

**Uniube**

UNIUBE – CAMPUS VIA CENTRO – Uberlândia/MG
Curso de Engenharia Elétrica e Engenharia de Computação
Disciplina: Sistemas Digitais

Aula 03

Circuitos Lógicos. Introdução à Álgebra de Boole.

Revisão 3, de 17/02/2025

Prof. João Paulo Seno
joao.seno@uniube.br

1

**Uniube**

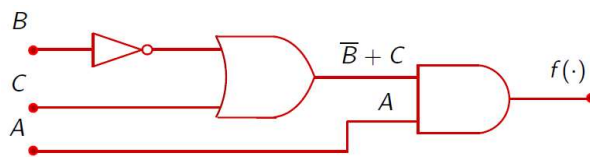
Vamos agora definir Circuito Lógico e estudar um pouco de Álgebra Booleana, para nos familiarizarmos com as portas lógicas e seu funcionamento.

Circuito lógico

- É o arranjo de um grupo de circuitos básicos padronizados conhecidos como portas lógicas, que realizam funções de lógica digital dentro da eletrônica digital.

- Exemplo:

Circuito lógico

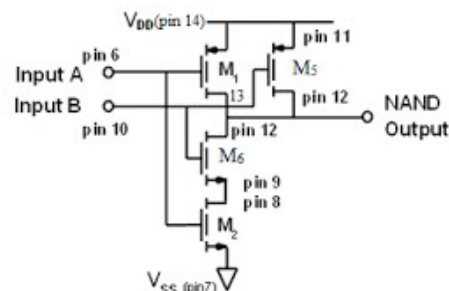
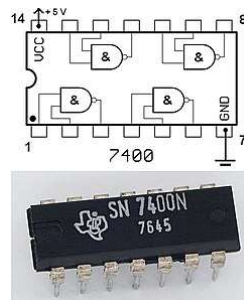


Função lógica

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

Portas lógicas

- Na prática, as portas lógicas são encontradas dentro de circuitos integrados comerciais específicos ou fazem parte da estrutura interna de dispositivos mais complexos, tais como microprocessadores, microcontroladores e outros circuitos integrados digitais.

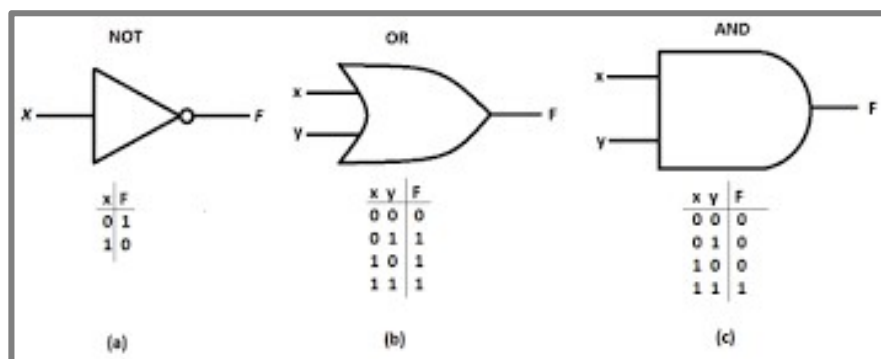


Álgebra booleana

- É um ramo especial da álgebra que é usado principalmente em eletrônica digital. A álgebra booleana foi criada por George Boole, em 1854 (ano de publicação de seu livro “Uma Investigação das Leis do Pensamento”).
- A álgebra booleana permite apenas dois estados em um circuito lógico, como Verdadeiro e Falso, Alto e Baixo, Sim e Não, Abrir e Fechar ou 0 e 1, daí sua aplicação se encaixar perfeitamente nos circuitos eletrônicos digitais.
- A álgebra booleana trabalha com dois operadores, o operador AND, simbolizado por $(.)$ e o operador OR, simbolizado por $(+)$. Há ainda um operador pra negação, o operador NOT, representado pela variável com uma barra sobre ela (ou pelo nome da variável seguido de apóstrofo).

Álgebra booleana

- A álgebra booleana é um método para simplificar circuitos lógicos (ou às vezes chamados de circuitos de comutação lógica) em eletrônica digital.
- Para relembrar o que já conversamos, veja o quadro abaixo:





Uniube

Propriedades da álgebra booleana

- **Postulados**

$A \cdot 0 = 0$	$A + 0 = A$	$A + 1 = 1$	$A \cdot 1 = A$
$A + \bar{A} = 1$	$A \cdot \bar{A} = 0$	$A + A = A$	$A \cdot A = A$

- **Propriedade Comutativa**

$A + B = B + A$	$A \cdot B = B \cdot A$
-----------------	-------------------------

- **Propriedade Associativa**

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

- **Propriedade Distributiva**

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

- **Teorema de De Morgan**

$\bar{A} \cdot \bar{B} \cdot \bar{C} \dots = \overline{A + B + C + \dots}$	$\overline{A + B + C + \dots} = \bar{A} \cdot \bar{B} \cdot \bar{C} \dots$
--	--



Uniube

Função lógica

- Uma função lógica admite uma ou mais entradas (variáveis lógicas), mas apenas uma saída.
- As variáveis lógicas, normalmente representadas por letras, podem assumir apenas dois valores mutuamente excludentes, chamados níveis lógicos, e seu uso permite que se escrevam expressões algébricas, que podem ser manipuladas matematicamente dentro da álgebra booleana.

