=	1	V
1	0	=	1	V
1	1	=	1	V

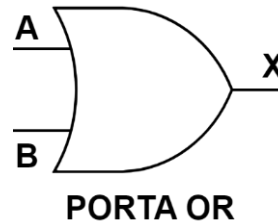


Tabela Verdade OR					
A	B	C		$x = A + B + C$	
0	0	0	=	0	F
0	0	1	=	1	V
0	1	0	=	1	V
0	1	1	=	1	V
1	0	0	=	1	V
1	0	1	=	1	V
1	1	0	=	1	V
1	1	1	=	1	V

Operador	Tipo	Significado
"OR"	Composto	<ul style="list-style-type: none"> » Utiliza o operador de soma lógica. » A saída é igual a 1 se pelo menos uma das entradas for 1. » A saída é igual a 0 se nenhuma entrada for 1.

Principais Operadores Lógicos - Tabela Verdade OR

- Operações lógicas **OR** são **muito** utilizadas na lógica digital ou na definição de condições de decisão em algumas linguagens de programação de alto ou baixo nível.
- No de trecho de programa mostrado pelo slide 21 para a operação **AND** (E) é possível alterar a condição do **Se**, tornando-a sintaxe mais flexível com o uso de operação **or**:
 - Leia X, Y e Z
 - $T = X + Y$
 - $R = Z + X$
 - **Se** ($T > 6$ **or** $R < 10$) 'condição do programa'
 - **Então** imprimir T
 - **Senão** Imprimir R 'expressa uma condição de inverso'
 - Fim

Para **T** ser impresso, basta que **uma das duas condições seja verdadeira**, " **$T > 6$ ou $R < 10$** ", não sendo necessário que ambas sejam verdadeiras.

Tabela Verdade – Exemplo Prático 5

→ Seja $A = 0110$, $B = 1110$.

→ Calcular $X = A + B$ (A or B).

Tabela Verdade OR (solução)									
						A	B	$X = A + B$	
0 1 1 0 ← A						0	1	1	V
or	1	1	1	0	← B	1	1	1	V
1 1 1 0 ← X						1	1	1	V
Resultado: X = 1110 ₂						0	0	0	F

Tabela Verdade – Exemplo Prático 6

Seja $A = 1100$, $B = 1111$ e $C = 0001$. Calcular $X = A + B + C$ (A or B or C).

→ O resultado será obtido através da realização das operações em **duas** etapas.

→ Na primeira parte, calcula-se $A + B$ ($X_{\text{parcial}} = A$ or B) e, em seguida, o resultado parcial obtido que deverá ser combinado com C **em outra operação lógica OR** (X_{parcial} or C).

Tabela Verdade OR (X_{parcial})				
	A	B	$X_p = A + B$	
1 1 0 0 ← A	1	1	1	V
or 1 1 1 1 ← B	1	1	1	V
1 1 1 1 ← X_{parcial}	0	1	1	V
Resultado: $X_{\text{parcial}} = 1111$	0	1	1	V

Tabela Verdade OR (X)				
	X_p	C	$X = X_p + C$	
1 1 1 1 ← X_{parcial}	1	0	1	V
or 0 0 0 1 ← C	1	0	1	V
1 1 1 1 ← X	1	0	1	V
Resultado: $X = 1111$	1	1	1	V

Tabela Verdade – Exemplo Prático 4 - Cálculo Direto

Seja $A = 1100$, $B = 1111$ e $C = 0001$. Calcular $X = A + B + C$ (A or B or C).

→ O resultado será obtido através da realização direta das operações em apenas **uma** etapa.

