Lógica Digital

- Álgebra de Boole
  - Fundamentação matemática para a lógica digital
- Portas Lógicas
  - Bloco fundamental de construção de circuitos lógicos digitais

Lógica Digital

- Circuitos Combinatórios
  - Interconexão de portas lógicas onde o sinal de saída é, em qualquer instante, função apenas de sinais de entrada
- Circuitos Seqüenciais
  - Interconexão de portas lógicas onde o sinal de saída depende também dos sinais anteriores de entrada

Contextualização

- Álgebra em Linguagem de programação
  - If (EXPRESSÃO) AND (EXPRESSÃO) {}
  - If (EXPRESSÃO) OR (EXPRESSÃO) {}
  - While (!X) {}
  - X = !X;

Contextualização

- A operação de computadores digitais é baseada no armazenamento e processamento de **dados binários**.
- Circuitos são capazes de **operar** sobre dados binários.
- As várias funções de um computador são **implementadas** através de sinais de controle e operações efetuadas por circuitos.
- **Álgebra booleana** constitui a fundamentação matemática para a lógica digital.

Álgebra Booleana

- Circuitos digitais são projetados e têm o seu comportamento analisado através da *álgebra booleana.*
- A álgebra booelana aplica-se em duas áreas:
  - **Análise**: descrição de um circuito digital
  - **Projeto**: dada uma função, pode-se utilizar a *álgebra booleana* para desenvolver uma implementação simplificada para essa função.

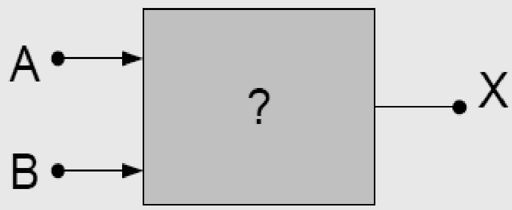
Álgebra Booleana

- Constantes e variáveis booleanas
  - Valores binários (0 – 1) ou (verdadeiro – falso) ou (alto – baixo), etc
- Apenas três operações básicas
  - AND (E)
  - OR (OU)
  - NOT (NÃO)

Tabela Verdade

- Determina os possíveis resultados de uma expressão algébrica em função dos valores das variáveis

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



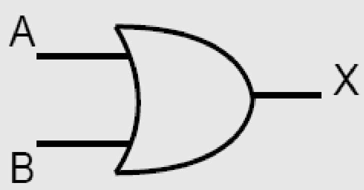
Operação OR

- A expressão booleana da operação OR é:

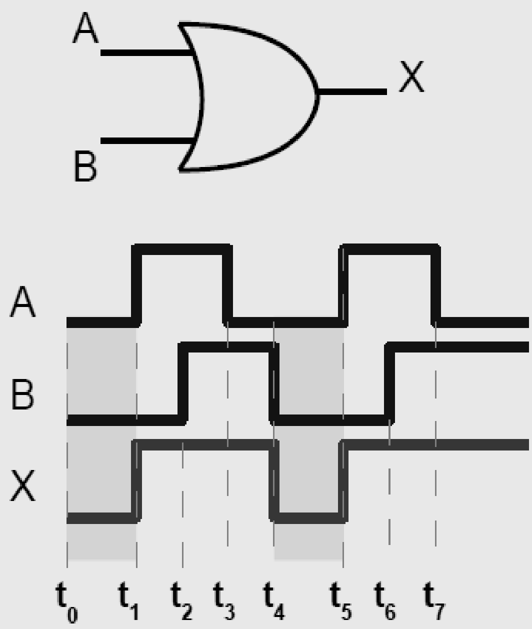
  x = A + B -> lê-se 'x é igual a A or B'