Na eletrônica digital, é comum representarmos os níveis lógicos pelos dígitos binários **0** e **1**. O nível lógico 0 pode, por exemplo, representar a ausência de tensão ou uma chave aberta. Nesses casos, obrigatoriamente, o nível lógico 1 representará a presença de tensão, ou uma chave fechada, respectivamente.



Variável lógica ou booleana

- É uma quantidade que pode ser, em diferentes momentos, igual a 0 ou 1, ou seja, podem assumir apenas dois valores mutuamente excludentes;
- Normalmente é representadas por uma letra;
- Podem representar situações da vida real.



Operações lógicas básicas

- Existem três operações lógicas básicas (AND, OR e NOT) e outras que são derivadas destas (NAND, NOR, XOR e XNOR).

Tabela verdade

A operação lógica de uma porta pode ser expressa com uma **tabela verdade** que apresenta uma coluna para cada entrada mais uma coluna para a saída. Nas linhas da tabela verdade são listadas todas as combinações de entrada com as saídas correspondentes de uma porta lógica, facilitando a representação e a análise delas. A Figura 1.21 mostra como uma tabela verdade é organizada.

Figura 1.21 | Como montar uma tabela verdade para três variáveis

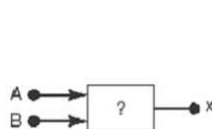
Variáveis lógicas			Variável de saída
A	B	C	S
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Todas as combinações possíveis das variáveis lógicas

Resultado da aplicação das variáveis à função

Tabela verdade

- Técnica para determinar como a saída lógica de um circuito depende dos níveis lógicos presentes nas entradas do circuito. Veja os exemplos.



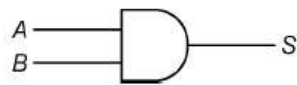
Entradas		Saída
A	B	x
0	0	1
0	1	0
1	0	1
1	1	0

A	B	C	x
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

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

Porta AND

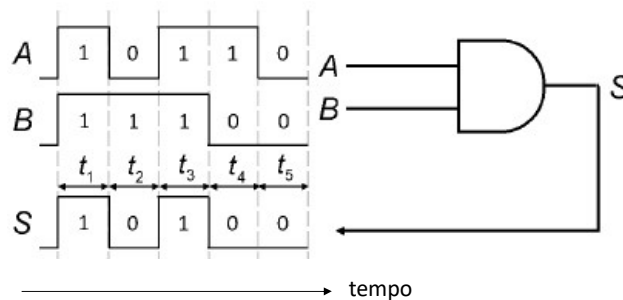
- Uma porta AND (ou E) de duas entradas é mostrada na figura abaixo;
- A saída de uma porta AND é 1 apenas quando todas as entradas forem 1. Quando qualquer uma das entradas for 0, a saída será 0.
- Veja a tabela verdade da porta lógica AND de duas entradas. Ela pode ser expandida para qualquer número de entradas.



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

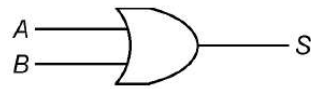
Situação prática envolvendo a porta AND

- Nas aplicações práticas, em geral, as entradas de uma porta são formas de onda de tensão que variam entre os níveis lógicos 1 e 0.
- Veja o exemplo abaixo:



Porta OR

- Uma porta OR (ou OU) de duas entradas é mostrada na figura abaixo;
- A porta OR (ou OU) produz uma saída 1 quando qualquer uma das entradas for 1. Apenas quando todas as entradas forem 0, a saída será 0.
- Veja a tabela verdade da porta lógica OR. Ela pode ser expandida para qualquer número de entradas.

