IC-UNICAMP

MC 602

Circuitos Lógicos e Organização de Computadores

IC/Unicamp

Prof Mario Côrtes

Capítulo 2

Introdução – Circuitos Combinacionais

IC-UNICAMP

Tópicos

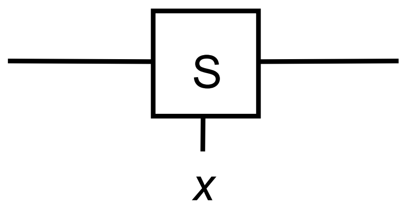
- Variáveis e funções
- Tabela verdade
- Portas lógicas
- Análise de circuitos digitais
- Álgebra Booleana
- Diagrama de Venn
- Síntese
- Mintermos e SOP
- Maxtermos e POS

Variáveis e funções

- Sinais digitais e representação por chaves
- Lógica feita com chaves

Uma chave binária

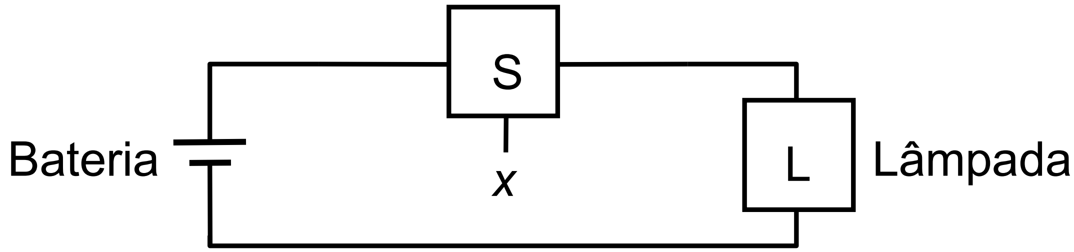
(a) Dois estados de uma chave



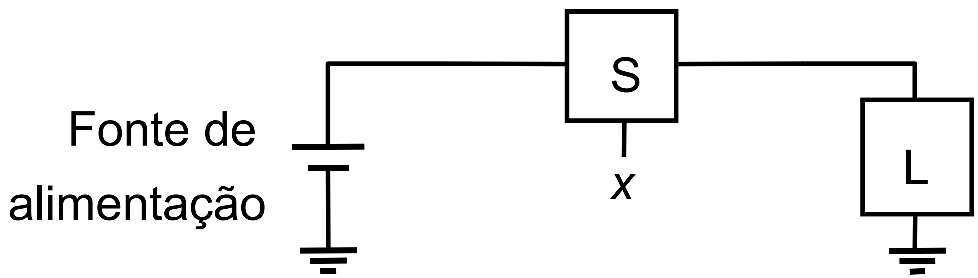
(b) Símbolo de uma chave

Lâmpada controlada por chave binária

IC-UNICAMP



(a) Conexão direta com a bateria



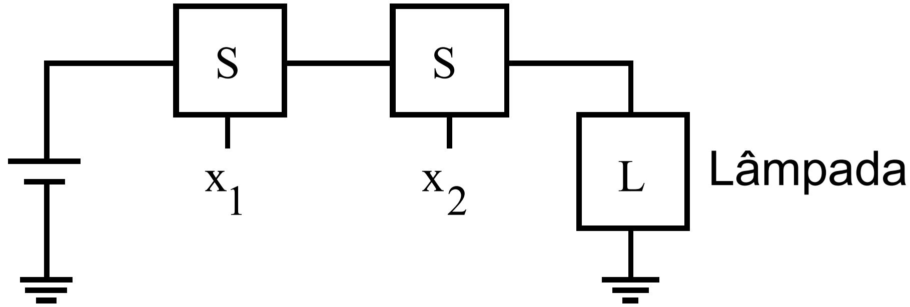
(b) Terra como interconexão

Se L(x) é o estado da lâmpada → ligado para L=1 e desligado para L=0, então
$L(x) = x$ liga se x = 1

L(x) é uma função lógica de uma variável

Duas funções lógicas básicas

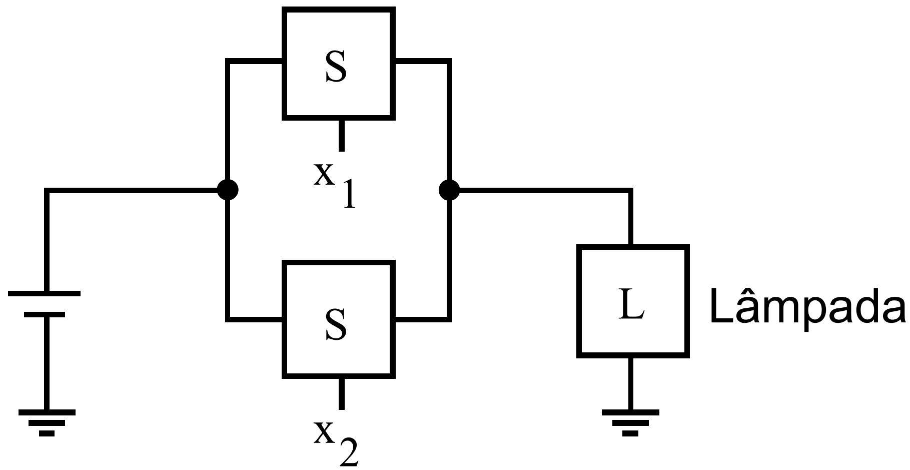
IC-UNICAMP



$L(x) = x_1 . x_2$
L=1 se $x_1 = x_2 = 1$
L=0 caso contrário

(a) AND lógico (E lógico): conexão em série

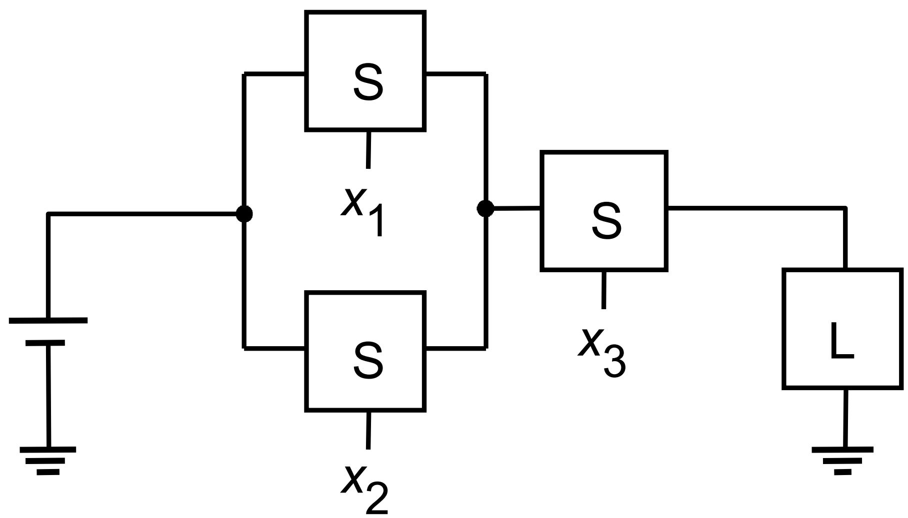
Símbolo para AND .
Símbolo para OR +



$L(x) = x_1 + x_2$
L=0 se $x_1 = x_2 = 0$
L=1 caso contrário

(b) OR lógico (OU lógico): conexão em paralelo

Conexão série-paralelo

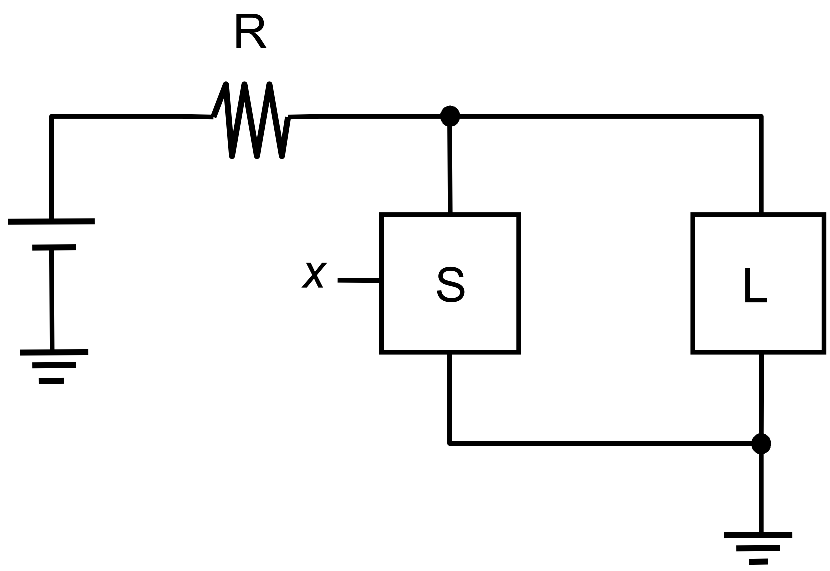


$L(x) = (x_1 + x_2) \cdot x_3$

L=1 se x_3 =1

E ou x_1 ou x_2 =1

Inversão



$L(x) = \overline{x}$

Tabela verdade (1)

entradas		saída
x1	**x2**	**x1 . x2**
0	0	0
0	1	0
1	0	0
1	1	1

entradas		saída
x1	**x2**	**x1 + x2**
0	0	0
0	1	1
1	0	1
1	1	1

AND e OR (2 entradas)

Tabela verdade (2)

IC-UNICAMP

entradas			saída
x1	**x2**	**x3**	**x1 . x2 . x3**
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

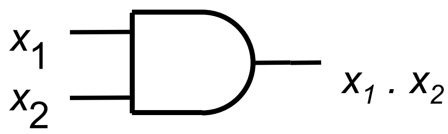
entradas			saída
x1	**x2**	**x3**	**x1 + x2 + x3**
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

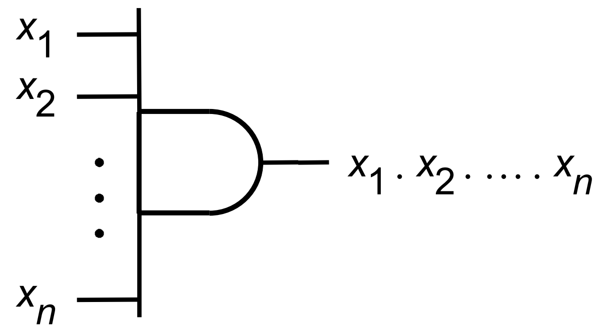
AND e OR (3 entradas)

