Sistemas Digitais
Portas Lógicas

Aula 03

Prof. Leandro Nogueira Couto
UFU – Monte Carmelo
05/2013

Funções e Portas Lógicas

- George Boole, 1854

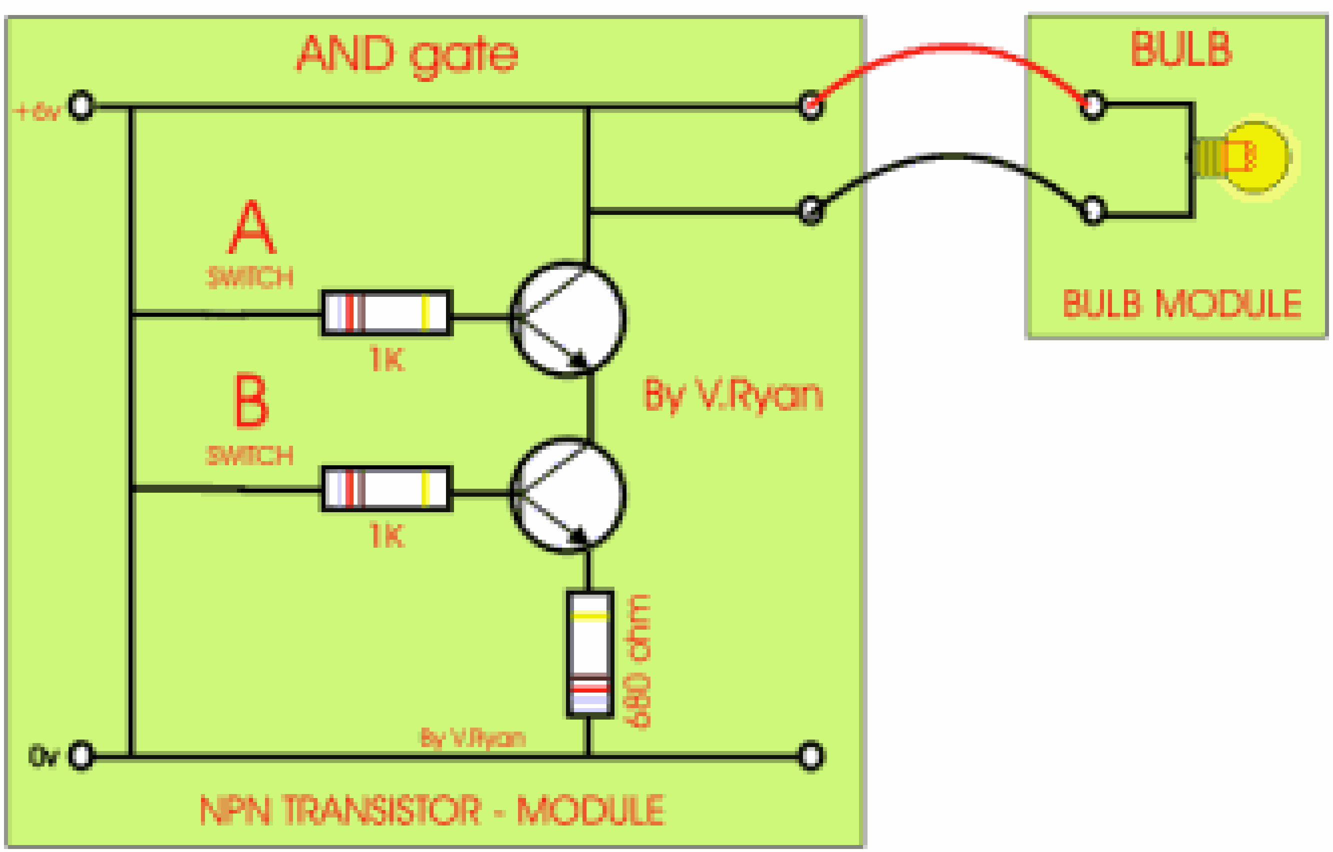
  Investigation on the Laws of Thought
- Fundamentos da álgebra Booleana
- Podemos representar essa álgebra com componentes eletrônicos
- Usamos os 2 estados possíveis, **0 e 1**, no lugar de **verdadeiro e falso**

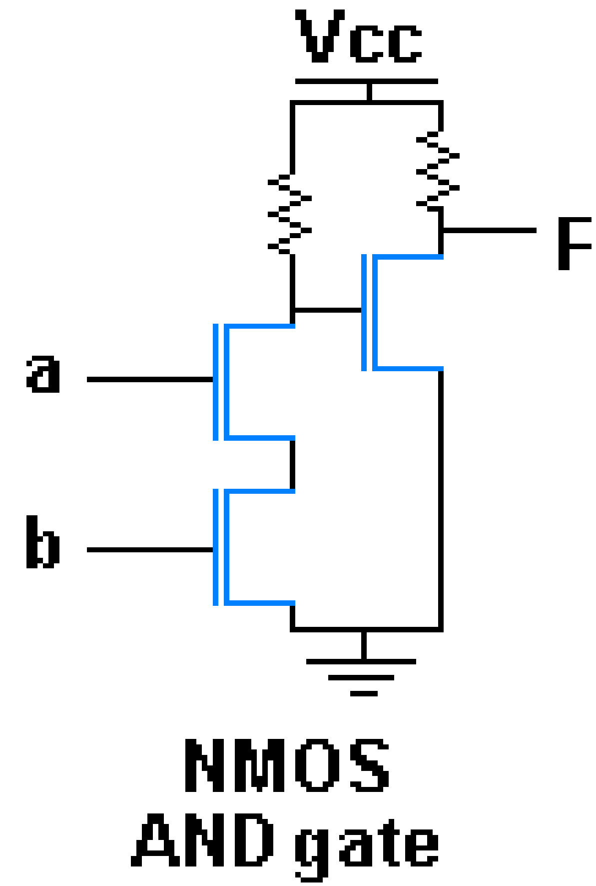
Funções e Portas Lógicas

- Com a invenção dos relays e **transistors** no século XX, foi possível reproduzir o comportamento dessas funções Booleanas em circuitos digitais
- Antes, era tudo analógico!
- Tamanho de um transistor no Xbox ONE: 28 nm
- Número de transistores no Xbox ONE: 5 bilhões

