PORTAS NOR e NAND

As portas NOR e NAND são obtidas a partir da complementação das funções OR e AND. Podemos então dizer que o operador booleano lógico NOR é a negação do operador booleano OR enquanto que o operador booleano lógico NAND é a negação do operador booleano lógico AND.

Como sabemos o sistema binário é um sistema de numeração para representar relações lógicas que só aceitam 2 condições para resposta: VERDADEIRO (1 em binário) e FALSO (0 em binário).

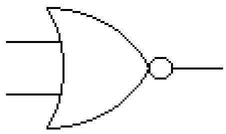
Levando-se em consideração que as portas NOR e NAND são o resultado da negação das funções OR e AND, concluímos então que qualquer circuito contendo portas lógicas pode ser descrito por meio da *“Álgebra Booleana”.*

A Álgebra Booleana tem 3 operações básicas:

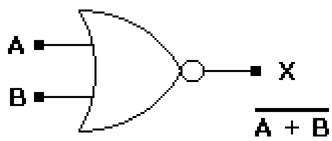
OR - AND - NOT

PORTA NOR

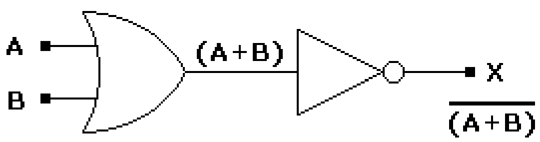
O símbolo de uma porta NOR é mostrado abaixo:



Considerando as entradas A e B, teremos na saída a complementação ou negação das mesmas.



A porta acima nada mais é do que uma porta OR complementada:

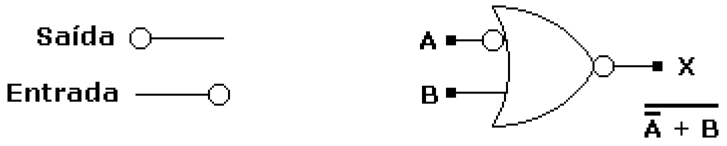


Veja a tabela da verdade a seguir:

ENTRADA		SAÍDA
A	B	
0	0	1
0	1	0
1	0	0
1	1	0

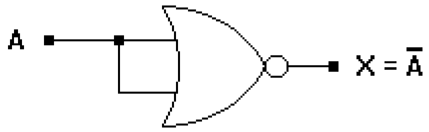
Com a finalidade de otimizar a simbologia e até mesmo simplificar o desenho dos circuitos lógicos, a porta NOT pode ser representada por um círculo, conforme mostram as figuras a seguir.

O exemplo mostrado é de uma porta NOR com apenas a entrada A, de forma que na saída teremos a expressão final complementada ou negada, bem como a variável de entrada A.

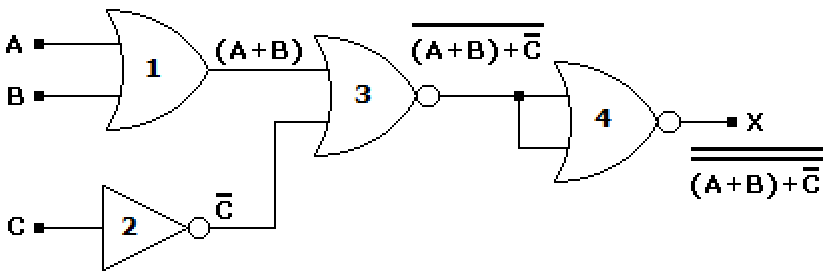


PORTA NOR COMO INVERSORA

Uma porta NOR com duas entradas pode ser utilizada como inversora, conforme ilustra a figura abaixo:



Exemplo de uma porta NOR usada como inversor:



Veja abaixo a tabela da verdade para o circuito acima:

ENTRADAS			A+B	$\overline{C}$	$\overline{(A+B)+\overline{C}}$		$\overline{\overline{(A+B)+\overline{C}}}$ (SAÍDA "X")
A	B	C					
0	0	0	0	1	*1*	0	1
0	0	1	0	0	*0*	1	0
0	1	0	1	1	*1*	0	1
0	1	1	1	0	*1*	0	1
1	0	0	1	1	*1*	0	1
1	0	1	1	0	*1*	0	1
1	1	0	1	1	*1*	0	1
1	1	1	1	0	*1*	0	1
C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**

Observações:

1 – A coluna C6 mostra o resultado das entradas da porta 3

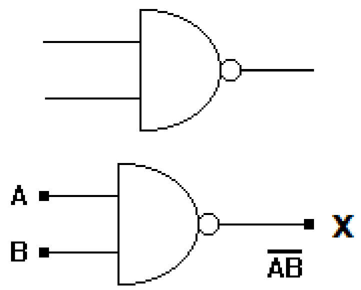
2 – A coluna C7 mostra esse resultado complementado (invertido) devido a ação da porta 3 NOR

3 – A coluna C8 mostra que os resultados retornam a condição da coluna C6, por força da porta 4 (NOR) que foi utilizada como inversor.

Logo neste caso, a porta 4 se comporta como uma porta NOT.

PORTA NAND

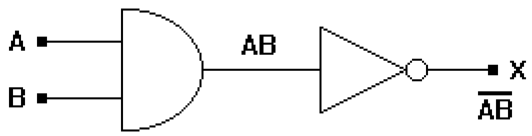
A figura abaixo mostra a simbologia de uma porta NAND com 2 entradas, sendo uma das mais utilizadas.



A saída X resulta no complemento do produto entre as entradas A e B, conforme indica a tabela da verdade abaixo:

ENTRADA		SAÍDA
A	B	
0	0	1
0	1	1
1	0	1
1	1	0

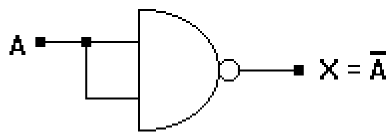
Uma porta NAND é o resultado de uma porta AND com um inversor na saída:



OPERAÇÕES DE UMA PORTA NAND

A porta NAND é uma das mais utilizadas e pode executar as operações: OR, AND e NOT, conforme mostram exemplos abaixo:

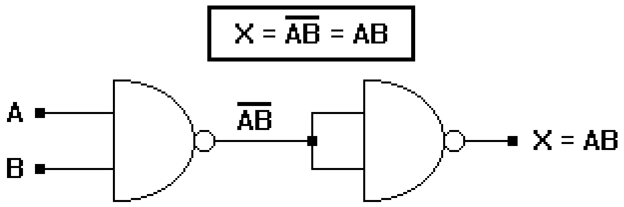
Função NOT (Inversor)



Temos uma porta NAND de 2 entradas, onde ambas estão conectadas juntas, assim, existem apenas duas possibilidades de entrada de nível lógico: 0 e 1. Observa-se então claramente que a porta NAND neste caso executa a função de um inversor lógico.

Função AND

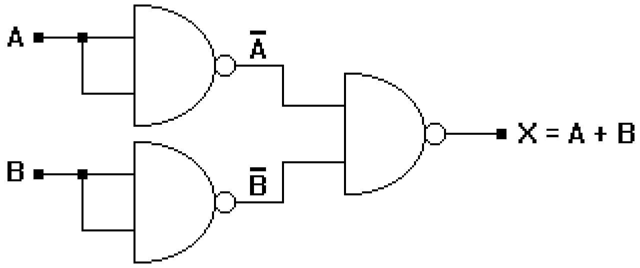
As duas portas NAND da figura abaixo são conectadas em série, onde a primeira executa a função NAND e a segunda a função de inversor lógico. Portanto “X” é o complemento da saída da primeira porta, isto é:



Função OR

A figura a seguir mostra um diagrama lógico que consiste em uma porta NAND de 2 entradas precedida de duas portas NAND de 1 entrada operando como inversores.