Tabela Verdade OR (X)								
				A	B	C	$X = A + B + C$	
1 1 0 0 ← A				1	1	0	1	V
or	1	1	1 1 ← B	1	1	0	1	V
0 0 0 1 ← C				0	1	0	1	V
<u>1 1 1 1</u> ← X				0	1	1	1	V
Resultado: X = 1111								

Principais Operadores Lógicos - Tabela Verdade NAND

Tabela Verdade NAND				
A	B		$x = AB$	
0	0	=	1	V
0	1	=	1	V
1	0	=	1	V
1	1	=	0	F

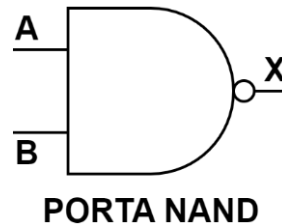


Tabela Verdade NAND					
A	B	C		$x = ABC$	
0	0	0	=	1	V
0	0	1	=	1	V
0	1	0	=	1	V
0	1	1	=	1	V
1	0	0	=	1	V
1	0	1	=	1	V
1	1	0	=	1	V
1	1	1	=	0	F

Operador	Tipo	Significado
“NAND” (Negação de AND)	Composto	<ul style="list-style-type: none"> » Utiliza o operador de produto lógico e de inversão (traço). » A saída for igual a 0 se todas as entradas for 1. » A saída é igual a 1 se ao menos uma entrada for 0.

Principais Operadores Lógicos - Tabela Verdade NOR

Tabela Verdade NOR				
A	B		$x = AB$	
0	0	=	1	V
0	1	=	0	F
1	0	=	0	F
1	1	=	0	F

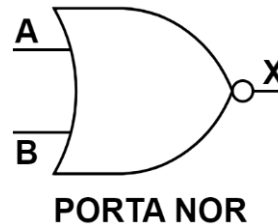


Tabela Verdade NOR					
A	B	C		$x = ABC$	
0	0	0	=	1	V
0	0	1	=	0	F
0	1	0	=	0	F
0	1	1	=	0	F
1	0	0	=	0	F
1	0	1	=	0	F
1	1	0	=	0	F
1	1	1	=	0	F

Operador	Tipo	Significado
“NOR” (Negação de OR)	Composto	<ul style="list-style-type: none"> » Utiliza o operador de soma lógica e inversão (traço). » A saída será igual a 0 se pelo menos uma das entradas for 1. » A saída será igual a 1 se nenhuma entrada for 1, todas igual a 0.

Principais Operadores Lógicos - Tabela Verdade XOR

Tabela Verdade XOR				
A	B		$x = A \oplus B$	
0	0	=	0	F
0	1	=	1	V
1	0	=	1	V
1	1	=	1	V

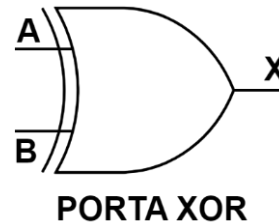


Tabela Verdade XOR					
A	B	C		$x = A \oplus B \oplus C$	
0	0	0	=	0	F
0	0	1	=	1	V
0	1	0	=	1	V
0	1	1	=	1	V
1	0	0	=	1	V
1	0	1	=	1	V
1	1	0	=	1	V
1	1	1	=	0	F

Operador	Tipo	Significado
"XOR" (EXCLUSIVO)	Composto	<ul style="list-style-type: none"> » Utiliza o operador de soma lógica com círculo (\oplus). » A saída é igual a 0 se as entradas forem iguais. » A saída é igual a 1 se as entradas não forem iguais.

Principais Operadores Lógicos – Tabela Verdade XNOR

Tabela Verdade XNOR				
A	B		$x = \overline{A \oplus B}$	
0	0	=	1	V
0	1	=	0	F
1	0	=	0	F
1	1	=	1	V

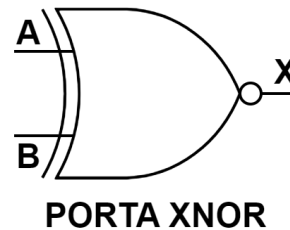
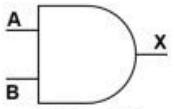


Tabela Verdade XNOR					
A	B	C		$x = \overline{A \oplus B \oplus C}$	
0	0	0	=	1	V
0	0	1	=	0	F
0	1	0	=	0	F
0	1	1	=	0	F
1	0	0	=	0	F
1	0	1	=	0	F
1	1	0	=	0	F
1	1	1	=	1	V

Operador	Tipo	Significado
“XNOR” (NOR Exclusivo)	Composto	<ul style="list-style-type: none"> » Utiliza o operador de soma lógica com círculo (\oplus) e a simbologia de inversão (traço). » A saída será verdadeiro ou 1 se todos os seus valores iguais. » A saída será falso ou 0 se ao menos um valor for diferente.