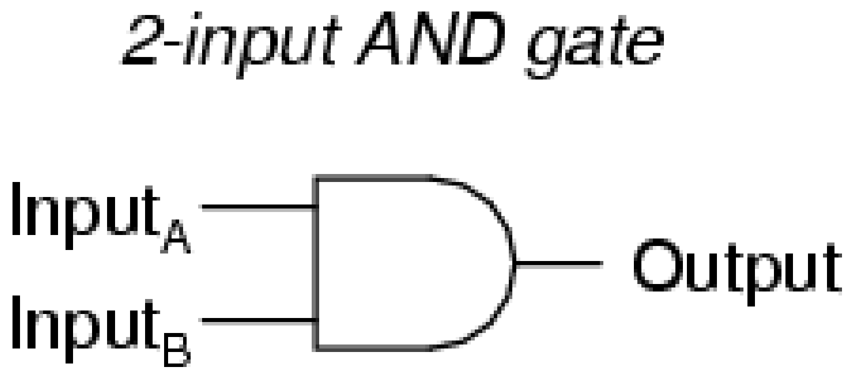
Funções e Portas Lógicas

- Por exemplo, dado 2 entradas diferentes A e B que podem ser 0 ou 1 será possível replicar a função lógica A∧B usando transistores? (simplificado!)





Porta E ou AND



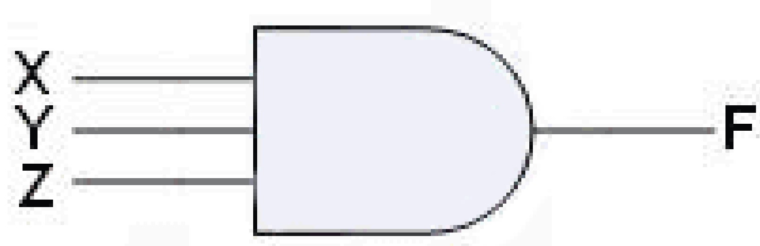
A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

- A função AND equivale à multiplicação
- Podemos escrever como A and B, ou **A.B**
- **Output** será 1 apenas quando todas as entradas (no caso **A** e **B**) forem 1

Porta E ou AND

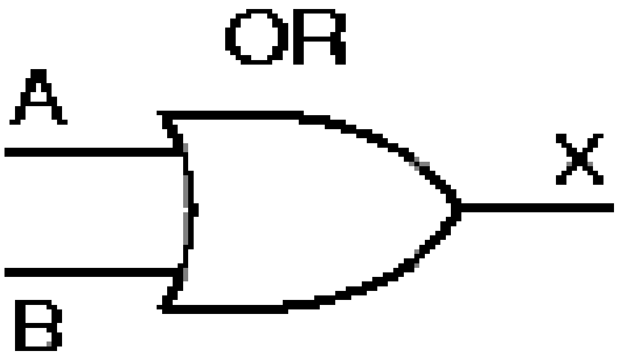
- Podemos representar AND como switches em série

- O número de entradas pode variar. Podemos ter uma AND de 3 entradas, por exemplo:



3 Input AND

Porta OU ou OR

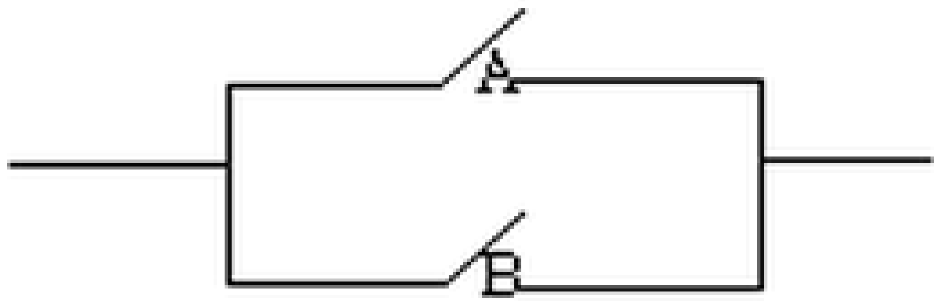


A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

- A função OU equivale à adição
- Podemos escrever como A or B, ou **A+B**
- **Output** será 0 apenas quando todas as entradas (no caso **A** e **B**) forem 0

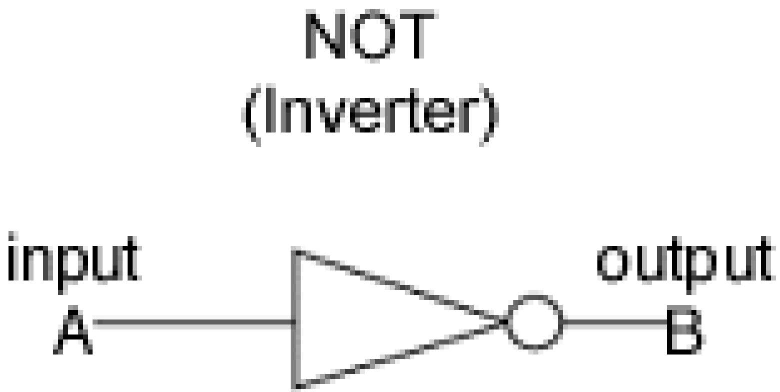
Porta OU ou OR

- Podemos representar OR como switches em paralelo



- Como na AND, o número de entradas pode variar

Porta NÃO ou NOT

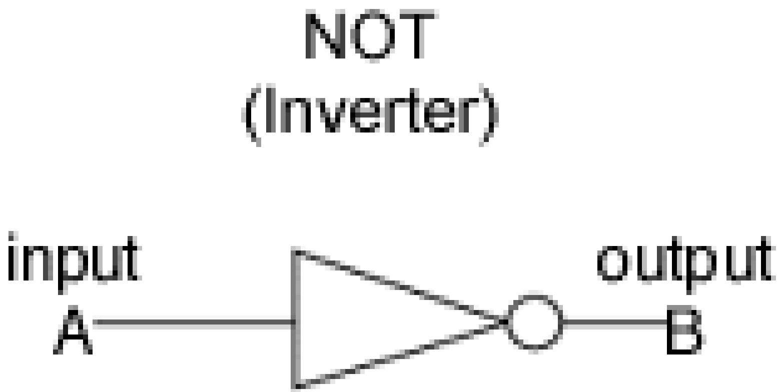


A	B
0	1
1	0

- Também chamamos a função NOT de **inversor**
- Note que a negação é representada pela "bolinha"
- O triângulo representa apenas um "buffer"