- Resultados possíveis
  - 0 + 0 = 0
  - 0 + 1 = 1
  - 1 + 0 = 1
  - 1 + 1 = 1

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

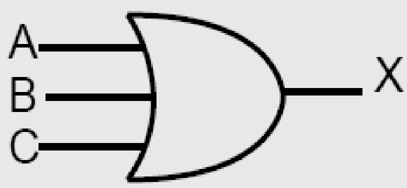


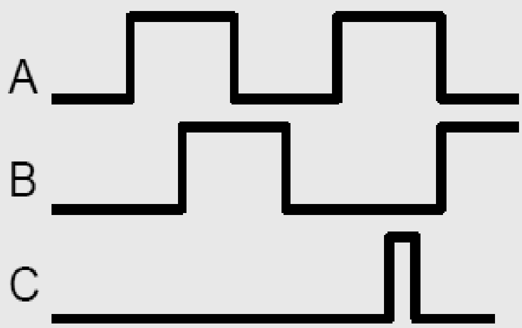
Porta OR



Prática

- Determine a forma de onda da saída X para o seguinte circuito





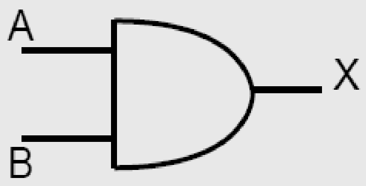
Operação AND

- A expressão booleana da operação AND é:

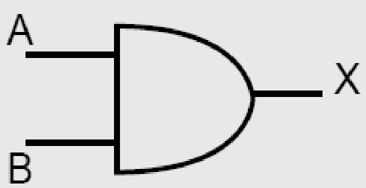
  x = A . B -> lê-se 'x é igual a A and B'

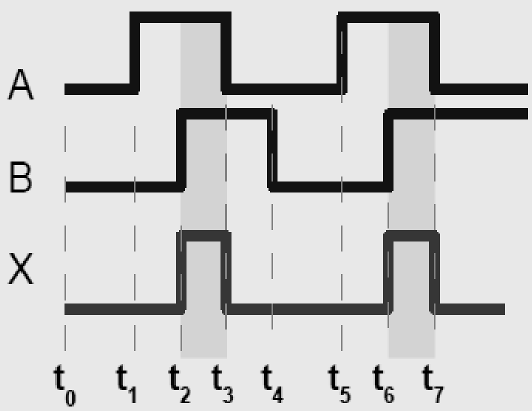
- Resultados possíveis
  - 0 . 0 = 0
  - 0 . 1 = 0
  - 1 . 0 = 0
  - 1 . 1 = 1

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



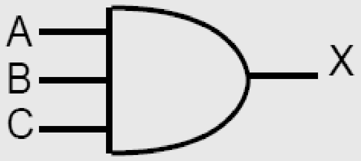
Porta AND

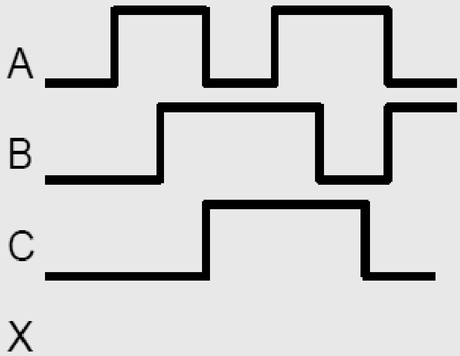




Prática

- Determine a forma de onda da saída X para o seguinte circuito





Operação NOT

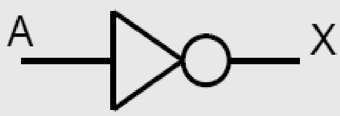
- A expressão booleana da operação NOT é:

  $x = \overline{A}$ -> lê-se 'x é igual a not A'

