Faculdade de Computação

Campus de Castanhal

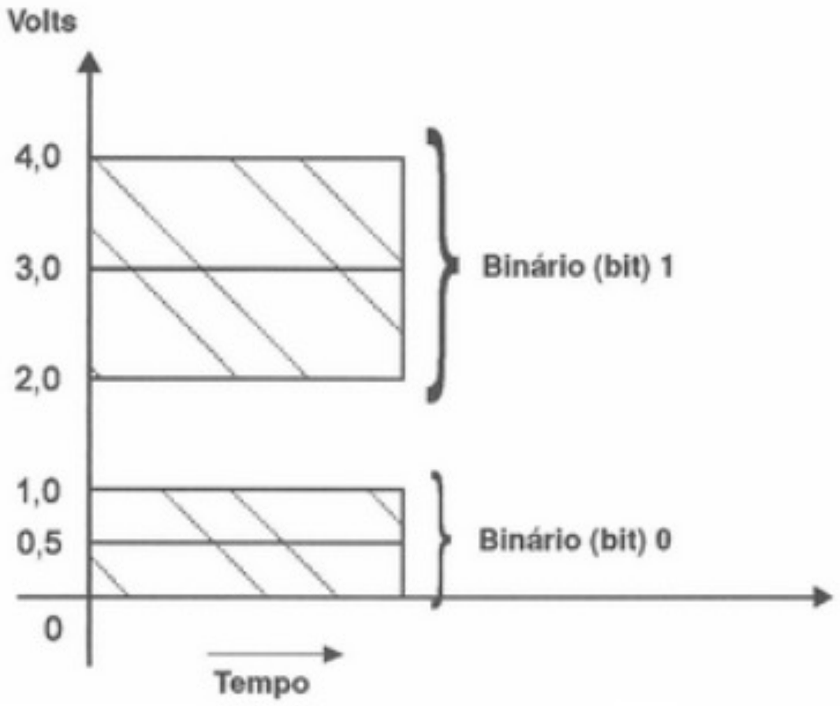
Eletrônica Digital

Prof. Dra. Evelin Cardoso

Aula 02 – Portas Lógicas e Introdução à álgebra booleana

Constantes e variáveis booleanas

- A principal diferença entre a álgebra booleana e a convencional é que, na booleana, as constantes e variáveis podem ter apenas dois valores possíveis, 0 ou 1. As variáveis booleanas são muitas vezes usadas para representar o nível de tensão presente em uma conexão ou em terminais de entrada/saída de um circuito.
- Por exemplo, em um determinado sistema digital, o valor booleano 0 pode representar qualquer tensão dentro da faixa de 0 a 1 V, enquanto o valor booleano 1 pode representar qualquer tensão dentro da faixa de 2 a 4 V, como no exemplo.



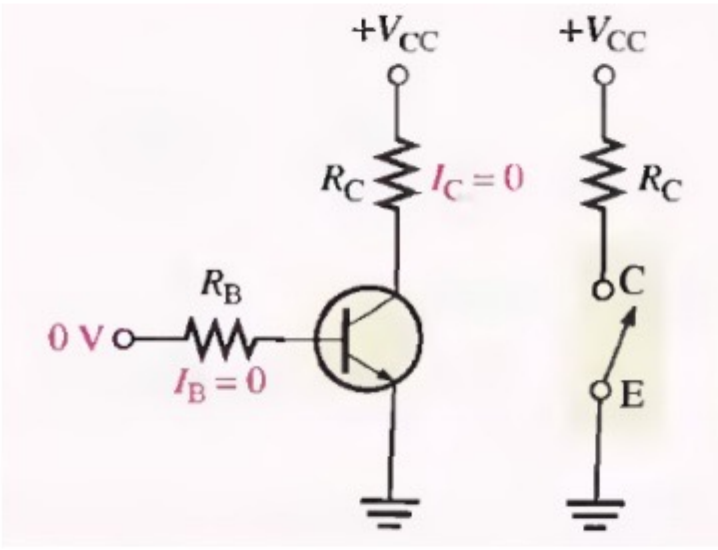
Exemplo de um sinal binário

O transistor

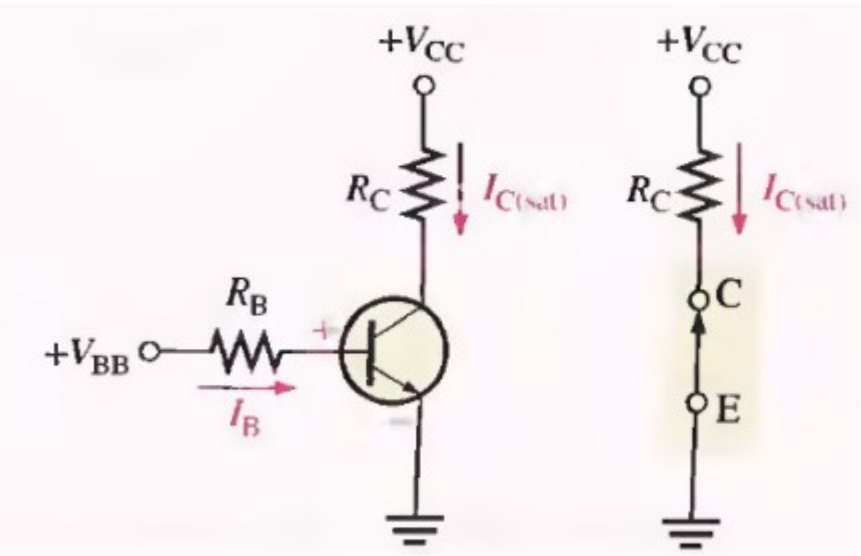
- O transistor é responsável pela amplificação de sinal elétrico, além de servir como um controlador que interrompe ou libera a passagem de corrente elétrica;
- Funcionamento

Todo transistor possui três terminais, que você pode ver na imagem abaixo.

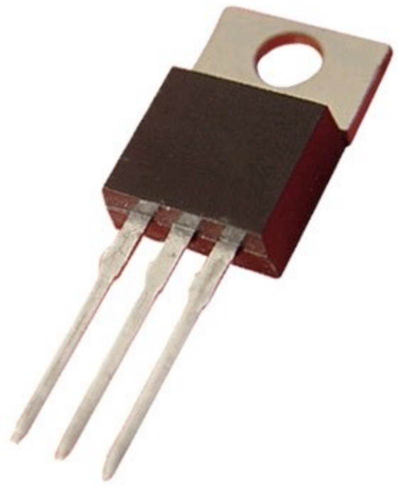
Um dos terminais recebe a tensão elétrica e o outro envia o sinal amplificado. O terminal do meio é o responsável pelo controle desse processo, pois a corrente elétrica entra e sai pelos outros dois terminais somente quando é aplicada tensão elétrica ao terminal do meio.



Operando na região de corte



Operando na região de saturação



Exemplo de transistor

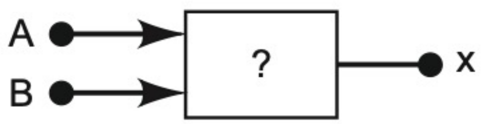
Portas lógicas

- Internamente, um computador moderno é constituído de elementos eletrônicos, tais como capacitores, resistores e **transistores.**
- Esses elementos são conhecidos como Portas Lógicas, por permitirem ou não a passagem de sinais.
- São a parte fundamental de qualquer circuito digital, do mais simples ao mais complexo.
- Em geral, os transistores são componentes de determinados circuitos eletrônicos que precisam armazenar os sinais binários e realizar certos tipos de operações com eles.
- Esses circuitos são chamados de **circuitos digitais**

Trabalham apenas com grandezas binárias

Trabalham com entradas e saídas binárias

Lógico 0	Lógico 1
Falso	Verdadeiro
Desligado	Ligado
BAIXO	ALTO
Não	Sim
Aberto	Fechado



Uma porta lógica é, então, um elemento de hardware (circuito eletrônico) que recebe um ou mais sinais de entrada e produz um sinal de saída, cujo valor é dependente do tipo de regra lógica estabelecida para o referido circuito.

Operações Lógicas

- A álgebra booleana tem, de fato, apenas *três* operações básicas: *OR* (*OU*), *AND* (*E*) e *NOT* (*NÃO*).
- Essas operações básicas são denominadas ***operações lógicas***.
- Os circuitos digitais, denominados *portas lógicas,* podem ser construídos a partir de diodos, transistores e resistores interconectados de modo que a saída do circuito seja o resultado de uma operação lógica básica (*OR, AND* ou *NOT*) realizada sobre as entradas.
- **Uma operação lógica (booleana) realizada sobre um ou mais valores lógicos (na entrada) produz um certo resultado (na saída), também lógico, conforme a regra definida para aquela operação**
- Portanto, esses circuitos possuem uma ou mais tensões de entrada, mas somente uma tensão de saída. **Os valores possíveis das tensões de entrada e da tensão de saída são somente dois: Tensão de alimentação do circuito e Tensão nula ou terra (GND).**

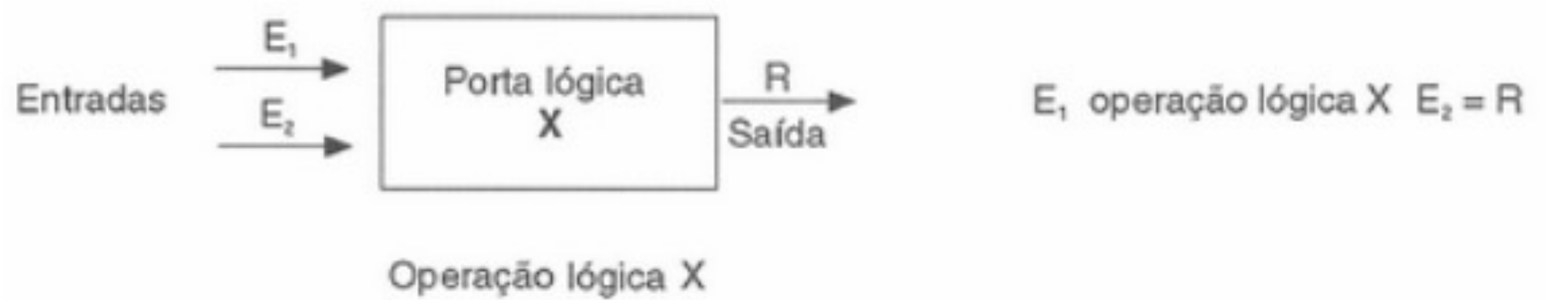


Tabela-verdade

- Se as variáveis de entrada só podem assumir os valores 0 (falso) ou 1 (verdadeiro) e o resultado (saída) também, então podemos definir previamente todos os possíveis valores de resultado de uma dada operação lógica conforme a combinação possível de valores de entrada.
- Essas possibilidades são representadas de forma tabular e o conjunto se chama **Tabela-verdade.**
- Cada operação lógica possui sua própria Tabela-verdade, estabelecida de acordo com a função que implementa. O número de linhas da tabela é igual ao número de combinações possíveis de valores de entrada.
- De um modo geral, a tabela-verdade possui 2^n linhas ou combinações de valores de entrada, sendo n igual à quantidade de elementos de entrada.