  (Se tivermos só o triângulo chamamos de buffer não-inversor)
- **Output** será sempre invertido

Porta NÃO ou NOT



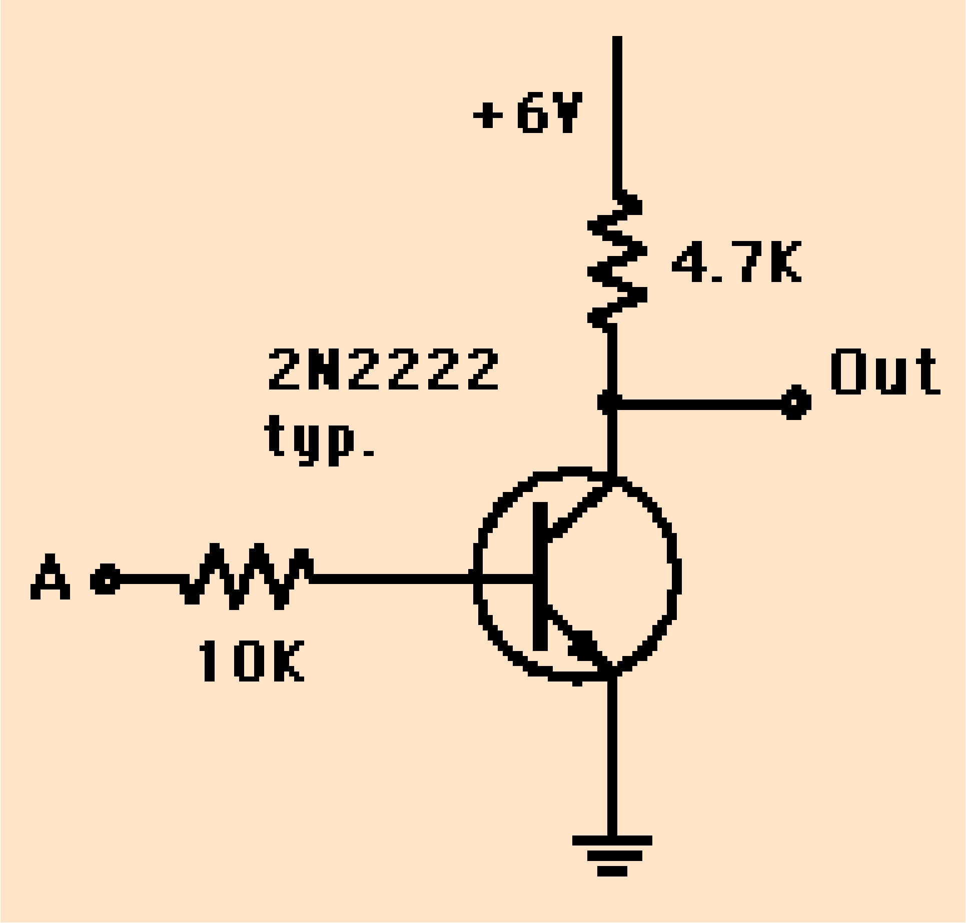
A	B
0	1
1	0

- A notação para o NOT nos Sistemas Digitais é uma barra sobre a variável de entrada ou um til (~):

$\bar{A}$ ou ~A

Porta OU ou OR

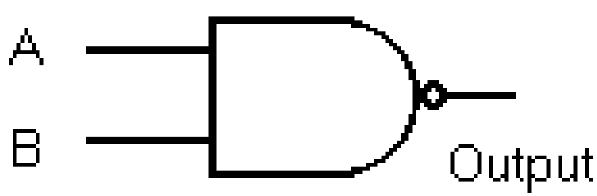
- O circuito para uma porta NOT, por curiosidade:

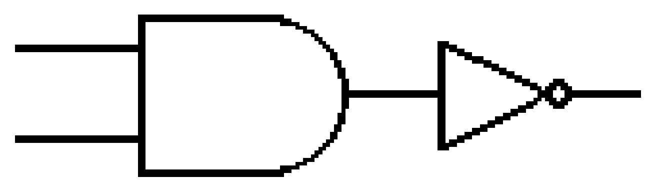


Porta NE ou NAND

- Como podemos implementar a expressão $\neg(A \land B)$?
- Já sabemos fazer o AND e sabemos fazer o NOT!

- Podemos desenhar a porta NÃO-AND, ou NAND, como na figura da esquerda
- Na notação de SD: $\overline{A . B}$