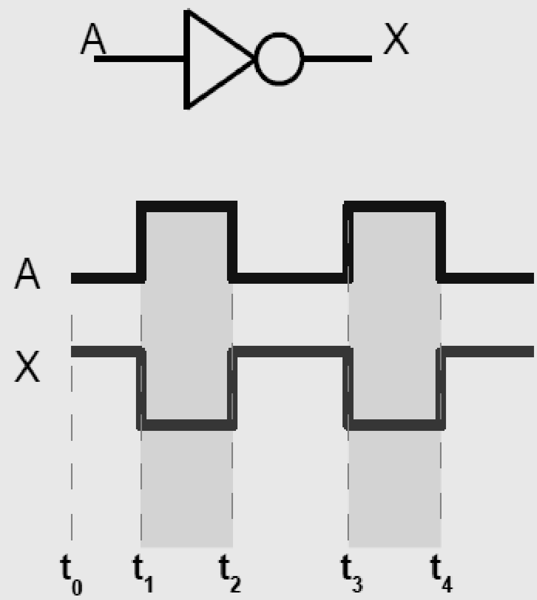
  $A' = \overline{A}$

- Resultados possíveis
  - $\overline{0} = 1$
  - $\overline{1} = 0$

A	X
0	1
1	0



Inversor

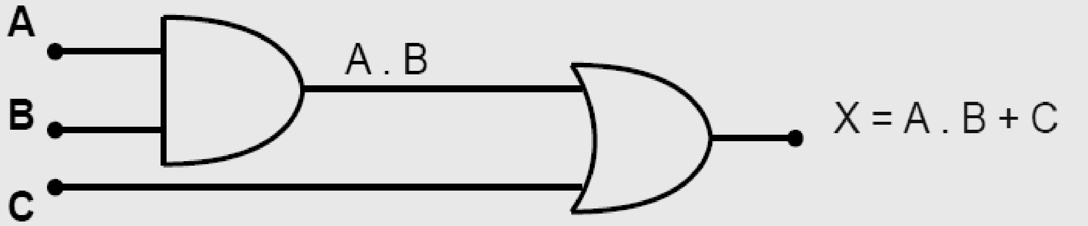


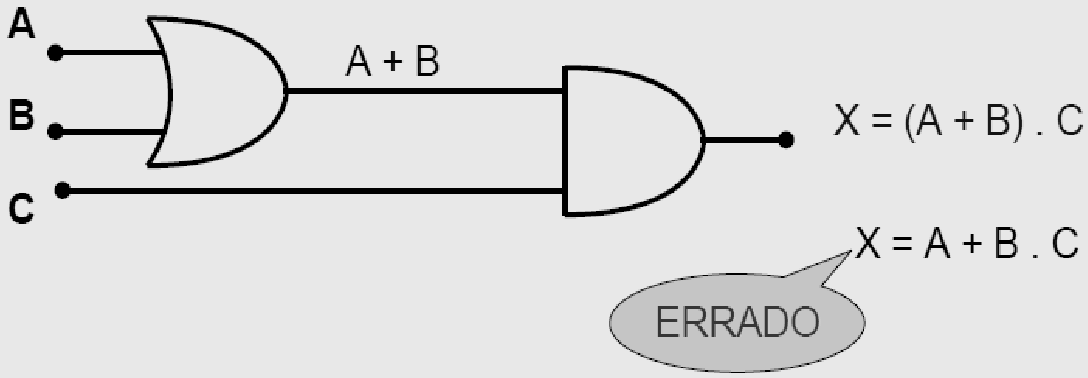
Precedência de Operações

A ordem de precedência de operações booleanas é a seguinte

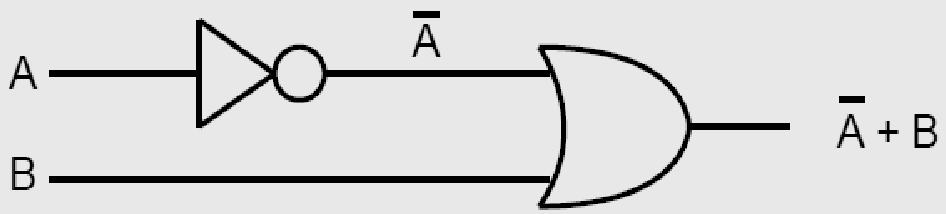
- ( ) mais alta precedência
- NOT
- AND
- OR mais baixa precedência

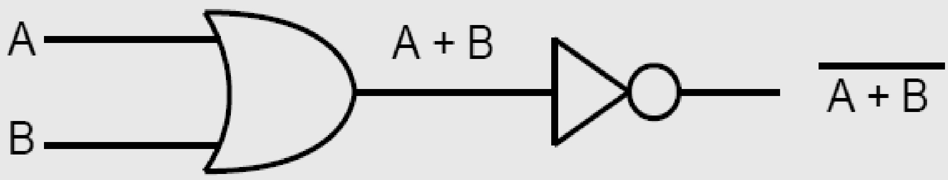
Descrevendo Circuitos Lógicos Algebricamente