Álgebra Booleana – Explicações

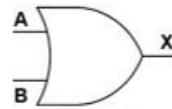
- Seria a **função booleana** de “ n ” variáveis com **2^n combinações** possíveis de valores de entrada recebendo o nome de “**tabela verdade**” contendo **2^n linhas** para **apresentar suas saídas e suas possíveis combinações**.
- **Cada linha** de uma tabela verdade procura informar o valor da função e produzir combinações de diferentes valores de entrada resultando nas possíveis saídas.



PORTA AND

AND

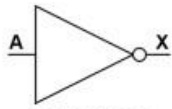
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1



PORTA OR

OR


A	B	X
0	0	1
0	1	0
1	0	0
1	1	0



PORTA NOT
OU PORTA INVERSORA

NOT


A	A
0	1
1	0



PORTA NAND

NAND


A	B	C	X
0	0	0	1
0	1	1	1
1	0	1	1
1	1	1	0



PORTA XOR

XOR


A	B	X
0	0	0
0	1	1
1	0	1
1	1	0



PORTA NOR

NOR

A	B	C	X
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

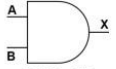

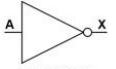


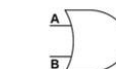
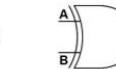


PORTA XNOR

XNOR

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

→ O sistema digital usa as **portas lógicas** como unidade básica para o seu desenvolvimento.

																																																																																																																
PORTA AND	PORTA OR	PORTA NOT OU PORTA INVERSORA	PORTA NAND	PORTA XOR	PORTA NOR	PORTA XNOR																																																																																																										
AND	OR	NOT	NAND	XOR	NOR	XNOR																																																																																																										
<table><tr><th>A</th><th>B</th><th>X</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	X	0	0	0	0	1	0	1	0	0	1	1	1	<table><tr><th>A</th><th>B</th><th>X</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	X	0	0	1	0	1	0	1	0	0	1	1	0	<table><tr><th>A</th><th>A</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	A	0	1	1	0	<table><tr><th>A</th><th>B</th><th>C</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	A	B	C	X	0	0	0	1	0	1	1	1	1	0	1	1	1	1	1	0	<table><tr><th>A</th><th>B</th><th>X</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	X	0	0	0	0	1	1	1	0	1	1	1	0	<table><tr><th>A</th><th>B</th><th>C</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	A	B	C	X	0	0	0	1	0	1	1	0	1	0	1	0	1	1	1	0	<table><tr><th>A</th><th>B</th><th>X</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	X	0	0	1	0	1	0	1	0	0	1	1	0
A	B	X																																																																																																														
0	0	0																																																																																																														
0	1	0																																																																																																														
1	0	0																																																																																																														
1	1	1																																																																																																														
A	B	X																																																																																																														
0	0	1																																																																																																														
0	1	0																																																																																																														
1	0	0																																																																																																														
1	1	0																																																																																																														
A	A																																																																																																															
0	1																																																																																																															
1	0																																																																																																															
A	B	C	X																																																																																																													
0	0	0	1																																																																																																													
0	1	1	1																																																																																																													
1	0	1	1																																																																																																													
1	1	1	0																																																																																																													
A	B	X																																																																																																														
0	0	0																																																																																																														
0	1	1																																																																																																														
1	0	1																																																																																																														
1	1	0																																																																																																														
A	B	C	X																																																																																																													
0	0	0	1																																																																																																													
0	1	1	0																																																																																																													
1	0	1	0																																																																																																													
1	1	1	0																																																																																																													
A	B	X																																																																																																														
0	0	1																																																																																																														
0	1	0																																																																																																														
1	0	0																																																																																																														
1	1	0																																																																																																														

- As portas lógicas estão presentes desde o nível mais alto de integração até os mais baixo as leituras, interpretações e saídas.
- Para a descrever dos circuitos e sua elaboração são combinados diversas portas lógicas, levando a **álgebra** para as variáveis e suas funções a assumirem somente dois valores “**0** ou **1**”, recebendo o nome de “**Álgebra Booleana** ou **Álgebra de Comutação**”, descoberto pelo o matemático inglês George Boole (1815- 1864).