Saída

Entradas

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

Tabela-verdade

- Se o circuito lógico possui apenas uma entrada, então a saída só pode assumir dois valores
- Nesse caso, tabela-verdade possuiria duas linhas: uma para a entrada igual a 0 e outra para a entrada igual a 1.
- Se houvesse duas entradas, haveriam quatro possíveis combinações na saída e a tabela possuiria quatro linhas

- **Quantas linhas teria uma tabela-verdade com 4 valores de entrada?**
- **E se houvesse 8 valores na entrada?**

Entradas → A, B; Saída → x

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

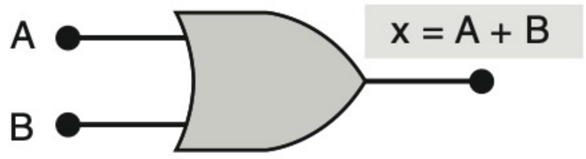
A	B	C	D	x
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

OPERAÇÃO OR ('OU') COM PORTA OR

- A tabela-verdade na Figura 3.2(a) mostra o que acontece quando duas entradas lógicas, *A* e *B*, são combinadas usando uma operação OR para produzir a saída *x*.
- A tabela mostra que *x* será um nível lógico 1 para cada combinação de níveis de entradas em que uma *ou* mais entradas forem 1. O único caso em que *x* é um nível 0 acontece quando ambas as entradas são 0.

OR

A	B	$x = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

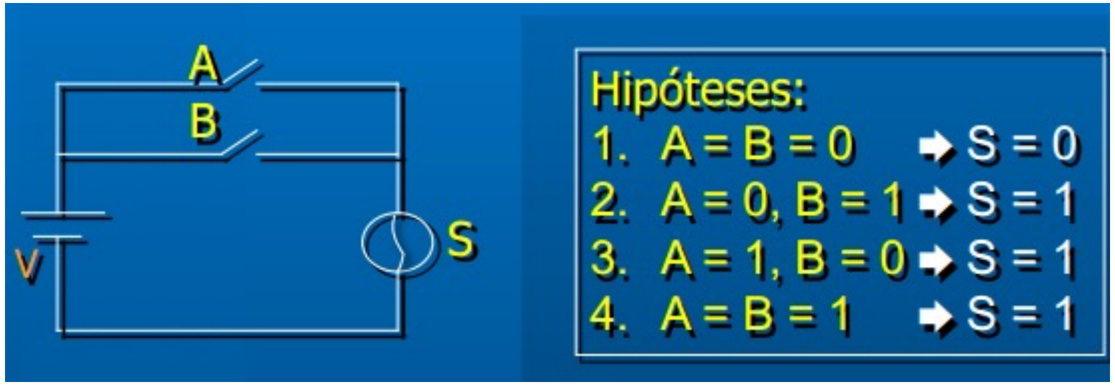


Porta OR

Nessa expressão, o sinal '+' não representa a adição convencional; ele representa a operação OR. Essa operação é semelhante à operação convencional de adição, exceto para o caso em que *A* e *B* forem 1

OPERAÇÃO OR (‘OU’) COM PORTA OR

- Exemplo 1: circuito para acender uma lâmpada:

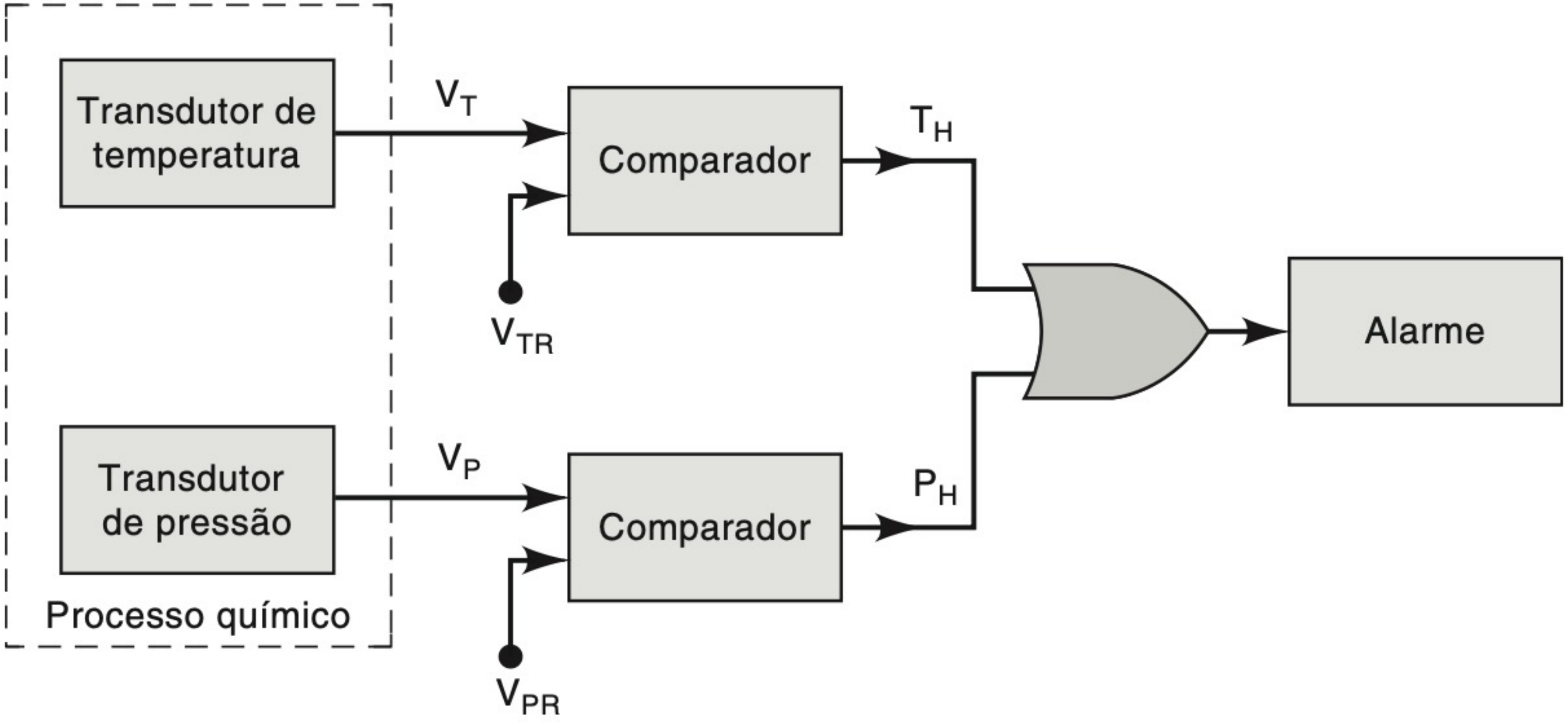


Exemplo 2: trecho de um programa

- SE (T > 6 **OU** (OR) R < 10)
  - ENTÃO IMPRIMIR T
  - SENÃO IMPRIMIR R

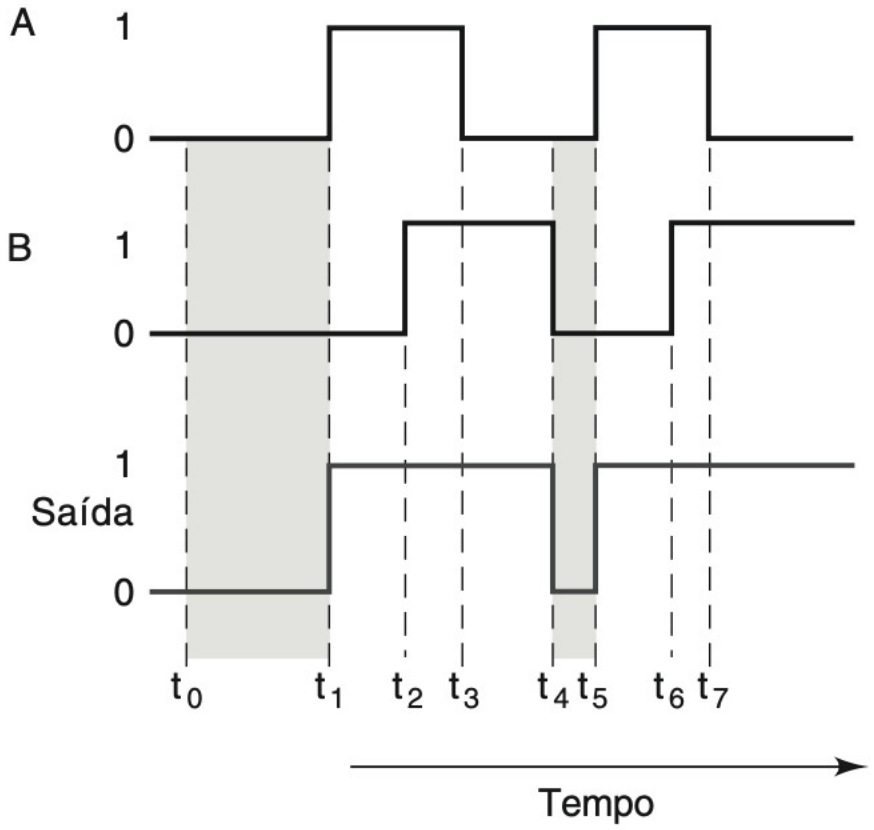
OPERAÇÃO OR ('OU') COM PORTA OR

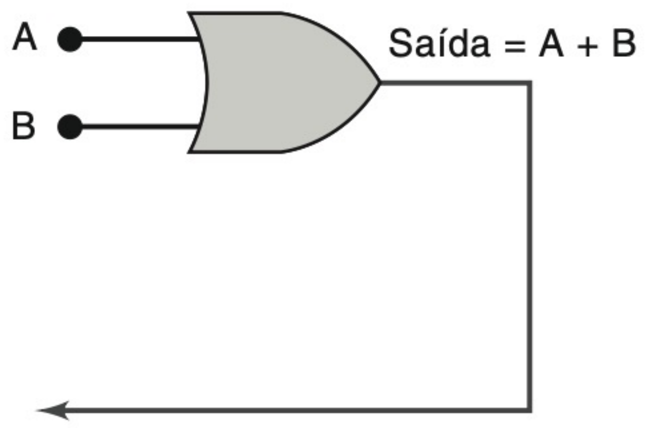
- Exemplo 3: sistema de alarme



OPERAÇÃO OR ('OU') COM PORTA OR

- Exemplo 4:





OPERAÇÃO OR ('OU') COM PORTA OR

- Exercício 1: Seja A = 1 e B = 0, calcular X=A+B (A or B)
- Exercício 2: Seja A = 0110 e B = 1110, calcular X=A+B (A or B)
- Exercício 3: Seja A = 1100, B = 1111 e C = 0001, calcular X=A+B+C (A or B or C)

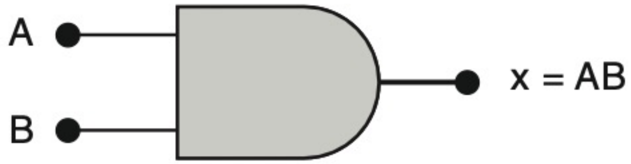
OPERAÇÃO AND (‘E’) COM PORTA AND

- A tabela-verdade mostra o que acontece quando duas entradas lógicas, *A* e *B*, são combinadas usando uma operação AND para gerar a saída *x*. A tabela mostra que *x* será nível lógico 1 apenas quando *A* e *B* forem 1. Para qualquer outro caso em que uma das entradas for 0, a saída será 0.

$$x = A \cdot B$$

AND

A	B	$x = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

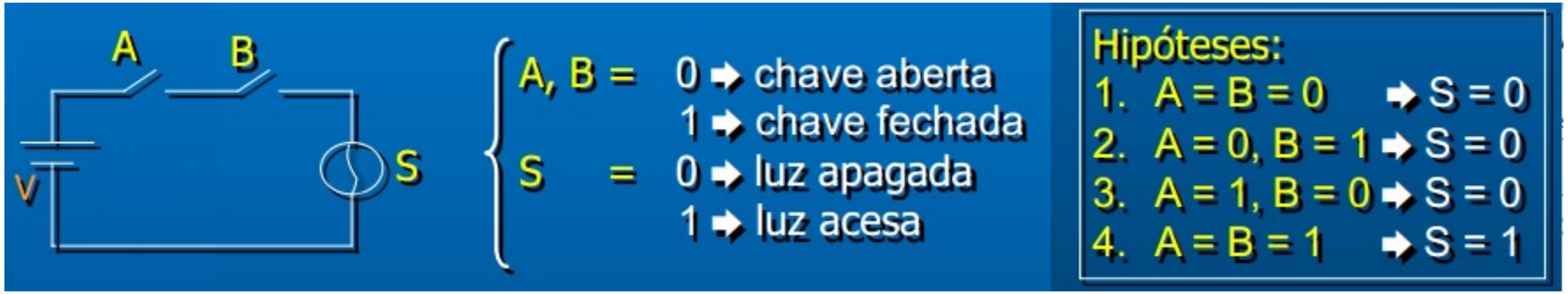


Porta AND

Nessa expressão, o sinal (·) representa a operação booleana AND; não é multiplicação. Entretanto, a operação AND sobre variáveis booleanas equivale à multiplicação convencional, conforme análise da tabela-verdade mostrada

OPERAÇÃO AND (‘E’) COM PORTA AND

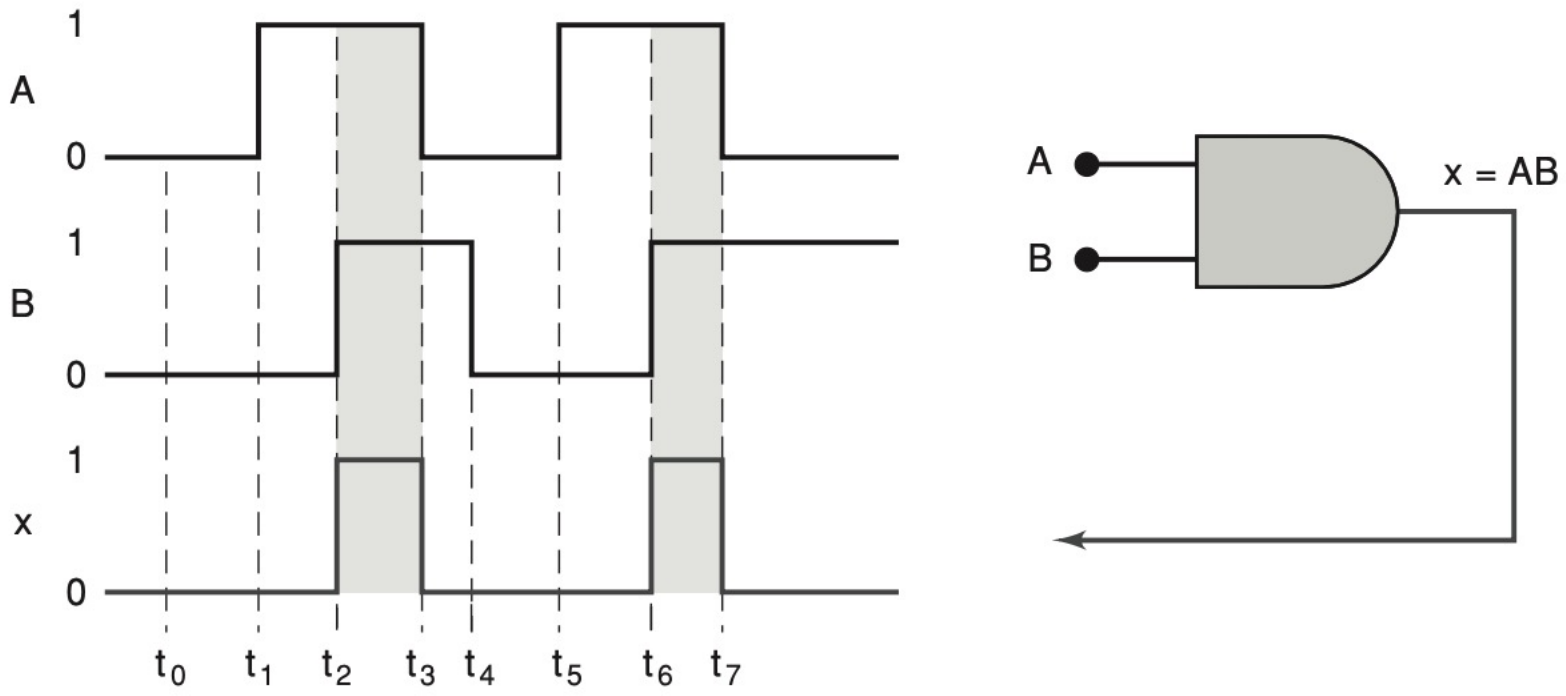
- Exemplo 1: circuito para acender uma lâmpada



- Exemplo 2: lógica de um determinado programa
  - Ler X, Y e Z
  - T = X + Y
  - R = Z + X
  - SE (T > 6 E(AND) R < 10)
    - ENTÃO IMPRIMIR T
    - SENÃO IMPRIMIR R

OPERAÇÃO AND (‘E’) COM PORTA AND

- Exemplo 3



OPERAÇÃO AND ('E') COM PORTA AND

- Exercício 4: Seja A =1 e B = 0, calcular X = A . B ou seja (A and B)
- Exercício 5: Seja A = 0101 e B = 1101, calcular X = A . B ou seja (A and B)
- Exercício 6: Seja A = 0101, B = 0011 e C = 1111, calcular x=A.B.C (A and B and C)

OPERAÇÃO NOT ('NÃO') OU INVERSÃO

- A **operação NOT**, também denominada **INVERSÃO**, é diferente das operações OR e AND pelo fato de poder ser realizada sobre uma única variável de entrada. Por exemplo, se a variável *A* for submetida à operação de inversão, o resultado *x* pode ser expresso como

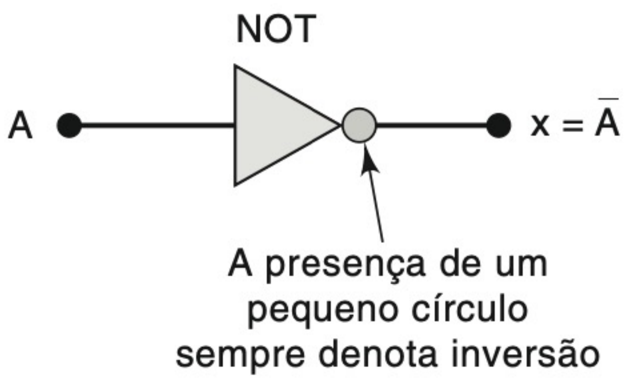
$$x = \overline{A}$$

onde a barra sobre o nome da variável representa a operação de inversão. Essa expressão é lida como '*x* é igual a *A* negado', o '*x* é igual ao *inverso* de *A*' ou '*x* é igual ao *complemento* de *A*'.