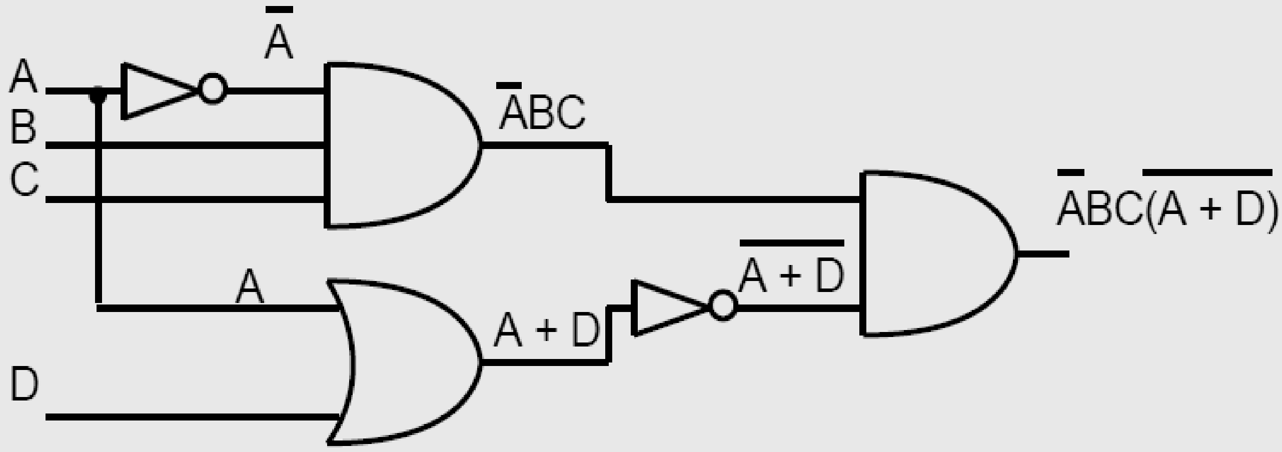




Circuitos com Inversores

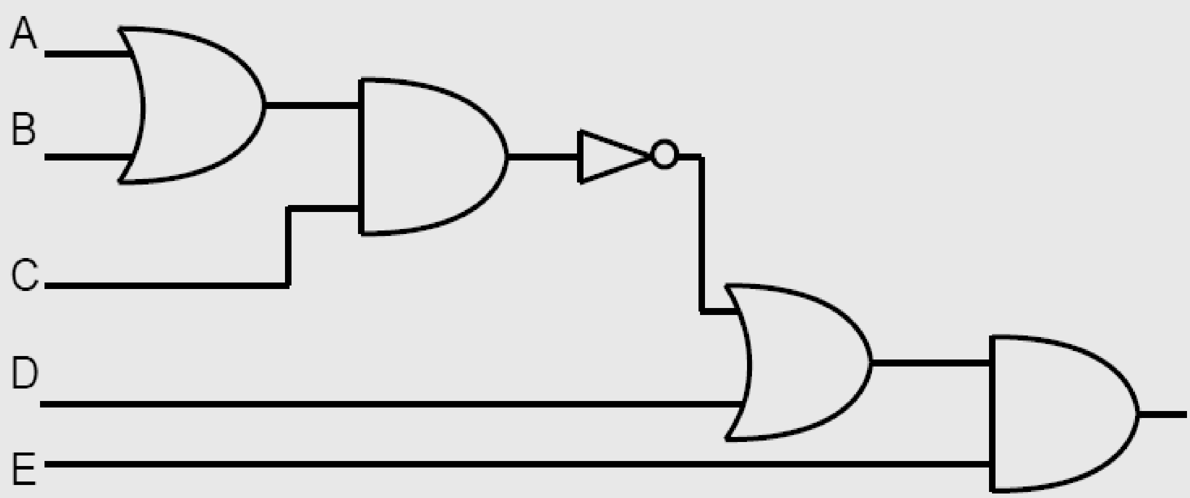






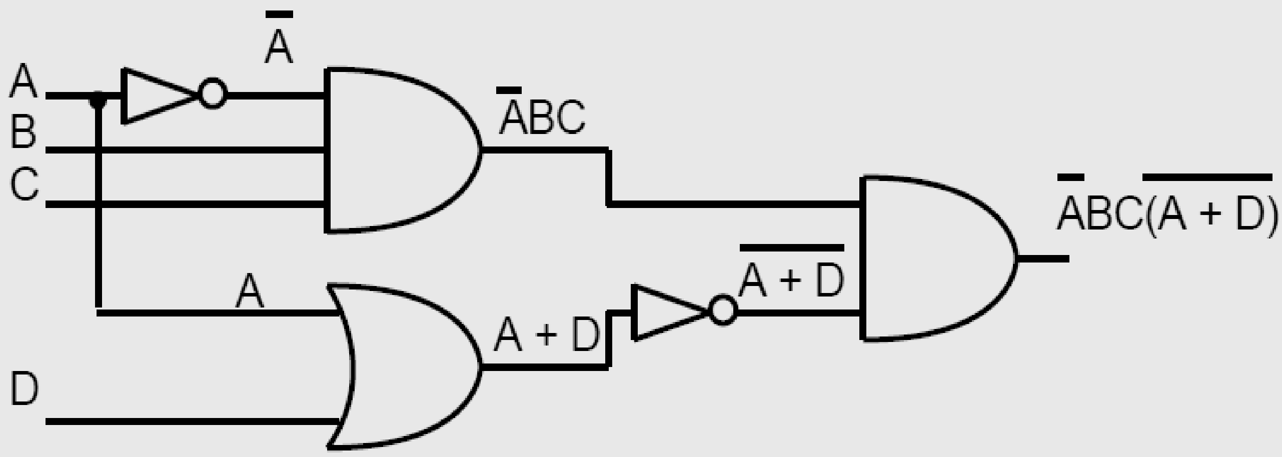
Prática

- Qual a expressão na saída do circuito abaixo?



Avaliando Saídas dos Circuitos

- Supondo que se queira saber o nível da saída de um circuito tendo os níveis de entrada das variáveis (sinais)
- Se os valores das entradas forem:
  - A=0
  - B=1
  - C=1
  - D=1



Avaliando Saídas dos Circuitos

- Para a expressão $x = \overline{A}BC\overline{(A + D)}$ teremos

$$
\begin{aligned}
x &= \overline{A}.B.C.\overline{(A + D)} \\
&= \overline{0}.1.1.\overline{(0 + 1)} \\
&= 1.1.1.\overline{(0 + 1)} \\
&= 1.1.1.(\overline{1}) \\
&= 1.1.1.0 \\
&= 0
\end{aligned}
$$

Tabela Verdade

- A tabela verdade da expressão booleana $x=\overline{A}BC\overline{(A + D)}$ é:

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

Tabela Verdade

- A tabela verdade da expressão booleana $x=\overline{A}BC(\overline{A+D})$ é:

A	B	C	D	$\overline{A}$
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
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	0

Tabela Verdade

