6. FUNÇÕES LÓGICAS (portas lógicas)

6.1. FUNÇÕES: E, OU, NÃO, NE e NOU

Nas funções lógicas, teremos apenas dois estados:
- o estado 0 (zero) = DESLIGADO
- o estado 1 (um) = LIGADO

O estado “0” representará, por exemplo: DESLIGADO, ou seja, portão fechado, aparelho desligado, ausência de tensão, chave aberta, não, etc;
O estado “1” representará, por exemplo: LIGADO, ou seja, portão aberto, aparelho ligado, presença de tensão, chave fechada, sim, etc;

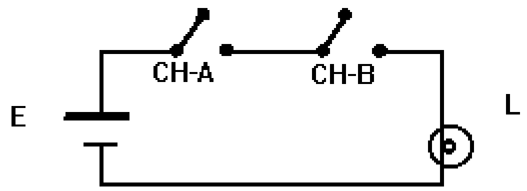
ESTADO “0”	**ESTADO “1”**
Desligado	Ligado
Portão Fechado	Portão Aberto
Aparelho Desligado	Aparelho Ligado
Ausência de Tensão	Presença de Tensão
Chave Aberta	Chave Fechada
Não	Sim

Notação: se representarmos por zero (0) uma situação, representaremos por um (1) a situação contrária.

6.1.1. Função “E” ou “AND” ($\wedge$)

A função “**E**” é aquela que executa a multiplicação de duas ou mais variáveis.

Representação Algébrica: S = A ● B, onde se lê: S = A **e** B.



Convenções: chave aberta = 0 chave fechada = 1
lâmpada apagada = 0 lâmpada acesa = 1

Situações Possíveis:
1ª) Se CH-A = 0 e CH-B = 0, lâmpada apagada. A=0, B=0, S= A ● B = 0
2ª) Se CH-A = 0 e CH-B = 1, lâmpada apagada. A=0, B=1, S= A ● B = 0
3ª) Se CH-A = 1 e CH-B = 0, lâmpada apagada. A=1, B=0, S= A ● B = 0
4ª) Se CH-A = 1 e CH-B = 1, lâmpada acesa. A=1, B=1, S= A ● B = 1

Portanto, só teremos a lâmpada acesa quando as chaves A e B estiverem fechadas (1 e 1).

Tabela da Verdade - “E” ou “AND”

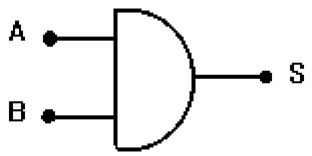
A	B	S
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

N = 2 → 2^N Linhas = 2^2 = 4 linhas

A tabela da verdade é um “mapa” onde colocamos todas as possíveis situações com seus respectivos resultados.

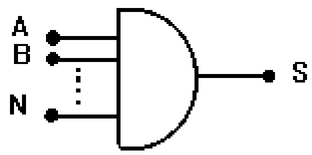
Porta “E” ou “AND”

A porta lógica é um circuito que executa a função “**E**”.



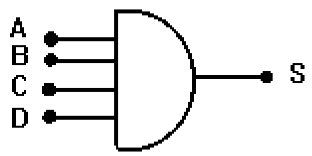
S = A ● B

Uma porta **AND** de N entradas terá saída “1”, se e somente se, todas as entradas forem iguais a “1”, e terá saída “0” nos demais casos.



S = A ● B ● C ● ... ● N

Exemplos: Vamos mostrar uma porta **AND** de quatro entradas e sua tabela da verdade.



A	B	C	D		S
0	0	0	0		**0**
0	0	0	1		**0**
0	0	1	0		**0**
0	0	1	1		**0**
0	1	0	0		**0**
0	1	0	1		**0**
0	1	1	0		**0**
0	1	1	1		**0**
1	0	0	0		**0**
1	0	0	1		**0**
1	0	1	0		**0**
1	0	1	1		**0**
1	1	0	0		**0**
1	1	0	1		**0**
1	1	1	0		**0**
1	1	1	1		**1**

6.1.2. Função "OU" ou "OR" ($\vee$)

A função "**OU**" é aquela que executa a adição de duas ou mais variáveis.