A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0

Output = $\overline{A . B}$

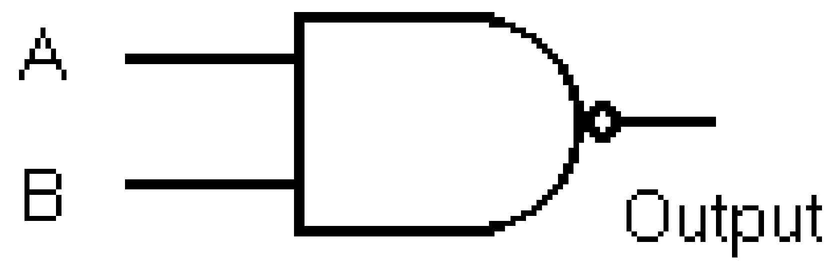
Porta NOU ou NOR

- O mesmo é válido pra porta NÃO OR, ou NOR

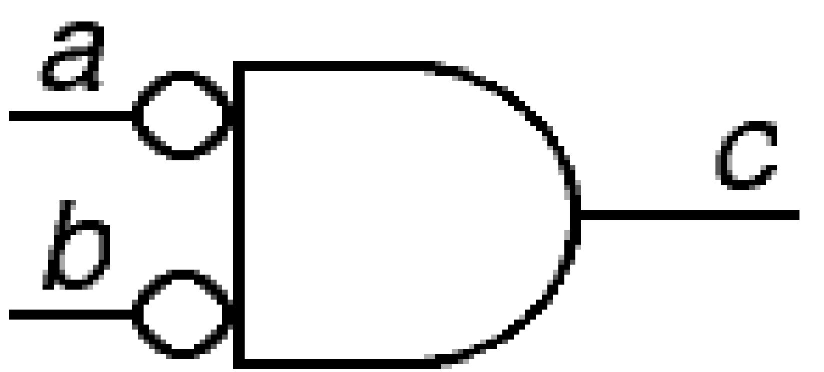
A	B	Out
0	0	1
0	1	0
1	0	0
1	1	0

Porta NOU ou NOR

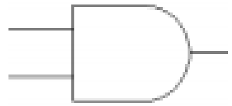
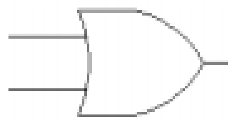
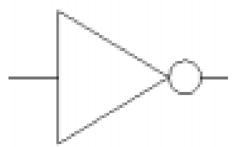
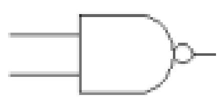
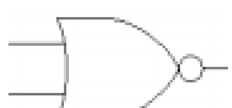
- Note que a **bolinha** significa **negação**, ou **NOT**
- Podemos colocá-la até mesmo logo antes de uma porta lógica, como no exemplo ao lado
- Como fica a tabela verdade ao lado? Note que não é a mesma coisa que NAND!
- Exemplo (circuito FSM)



$\overline{A.B}$

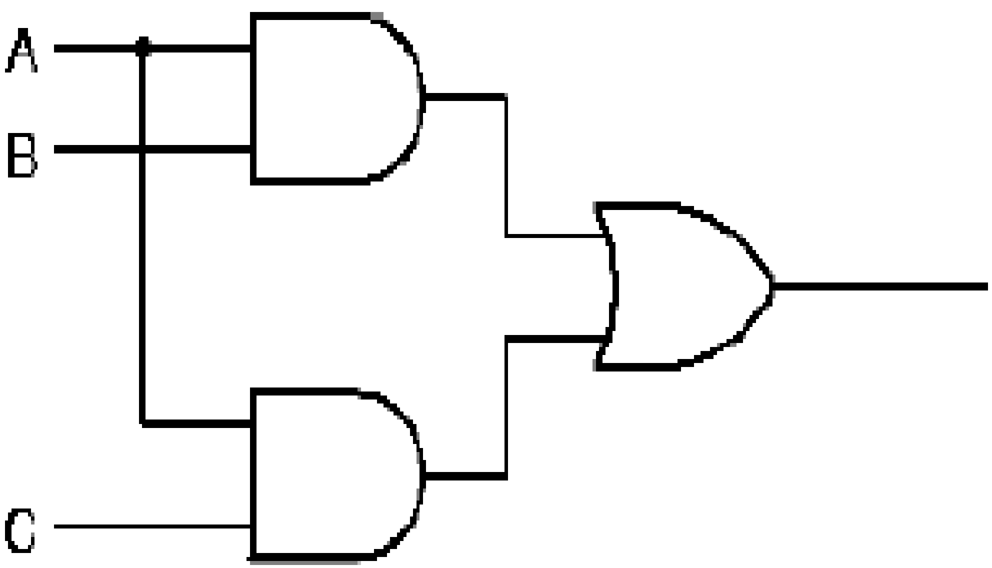


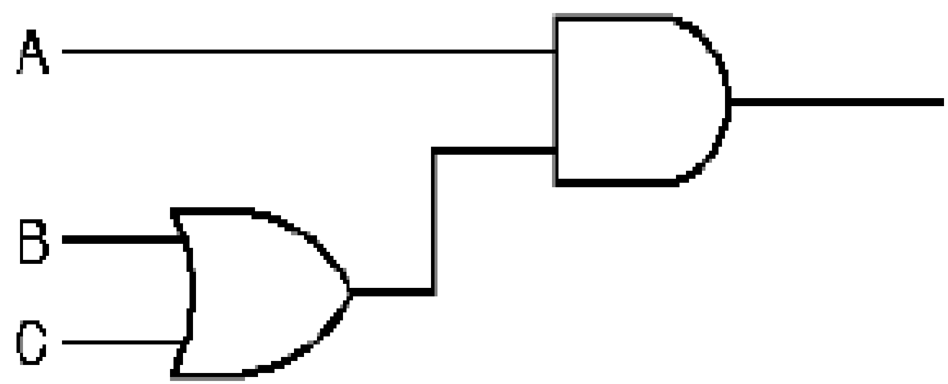
$\overline{A}.\overline{B}$

PORTA	Simbologia	Tabela da Verdade	Função Lógica	Expressão
E **AND**		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.B**
OU **OR**		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**
NÃO **NOT**		A S 0 1 1 0	**Função NÃO:** Inverte a variável aplicada à sua entrada.	**S=$\overline{A}$**
NE **NAND**		A B S 0 0 1 0 1 1 1 0 1 1 1 0	**Função NE:** Inverso da função E.	**S=$\overline{(A.B)}$**
NOU **NOR**		A B S 0 0 1 0 1 0 1 0 0 1 1 0	**Função NOU:** Inverso da função OU.	**S=$\overline{(A+B)}$**