- A tabela verdade da expressão booleana $x=\overline{A}BC(\overline{A+D})$ é:

A	B	C	D	$\overline{A}$	A+D
0	0	0	0	1	0
0	0	0	1	1	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	1	0
0	1	0	1	1	1
0	1	1	0	1	0
0	1	1	1	1	1
1	0	0	0	0	1
1	0	0	1	0	1
1	0	1	0	0	1
1	0	1	1	0	1
1	1	0	0	0	1
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	0	1

Tabela Verdade

- A tabela verdade da expressão booleana $x=\overline{A}BC(\overline{A + D})$ é:

A	B	C	D	$\overline{A}$	A+D	$\overline{A+D}$
0	0	0	0	1	0	1
0	0	0	1	1	1	0
0	0	1	0	1	0	1
0	0	1	1	1	1	0
0	1	0	0	1	0	1
0	1	0	1	1	1	0
0	1	1	0	1	0	1
0	1	1	1	1	1	0
1	0	0	0	0	1	0
1	0	0	1	0	1	0
1	0	1	0	0	1	0
1	0	1	1	0	1	0
1	1	0	0	0	1	0
1	1	0	1	0	1	0
1	1	1	0	0	1	0
1	1	1	1	0	1	0

Tabela Verdade

- A tabela verdade da expressão booleana $x=\overline{A}BC(\overline{A + D})$ é:

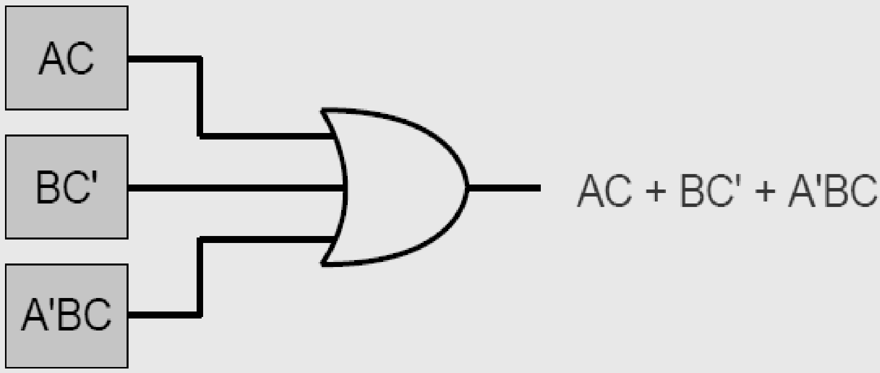
A	B	C	D	$\overline{A}$	A+D	$\overline{A+D}$	$\overline{A}BC(\overline{A+D})$
0	0	0	0	1	0	1	0
0	0	0	1	1	1	0	0
0	0	1	0	1	0	1	0
0	0	1	1	1	1	0	0
0	1	0	0	1	0	1	0
0	1	0	1	1	1	0	0
0	1	1	0	1	0	1	1
0	1	1	1	1	1	0	0
1	0	0	0	0	1	0	0
1	0	0	1	0	1	0	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	0	0
1	1	0	0	0	1	0	0
1	1	0	1	0	1	0	0
1	1	1	0	0	1	0	0
1	1	1	1	0	1	0	0