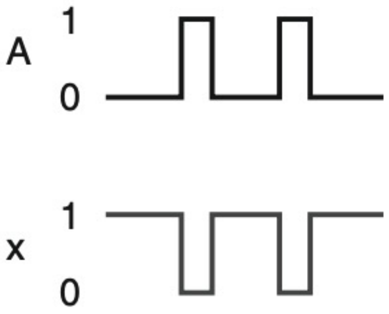
NOT

A	$x = \overline{A}$
0	1
1	0

(a)



(b)



(c)

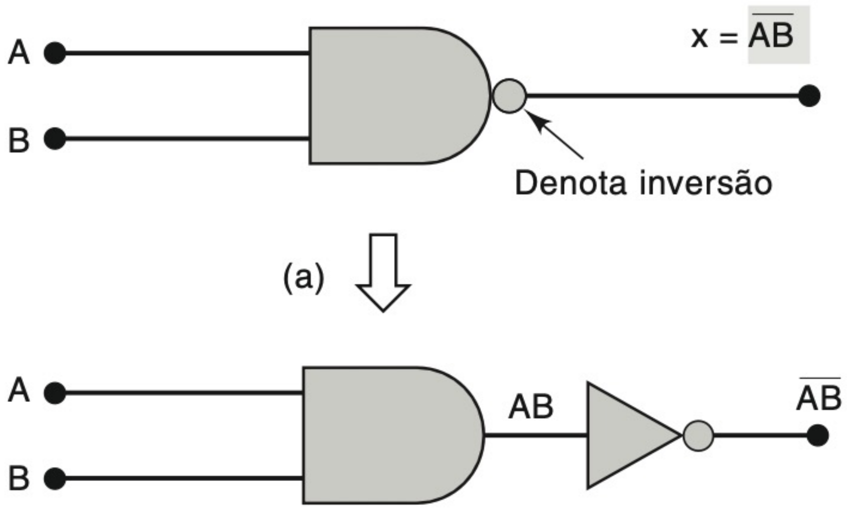
Ele inverte (complementa) o sinal de entrada em todos os pontos da forma de onda, de maneira que se a entrada = 0, a saída = 1, e vice-versa.

OPERAÇÃO NOT ('NÃO') OU INVERSÃO

- Exercício 7: Seja A = 0, calcular $X = \overline{A}$
- Exercício 8: Seja A = 10011, calcular $X = \overline{A}$
- Exercício 9: Seja A = 10010 e B = 11110, calcular $X = \overline{A.B}$

Operação lógica NAND (NOT AND)

- É definida como o complemento da porta AND
- A saída de um circuito lógico NAND é obtida ao se aplicar a regra da operação AND e inverter o resultado
- A porta NAND produzirá uma saída falsa ***se e somente se todas as entradas forem verdade***. Do contrário, a saída será verdade, se e pelo menos uma entrada for falsa.



(a)
(b)

		AND	NAND
A	B	AB	$\overline{AB}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

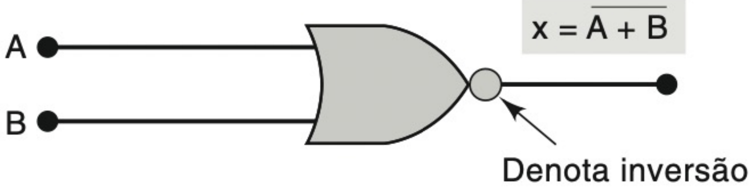
(c)

Operação lógica NAND (NOT AND)

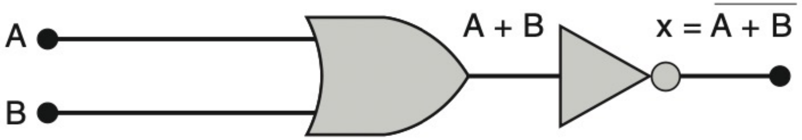
- Exercício 10: Seja A = 1 e B = 1, calcular o valor de X em $X = \overline{A \,.B}$
- Exercício 11: Seja A = 10110 e B = 00011, calcular o valor de X em $X = \overline{A \,.\, B}$
- Exercício 12: Seja A = 11110, B = 01001 e C = 10000, calcular $X = \overline{A \,.\, B \,.\, C}$

Operação lógica NOR (NOT OR)

- A porta NOR é o complemento ou inverso da porta OR
- A saída é obtida ao se efetuar a operação lógica OR sobre as entradas e inverter o resultado
- A porta NOR apresentará uma saída verdade ***se e somente se todas as entradas forem falsas***. Caso contrário, a saída será falsa se, pelo menos, uma das entradas for verdade.



(a)



(b)

		OR	NOR
A	B	$A + B$	$\overline{A + B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

(c)

Operação lógica NOR (NOT OR)

Exercício 13: Seja A =1 e B = 0, calcular o valor de X e Y nas expressões:

a) $X = \overline{A + B}$

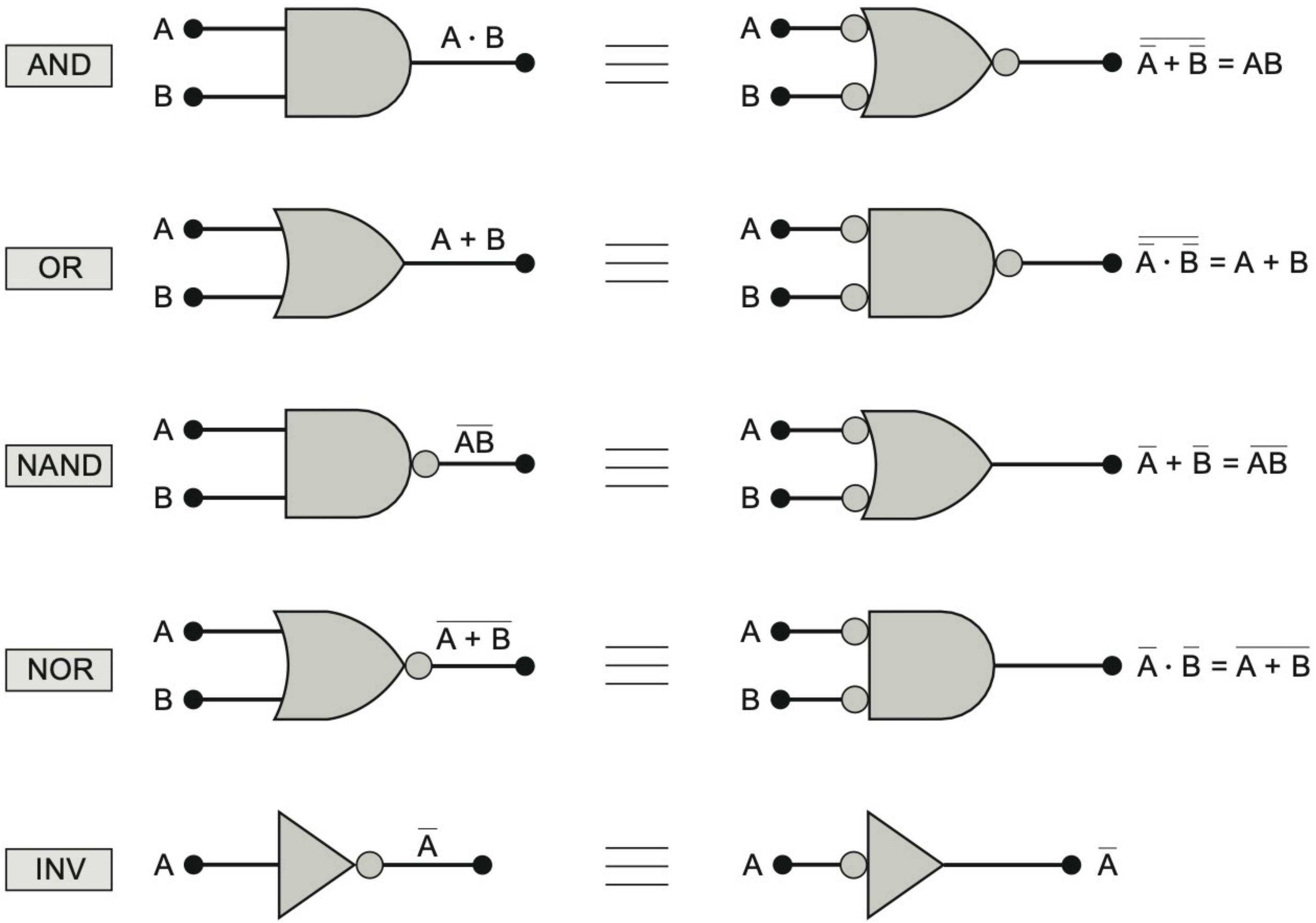
b) $Y = \overline{A} + \overline{B}$

Exercício 14: A = 0 e B = 1, calcular $X = \overline{A + B}$

Exercício 15: Seja A = 10001 e B = 01010, calcular $X = \overline{A + B}$

Exercício 16: Seja A = 11110, B = 10011 e C = 00100, calcular $X = \overline{A + B + C}$

Símbolos gráficos e matemáticos de algumas portas lógicas



Implementando circuitos a partir de expressões booleanas

- Se a operação de um circuito lógico é definida por meio de uma expressão booleana, então o **diagrama do circuito lógico** pode ser implementado diretamente a partir desta expressão.
- Por exemplo, se precisamos de um circuito que é definido pela expressão $X = ABC$, percebemos imediatamente que tudo de que precisamos é uma porta AND de três entradas.
- Se precisamos de um circuito definido por $\overline{X} = A+B$, poderíamos usar uma porta OR de duas entradas com um INVERSOR em uma de suas entradas.
- Esse mesmo raciocínio pode ser estendido para circuitos mais complexos.