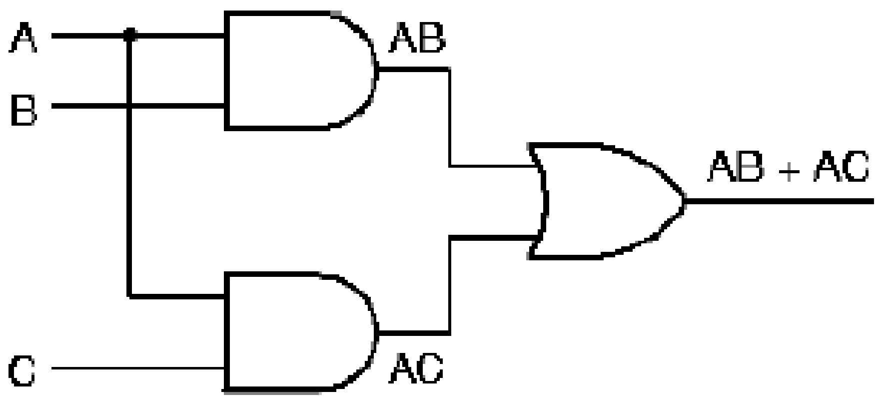
Circuitos Lógicos

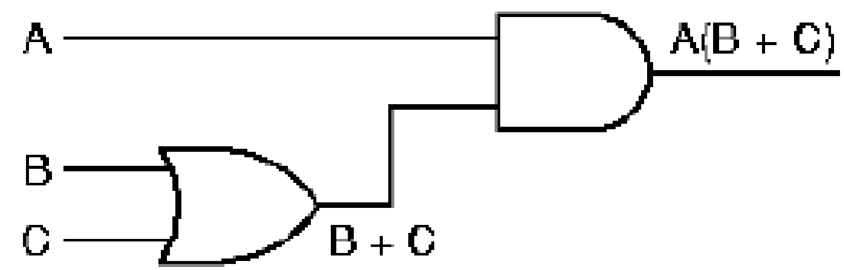
- Vimos antes que podemos combinar portas lógicas.
- Como saber qual a fórmula que determinado circuito implementa?
- Resolvamos os exemplos:





Circuitos Lógicos



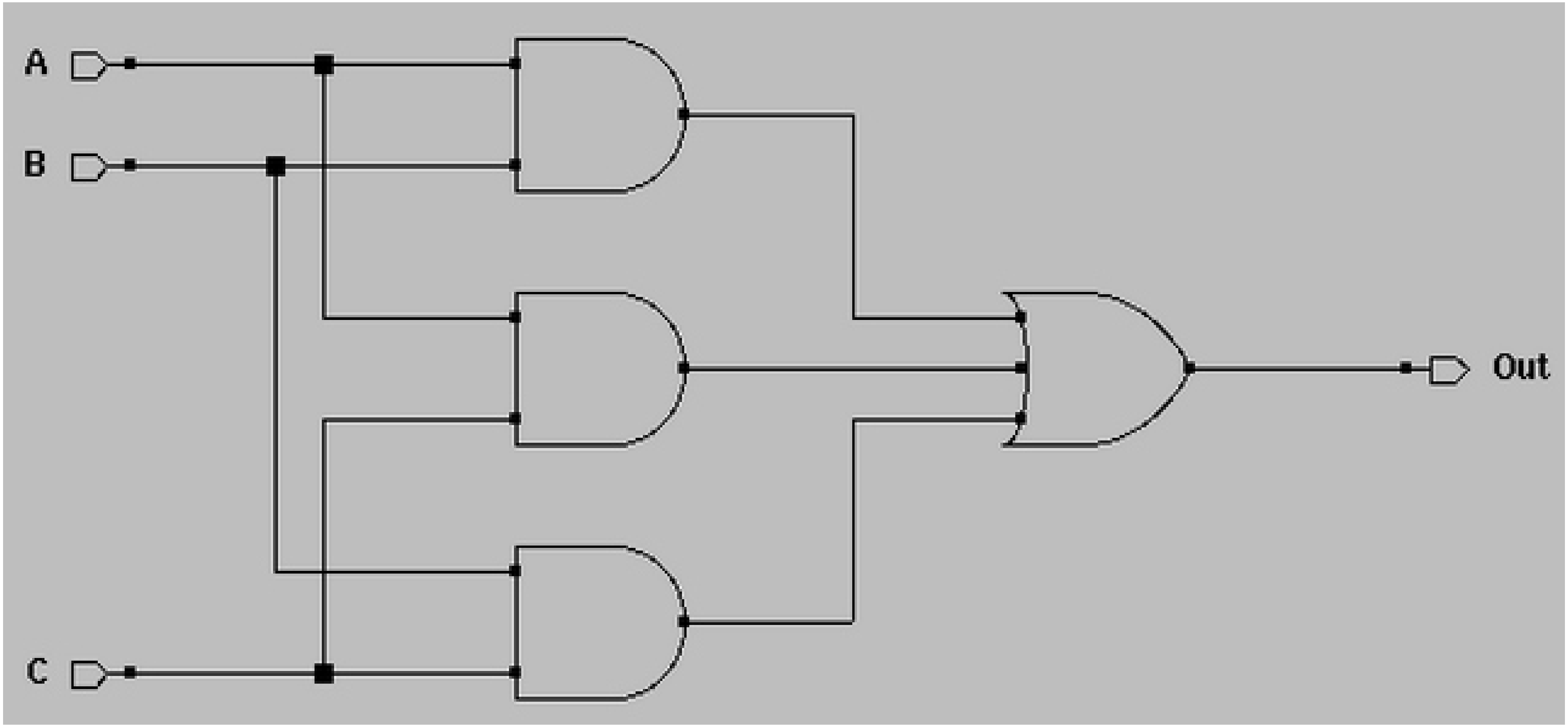


A	B	C	AB	AC	AB + AC
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

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

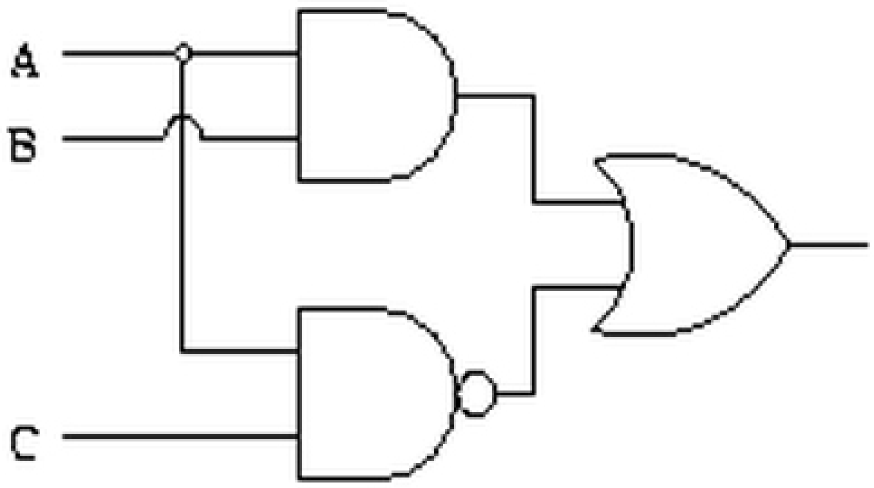
Circuitos Lógicos

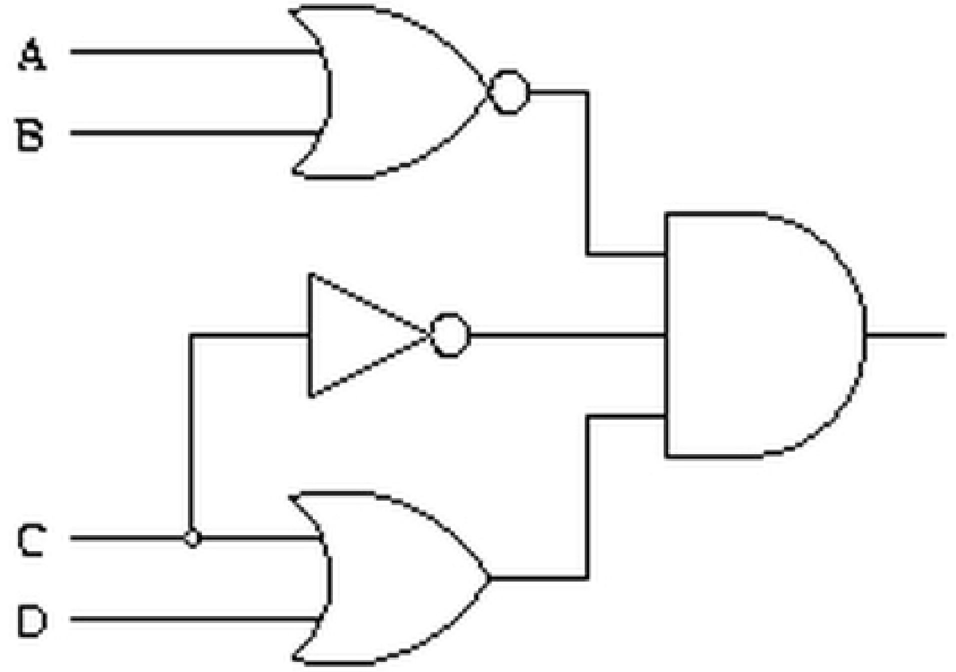
- Pelo mesmo raciocínio extraímos a tabela-verdade a partir do circuito (esse é conhecido como **Circuito da Maioria**):



Circuitos Lógicos

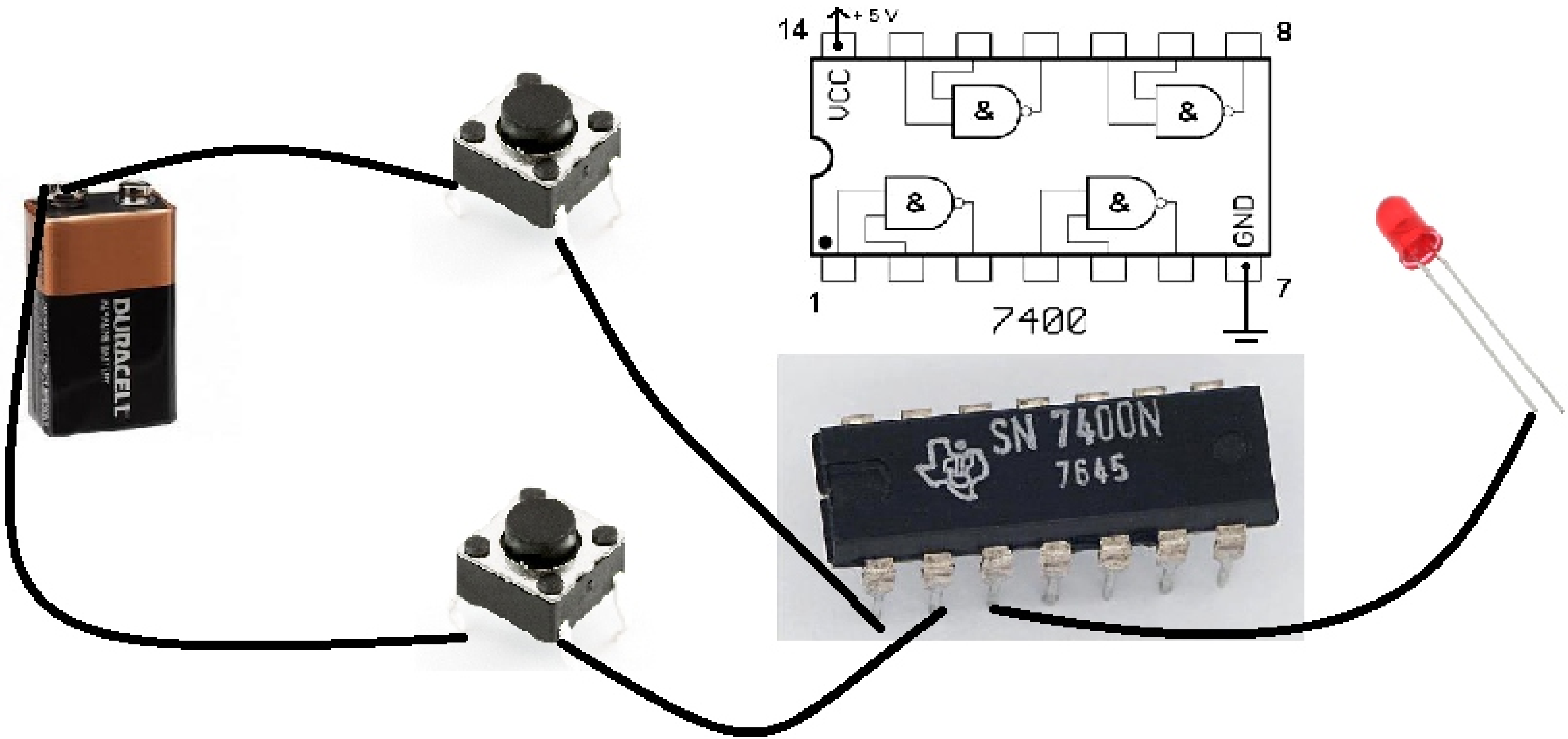
- Mais dois exemplos (tente resolver):