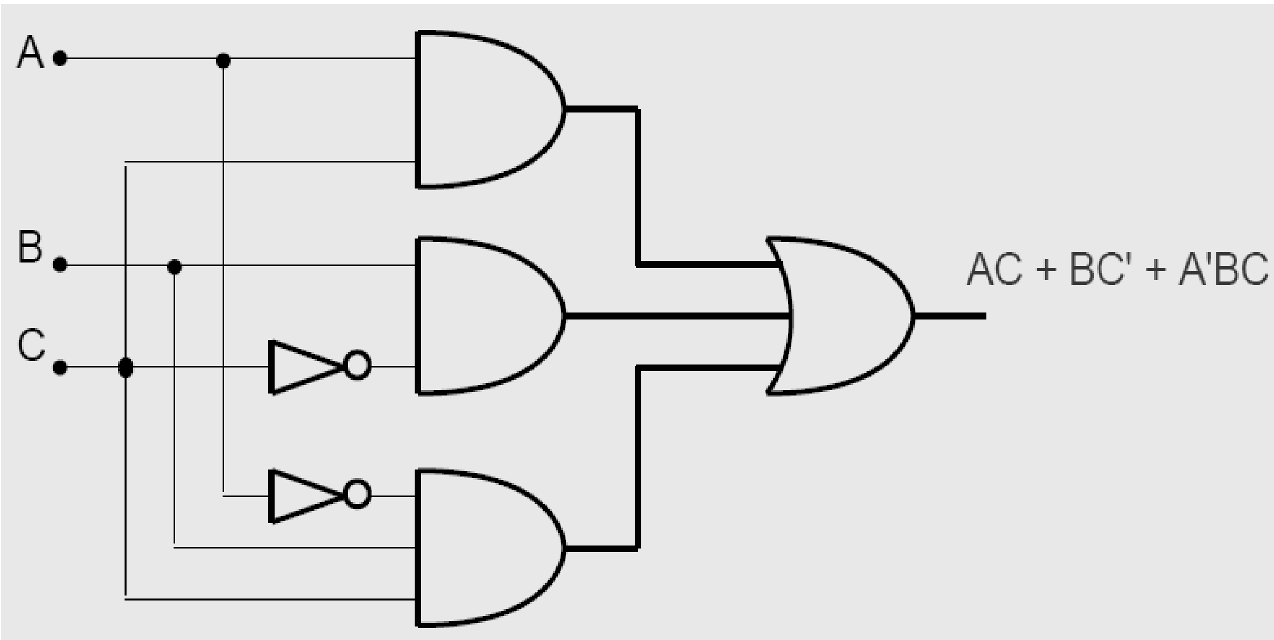
Implementando Circuitos a partir de Expressões Booleanas