Em uma porta NAND se ambas as entradas forem 0 a saída será 1. Agora, como as duas entradas da porta NAND são complementadas podemos descrever a operação total do circuito como: se ambas as entradas A ou B forem 1, então a saída “X” será 1, o que é exatamente a operação de uma porta OR.



Logo, a combinação de portas NAND conforme mostra a figura acima pode ser usada como uma única porta OR.

Tabela da verdade:

Entradas		Inversão	Entradas NAND		Saída NAND	Equivalência das funções: AB = A+B
A	**B**	**A B**	**A**	**B**		
0	**0**	**1 1**	**1**	**1**	**0**	**0+0 = 0**
0	**1**	**1 0**	**1**	**0**	**1**	**0+1 = 1**
1	**0**	**0 1**	**0**	**1**	**1**	**1+0 = 1**
1	**1**	**0 0**	**0**	**0**	**1**	**1+1 = 1**
C1	***C2***	***C3***	***C4***	***C5***	***C6***	***C7***

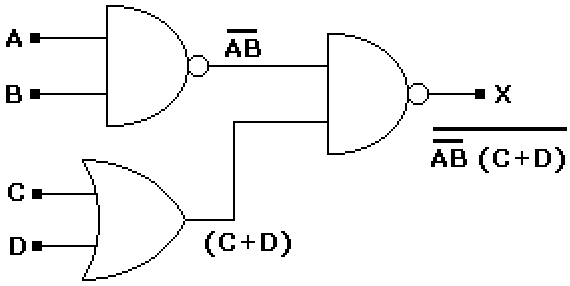
A coluna C3 mostra a inversão dos níveis lógicos aplicados às entradas A e B por conta das duas portas AND que estão atuando como inversoras. Esses valores são então aplicados nas entradas A e B da porta NAND.

Concluindo: *Observa-se que o resultado da saída X, mostrado na coluna 7 (C7) é equivalente a uma operação OR.*

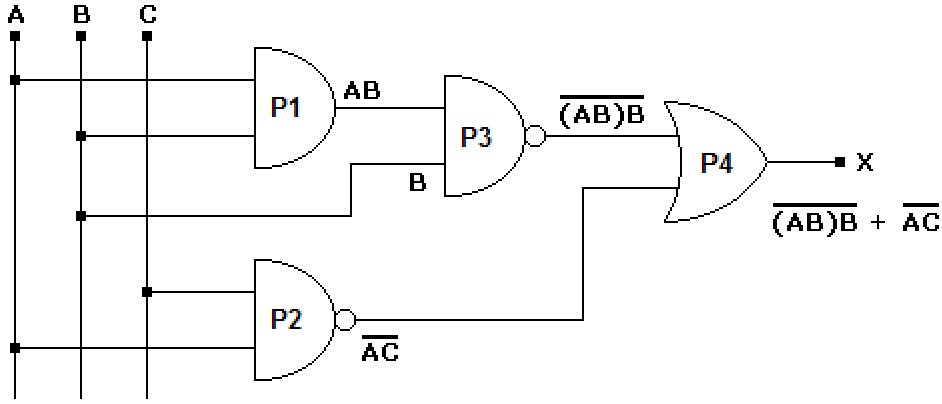
Mais adiante veremos que uma função AND (E) pode ser convertida em uma função OR (OU) e vice-versa, através dos teoremas de *“De Morgan”.*

Os teoremas do matemático De Morgan são propostas de simplificação de expressões em álgebra booleana de grande contribuição para a otimização de circuitos lógicos e blocos lógicos.
Definem regras usadas para converter operações lógicas OU em E e vice versa.

Veja a seguir um bloco lógico e respectiva expressão lógica na saída, com portas lógicas.