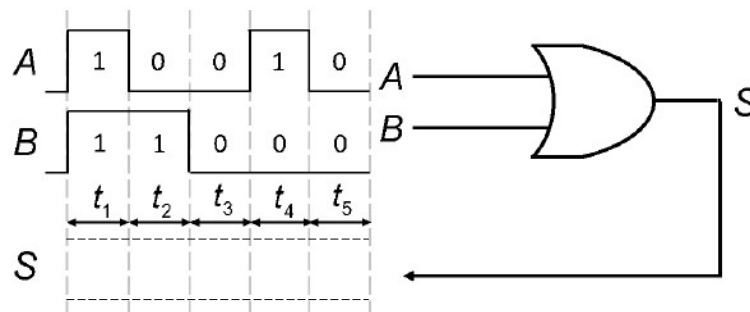


A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

Exercício

Formas de onda digitais e a porta OR

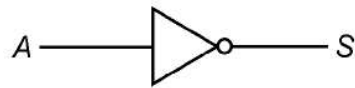
- Desenhe a forma de onda na saída (S):



Solução durante a aula!

Porta NOT

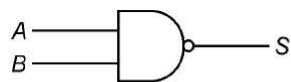
- A porta NOT (ou NÃO) realiza a operação denominada inversão lógica, e por isso pode ser chamada também de porta inversora. A porta NOT troca um nível lógico para o nível lógico oposto. Ou seja, em termos de bits, ele troca 1 por 0 e 0 por 1.
- A porta NOT tem apenas uma entrada, e seu símbolo lógico é mostrado na figura abaixo, junto com a tabela verdade.



A	S
0	1
1	0

Porta NAND

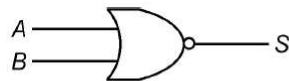
- A porta NAND (ou NÃO-E) é um elemento lógico importante, pois é considerada uma porta universal, ou seja, as portas NAND podem ser usadas em diferentes combinações para realizarem operações básicas AND, OR ou NOT.
- O termo NAND é uma contração da NOT-AND, portanto, essa porta funciona como uma porta AND com sua saída negada (invertida).
- O símbolo lógico dessa porta é apresentado na figura abaixo (o pequeno círculo na saída indica a inversão do sinal), junto com a tabela verdade.



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

Porta NOR

- A porta NOR (ou NÃO-OU) é um elemento lógico bastante útil, porque ela também é considerada uma porta universal e pode ser usada em determinadas combinações para realizar as operações básicas AND, OR e NOT.
- O termo NOR é a contração de NOT e OR, e indica que essa porta funciona como uma porta OR com sua saída invertida. O símbolo lógico padrão para essa porta está representado na figura, junto com a tabela verdade. Para uma porta NOR, a saída será nível 0 quando pelo menos uma das entradas estiver em nível lógico 1.



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

Portas XOR e XNOR

- As portas XOR (ou OU exclusivo) e XNOR (ou NÃO-OU exclusivo) são formadas pela combinação de algumas das portas já estudadas, mas, devido à sua importância em diversas aplicações, elas são tratadas como um elemento básico, e, por isso, têm seu próprio símbolo lógico.
- A saída de uma **porta XOR** é o nível lógico 1 apenas quando as duas entradas estão em níveis lógicos opostos. A operação de uma porta XOR está resumida na tabela verdade abaixo, junto com seu símbolo.



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

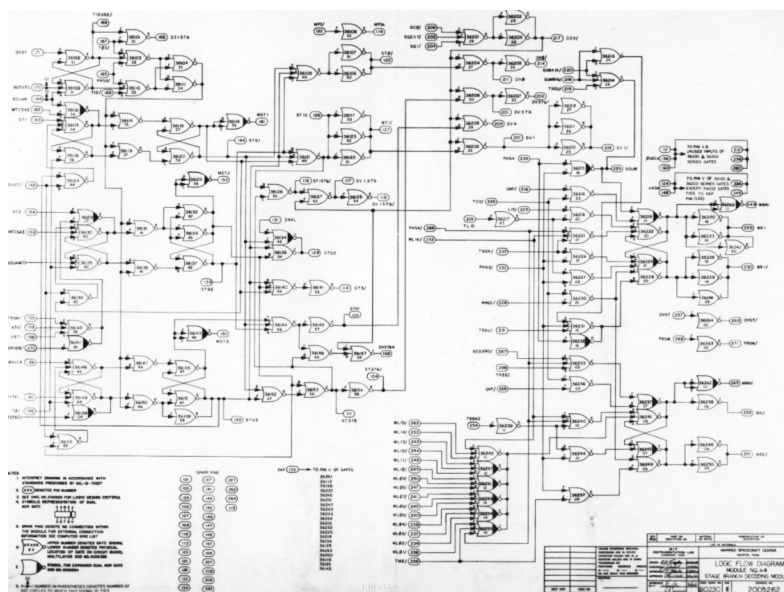
Portas XOR e XNOR

- A **porta XNOR** funciona de maneira oposta à porta XOR, ou seja, sua saída terá nível lógico 1 quando todas as entradas estiverem no mesmo nível lógico, como podemos ver na tabela verdade. O símbolo lógico padrão para essa porta está representado abaixo, também.