- Peguemos uma expressão como exemplo

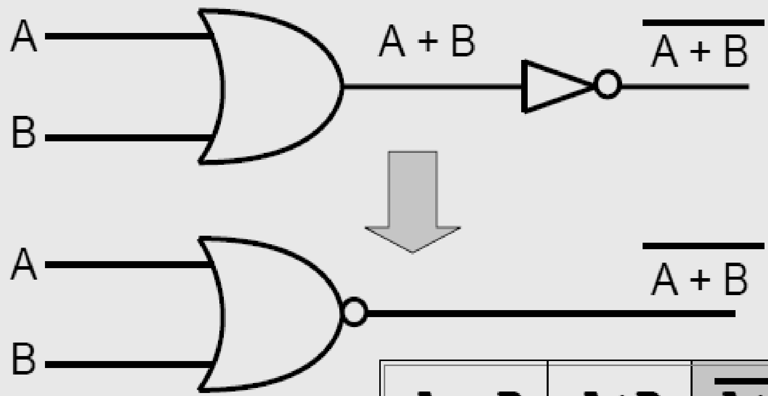
$y = AC + BC' + A'BC$

uma porta OR com três entradas



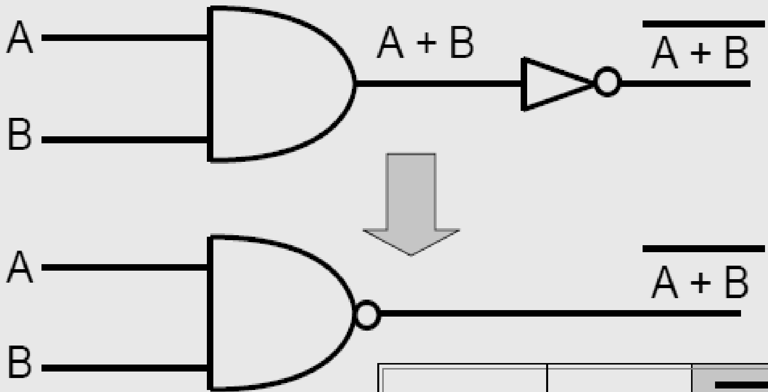


Porta NOR



A	B	A+B	$\overline{A+B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Porta NAND



A	B	A.B	$\overline{A.B}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Simplificação Algébrica

Um problema comum no projeto de lógica combinacional é a simplificação da expressão lógica para a sua forma mais simples possível

A simplificação permite a redução do número de componentes, o que implica em menor consumo de energia e melhor desempenho

Exemplos:

F = A.B + A.B’ /* 1-propriedade distributiva */

F= A. (B + B’) /* 2- expressão entre parênteses =1 */

F= A. 1

F = A

Teoremas Booleanos

(1) $x.0 = 0$

(2) $x.1 = x$

(3) $x.x = x$

(4) $x.\bar{x} = 0$

(5) $x+0 = x$

(6) $x+1 = 1$

(7) $x+x = x$

(8) $x+\bar{x} = 1$

(9) $x+y = y+x$

(10) $x.y = y.x$

(11) $x+(y+z) = (x+y)+z$

(12) $x.(y.z) = (x.y).z$

(13) $x(y + z) = xy + xz$

(14) $x + xy = x$

(15) $x + \bar{x}y = x + y$

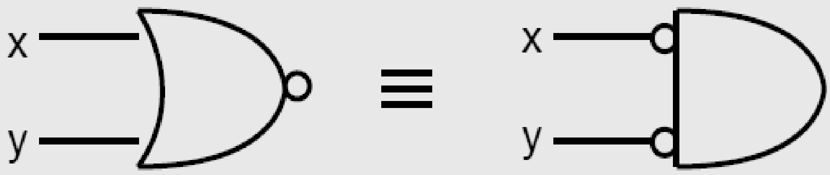
(16) $\bar{x} + xy = \bar{x} + y$

(17) $(w+x)(y+z) = wy + wz + xy + xz$

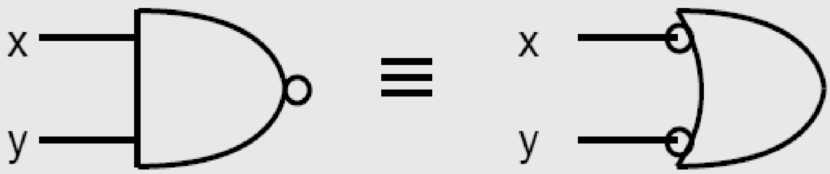
Teoremas de DeMorgan

- Dois dos mais importantes teoremas da álgebra booleana

(18) $\overline{(x + y)} = \bar{x} \cdot \bar{y}$



(19) $\overline{(x \cdot y)} = \bar{x} + \bar{y}$



Simplificação de Expressão Booleana

- Simplificar a expressão

$$z = \overline{(\bar{A} + C) \cdot (B + \bar{D})}$$

considerando (A'+C) = x e (B+D') = y, teremos

z = (x . y)'

aplicando DeMorgan a esta expressão teremos

z = (x' + y')