Implementando circuitos a partir de expressões booleanas

- Exemplo 1: Desenhe o circuito que implementa a expressão

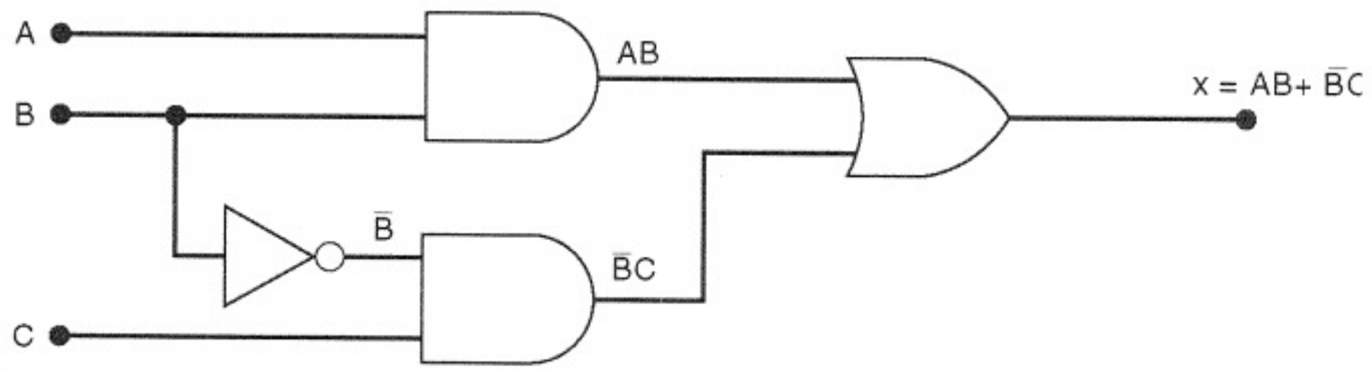
$X = AB + \overline{B}C$

Implementando circuitos a partir de expressões booleanas

- Exemplo 1: Desenhe o circuito que implementa a expressão

$X = AB + \overline{B}C$

Solução: esta expressão indica que os termos AB e $\overline{B}C$ são entradas de uma porta OR, e cada um destes termos pode ser gerado por uma porta AND.



Implementando circuitos a partir de expressões booleanas

- Exercício 1: Desenhe o circuito que implementa a expressão, usando portas lógicas com **no máximo três entradas**:

  $X = ABC(A+D)$

Implementando circuitos a partir de expressões booleanas

- Exercício 2: Desenho o circuito para a expressão $Y = \overline{A}C + BC + \overline{ABC}$

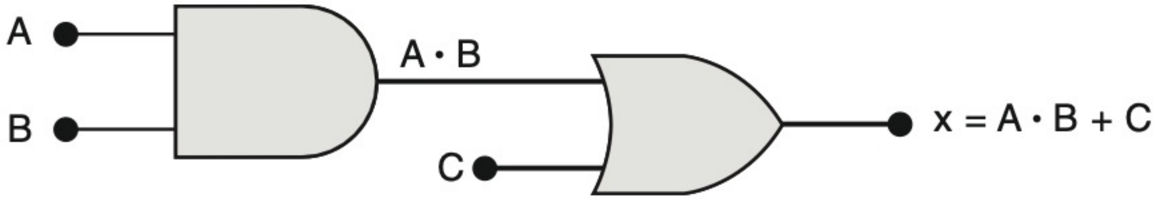
Implementando circuitos a partir de expressões booleanas

- Exercício 3: Desenhe o circuito que implementa a expressão

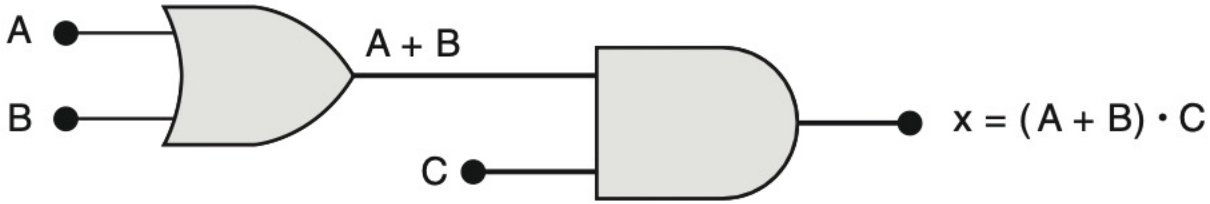
$X = [D + (\overline{A+B}).C].E$

Descrevendo circuitos lógicos algebricamente

- Qualquer circuito lógico, independente de sua complexidade, pode ser descrito por operações booleanas, as mesmas que vimos anteriormente (OR, AND e NOT), pois as operações que conhecemos na álgebra booleana são básicas e formam a base para outros teoremas que veremos posteriormente.
- Seja o circuito lógico a seguir formado por uma porta AND e uma OR, qual a expressão algébrica que o descreve?



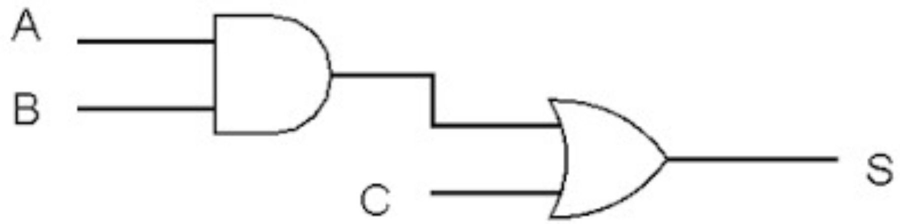
(a)



(b)

Descrevendo circuitos lógicos algebricamente

- A fim de representar um circuito lógico algebricamente, estamos interessados em **expressar a saída em função das entradas.** Assim, na figura abaixo estamos interessados na saída S em função das entradas A, B e C do **exemplo 1**.

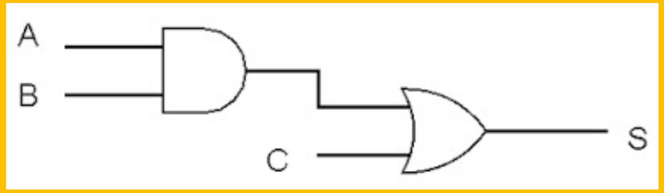


- Deve-se observar para onde as entradas estão direcionadas, ou seja, a quais portas elas estão conectadas. No exemplo:

  A e B estão sendo operados em uma porta *AND* -> A . B

  O resultado (A . B) é enviado para a porta lógica *OR* e é operado junto com a entrada C de acordo com a operação booleana *OR* -> A . B + C

- O resultado da expressão A . B + C representa a saída do circuito. Assim, podemos finalmente escrever:



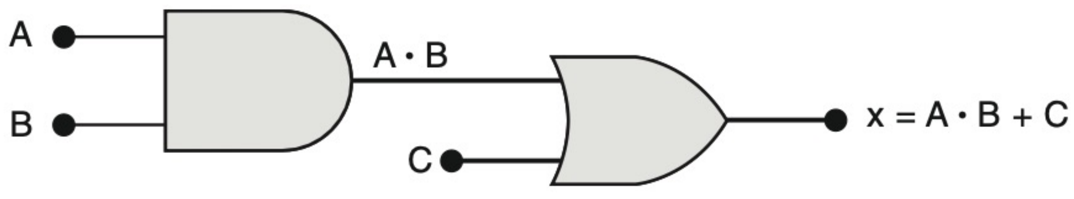
S = A . B + C

Precedência de operador

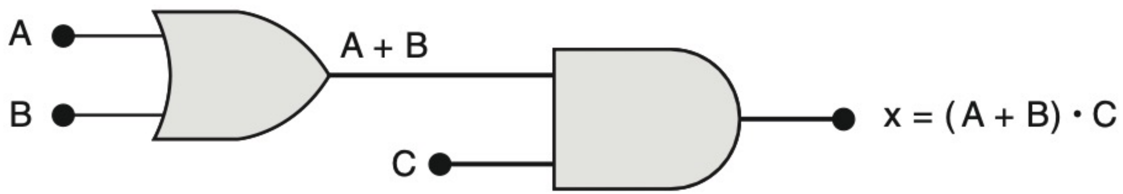
Se nós não tivéssemos conhecimento do circuito, que operação lógica seria realizado primeiro na expressão abaixo?

$$S = A \,.\, B + C$$

Observação: Fica definido que, caso o circuito possua operações AND e OR, a operação AND é realizada primeiro a não ser que haja parênteses, sendo que as operação entre parênteses tem prioridade assim como na álgebra convencional.



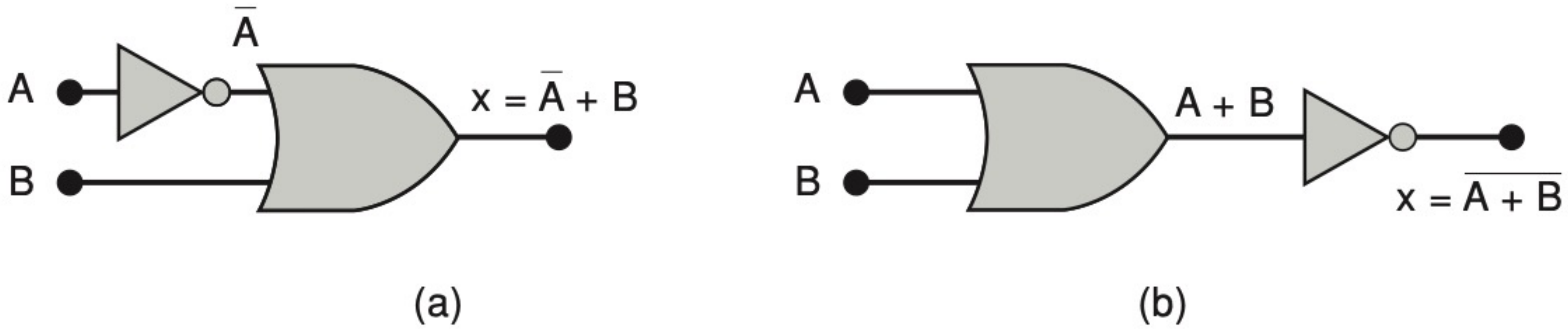
(a)



(b)

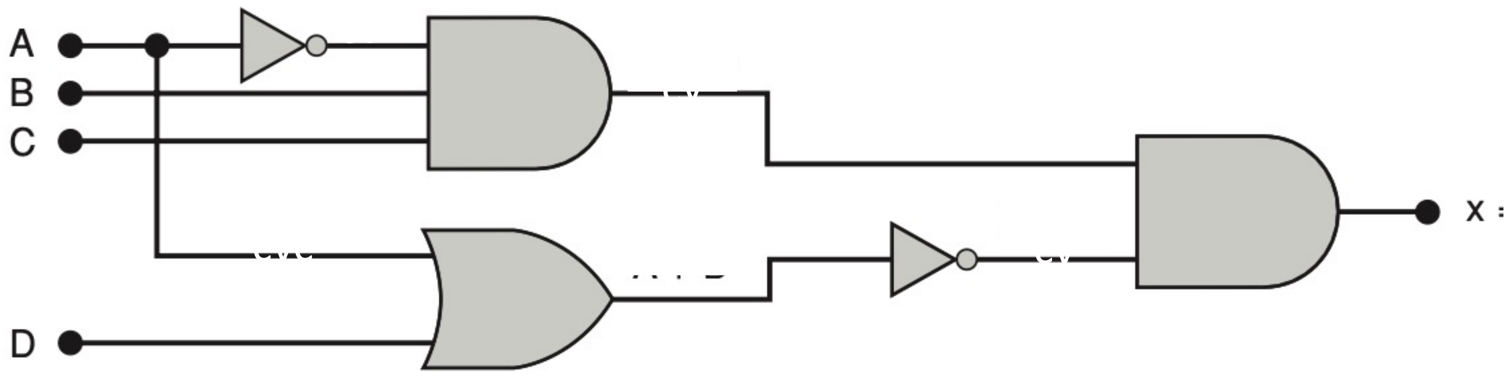
Circuitos com INVERSORES lógicos

- Sempre que um INVERSOR estiver presente em um circuito lógico, a expressão para a saída do INVERSOR será igual à expressão de entrada com uma barra sobre ela. Veja os exemplos:



Circuitos com INVERSORES lógicos

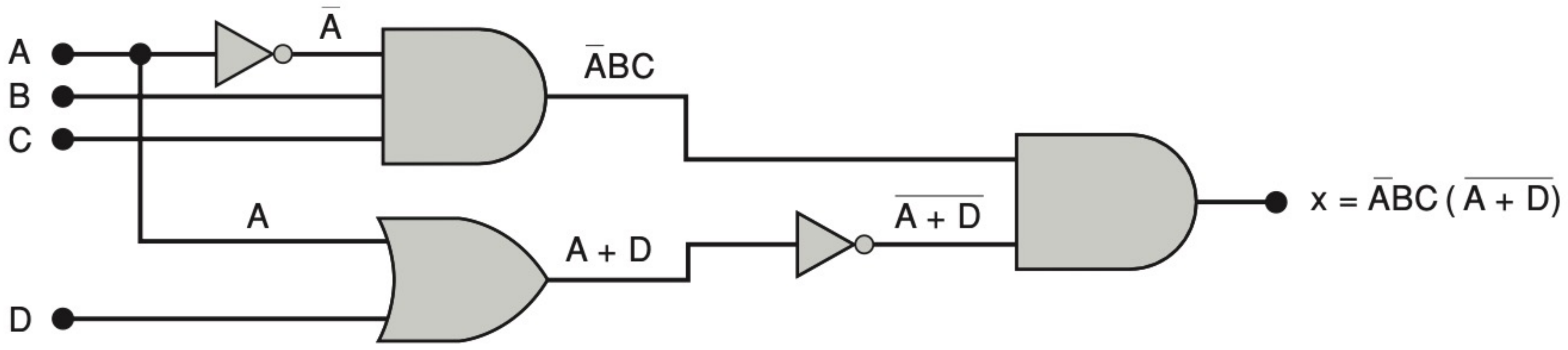
Exercício 1: Descreva a saída do circuito abaixo:



(a)

Circuitos com INVERSORES lógicos

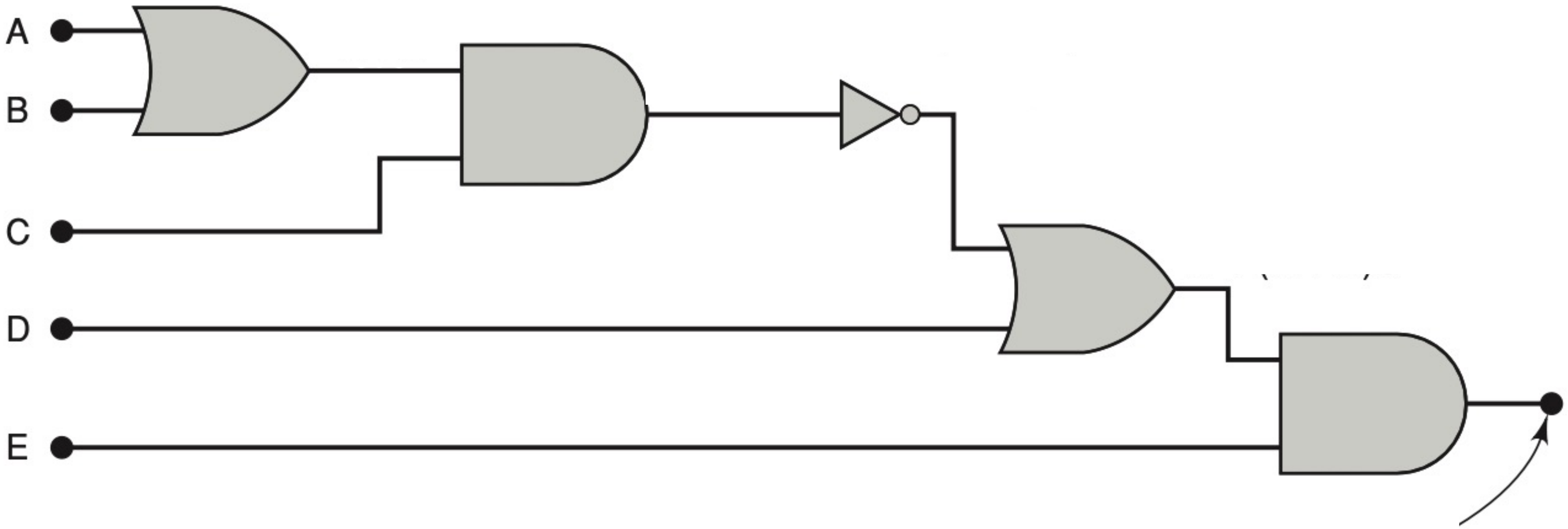
Exercício 1: Descreva a saída do circuito abaixo:



(a)

Circuitos com INVERSORES lógicos

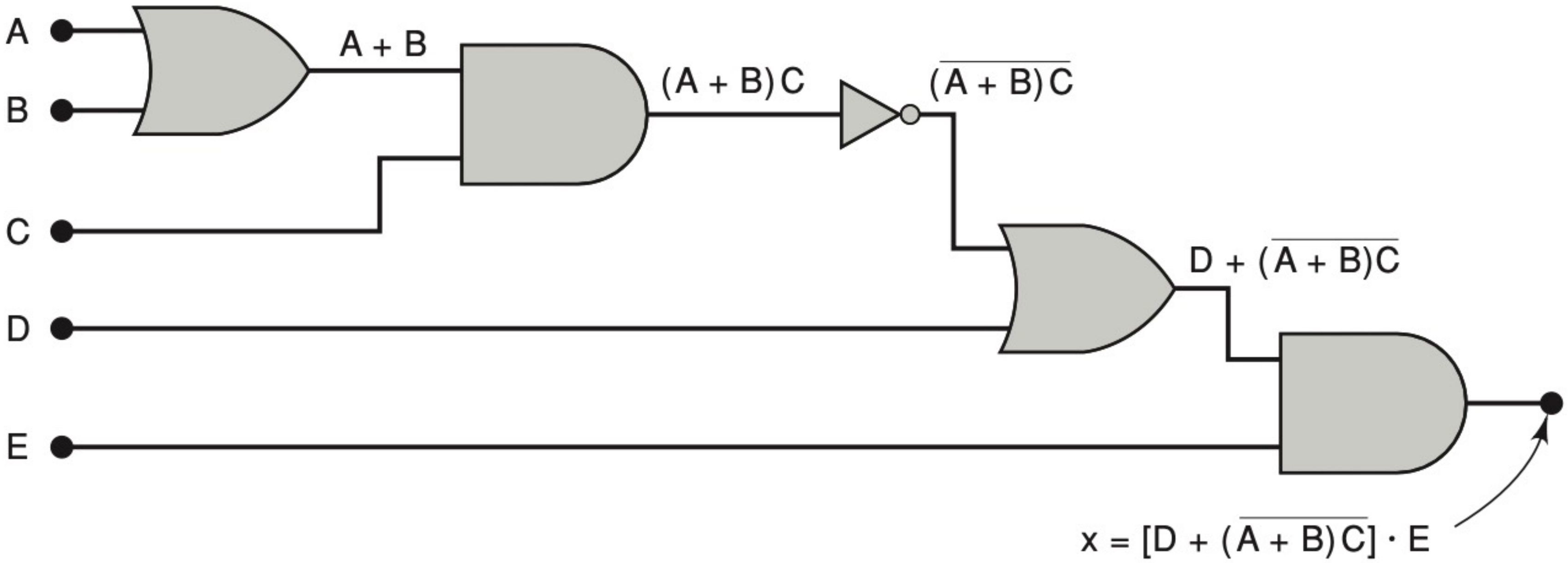
Exercício 2: Descreva a saída do circuito abaixo:



(b)

Circuitos com INVERSORES lógicos

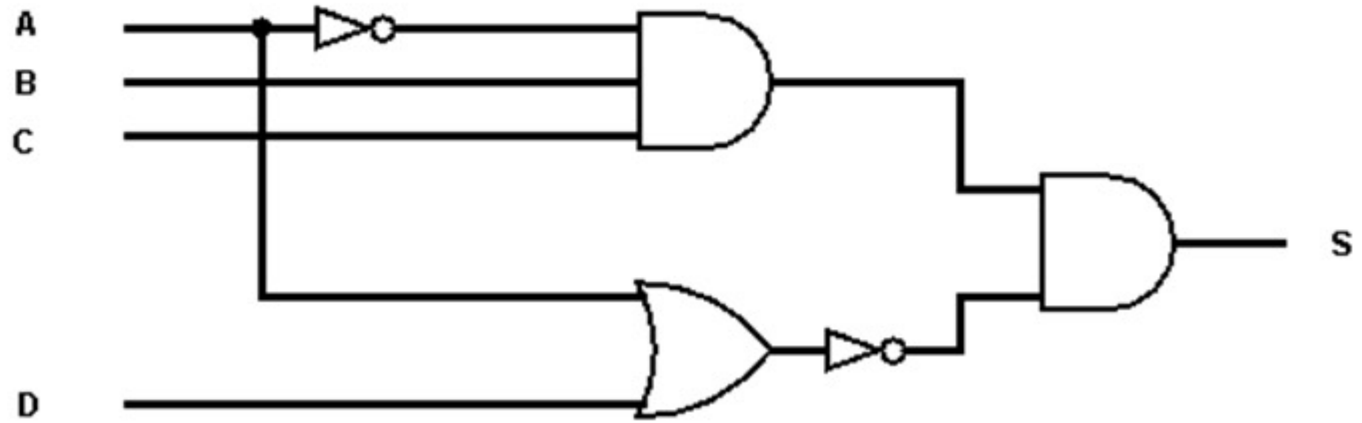
Exercício 2: Descreva a saída do circuito abaixo:



(b)

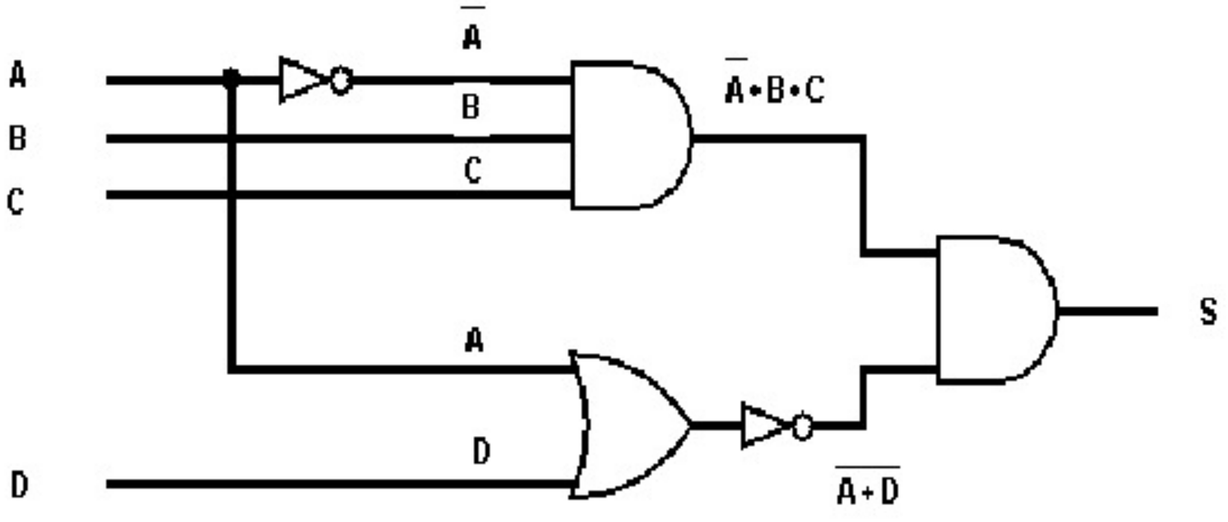
Descrevendo circuitos lógicos algebricamente

Exercício 3: Descreva a expressão do circuito abaixo



Descrevendo circuitos lógicos algebricamente

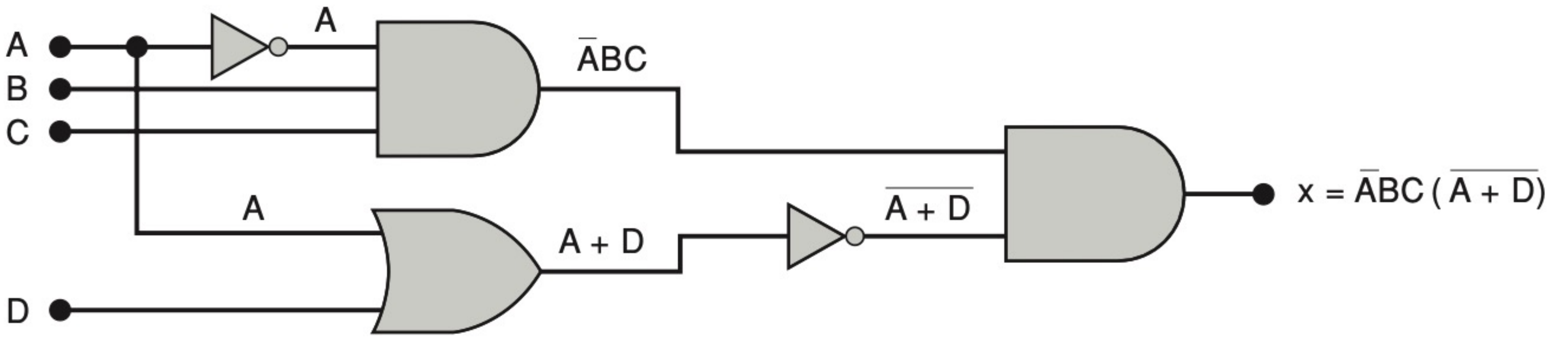
Exercício 3: Descreva a expressão do circuito abaixo



Solução: $S = (\overline{A}\ .\ B\ .\ C).\overline{(A+D)}$

Avaliando as saídas dos circuitos lógicos

- De posse da expressão booleana para a saída de um circuito, podemos obter o nível lógico da saída para qualquer conjunto de níveis lógicos de entrada. Por exemplo, suponha que desejemos saber o nível lógico da saída *x* para o circuito da figura abaixo para o caso em que *A* = 0, *B* = 1, *C* = 1 e *D* = 1. Assim como na álgebra convencional, o valor de *x* pode ser encontrado com a 'substituição' dos valores das variáveis na expressão e realizando a operação indicada, conforme mostrado a seguir:



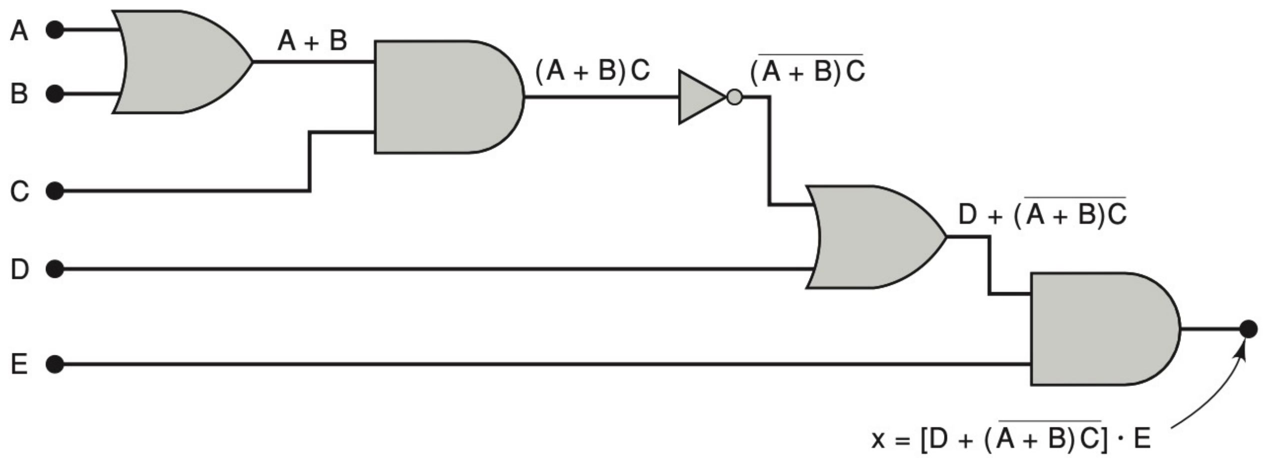
$$
\begin{aligned}
x &= \overline{A}BC(\overline{A + D}) \\
&= \overline{0} \cdot 1 \cdot 1 \cdot (\overline{0 + 1}) \\
&= 1 \cdot 1 \cdot 1 \cdot (\overline{0 + 1}) \\
&= 1 \cdot 1 \cdot 1 \cdot (\overline{1}) \\
&= 1 \cdot 1 \cdot 1 \cdot 0 \\
&= 0
\end{aligned}
$$

Avaliando as saídas dos circuitos lógicos

- Em geral, as regras a seguir têm de ser obedecidas quando avaliamos uma expressão booleana:
  1. Primeiro, realize as inversões de termos simples; ou seja, 0 = 1 ou 1 = 0.
  2. Em seguida, realize as operações dentro de parênteses.
  3. Realize as operações AND antes das operações OR, a menos que os parênteses indiquem o contrário.
  4. Se uma expressão tiver uma barra sobre, realize a operação indicada pela expressão e, em seguida, inverta o resultado.

Avaliando as saídas dos circuitos lógicos

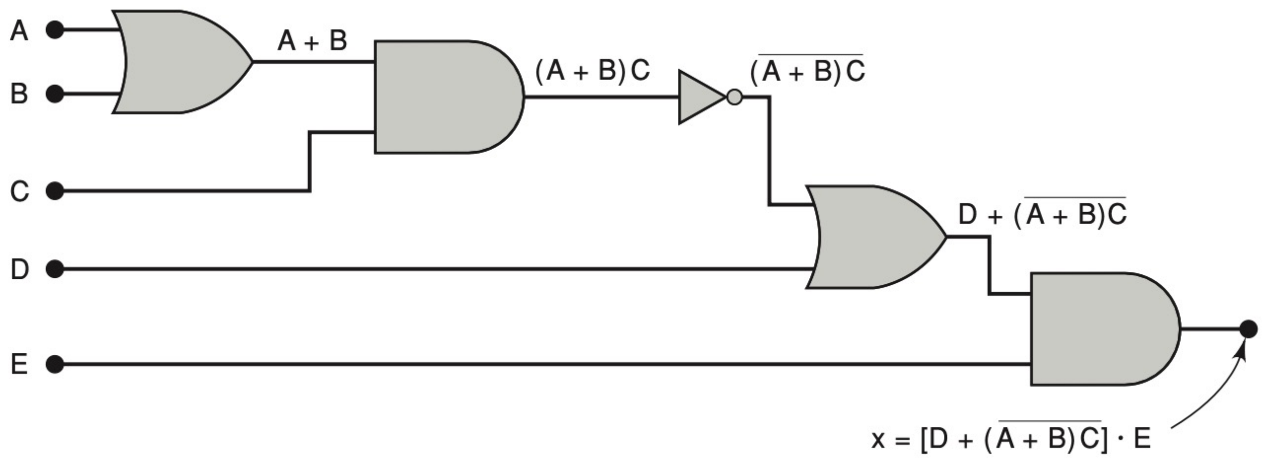
Exercício 4: Determine a saída do circuito da figura abaixo para *A* = 0, *B* = 0, *C* = 1, *D* = 1 e *E* = 1