- Assim como as funções na álgebra “**ordinária**”, a álgebra booleana também possui funções com uma ou mais variáveis de entrada que produzirá uma saída com base nessas **variáveis**.
- Uma função (f) booleana poderá possuir $2n$ (base 2) as combinações das variáveis para os valores de entrada, descrita por uma **tabela verdade**, na qual cada linha da tabela informa o valor da *função (f)* para possíveis combinação dos valores de entrada.
- **Assim**, as linhas de uma tabela verdade seguem uma ordem numérica de “**base 2**” que procuram descrever **duas possíveis variáveis (00, 01, 10 e 11)**, descrita pela *função (f)* em um valor **binário de $2n$ bits com leitura coluna a coluna**.



→ Comparando com as portas lógicas AND, NOR E OR, a porta lógica NAND de valor “**1110₂**” seria descrita da seguinte forma as demais:

▪ **NAND** = 1110₂, então:

☺ **AND** = 0001₂;

☺ **NOR** = 1000₂;

☺ **OR** = 0111₂.

→ Ainda temos na álgebra booleana **16 funções booleanas de duas variáveis**, que são correspondentes às **16 possíveis sequências de 4 bits** de resultantes.

→ Ao contrário a álgebra ordinária a álgebra booleana possui **um número infinito de funções de duas variáveis** e não sendo possível que seja descrita por uma tabela verdade de possíveis saídas e entradas, porque, nestes casos cada **variável poderá assumir qualquer valor** de um número infinito com possíveis valores de saída.

Álgebra Booleana – Explicações – Lógica Majoritária

- » Vamos analisar a tabela verdade para uma **função booleana de três variáveis**: $M = f(A, B, C)$.
- » Chamamos o modelo de função de três variáveis de **lógica majoritária**, porque:
 - somente será 0 se a maioria de suas entradas for 0, e;
 - somente será 1 se a maioria de suas entradas for 1.
- » Embora qualquer função booleana possa ser completamente especificada **conforme sua tabela verdade**, à medida que aumenta o número de variáveis, essa notação fica cada vez mais trabalhosa, e costuma-se usar outra forma notação no lugar dela.

Vamos analisar a tabela verdade:

Álgebra Booleana – Explicações – Lógica Majoritária

Lógica Majoritária					
Somente será 0 se a maioria de suas entradas for 0.					
Somente será 1 se a maioria de suas entradas for 1.					
Tabela Verdade $M = f(A, B, C)$					
A	B	C		M	
0	0	0	=>	0	F
0	0	1	=>	0	F
0	1	0	=>	0	F
0	1	1	=>	1	V
1	0	0	=>	0	F
1	0	1	=>	1	V
1	1	0	=>	1	V
1	1	1	=>	1	V