ou

$$z = \overline{(\bar{A} + C)} + \overline{(B + \bar{D})}$$

Simplificação de Expressão Booleana (cont)

$$z = \overline{(\overline{A} + C)} + \overline{(B + \overline{D})}$$

aplicando DeMorgan novamente em cada parcela teremos

$$z = (\overline{\overline{A}} \,.\, \overline{C}) + (\overline{B} \,.\, \overline{\overline{D}})$$

que é:

$$z = (A \,.\, \overline{C}) + (\overline{B} \,.\, D)$$

ou

$$\boxed{z = A\overline{C} + \overline{B}D}$$

Simplificação Algébrica

Exemplo
H = (A+B+C). (A+B) /*propriedade distributiva */
H = A.A + A.B + A.C + A.B + B.B + B.C
H = A + A.B + B + A.C + A.B + B.C /*A +A.B = A*/
H = A + B + C.A + B.C
H = (A + C.A) + (B + B.C)
H = A + B

Exemplo
T= A. B + A.B.C + A.B.C’ /*distributiva */
T = A.B + A.B.(C + C’)
T = A.B + A.B.(1)
T = A.B + A.B
T = A.B

Simplificação Algébrica

Exemplo
F = A.B.C + B.C /* agrupa-se o fator B.C*/
F = A. (B.C) + (B.C) /* propriedade distributiva*/
F = (A +1) . (B.C) /* A + 1 = 1*/
F = 1.(B.C)
F = B.C

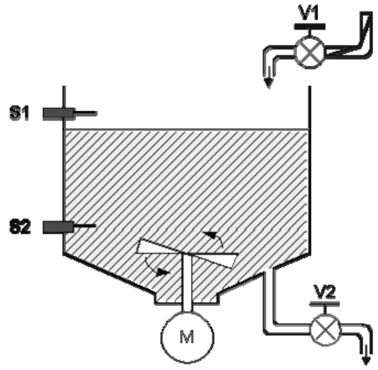
Exemplo
G = ((A + B’ + C)+ (B + C’))’ /* Morgan */
G = (A + B’+ C)’ . (B + C’)’ /* Morgan */
G = A’. B’’. C’ . B’ . C’’
G = A’. B . C’ . B’. C
G = A’. (B. B’). (C.C’)
G= 0

Circuitos sequenciais

Circuitos sequenciais assíncronos

Exemplo - Tanque

- O líquido no tanque deve ser mantido entre os limites S_1 e S_2, abrindo ou fechando V_1.
  - Se considerarmos que os sensores $\mathbf{S_1}$ e $\mathbf{S_2}$ tem valor 1 sempre que em contacto com o líquido. A tabela exprime o que desejamos do circuito de controlo.
  - Da tabela constatamos que para a combinação de valores de entrada $S_2 = 0$, $S_1= 1$ a saída pode tomar dois valores distintos.
  - Se analisarmos o problema como um circuito combinatório, a saída é função das entradas (S1 e S2), ou seja sobre os valores das entradas, por aplicação dos operadores AND, OR ou NOT, obtemos o valor da saída. A aplicação destes operadores conduz sempre ao mesmo valor de saída, para a mesma combinação de entradas.
  - Como resolver a incongruência?



S1	S2	V1	
1	0	X	Impossível
0	0	1	Tanque Vazio, encher
0	1	1	A encher, deve continuar até S1=1
1	1	0	Tanque cheio, fechar V1
0	1	0	A esvaziar, manter V1 fechado

Circuitos sequenciais

Circuitos sequenciais assíncronos

Exemplo - Tanque

- O comportamento do sistema não é aleatório:
  - Se tanque está a encher, continua até atingir
  - Se está a esvaziar, continua até atingir S_2.
- O comportamento do nosso sistema depende de:
  - **S1 e S2** (Variáveis externas).
  - **Do estado**, “a encher” “a esvaziar”.
- Representar o estado por uma variáve à qual atribuímos
  - 0 “a esvaziar”,
  - 1 “a encher”
- O sistema é agora um circuito combinatório definido pela tabela de verdade, pode ser tratado com as ferramentas já estudadas.

y	S1	S2	V1	
0	0	0	1	Tanque vazio, enche
0	0	1	0	A esvaziar
0	1	0	x	Impossivel
0	1	1	0	Limite máximo
1	0	0	1	A encher
1	0	1	1	A encher
1	1	0	x	Impossivel
1	1	1	0	Limite máximo