IC-UNICAMP

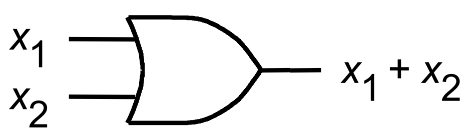
Portas lógicas AND, OR, NOT símbolos

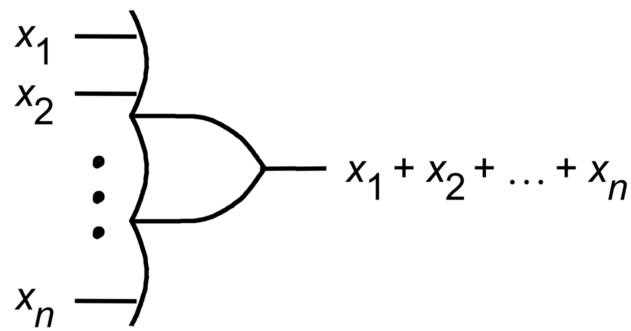
Portas AND



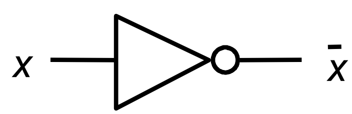


Portas OR

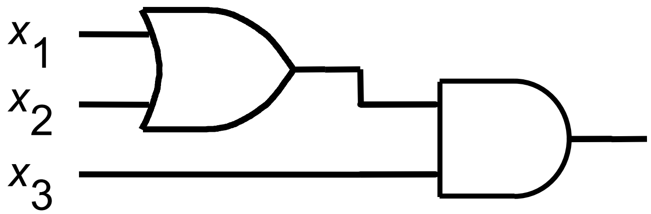




Inversor (not)



Uma função lógica com AND e OR



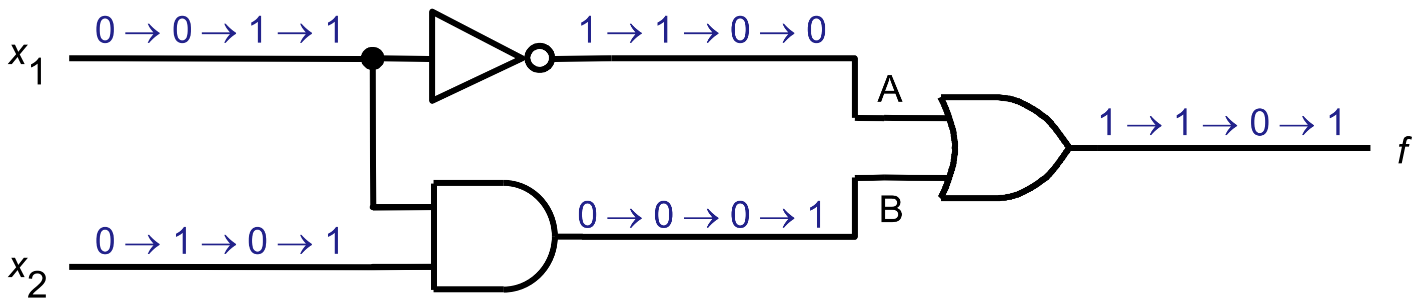
$f = (x_1 + x_2) \cdot x_3$

Análise e Síntese de Sistemas Digitais

- Processo de análise
  - Dado um circuito digital →
    - Descobrir a função lógica que ele implementa

- Processo de síntese
  - Dado um comportamento lógico esperado →
    - Sintetizar um circuito que se comporte desta maneira

Uma rede lógica



(a) A rede implementa $f = \bar{x}_1 + x_1 \cdot x_2$

x_1	x_2	$f(x_1, x_2)$	A	B
0	0	1		
0	1	1		
1	0	0		
1	1	1		

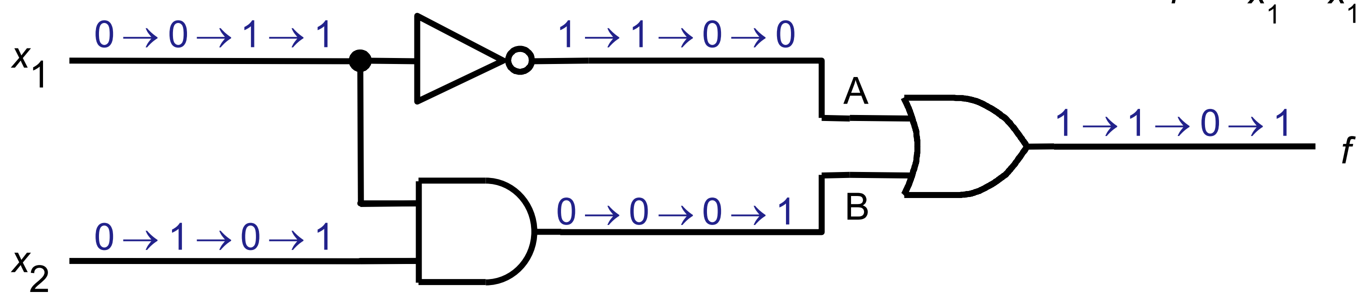
completar

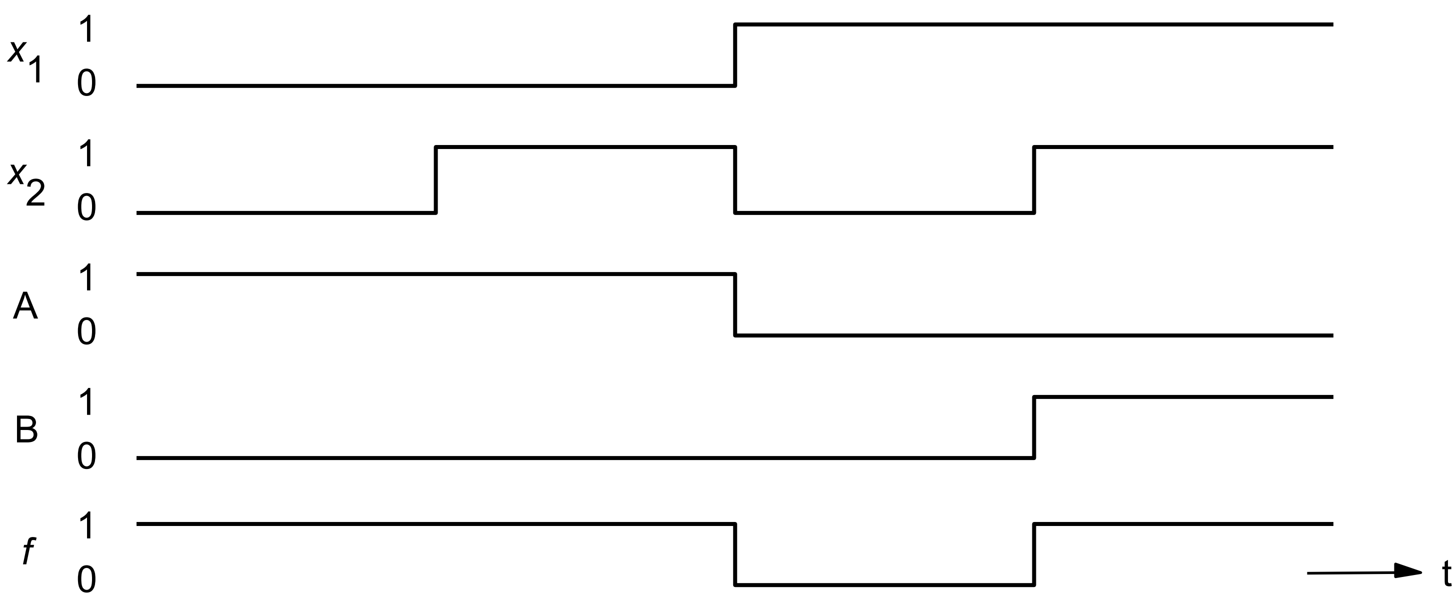
(b) Tabela verdade de f

Timing diagram

x_1	x_2	$f(x_1, x_2)$
0	0	1
0	1	1
1	0	0
1	1	1

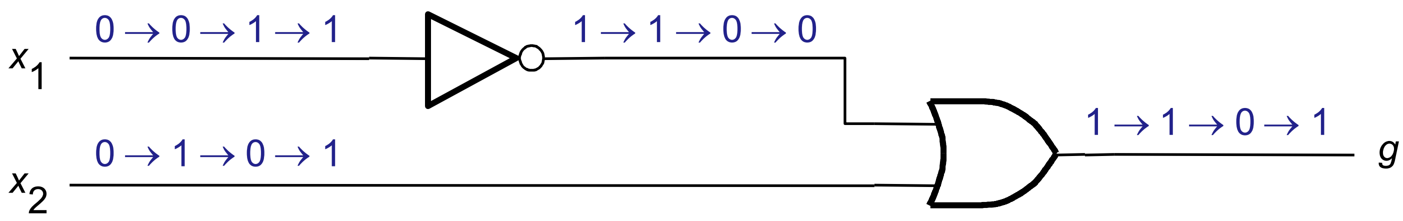
$f = \bar{x}_1 + x_1 \cdot x_2$



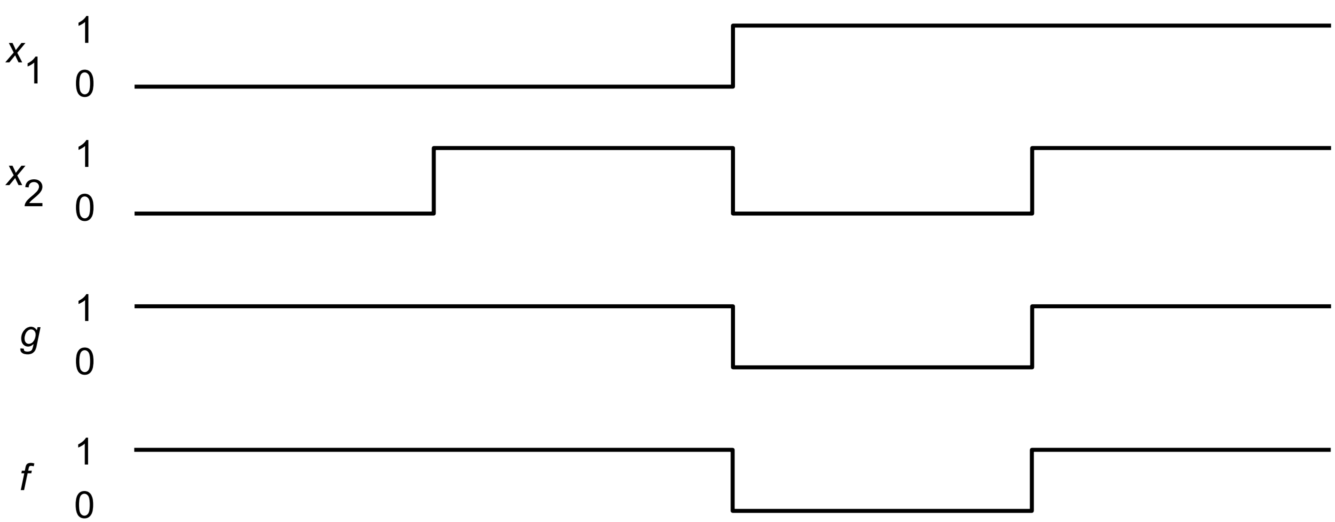


Timing diagram = diagrama de tempo = waveform = forma de onda

Circuitos equivalentes



Outra rede que implementa $g = \bar{x}_1 + x_2$



$g = \bar{x}_1 + x_2$

$f = \bar{x}_1 + x_1 \cdot x_2$

Então $g = \bar{x}_1 + x_2$ é equivalente a $f = \bar{x}_1 + x_1 \cdot x_2$

Implementação ótima

- Se $g = f \rightarrow \overline{x_1} + x_2 = \overline{x_1} + x_1 . x_2$
- Mas g é melhor → mais barato → menos hardware
- Existem muitas implementações para a mesma função
- Como encontrar a melhor?
- Método formal no capítulo 4
- Manipulações algébricas podem também gerar solução minimizada → em seguida