(b)

Avaliando as saídas dos circuitos lógicos

Exercício 4: Determine a saída do circuito da figura abaixo para *A* = 0, *B* = 0, *C* = 1, *D* = 1 e *E* = 1



(b)

$$
\begin{aligned}
x &= [D + \overline{(A+B)C}] \cdot E \\
&= [1 + \overline{(0+0)\cdot 1}] \cdot 1 \\
&= [1 + \overline{0 \cdot 1}] \cdot 1 \\
&= [1 + \overline{0}] \cdot 1 \\
&= [1 + 1] \cdot 1 \\
&= 1 \cdot 1 \\
&= 1
\end{aligned}
$$

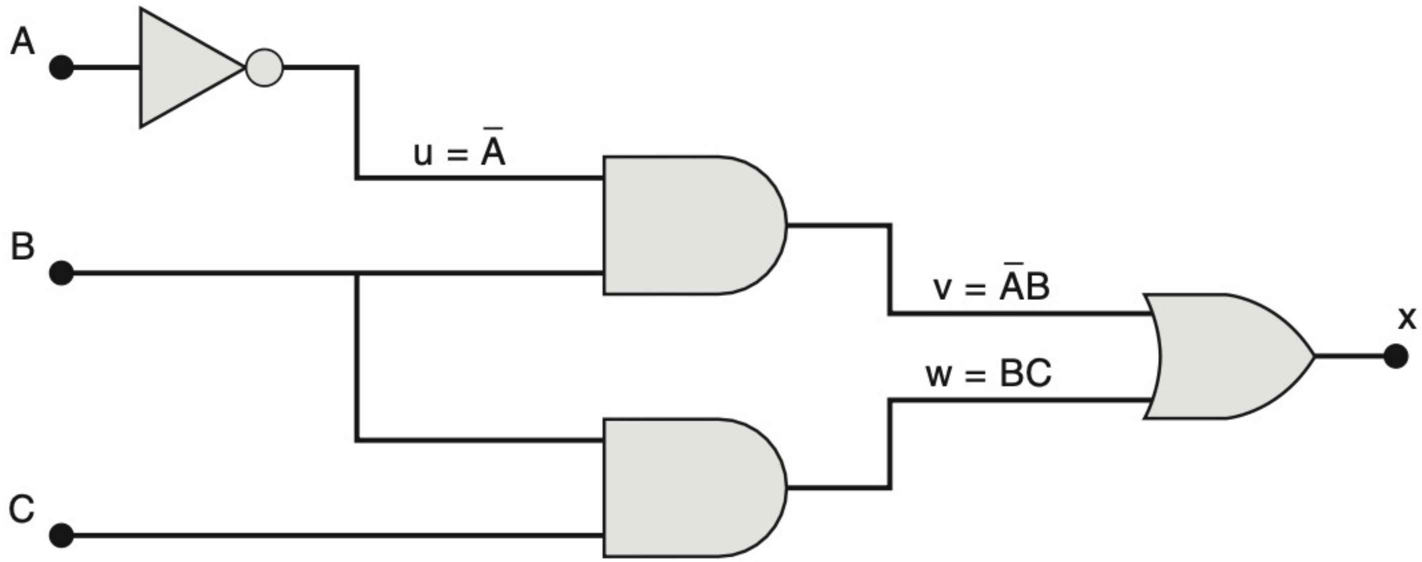
Análise utilizando uma tabela-verdade

- Quando se tem um circuito lógico combinacional e se deseja saber como ele funciona, a melhor maneira de analisá-lo é utilizar uma tabela-verdade. Esse método:
  - Permite que se analise uma porta ou combinação lógica de cada vez.
  - Permite que se confira facilmente o trabalho.
  - Quando o trabalho se encerra, há uma tabela que ajuda a verificação de erros do circuito lógico.

Lembre-se de que uma tabela-verdade lista as possíveis combinações de entrada em ordem numérica.
Para cada possível combinação de entrada, podemos determinar o estado lógico em cada ponto (nó) do circuito lógico, inclusive a saída.

Análise utilizando uma tabela-verdade

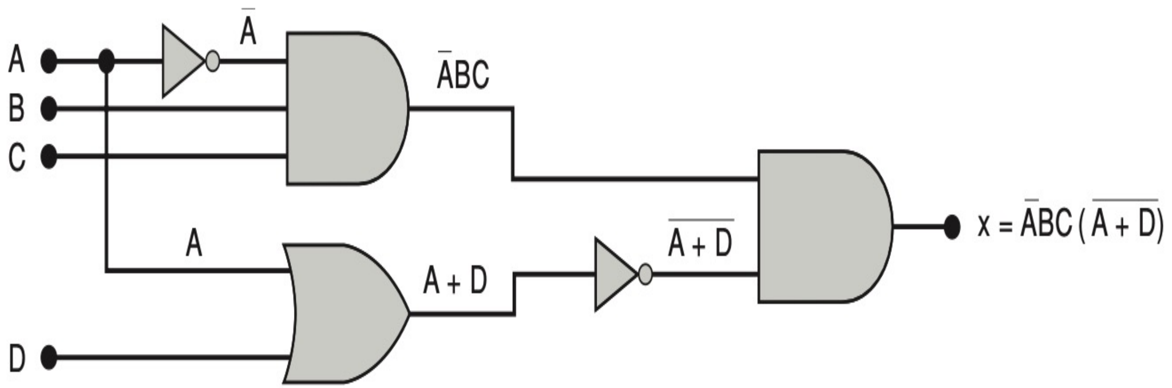
- Exemplo 3:



A	B	C	$u=\bar{A}$	$v=\bar{A}B$	$w=BC$	$x=v+w$
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

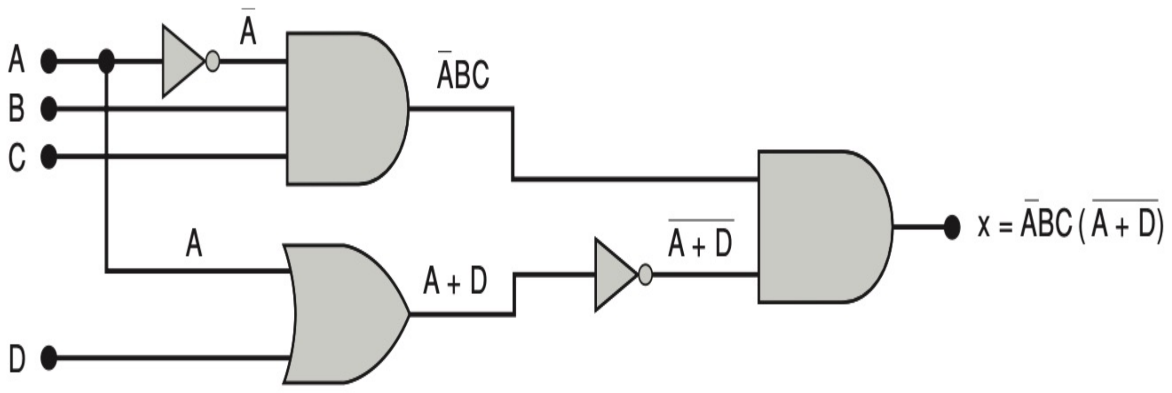
Análise utilizando uma tabela-verdade

- **Exercício 5: Analise a operação da figura abaixo criando uma tabela que mostre o estado lógico em cada nó do circuito.**



Análise utilizando uma tabela-verdade

- **Exercício 5: Analise a operação da figura abaixo criando uma tabela que mostre o estado lógico em cada nó do circuito.**



A	B	C	D	$t = \overline{A}BC$	$u = A + D$	$v = \overline{A + D}$	$x = tv$
0	0	0	0	0	0	1	0
0	0	0	1	0	1	0	0
0	0	1	0	0	0	1	0
0	0	1	1	0	1	0	0
0	1	0	0	0	0	1	0
0	1	0	1	0	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

Exercícios propostos de fixação

TOCCI, R. J.; WIDMER, N. S. & MOSS, G. L. Sistemas Digitais – Princípios e Aplicações. 11ª. Edição. São Paulo: Pearson Prentice Hall, 2011.

- Pag 51 Exercício 1 e 3
- Pag 54 Exercícios 2 e 3
- Pag 56 Exercicios 1 e 3
- Pag 58 Exercício 1
- Pag 67 Exercício 1

Exercícios recomendados

Esta aula foi baseada no Capítulo 3 (3.1, 3.2, 3.3, 3.4, 3.5 3.6, 3.7, 3.8, 3.9, 3.10 e 3.12) do livro:

TOCCI, R. J.; WIDMER, N. S. & MOSS, G. L. Sistemas Digitais – Princípios e Aplicações. 11ª. Edição. São Paulo: Pearson Prentice Hall, 2011.

Recomenda-se fortemente a realização dos exercícios de revisão referentes a essas seções.

Alguns livros da Bibliografia do curso podem ser encontrados em:

https://1drv.ms/u/s!At_6efeWN7EmknVhEyWUob-8o3O2?e=UHhhz1

Leituras recomendadas

Capítulo 3 (3.1, 3.2, 3.3, 3.4, 3.5 e 3.9)

- TOCCI, R. J.; WIDMER, N. S. & MOSS, G. L. Sistemas Digitais – Princípios e Aplicações. 11ª. Edição. São Paulo: Pearson Prentice Hall, 2011.

Capítulo 3 (3.1, 3.2, 3.3, 3.4, 3.5)

- FLOYD, T. L. Sistemas Digitais – Fundamentos e Aplicações. 9ª. Edição. Porto Alegre: Bookman, 2007.

Capítulo 3 (3.6, 3.7, 3.8, 3.9, 3.10 e 3.12)

- TOCCI, R. J.; WIDMER, N. S. & MOSS, G. L. Sistemas Digitais – Princípios e Aplicações. 11ª. Edição. São Paulo: Pearson Prentice Hall, 2011.

Capítulo 4 (4.4, 4.1, 4.2, 4.5, 5.3)

- FLOYD, T. L. Sistemas Digitais – Fundamentos e Aplicações. 9ª. Edição. Porto Alegre: Bookman, 2007.

Dúvidas??

Obrigada ☺