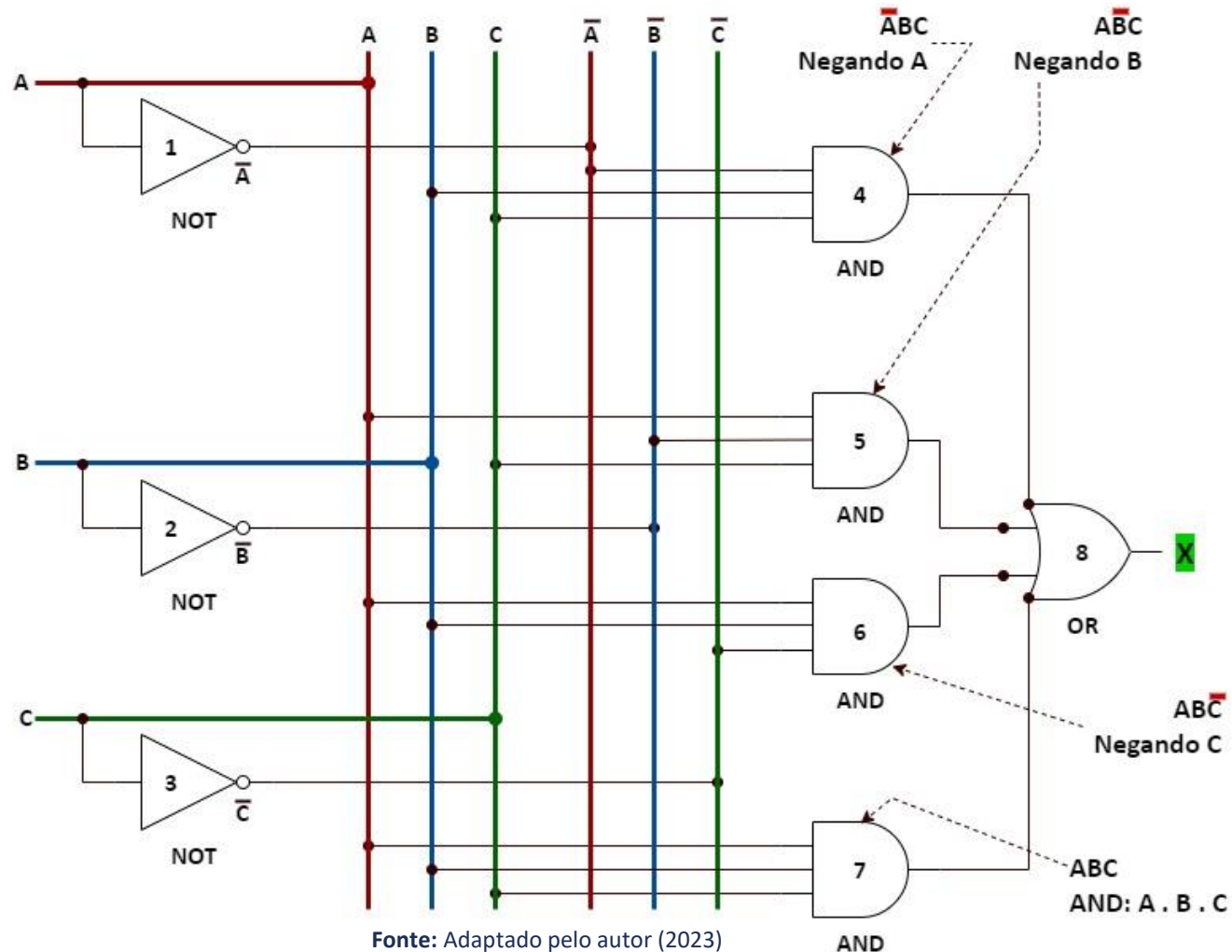
Álgebra Booleana - Exemplo de Circuito

Tabela Verdade de três variáveis

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

**A saída na porta OR será
"X = 00010111"**

Circuito Digital das Portas Lógicas



Fonte: Adaptado pelo autor (2023)

Álgebra Booleana – Explicações – Lógica Majoritária

- » No exemplo vemos que qualquer função booleana pode ser especificada ao se dizer quais combinações de variáveis de entrada dão um valor de saída igual a 1.
- » Na função tem-se quatro combinações de variáveis de entrada que fazem com que M seja 1.
- » Também temos diferenças entre uma função booleana abstrata e sua execução por um circuito eletrônico:
 - Uma função booleana consiste em variáveis, como **A, B e C**, e operadores booleanos, como **AND, OR e NOT** descrita em **maiúsculo** por uma tabela verdade ou por uma função booleana como **$F = ABC + ABC$** .
 - Já o circuito eletrônico é a execução de uma função booleana usando sinais que representam as **variáveis de entrada, saída e portas** como **and, or e not** em **minúsculo**.