Álgebra Booleana

- Geoge Boole (1849): descrição algébrica do processo de raciocínio lógico
  - Verdadeiro / Falso → 1/0 → ligado/desligado
- Shannon (1930): uso da álgebra booleana para descrever o comportamento de circuitos feitos com chaves (1ª impl. de circ digitais)
- Vamos ver
  - Propriedades
  - Axiomas
  - Teoremas

Álgebra Booleana: primitivas

- Valores:
  - 0 e 1
- Operadores:
  - AND
  - OR
  - NOT

Álgebra Booleana: axiomas

- Como em matemática, axiomas são hipóteses básicas (não demonstradas)

1a $0 . 0 = 0$

1b $1 + 1 = 1$

2a $1 . 1 = 1$

2b $0 + 0 = 0$

3a $0 . 1 = 1 . 0 = 0$

3b $1 + 0 = 0 + 1 = 1$

4a se $x = 0$ então $\overline{x} = 1$

4b se $x = 1$ então $\overline{x} = 0$

Álgebra Booleana: teoremas de 1 variável

IC-UNICAMP

- Podem ser demonstrados por indução perfeita, substituindo-se valores possíveis de x, ou seja 0 ou 1, e usando os axiomas

5a $x \cdot 0 = 0$

5b $x + 1 = 1$

6a $x \cdot 1 = x$

6b $x + 0 = x$

7a $x \cdot x = x$

7b $x + x = x$

8a $x \cdot \overline{x} = 0$

8b $x + \overline{x} = 1$

9 $\overline{(\overline{x})} = x$

exemplo de prova T8b, para x=0

$0 + \overline{0} \overset{A4a}{=} 0 + 1 \overset{A3b}{=} 1$

repetir para x = 1

alunos → demonstrar todos

IC-UNICAMP

Princípio da dualidade

- Observar axiomas/teoremas (a) e (b)
- Dualidade $0 \Leftrightarrow 1$ $+ \Leftrightarrow .$
- Toda igualdade tem sua igualdade dual
- Há (no min.) 2 alternativas para implementar uma função→ adotar a mais barata

1a	$0 . 0 = 0$	1b	$1 + 1 = 1$
2a	$1 . 1 = 1$	2b	$0 + 0 = 0$
3a	$0 . 1 = 1 . 0 = 0$	3b	$1 + 0 = 0 + 1 = 1$
4a	se $x = 0$ então $\overline{x} = 1$	4b	se $x = 1$ então $\overline{x} = 0$
5a	$x . 0 = 0$	5b	$x + 1 = 1$
6a	$x . 1 = x$	6b	$x + 0 = x$
7a	$x . x = x$	7b	$x + x = x$
8a	$x . \overline{x} = 0$	8b	$x + \overline{x} = 1$

IC-UNICAMP

Propriedades

10a $x \cdot y = y \cdot x$	10b $x + y = y + x$	comutativa
11a $x.(y.z) = (x.y).z$	11b $x+(y+z) = (x+y)+z$	associativa
12a $x.(y+z) = x.y + x.z$	12b $x+y.z = (x+y).(x+z)$	distributiva
13a $x + x.y = x$	13b $x.(x+y) = x$	absorção
14a $x.y + x.\overline{y} = x$	14b $(x+y).(x+\overline{y}) = x$	combinação
15a $\overline{(x.y)} = \overline{x} + \overline{y}$	15b $\overline{(x+y)} = \overline{x} . \overline{y}$	Teor. DeMorgan
16a $x + \overline{x}.y = x + y$	16b $x.(\overline{x}+y) = x . y$	
17a $x.y+y.z+\overline{x}.z = x.y+\overline{x}.z$	17b $(x+y).(y+z).(\overline{x}+z) = (x+y).(\overline{x}+z)$	consenso

- Postulados básicos para derivar todos os teoremas e propriedades
  - operações + e . definidas
  - teoremas 5 e 8
  - propriedades 10 e 12

IC-UNICAMP

Prova de igualdades

- Indução perfeita:
  - substituir todas possibilidades de variáveis → como tabela verdade LHS e RHS
- Manipulação algébrica
  - manipular LHS até que ele se iguale ao RHS; ou
  - manipular LHS e RHS até que cheguem à mesma expressão
- Diagrama de Venn

IC-UNICAMP

Prova do teorema de DeMorgan

- Por indução perfeita
- $\overline{(x.\ y)} = \overline{x} + \overline{y}$

x	y	$x \cdot y$	$\overline{x \cdot y}$	$\overline{x}$	$\overline{y}$	$\overline{x} + \overline{y}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

LHS RHS

Exemplos: provar igualdades por manipulação algébrica

- Exemplo 2.1:

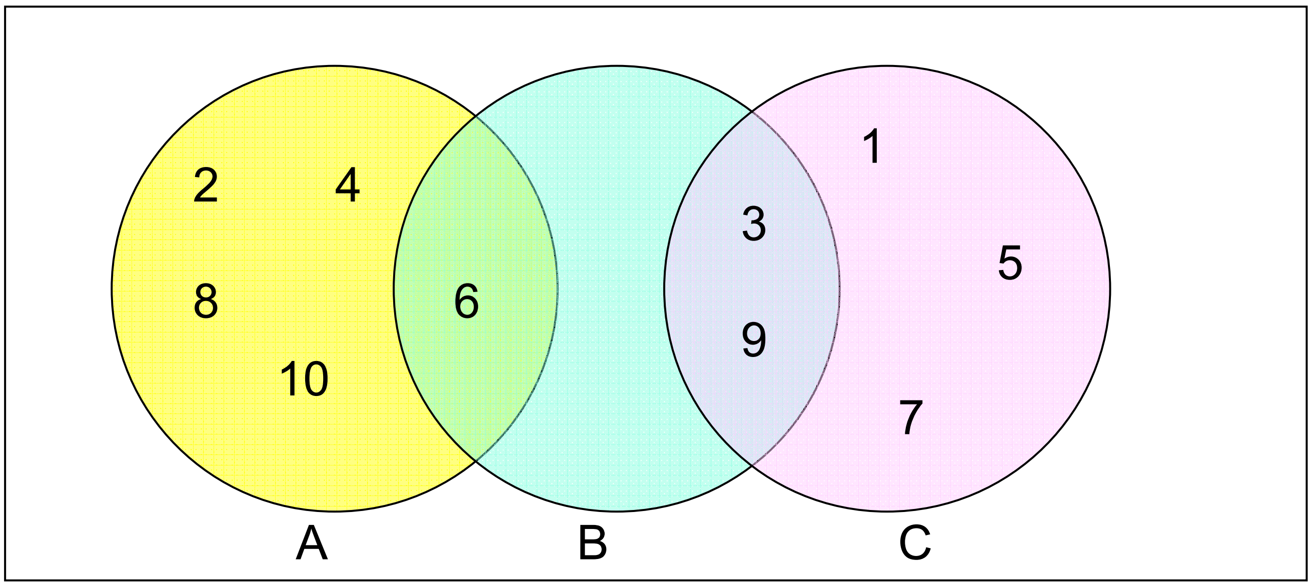
  $(x_1+x_3)\,.\,(\overline{x_1} + \overline{x_3}) = x_1.\overline{x_3} + \overline{x_1}.x_3$

- Exemplo 2.2:

  $x_1\,.\overline{x_3} + \overline{x_2}\,.\,\overline{x_3} + x_1\,.\,x_3 + \overline{x_2}\,.\,x_3 = \overline{x_1}\,\overline{x_2} + x_1\,x_2 + x_1\,\overline{x_2}$

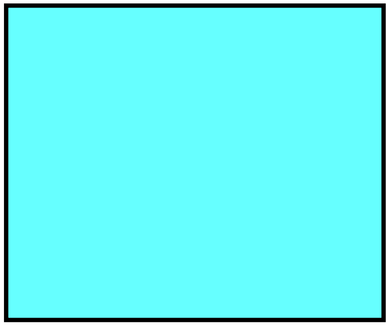
Diagrama de Venn

- Originou-se da teoria dos conjuntos e de lógica
- Exemplo: inteiros de 1 a 10
  - A: pares B: múltiplo de 3 C: ímpares