Representação Algébrica: S = A **+** B, onde se lê => S = A **ou** B.

Situações Possíveis:
1ª) Se CH-A = 0 e CH-B = 0, lâmpada apagada. A=0, B=0, S= A + B = 0
2ª) Se CH-A = 0 e CH-B = 1, lâmpada acesa. A=0, B=1, S= A + B = 1
3ª) Se CH-A = 1 e CH-B = 0, lâmpada acesa. A=1, B=0, S= A + B = 1
4ª) Se CH-A = 1 e CH-B = 1, lâmpada acesa. A=1, B=1, S= A + B = 1

Portanto, só teremos a lâmpada apagada quando as chaves A e B estiverem abertas (0 e 0).

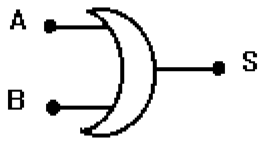
Tabela da Verdade - "**OU**" ou "**OR**"

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

N = 2 → 2^N Linhas = 2^2 = 4 linhas

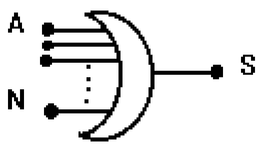
Porta “OU” ou “OR”

A porta lógica é um circuito que executa a função “OU”.



S = A + B

Uma porta **OR** de N entradas terá saída “0”, se e somente se, todas as entradas forem iguais a “0”, e terá saída “1” nos demais casos.



S = A + B + C + ... + N

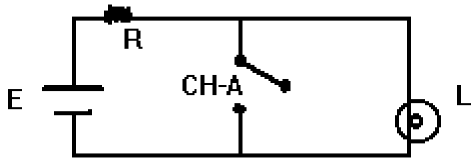
Exemplos: Vamos mostrar uma porta **OR** de três entradas e sua tabela da verdade.

A	B	C		S
0	0	0		**0**
0	0	1		**1**
0	1	0		**1**
0	1	1		**1**
1	0	0		**1**
1	0	1		**1**
1	1	0		**1**
1	1	1		**1**

6.1.3. Função “NÃO” ou “NOT” (¬)

A função **NÃO** ou função complemento é aquela que inverte o estado da variável.

Representação: S = A ou S = A’, onde se lê (A barra ou apóstrofo) ou (NÃO A).



Situações Possíveis:
1ª) Se CH-A = 0, então A = 0, A = 1, lâmpada acesa.
2ª) Se CH-A = 1, haverá um curto circuito A = 1, A = 0, lâmpada apagada.

Tabela da Verdade - "NÃO" ou "NOT"

A	A
0	**1**
1	**0**

Inversor

É o bloco lógico que executa a função **NOT** (**NÃO**) (ʽ) (¯) (¬)

Representação: A ------>O---------- A' ou Ā

O---------- após um bloco lógico ----------O antes de um bloco lógico

No caso do inversor, podemos ter somente uma entrada e uma saída.

6.1.4. Função "NÃO E" ou "NE" ou "NAND"

É uma composição da função **E** com a função **NÃO**, ou seja, teremos a função **E** invertida.

Representação Algébrica: S = (A ● B)'

Situações Possíveis:
1ª) Se CH-A = 0 e CH-B = 0, lâmpada acesa. A=0, B=0, S= (A ● B)' = 1
2ª) Se CH-A = 0 e CH-B = 1, lâmpada acesa. A=0, B=1, S= (A ● B)' = 1
3ª) Se CH-A = 1 e CH-B = 0, lâmpada acesa. A=1, B=0, S= (A ● B)' = 1
4ª) Se CH-A = 1 e CH-B = 1, lâmpada apagada. A=1, B=1, S= (A ● B)' = 0

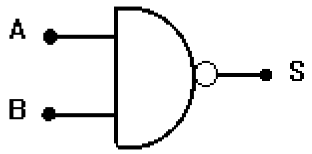
Portanto, só teremos a lâmpada apagada quando as chaves A e B estiverem abertas (1 e 1).