Lógica Majoritária					
Somente será 0 se a maioria de suas entradas for 0.					
Somente será 1 se a maioria de suas entradas for 1.					
Tabela Verdade $M = f(A, B, C)$					
A	B	C		M	
0	0	0	=>	0	F
0	0	1	=>	0	F
0	1	0	=>	0	F
0	1	1	=>	1	V
1	0	0	=>	0	F
1	0	1	=>	1	V
1	1	0	=>	1	V
1	1	1	=>	1	V

EXPRESSÕES LÓGICAS

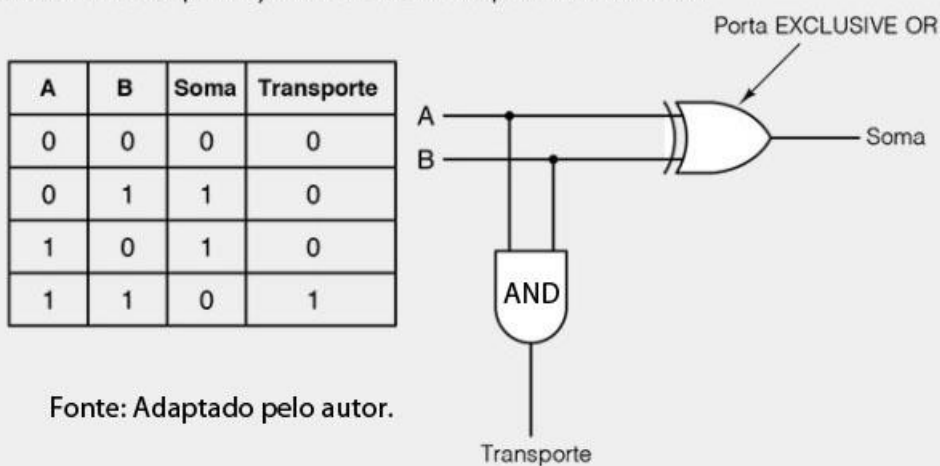
CONCEITOS

Somadores

A soma de números inteiros de um circuito de hardware e somar é essencial a CPU.

- Na tabela verdade para adição de inteiros de 1 *bit*, imagem, posição “a”, possui duas saídas presentes: a somatória das entradas “A e B” e o transporte (vai-um) que será apresentado na coluna “Soma”.
- No circuito para calcular o *bit* de soma e o de transporte imagem, posição “b”, tem-se um circuito simples, conhecido como um “meio-somador”.

(a) Tabela verdade para adição de 1 bit. (b) Circuito para um meio-somador.

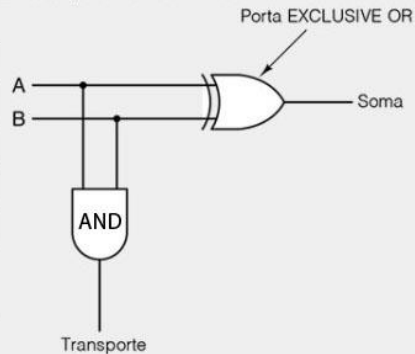


Para se obter o valor do “**Exclusive OR**” faz-se um inversão das portas obtidas com a soma.

Entendendo o cálculo de uma porta AND com a soma e inversão do binário “A = 0011 e B = 0101”:

(a) Tabela verdade para adição de 1 bit. (b) Circuito para um meio-somador.

A	B	Soma	Transporte
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Fonte: Adaptado pelo autor.

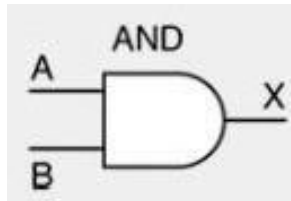
Tabela Verdade			
A	B	Soma	<i>Exclusive OR</i>
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

REGRAS DA SOMA BINÁRIA – TABELA PADRÃO

0 + 0 = 0	⇒	Sem carry ou vai um para a próxima coluna/posição.
0 + 1 = 1	⇒	Sem carry ou vai um para a próxima coluna/posição.
1 + 0 = 1	⇒	Sem carry ou vai um para a próxima coluna/posição.
1 + 1 = 0	⇒	Com carry ou vai um para a próxima coluna/posição.
1 + 1 + 1 = 1	⇒	Com carry ou vai um para a próxima coluna/posição.