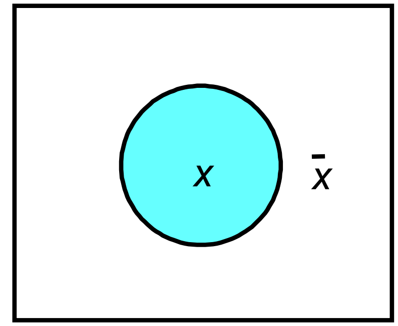
A.B = A ∩ B A+B = A ∪ B

Diagrama de Venn e álgebra de Boole

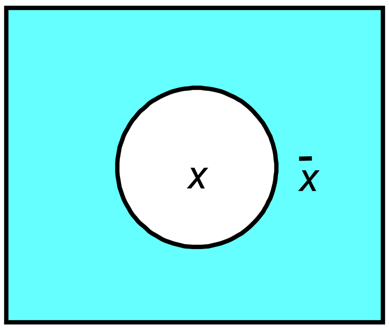


(a) Constante 1

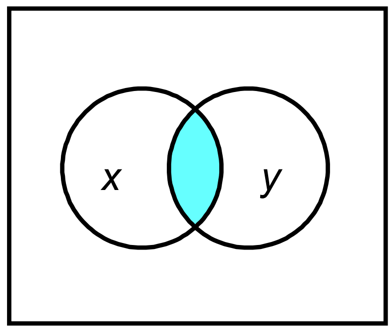
(b) Constante 0



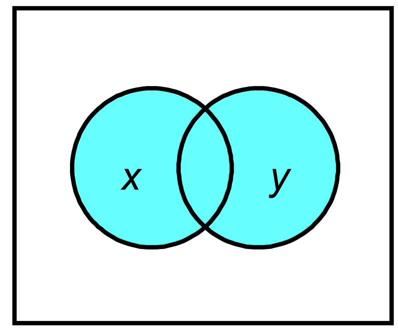
(c) Variável x



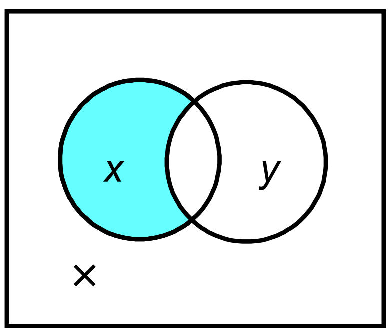
(d) $\bar{x}$



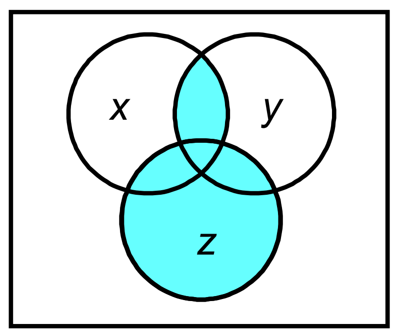
(e) $x \cdot y$



(f) $x + y$

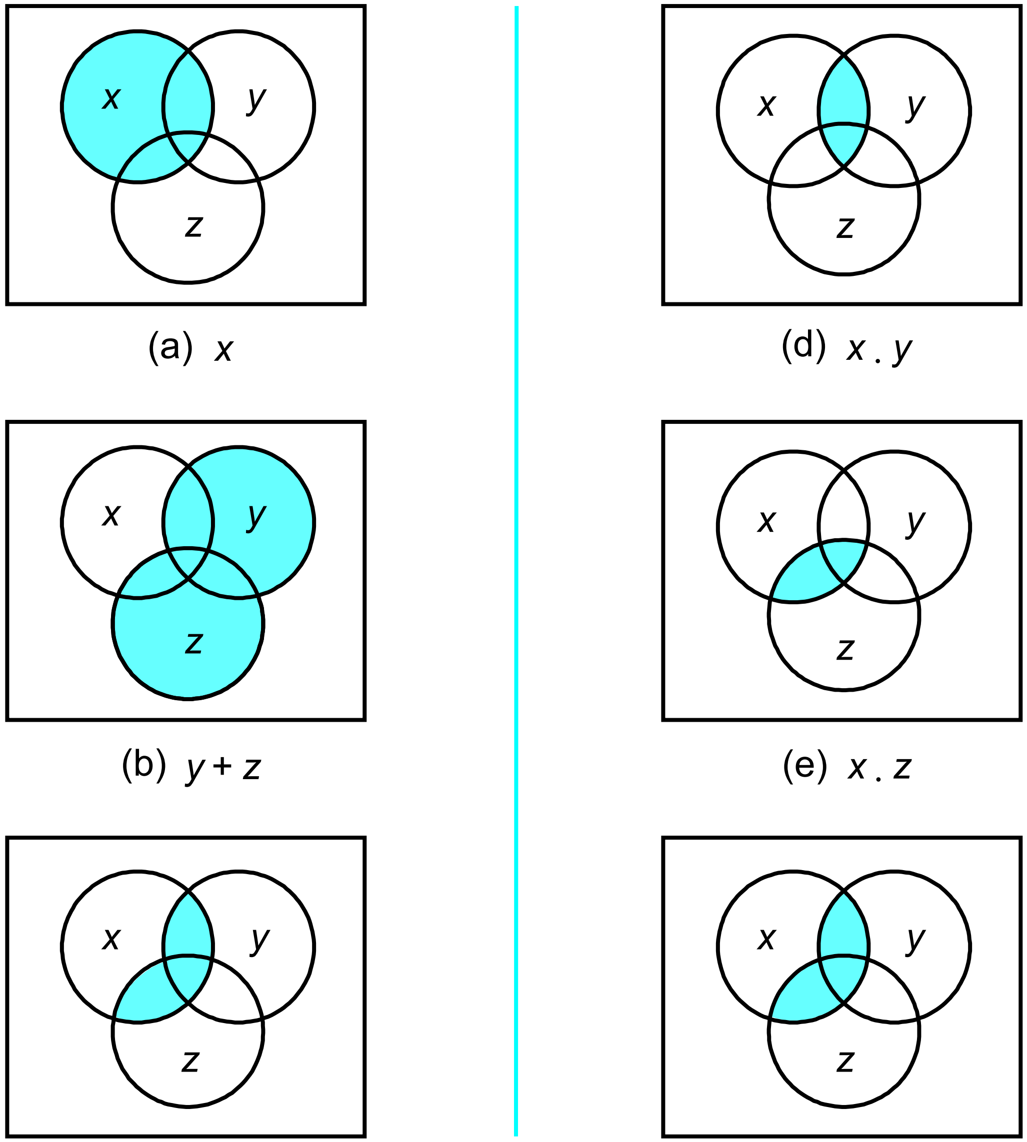


(g) $x \cdot \bar{y}$



(h) $x \cdot y + z$

Propriedade distributiva (T12a)



(a) x

(b) $y + z$

(c) $x \, . \, (y + z)$

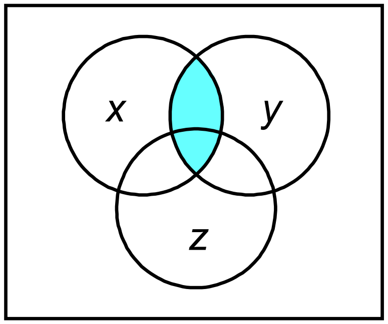
(d) $x \, . \, y$

(e) $x \, . \, z$

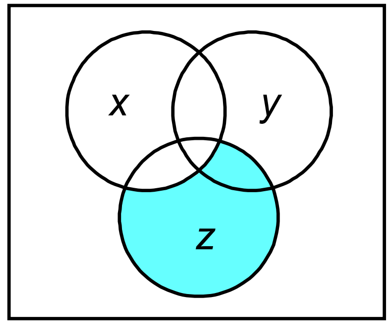
(f) $x \, . \, y + x \, . \, z$

IC-UNICAMP

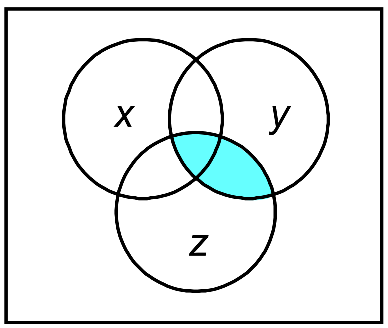
Verificação: $xy + \bar{x}z + yz = xy + \bar{x}z$



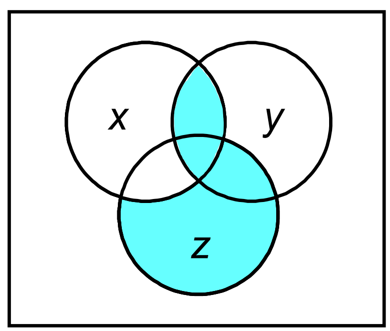
$x . y$



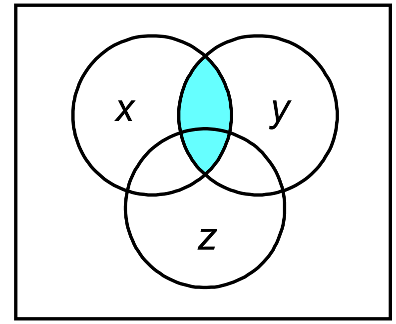
$\bar{x} . z$



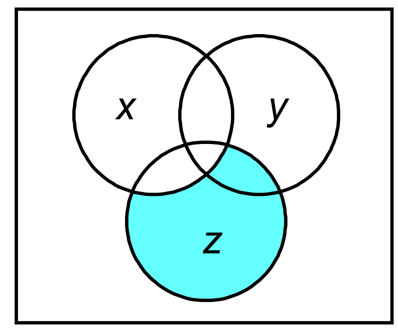
$y . z$



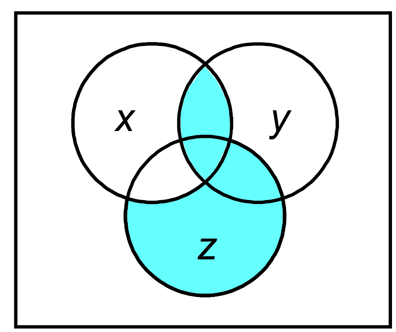
$x . y + \bar{x} . z + y . z$



$x . y$



$\bar{x} . z$



$x . y + \bar{x} . z$

Precedência das operações

- Com AND, OR e NOT é possível construir infinitas expressões
- Parênteses podem indicar precedência, mas a convenção seguinte é usada:
- NOT depois AND depois OR
- A expressão

$$x_1 \cdot x_2 + \overline{x_1} \cdot \overline{x_2}$$

- é calculada na seguinte ordem

$$(x_1 \cdot x_2) + ((\overline{x_1}) \cdot (\overline{x_2}))$$

Síntese usando portas AND, OR, NOT

x_1	x_2	$f(x_1, x_2)$
0	0	1
0	1	1
1	0	0
1	1	1

"produtos"

$\overline{x}_1.\overline{x}_2$

$\overline{x}_1.x_2$

$x_1.\overline{x}_2$

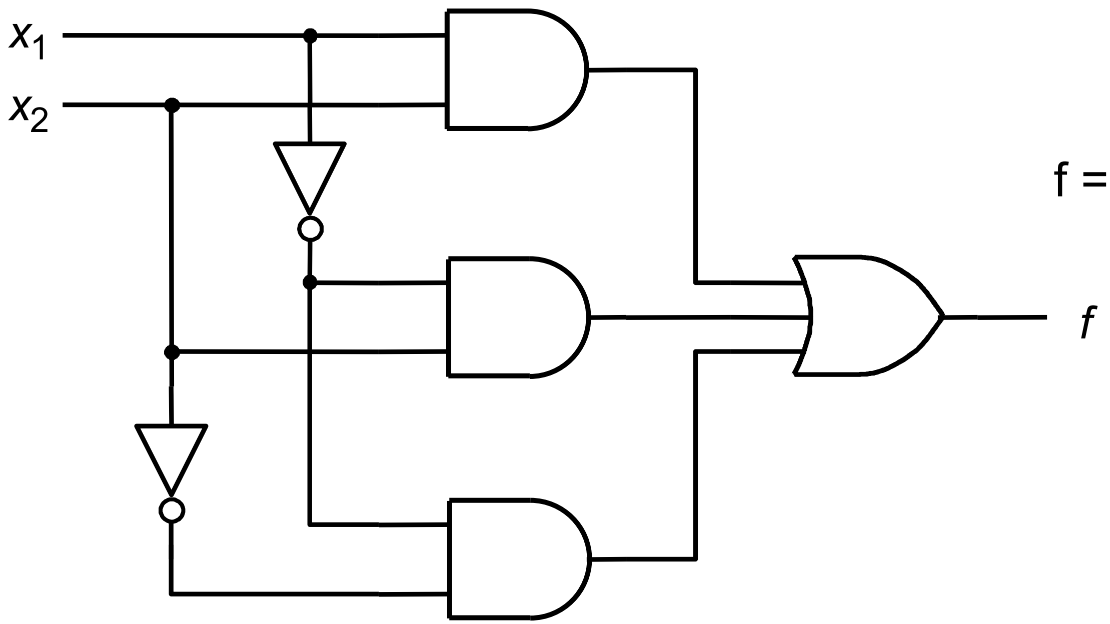
$x_1.x_2$

f=1

- soma de produtos "canônica" (um produto por linha da tabela verdade, com todos os literais)

$$f = \overline{x}_1.\overline{x}_2 + \overline{x}_1.x_2 + x_1.x_2$$

Implementação canônica e mínima

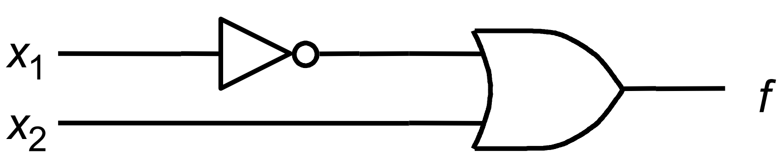


implementação canônica de soma de produtos

$$f = \overline{x}_1.\overline{x}_2 + \overline{x}_1.x_2 + x_1.x_2$$

demonstrar algebricamente

$$f = \overline{x}_1 + x_2$$



implementação mínima

Mintermos e Soma de Produtos (SOP)