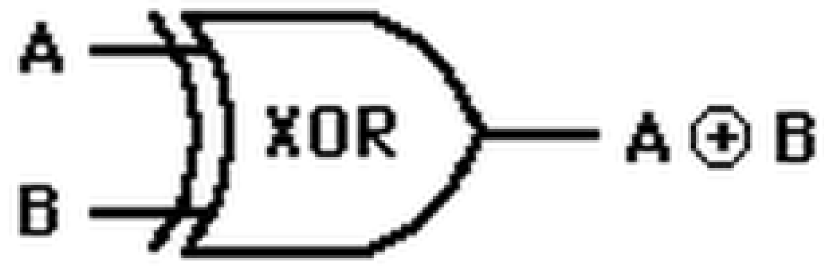
Circuitos Lógicos

- Um circuito na prática (bem simplificado)



Porta ou-exclusivo ou XOR

- No ou-exclusivo ou **XOR**, a diferença é que “queremos um ou outro, mas não os 2”
- Diferente do OR
- Note a tabela-verdade, c bloco de circuito lógico e o símbolo lógico usado para representar o XOR

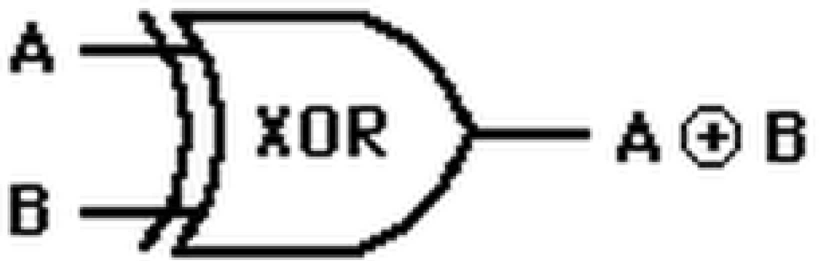


A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0

Porta ou-exclusivo ou XOR

- Problema:
  - Como fica o XOR para muitas entradas?
  - Simples:

    XOR é 1 se e somente se o número de 1s na entrada é **ímpar**



A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0

Porta Coincidência ou NOU-exclusivo

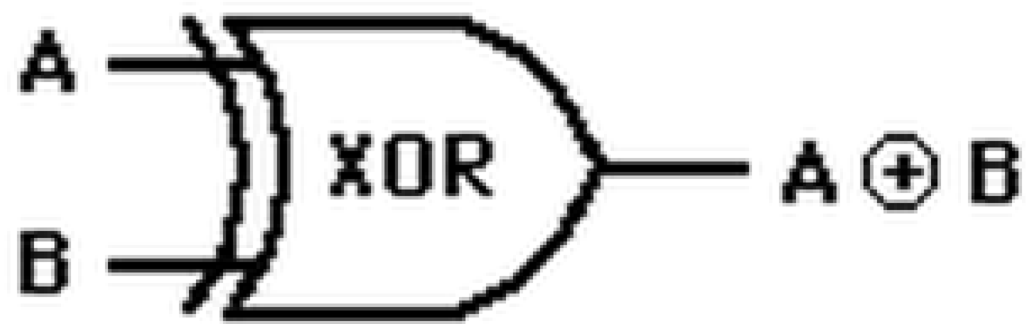
- Simplesmente a negação do XOR (inclusive na notação do bloco lógico)
- A tabela verdade é o XOR invertido
- Porque se chama coincidência (e às vezes se chama equivalência?)?
- Como fica para mais de 2 variáveis?

A	B	Out
0	0	1
0	1	0
1	0	0
1	1	1

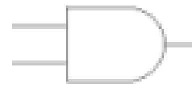
Circuitos Lógicos

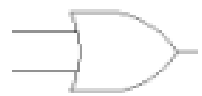
- Exemplo

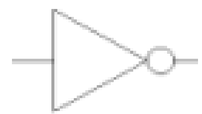
  Dados os sinais de entrada, desenhe o sinal de saída

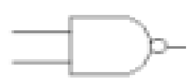


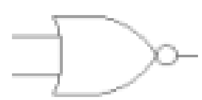
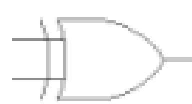
<table>
<tr><td>E

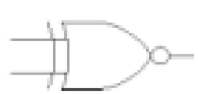
AND</td><td></td><td><table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td><td>Função E:
Assume 1 quando todas as variáveis forem 1 e 0 nos outros casos.</td><td>S=A.B</td></tr>
<tr><td>OU

OR</td><td></td><td><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></td><td>Função OU:
Assume 0 quando todas as variáveis forem 0 e 1 nos outros casos.</td><td>S=A+B</td></tr>
<tr><td>NÃO

NOT</td><td></td><td><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></td><td>Função NÃO:
Inverte a variável aplicada à sua entrada.</td><td>$S=\overline{A}$</td></tr>
<tr><td>NE

NAND</td><td></td><td><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></td><td>Função NE:
Inverso da função E.</td><td>$S=\overline{(A.B)}$</td></tr>
<tr><td>NOU