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

Imagem do circuito lógico de um dos módulos do computador da nave Apollo 11 (módulo Lunar) – Usando apenas portas **NOR**

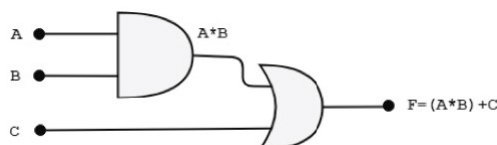


Quadro resumo

BLOCOS LÓGICOS BÁSICOS																			
PORTA	Simbologia	Tabela da Verdade	Função Lógica	Expressão															
E AND		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	S	0	0	0	0	1	0	1	0	0	1	1	1	Função E: Assume 1 quando todas as variáveis forem 1 e 0 nos outros casos.	$S=A \cdot B$
A	B	S																	
0	0	0																	
0	1	0																	
1	0	0																	
1	1	1																	
OU OR		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	S	0	0	0	0	1	1	1	0	1	1	1	1	Função OU: Assume 0 quando todas as variáveis forem 0 e 1 nos outros casos.	$S=A+B$
A	B	S																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	1																	
NÃO NOT		<table><tr><th>A</th><th>S</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	S	0	1	1	0	Função NÃO: Inverte a variável aplicada à sua entrada.	$S=\overline{A}$									
A	S																		
0	1																		
1	0																		
NE NAND		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	S	0	0	1	0	1	1	1	0	1	1	1	0	Função NE: Inverso da função E.	$S=\overline{(A \cdot B)}$
A	B	S																	
0	0	1																	
0	1	1																	
1	0	1																	
1	1	0																	
NOU NOR		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	S	0	0	1	0	1	0	1	0	0	1	1	0	Função NOU: Inverso da função OU.	$S=\overline{(A+B)}$
A	B	S																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	0																	
OU EXCLUSIVO		<table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	S	0	0	0	0	1	1	1	0	1	1	1	0	Função OU Exclusivo: Assume 1 quando as variáveis assumirem valores diferentes entre si.	$S=A \oplus B$ $S=\overline{A} \cdot B + A \cdot \overline{B}$
A	B	S																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	0																	
COINCIDÊNCIA		<table><tr><th>A</th><th>B</th><th>S</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>1</td></tr></table>	A	B	S	0	0	1	0	1	0	1	0	0	1	1	1	Função Coincidência: Assume 1 quando houver coincidência entre os valores das variáveis.	$S=A \odot B$ $S=\overline{A} \cdot \overline{B} + A \cdot B$
A	B	S																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	1																	

Circuitos com portas AND, OR e NOT

- Qualquer circuito lógico, independente da sua complexidade, pode ser construído usando as três portas lógicas básicas: AND, OR e NOT (ou inversora).
- Exemplo de um circuito lógico (já vimos na aula anterior): O circuito abaixo implementa a função F, que depende de três variáveis, A, B e C.





Uniube

Circuito lógico combinacional com portas AND, OR, e NOT e sua tabela verdade

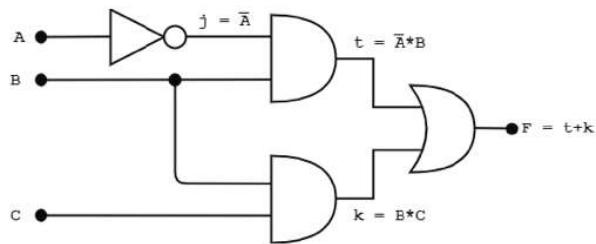


Tabela verdade

A	B	C	$j = \bar{A}$	$t = \bar{A} * B$	$k = B * C$	$F = t + k$
0	0	0	1	0	0	0
0	0	1	1	0	0	0
0	1	0	1	1	0	1
0	1	1	1	1	1	1
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	0	0	1	1



Uniube

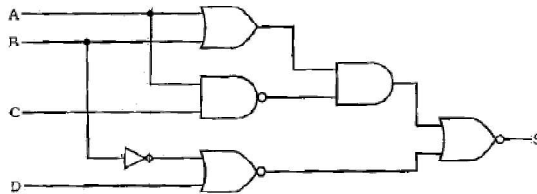
Precedência de operador

Precedência de operador: é comum um pouco de confusão quando se usam os operadores AND e OR. Existem algumas regras que devem ser obedecidas. Sempre que em uma expressão existir parênteses, essa será a operação realizada primeiro, por exemplo: $(A * B) + C$; a expressão $(A * B)$ será realizada primeiro. Outra regra: quando em uma expressão tiver lógica AND e OR, a operação que tiver a porta AND será realizada primeiro – a menos que não conflite com a regra anterior. Por exemplo: $A * B + C$



Exercício

- Determine a função lógica $S = f(A,B,C,D)$ executada pelo circuito abaixo.



Exercício

- Desenhe o circuito que executa a expressão lógica abaixo.

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



Fim