Cálculo					
Soma A+B					
+	0	0	1	1	← A
	0	1	0	1	← B
	0	1	1	0	← Soma
	0	0	0	1	← Transporte da Soma
	0	0	0	1	<i>Exclusive OR (inversão)</i>

Calcular a soma e inversão da porta **AND** dos binários “A = 1111 e B = 0101”:

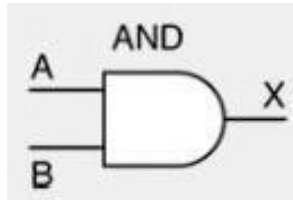


REGRAS DA SOMA BINÁRIA – TABELA PADRÃO		
0 + 0 = 0	⇒	Sem carry ou vai um para a próxima coluna/posição.
0 + 1 = 1	⇒	Sem carry ou vai um para a próxima coluna/posição.
1 + 0 = 1	⇒	Sem carry ou vai um para a próxima coluna/posição.
1 + 1 = 0	⇒	Com carry ou vai um para a próxima coluna/posição.
1 + 1 + 1 = 1	⇒	Com carry ou vai um para a próxima coluna/posição.

Tabela Verdade			
A	B	Soma	<i>Exclusive OR</i>
1	0	1	0
1	1	0	1
1	0	1	0
1	1	0	1

Cálculo						
Soma A+B						
+	1	1	1	1	← A	
	0	1	0	1	← B	
	1	0	1	0	← Soma	
	0	1	0	1	← Transporte da Soma	
	0	1	0	1	<i>Exclusive OR</i> (inversão = soma)	

Mas, se fossemos calcular a soma e inversão da porta AND de 5 bits de quatro binário: “A = 00111, B = 01101, C = 11001 e D = 11011”:

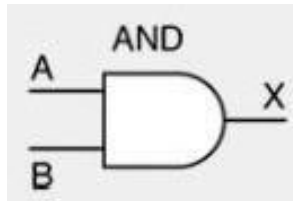


REGRAS DA SOMA BINÁRIA – TABELA PADRÃO		
0 + 0 = 0	⇒	Sem carry ou vai um para a próxima coluna/posição.
0 + 1 = 1	⇒	Sem carry ou vai um para a próxima coluna/posição.
1 + 0 = 1	⇒	Sem carry ou vai um para a próxima coluna/posição.
1 + 1 = 0	⇒	Com carry ou vai um para a próxima coluna/posição.
1 + 1 + 1 = 1	⇒	Com carry ou vai um para a próxima coluna/posição.

Tabela Verdade					
A	B	C	D	Soma	Exclusive OR
0	0	1	1	0	1
0	1	1	1	1	1
1	1	0	0	0	1
1	0	0	1	0	1
1	1	1	1	1	1

Cálculo						
Soma A + B + C + D						
+	0	0	1	1	1	← A
	0	1	1	0	1	← B
	1	1	0	0	1 ¹	← C
	1	1	0	1	1	← D
	0	1	0	0	1	← Soma
	1	1	1	1	1	← Transporte da Soma
	1	1	1	1	1	Exclusive OR (inversão = soma)

Calcular a soma e inversão da porta AND de 6 bits do binário “A = 100111, B = 101101, C = 111001 e D = 001100”:



REGRAS DA SOMA BINÁRIA – TABELA PADRÃO		
0 + 0 = 0	⇒	Sem carry ou vai um para a próxima coluna/posição.
0 + 1 = 1	⇒	Sem carry ou vai um para a próxima coluna/posição.
1 + 0 = 1	⇒	Sem carry ou vai um para a próxima coluna/posição.
1 + 1 = 0	⇒	Com carry ou vai um para a próxima coluna/posição.
1 + 1 + 1 = 1	⇒	Com carry ou vai um para a próxima coluna/posição.

Tabela Verdade					
A	B	C	D	Soma	Exclusiv e OR
1	1	1	0	1	1
0	0	1	0	1	0
0	1	1	1	1	1
1	1	0	1	1	1
1	0	0	0	1	0
1	1	1	0	1	1

Cálculo							
Soma A + B + C							
+	1	0	0	1	1	1	← A
	1	0	1	1	0	1	← B
	1	1	1	0	0	1	← C
	0	0	1	1	0	0	← D
	1	1	1	1	1	1	← Soma
	1	0	1	1	0	1	← Transporte da Soma
	1	0	1	1	0	1	Exclusive OR (inversão = SOMA)

Conceitos Rápidos - *Clock*

***Clock* é um circuito que emite uma série de pulsos com uma largura de pulso precisa e intervalos precisos entre pulsos consecutivos.**