IC-UNICAMP

- Para uma função de n variáveis
- Mintermo: produto (ANDs) de n variáveis, complementadas ou não
  - existe um mintermo para cada linha da tabela verdade
- Forma canônica de SOP
  - f = soma (ORs) de todos os mintermos para os quais a função é igual a um

	x_1	x_2	$f(x_1, x_2)$
m_0	0	0	1
m_1	0	1	1
m_2	1	0	0
m_3	1	1	1

$$f(x_1,x_2) = \overline{x_1}.\overline{x_2} + \overline{x_1}.x_2 + x_1.x_2$$

$$= \Sigma(m_0, m_1, m_3)$$

$$= \Sigma(0, 1, 3)$$

Exemplo: função de 3 variáveis

IC-UNICAMP

Row number	x_1	x_2	x_3	Minterm
0	0	0	0	$m_0 = \overline{x}_1\overline{x}_2\overline{x}_3$
1	0	0	1	$m_1 = \overline{x}_1\overline{x}_2 x_3$
2	0	1	0	$m_2 = \overline{x}_1 x_2\overline{x}_3$
3	0	1	1	$m_3 = \overline{x}_1 x_2 x_3$
4	1	0	0	$m_4 = x_1\overline{x}_2\overline{x}_3$
5	1	0	1	$m_5 = x_1\overline{x}_2 x_3$
6	1	1	0	$m_6 = x_1 x_2\overline{x}_3$
7	1	1	1	$m_7 = x_1 x_2 x_3$

Outro exemplo com 3 variáveis

IC-UNICAMP

Row number	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

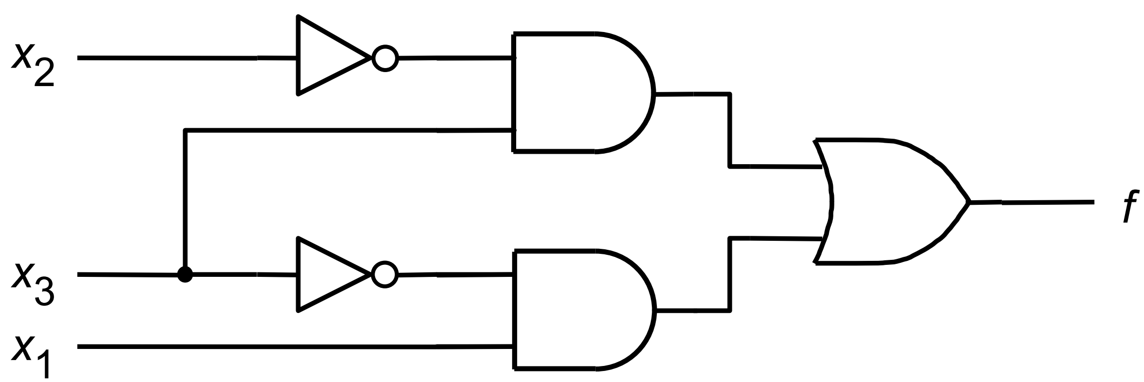
$$f(x_1,x_2,x_3) = \Sigma(m_1,m_4,m_5,m_6)$$

$$f(x_1,x_2,x_3) = \overline{x_1}.\overline{x_2}.x_3+x_1.\overline{x_2}.\overline{x_3}+x_1.\overline{x_2}.x_3+x_1.x_2.\overline{x_3}$$

$$f(x_1,x_2,x_3) = (\overline{x_1}+x_1).\overline{x_2}.x_3+x_1.(\overline{x_2}+x_2).\overline{x_3}$$

$$f(x_1,x_2,x_3) = 1.\overline{x_2}.x_3+x_1.1.\overline{x_3}$$

$$f(x_1,x_2,x_3) = \overline{x_2}.x_3+x_1.\overline{x_3}$$



Uma implementação SOP mínima

custo da implementação= n. de portas + n. entradas

custo = 5+8 = 13

IC-UNICAMP

Produto de Soma, Maxtermos

- Pelo princípio da dualidade: se é possível implementar como soma de mintermos (f=1) deve haver outra implementação com mintermos (f=0)
- Consideremos as linhas da tabela onde f=0
  - $\overline{f} = \Sigma m_i$ (m das linhas onde f=0)
  - $f = \overline{\Sigma m_i}$
  - por de Morgan: $f = \Pi \overline{m_i}$
  - Os complementos de m_i são os Maxtermos M_i
  - $f = \Pi M_i$

Produto de Soma, Maxtermos: exemplo

IC-UNICAMP

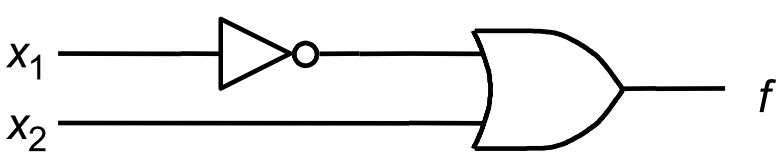
x_1	x_2	$f(x_1, x_2)$
0	0	1
0	1	1
1	0	0
1	1	1

$\overline{f(x_1,x_2)} = \Sigma(m_2) = x_1.\overline{x_2}$

Por de Morgan

$f(x_1,x_2) = \overline{x_1.\overline{x_2}} = \overline{x_1}+x_2$

$f(x_1,x_2) = \overline{m}_2 = M_2$



implementação mínima

custo da implementação=
n. de portas + n. entradas

custo = 2+3 = 5

IC-UNICAMP

Refazer com POS e maxtermos

Row number	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

$$\overline{f(x_1,x_2,x_3)} = \Sigma(m_0,m_2,m_3,m_7)$$

$$\overline{f} = \bar{x}_1.\bar{x}_2.\bar{x}_3+\bar{x}_1.x_2.\bar{x}_3+\bar{x}_1.x_2.x_3+x_1.x_2.x_3$$

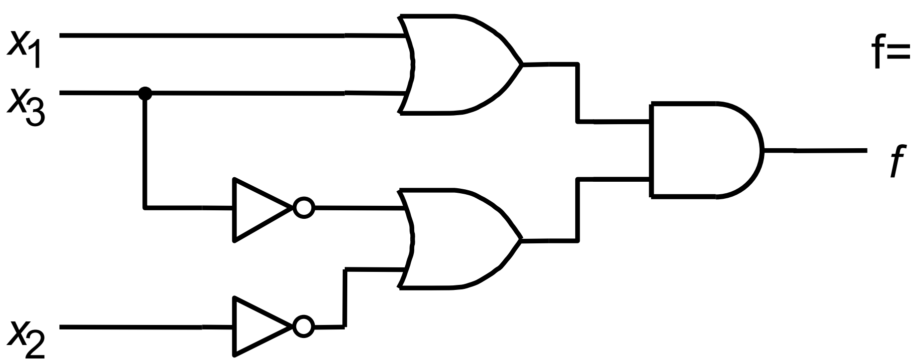
$$f = \overline{\bar{x}_1.\bar{x}_2.\bar{x}_3+\bar{x}_1.x_2.\bar{x}_3+\bar{x}_1.x_2.x_3+x_1.x_2.x_3}$$

$$f = \overline{m_0+m_2+m_3+m_7}$$

$$f = \bar{m}_0.\bar{m}_2.\bar{m}_3.\bar{m}_7$$

$$f = M_0.M_2.M_3.M_7$$

$$f=(x_1+x_2+x_3).(x_1+\bar{x}_2+x_3).(x_1+\bar{x}_2+\bar{x}_3).(\bar{x}_1+\bar{x}_2+\bar{x}_3)$$



provar que $f=(x_1+x_3).(\bar{x}_2+\bar{x}_3)$

custo = 5+8 = 13

Uma implementação POS mínima

Exemplo: mintermos e maxtermos

IC-UNICAMP

Row number	x_1	x_2	x_3	Minterm	Maxterm
0	0	0	0	$m_0 = \overline{x}_1\overline{x}_2\overline{x}_3$	$M_0 = x_1 + x_2 + x_3$
1	0	0	1	$m_1 = \overline{x}_1\overline{x}_2 x_3$	$M_1 = x_1 + x_2 + \overline{x}_3$
2	0	1	0	$m_2 = \overline{x}_1 x_2\overline{x}_3$	$M_2 = x_1 + \overline{x}_2 + x_3$
3	0	1	1	$m_3 = \overline{x}_1 x_2 x_3$	$M_3 = x_1 + \overline{x}_2 + \overline{x}_3$
4	1	0	0	$m_4 = x_1\overline{x}_2\overline{x}_3$	$M_4 = \overline{x}_1 + x_2 + x_3$
5	1	0	1	$m_5 = x_1\overline{x}_2 x_3$	$M_5 = \overline{x}_1 + x_2 + \overline{x}_3$
6	1	1	0	$m_6 = x_1 x_2\overline{x}_3$	$M_6 = \overline{x}_1 + \overline{x}_2 + x_3$
7	1	1	1	$m_7 = x_1 x_2 x_3$	$M_7 = \overline{x}_1 + \overline{x}_2 + \overline{x}_3$