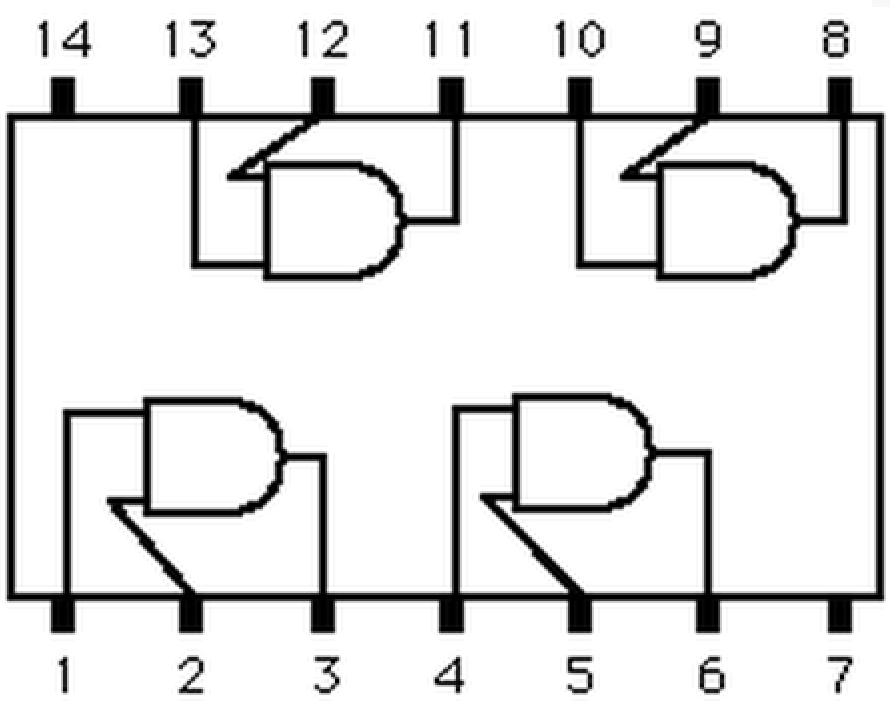
NOR</td><td></td><td><table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></td><td>Função NOU:
Inverso da função OU.</td><td>$S=\overline{(A+B)}$</td></tr>
<tr><td>OU EXCLUSIVO</td><td></td><td><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></td><td>Função OU Exclusivo:
Assume 1 quando as variáveis assumirem valorem diferentes entre si.</td><td>$S=A\oplus B$

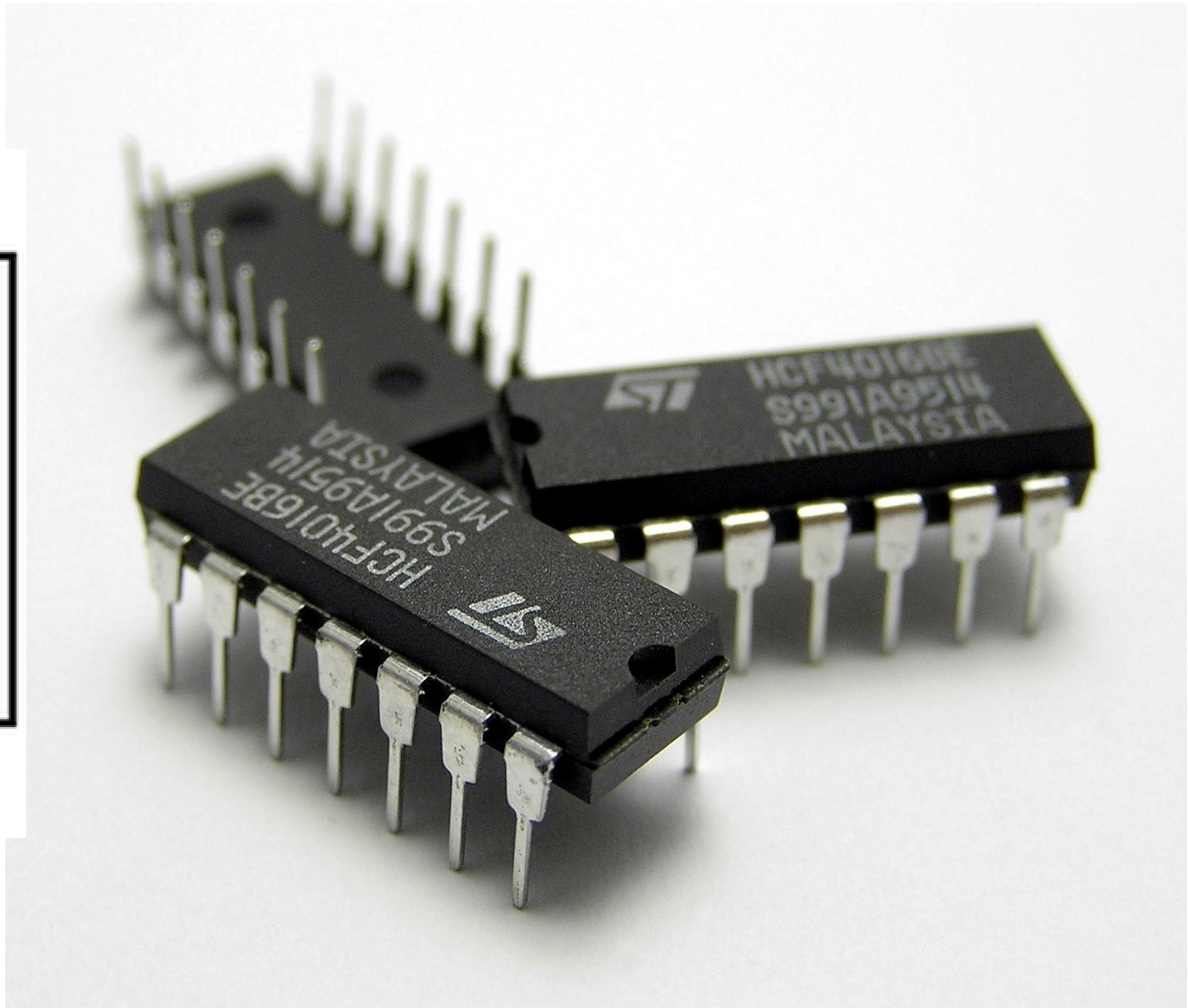
$S=\overline{A}.B+A.\overline{B}$</td></tr>
<tr><td>COINCIDÊNCIA</td><td></td><td><table><tr><th>A</th><th>B</th><th>S</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td><td>Função Coincidência:
Assume 1 quando houver coincidência entre os valores das</td><td>$S=A\odot B$

$S=\overline{A}.\overline{B}+A.B$</td></tr>
</table>

Circuitos Lógicos

- Na prática





Circuitos Lógicos

- Na prática