IC-UNICAMP

MC 613

IC/Unicamp

Prof Guido Araújo

Prof Mario Côrtes

Circuitos combinacionais

Tópicos

- Circuitos digitais:
  - Níveis e margem de ruído
  - Tri-state
  - Wired AND, Wired OR
- Conceitos básicos de projeto de circuitos combinacionais
  - Tabela verdade
  - Soma de produtos
  - Álgebra Booleana
  - Mintermos e implicantes principais
  - Mapa de Karnaugh e minimização

Níveis Lógicos

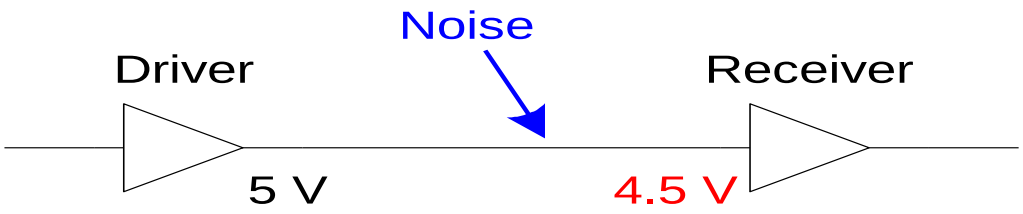
- Define as voltagens para representar o 1 e o 0
- Exemplo:
  - 0 : terra ou 0 volts
  - 1 : V_{DD} ou 5 volts
- Qual o valor produzido por uma porta (gate)?
- Se produzir 4.99 volts? Isso é um 0 ou um 1?
- E se 3.2 volts?

Níveis Lógicos

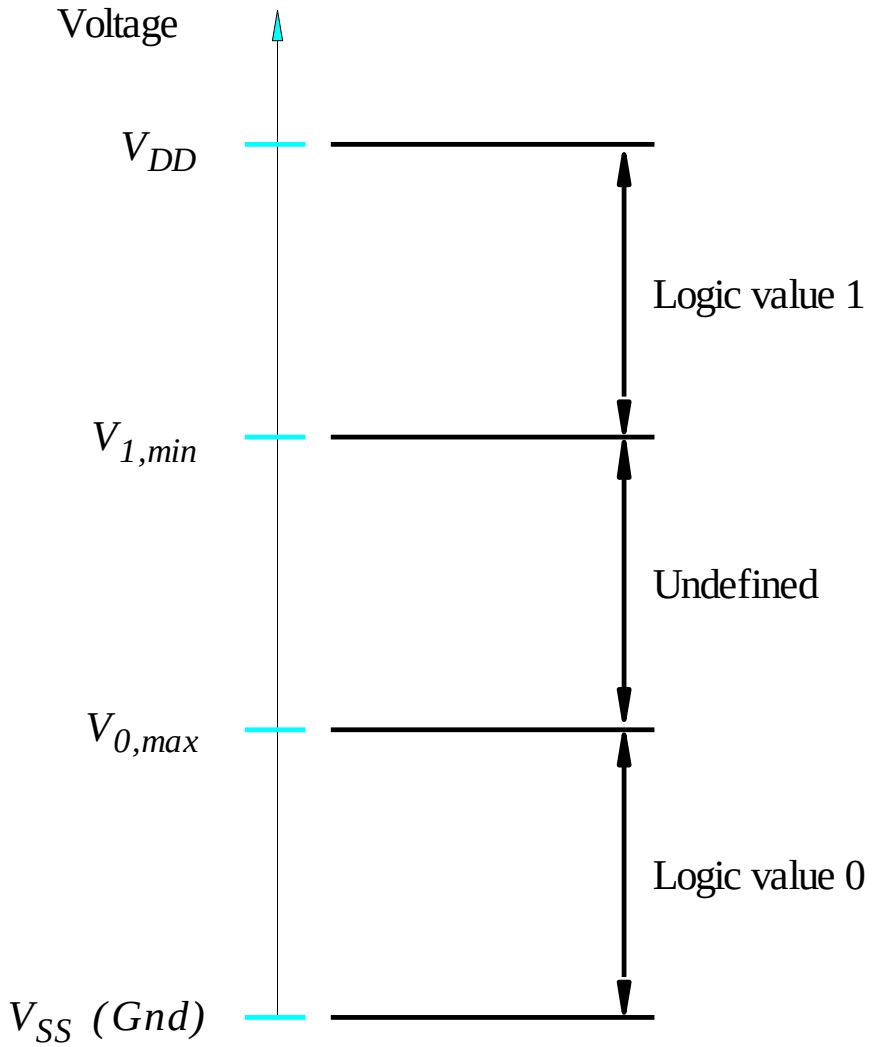
- Definem-se intervalos de voltagens para representar o 1 e o 0