Exemplos

- Ex 2.3: gerar SOP e simplificar

  $f(x_1,x_2,x_3) = \Sigma\ m(2, 3, 4, 6,7)$

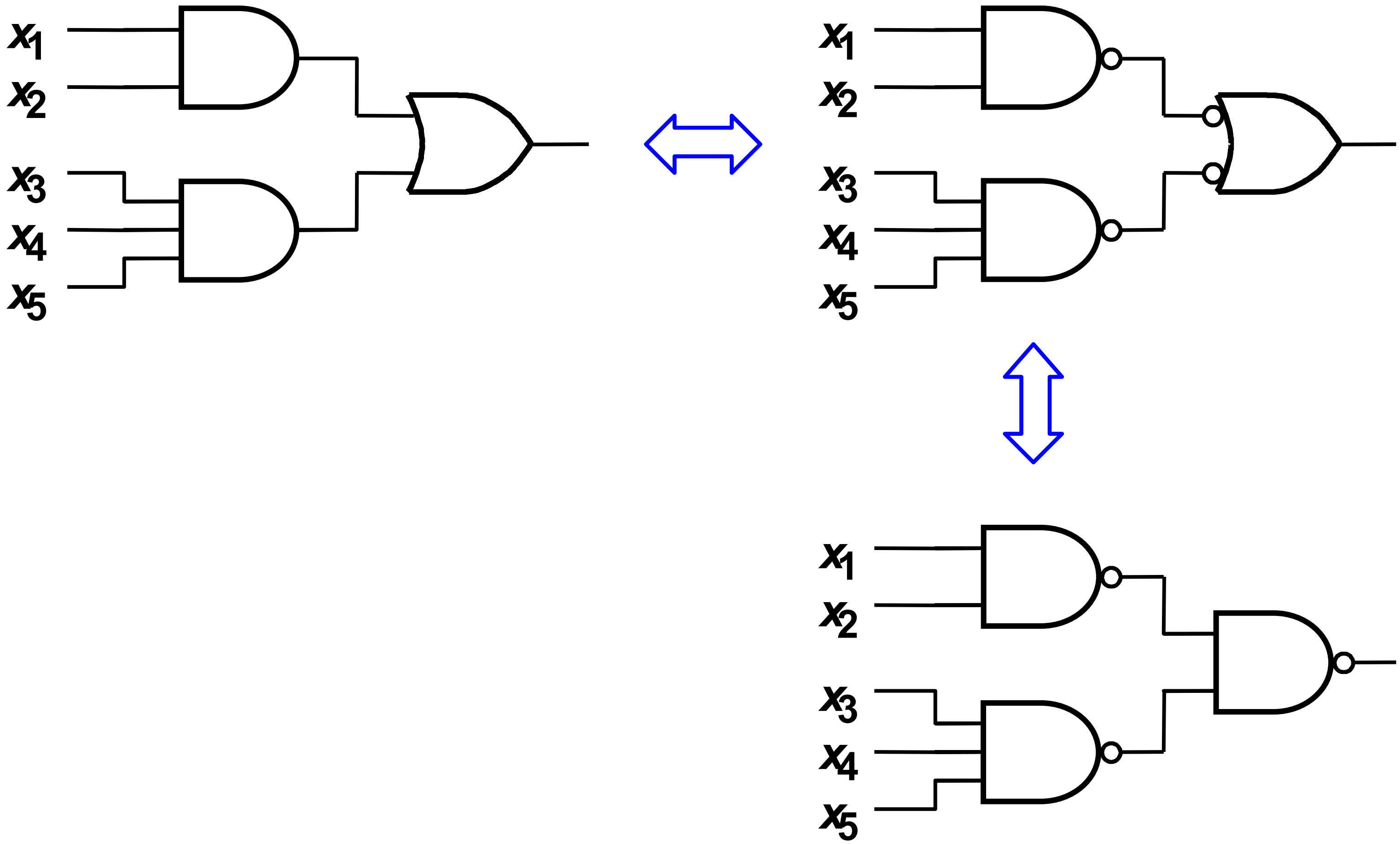
- Ex 2.4: gerar POS e simplificar

  $f(x_1,x_2,x_3) = \Pi\ M(0, 1, 5)$

- Ex 2.5: gerar SOP e simplificar

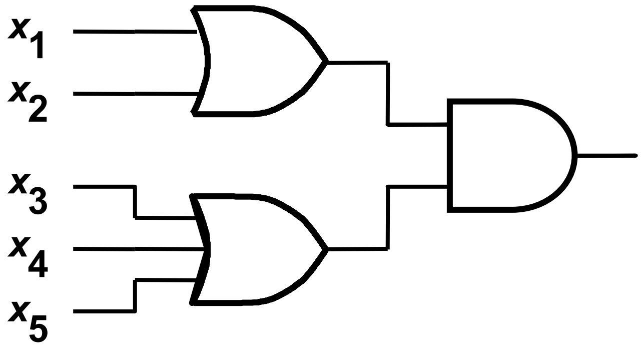
  $f(x_1,x_2,x_3,x_4) = \Sigma\ m(3, 7, 9, 12, 13, 14, 15)$