A figura a seguir mostra um bloco lógico onde as entradas podem ser comuns para algumas portas (neste exemplo as portas 1 e 3):



$$X = \overline{(AB)B} + \overline{AC}$$

Veja como construir a tabela da verdade para o circuito acima:

Entradas			**Saídas**		**Entradas da P3**		**Saída P3**	**Entradas da P4**		**X**
A	**B**	**C**	**P1**	**P2**	**Saída P1**	**Variável B**		**Saída P2**	**Saída P3**	
0	0	0	0	1	0	0	1	1	1	1
0	0	1	0	1	0	0	1	1	1	1
0	1	0	0	1	0	1	1	1	1	1
0	1	1	0	1	0	1	1	1	1	1
1	0	0	0	1	0	0	1	1	1	1
1	0	1	0	0	0	0	1	0	1	1
1	1	0	1	1	1	1	0	1	0	1
1	1	1	1	0	1	1	0	0	0	0

A porta P1 é uma função AND

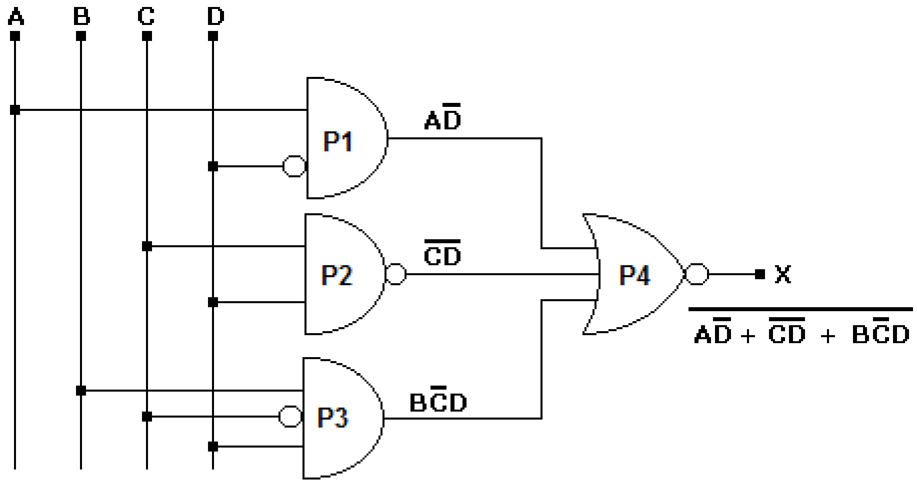
A porta P2 é uma função NAND

A porta P2 é uma função NAND

A porta P3 é uma função OR

Ao circuito acima denominamos “bloco lógico” o qual gera uma expressão lógica ou expressão booleana.

Mais um exemplo de um bloco lógico com respectiva expressão na saída:



Podemos então construir a tabela da verdade:

Entradas				Saída P1	Saída P2	Saída P3	Entradas da P4			X
A	B	C	D				Saída P1	Saída P2	Saída P3	
0	0	0	0	0	1	0	0	1	0	0
0	0	0	1	0	1	0	0	1	0	0
0	0	1	0	0	1	0	0	1	0	0
0	0	1	1	0	0	0	0	0	0	1
0	1	0	0	0	1	0	0	1	0	0
0	1	0	1	0	1	1	0	1	1	0
0	1	1	0	0	1	0	0	1	0	0
0	1	1	1	0	0	0	0	0	0	1
1	0	0	0	1	1	0	1	1	0	0
1	0	0	1	0	1	0	0	1	0	0
1	0	1	0	1	1	0	1	1	0	0
1	0	1	1	0	0	0	0	0	0	1
1	1	0	0	1	1	0	1	1	0	0
1	1	0	1	0	1	1	0	1	1	0
1	1	1	0	1	1	0	1	1	0	0
1	1	1	1	0	0	0	0	0	0	1

A porta P1 é uma função AND

A porta P2 é uma função NAND

A porta P3 é uma função AND

A porta P4 é uma função NOR