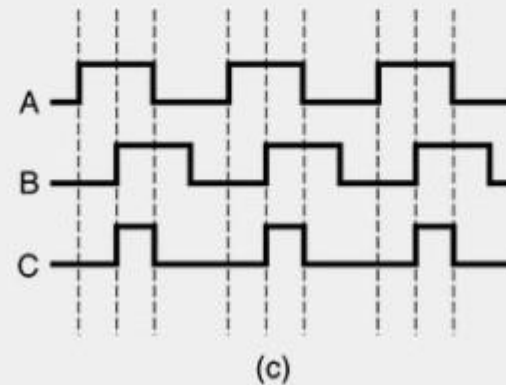
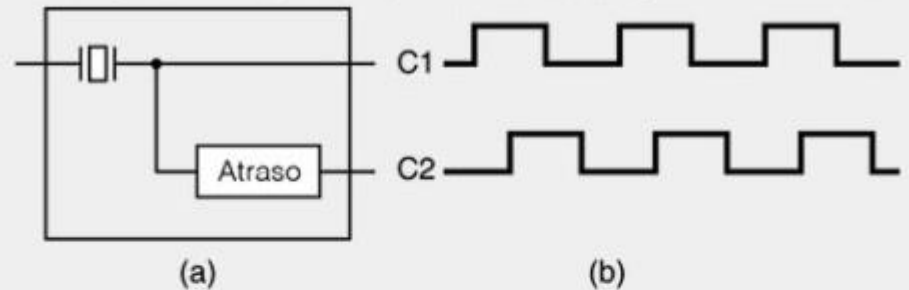
Por exemplo – O intervalo de tempo entre as **arestas** correspondentes de dois pulsos consecutivos é denominado **tempo de ciclo de *clock***.

- No geral *clocks* são frequências de pulso entre 100 MHz e 4 GHz, correspondendo a ciclos de *clock* de 10 nanossegundos a 250 picossegundos, controlados por um **oscilador de cristal**.
- Quando os *clocks* acontecem em uma ordem específica são divididos em **subciclos**.
- **Clocks são simétricos**, com o tempo gasto no estado alto igual ao tempo gasto no estado baixo.
- Toda vez que se gerar um conjunto de **pulsos assimétricos**, o **clock básico** (simétrico) é deslocado usando um circuito de atraso através de operação AND com o sinal original.

Conceitos Rápidos - *Clock*

Toda vez que se gerar um conjunto de pulsos assimétricos, o clock básico (simétrico) é deslocado usando um circuito de atraso através de operação AND com o sinal original.

(a) Um *clock*. (b) Diagrama de temporização para o *clock*. (c) Geração de um *clock* assimétrico.



Bibliografia do Curso

Bibliografia Básica

TANENBAUM, A. S. Organização estruturada de computadores. 6. ed. São Paulo: Pearson Prentice Hall, 2013 (e-book).

MONTEIRO, M. A. Introdução à organização de computadores. 4. ed. Rio de Janeiro: LTC, 2002.

STALLINGS, W. Arquitetura e organização de computadores: projeto para o desempenho. 5. ed. São Paulo: Prentice-Hall, 2002.

Bibliografia Complementar

CORRÊA, A. G. D. [org.]. Organização e arquitetura de computadores. São Paulo: Pearson Education do Brasil, 2016 (e-book).

DELGADO, J.; RIBEIRO, C. Arquitetura de computadores. 5. ed. Rio de Janeiro: LTC, 2017 (e-book).

PAIXÃO, R. R. Arquitetura de computadores - PCs. São Paulo: Érica, 2014 (e-book).

WEBER, R. F. Fundamentos de arquitetura de computadores. 4. ed. Porto Alegre: Bookman, 2012 (e-book).

WIDMER, N. S.; MOSS, G. L.; TOCCI, R. J. Sistemas digitais: princípios e aplicações. 12. ed. São Paulo: Pearson Education do Brasil, 2018 (e-book).

Conteúdo elaborado por:

Prof. Ms. Celso Candido
celsoc@unicid.edu.br

OneDrive: https://cutt.ly/Alunos_Unicid_Aulas

Fim da Apresentação