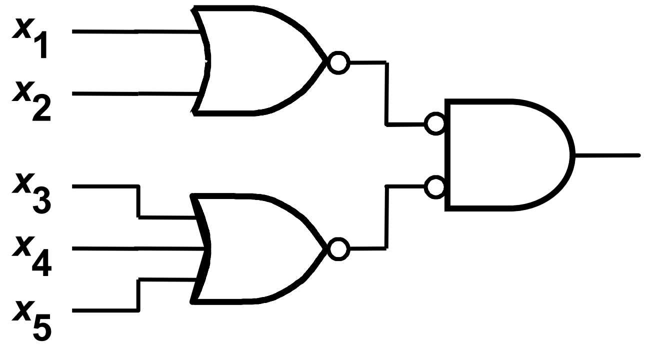
Exemplo de Síntese Só com NANDs

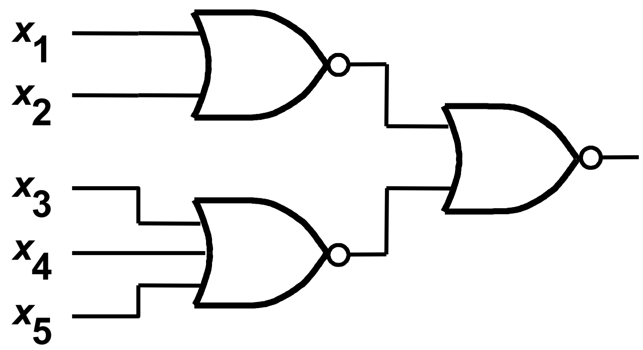


Exemplo de Síntese Só com NORs

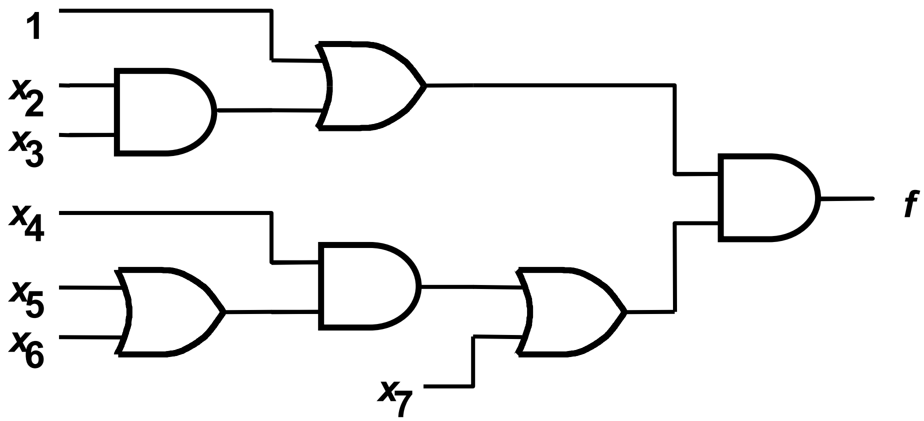
IC-UNICAMP



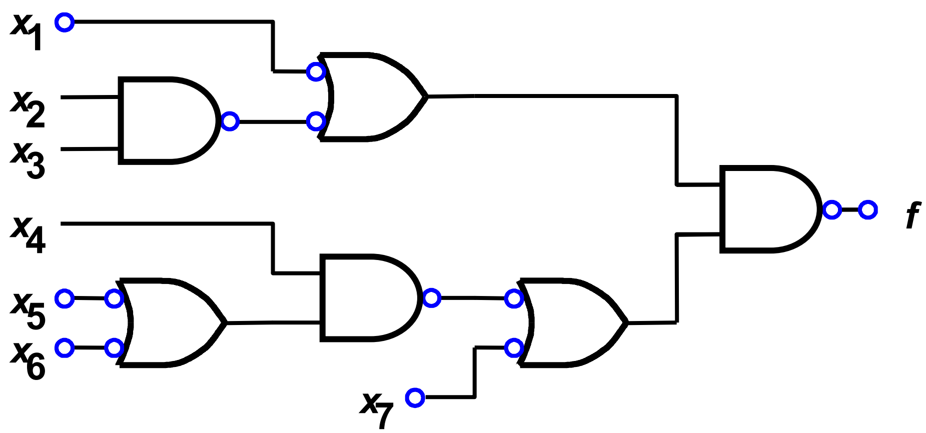




Exemplo

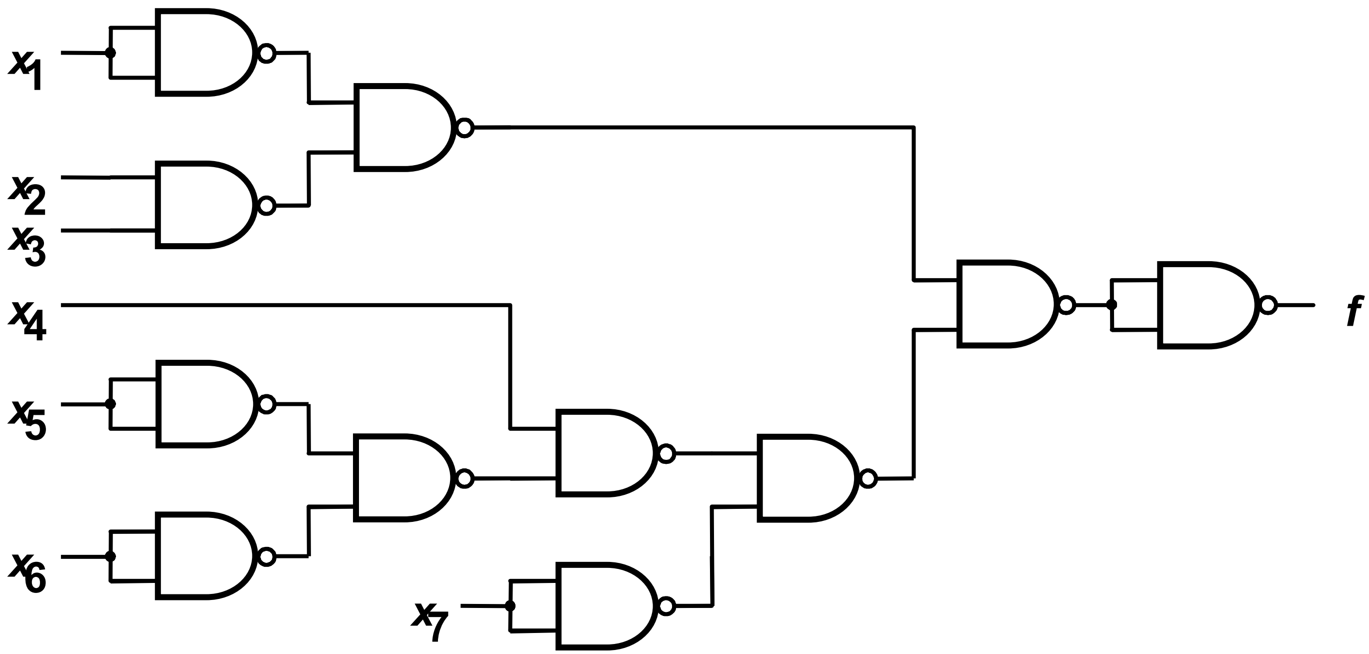


Circuit with AND and OR gates

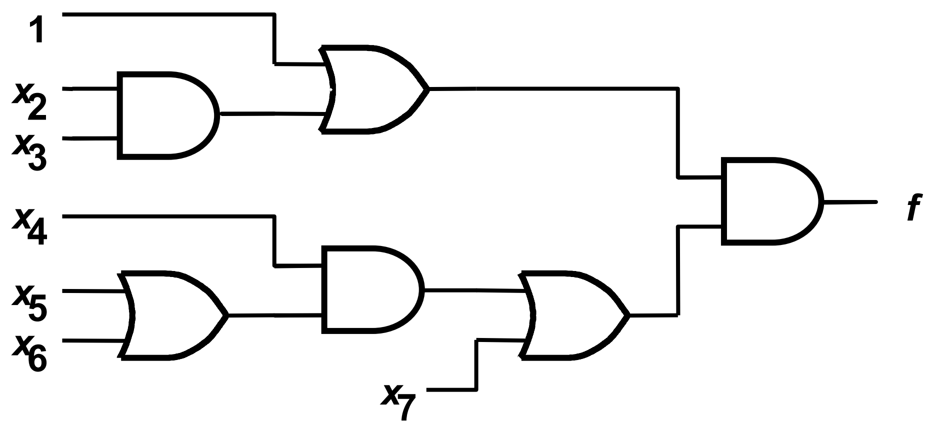


Convertendo para NANDs

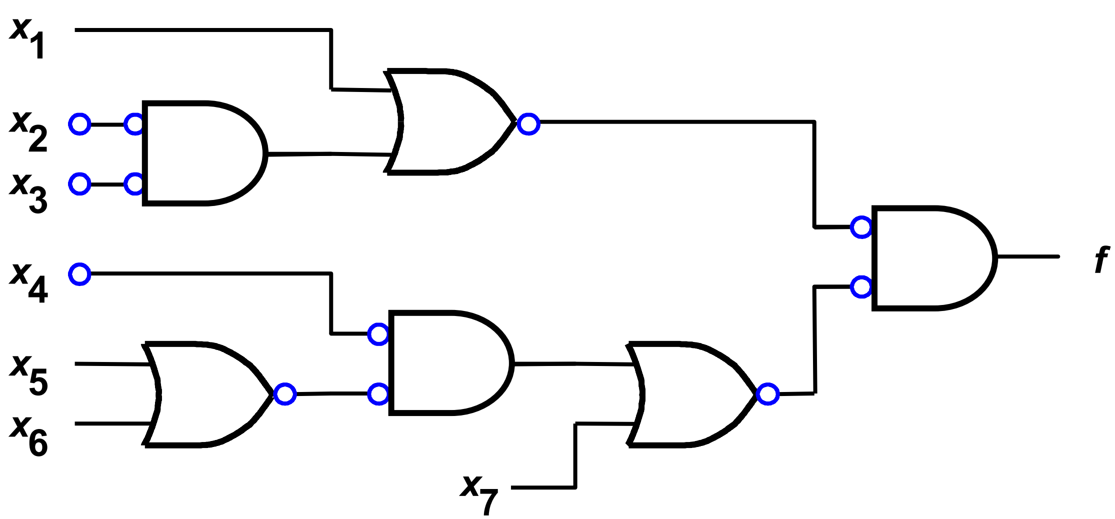
Exemplo (cont.)



Exemplo (cont.)



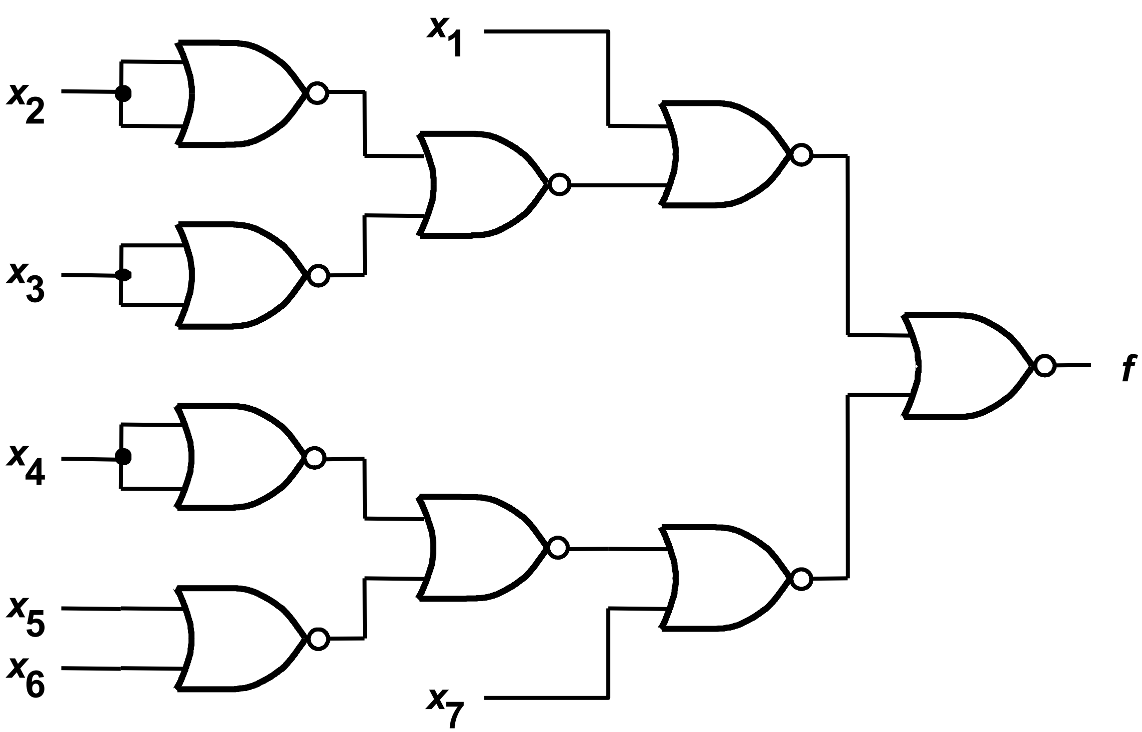
Circuit with AND and OR gates



Convertendo para NORs

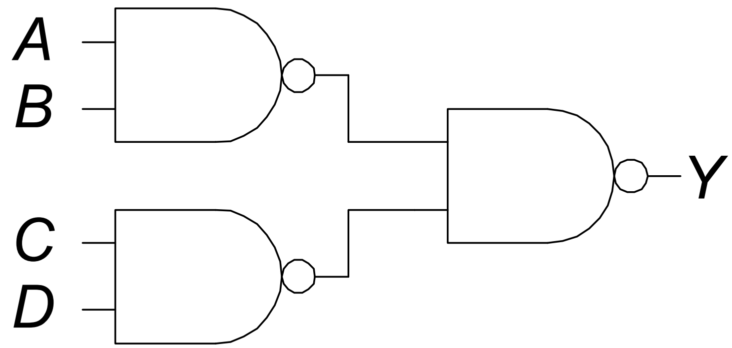
IC-UNICAMP

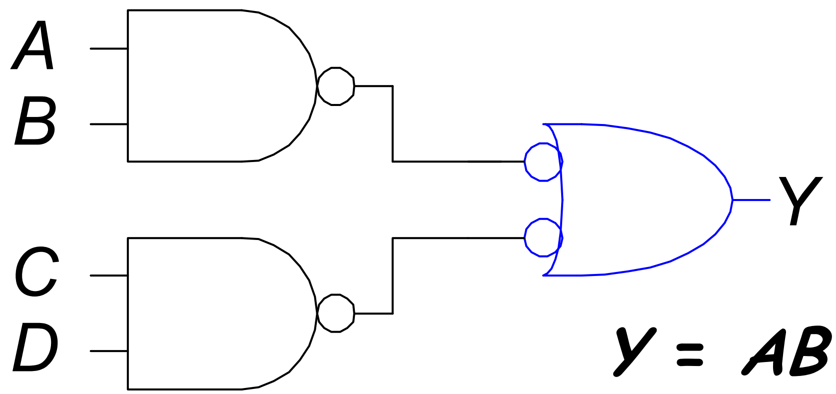
Exemplo (cont.)



Exercício:

- Qual é a expressão booleana para o circuito abaixo?

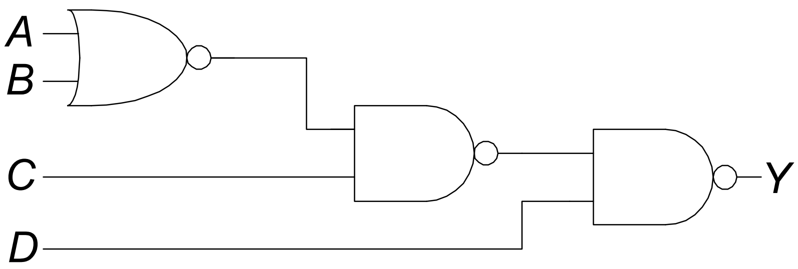


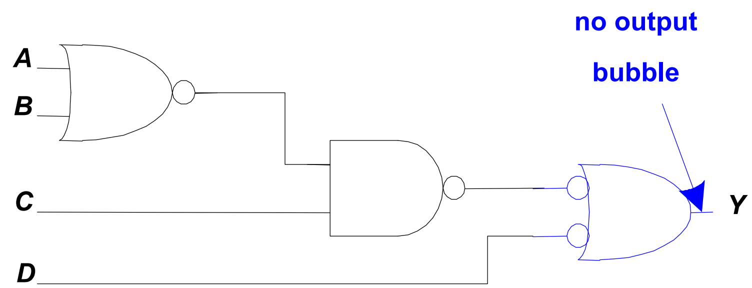


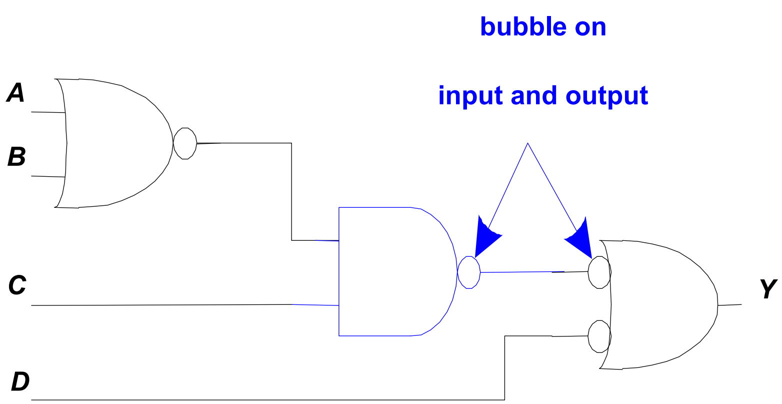
$Y = AB + CD$

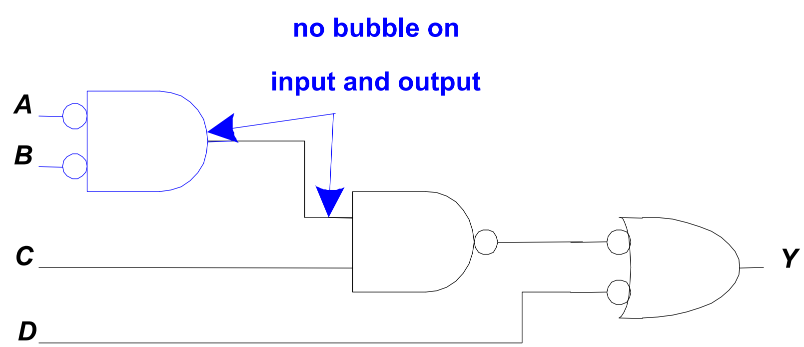
IC-UNICAMP

Técnica Bubble Pushing









$$Y = \overline{A}\ \overline{B}\ C + D$$

Exemplos

- Ex 2.6: implementar somente com NORs (ver exemplo 2.4)