Tabela da Verdade - "NE" ou "NAND"

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

Porta "NE" ou "NAND"

A porta lógica é um circuito que executa a função "**NE**"



S= (A ● B)'

A porta **NAND** é a composição da porta **AND** com um inversor ligado à sua saída. A porta **NAND** pode ter duas ou mais entrada, assim como os outros blocos lógicos.

6.1.5. Função "NÃO OU" ou "NOU" ou "NOR"

É uma composição da função **NÃO** com a função **OU**, ou seja, a função **NOU** será o inverso da função **OU**.

Representação Algébrica: S = (A **+** B)'

Situações Possíveis:
1ª) Se CH-A = 0 e CH-B = 0, lâmpada acesa. A=0, B=0, S= (A + B)' = 1
2ª) Se CH-A = 0 e CH-B = 1, lâmpada apagada. A=0, B=1, S= (A + B)' = 0
3ª) Se CH-A = 1 e CH-B = 0, lâmpada apagada. A=1, B=0, S= (A + B)' = 0
4ª) Se CH-A = 1 e CH-B = 1, lâmpada apagada. A=1, B=1, S= (A + B)' = 0

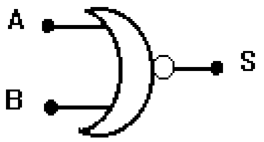
Portanto, só teremos a lâmpada acesa quando as chaves A e B estiverem abertas (0 e 0).

Tabela da Verdade - “NOU” ou “NOR”

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

Porta “NOU” ou “NOR”

A porta lógica é um circuito que executa a função “**NOR**”

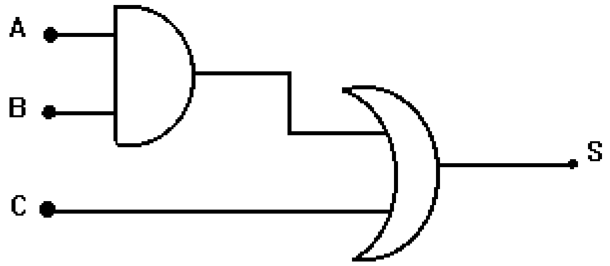


S= (A **+** B)’

A porta **NOR** é a composição da porta **OR** com um inversor ligado à sua saída. A porta **NOR** pode ter duas ou mais entrada, assim como os outros blocos lógicos.

6.2. EXPRESSÕES BOOLEANAS GERADAS POR CIRCUITOS LÓGICOS:

Vejamos qual a expressão que o circuito abaixo executa:



Circuitos Obtidos de Expressões Booleanas.

Vimos até agora que podemos obter uma expressão booleana que um circuito lógico executa. Vamos estudar que a partir de uma expressão booleana podemos desenhar um circuito.
Por exemplo, obter o circuito lógico da seguinte expressão:
S = (A + B) ● C ● (B + D)

EXERCÍCIOS (valendo pontos para a avaliação/prova)

1) Desenhe os circuitos lógicos que executam as seguintes expressões booleanas:

a) S = [(A●B) + (C●D)]’

b) S = (A●B) + C + (C●D)

c) S = (A●B●C) + ((A+B) ● C)

d) S = {[(A●B●C) + B + C]’ ● [A’+C+D + (B●C’●D)] ● [A+B+C + (B●C’●D)]}’

e) S = [(A+B’+C)’ ● (A+D’+B)]’●A’●B●C’

f) S = [(A’+B) ● (C+D’)]’ + D’ + [B●D + D’]’

g) S = [(A’●B) ● (B●C)’ ● (B+D)’]’

h) S = [(A’●B)’ + (A●B’)’ + C’]’ ● (C+D)

i) S = A●B●C + A●D + A●B●D

j) S = (A’+B) + (A●B●C’)

l) S = A●B●C + A●B’●C + A’●B’●C + A’●B’●C’

m) S = [(A+B)●C]’ + [D●(C+B)]’

n) S = [A●C’ + D + B]’ + [C ● (A●C●D)’]