  $f(x_1,x_2,x_3) = \Sigma\ m(2,3,4,6,7)$

- Ex 2.7: implementar somente com NANDs (ver exemplo 2.3)

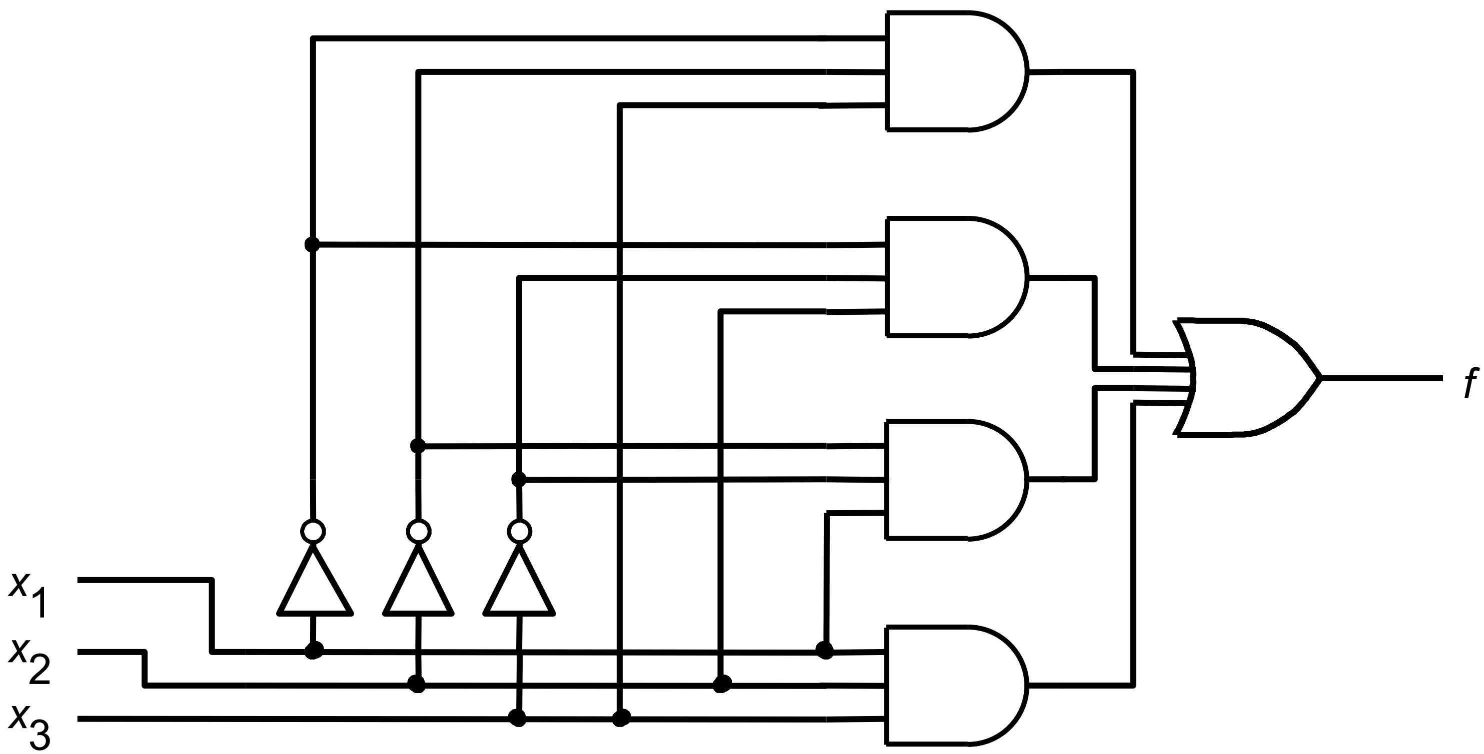
  $f(x_1,x_2,x_3) = \Sigma\ m(2,3,4,6,7)$

Exemplos: 3-way light controller

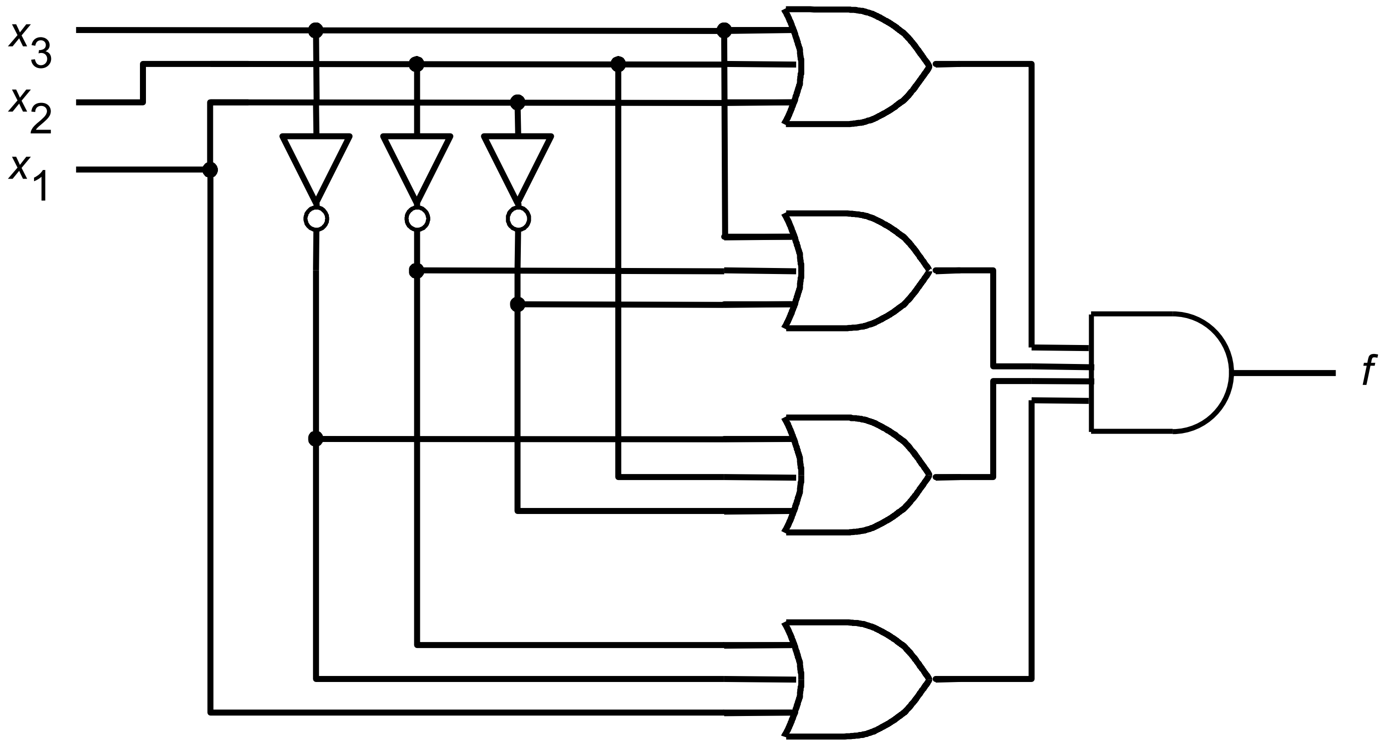
- Três interruptores de luminária: qualquer um que seja acionado causa mudança de estado da lâmpada (on ↔ off)
- $f(x_1,x_2,x_3) = \Sigma\ m(1,2,4,7)$
- f(x1,x2,x3) = Π M(0, 3, 5, 6)
- circuitos minimizados no próximo slide

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

3-way light controller: SOP



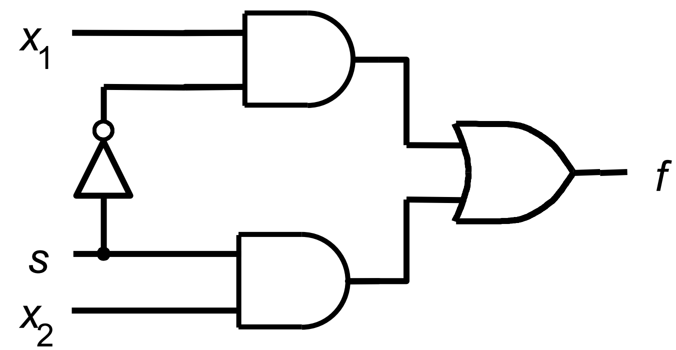
3-way light controller: POS



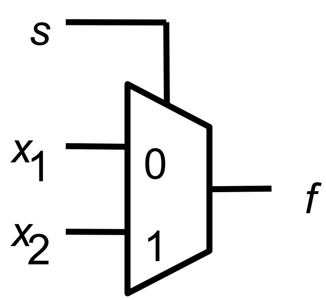
Exemplos: multiplexador

$s\ x_1\ x_2$	$f(s, x_1, x_2)$
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

(a)Truth table



(b) Circuit



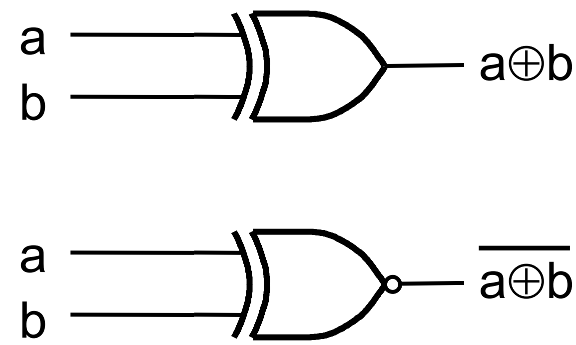
(c) Graphical symbol

s	$f(s, x_1, x_2)$
0	x_1
1	x_2

(d) More compact truth-table representation

Função lógica: XOR – ou exclusivo

a	b	XOR (a, b)	XNOR (a, b)
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1



- XOR = 0 se entradas iguais e =1 caso contrário
- XNOR = complemento de XOR = det. de igualdade
- Demonstrar que
  - $XOR(a,b) = a \oplus b = \bar{a} .\, b + a \,.\, \bar{b}$
  - $a \oplus b = b \oplus a$ e $a \oplus (b \oplus c) = (a \oplus b) \oplus c = a \oplus b \oplus c$
  - $\bar{a} \oplus b = \overline{a \oplus b}$
  - $XOR(a_1,a_2,..,a_n) = 1$ nº ímpar de 1's em $(a_1,a_2,..,a_n)$

Algumas implementações de XOR