- Define-se diferentes intervalos para saídas e entradas para permitir tolerância a ruídos

- Ruído é qualquer coisa que degrada o sinal

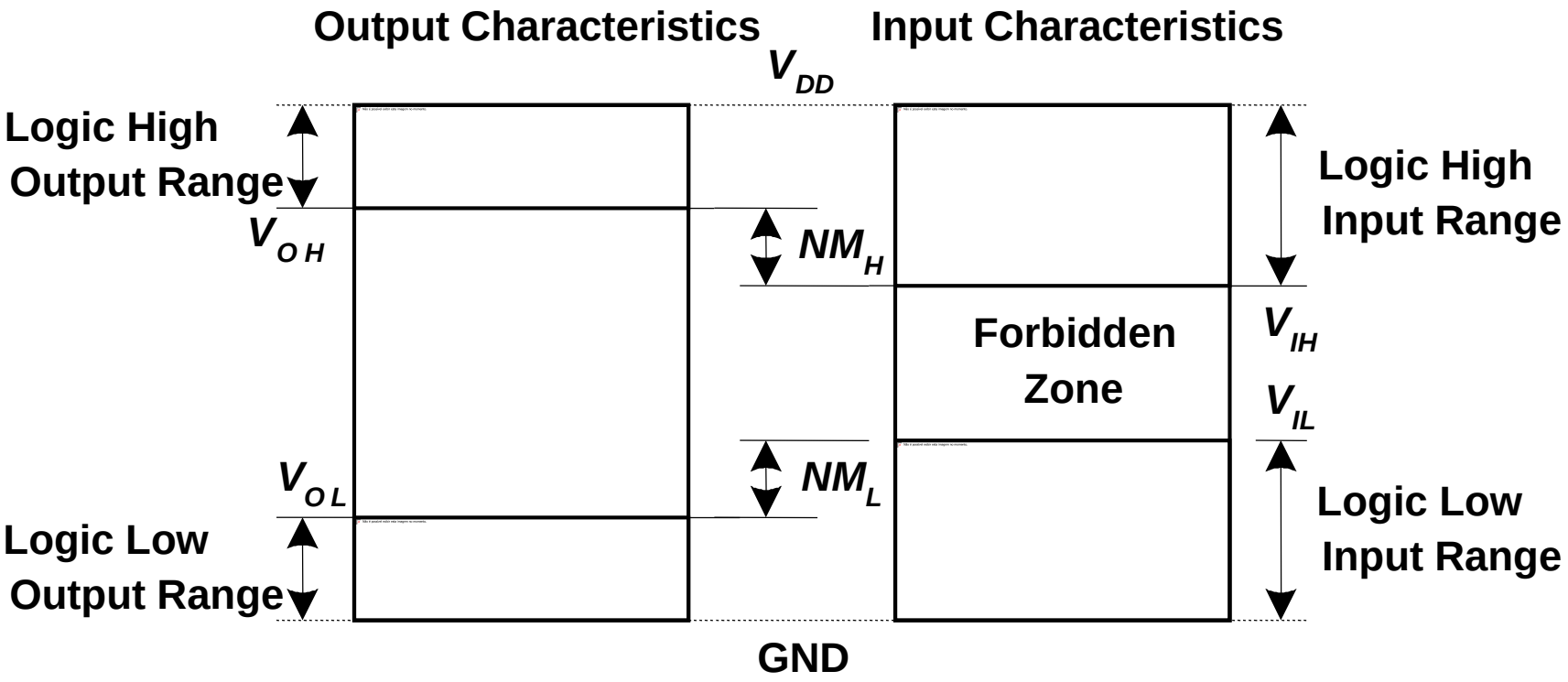


Níveis Lógicos



Níveis Lógicos: Margem de Ruído

IC-UNICAMP

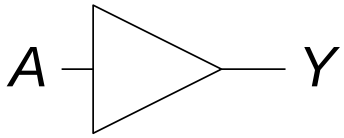


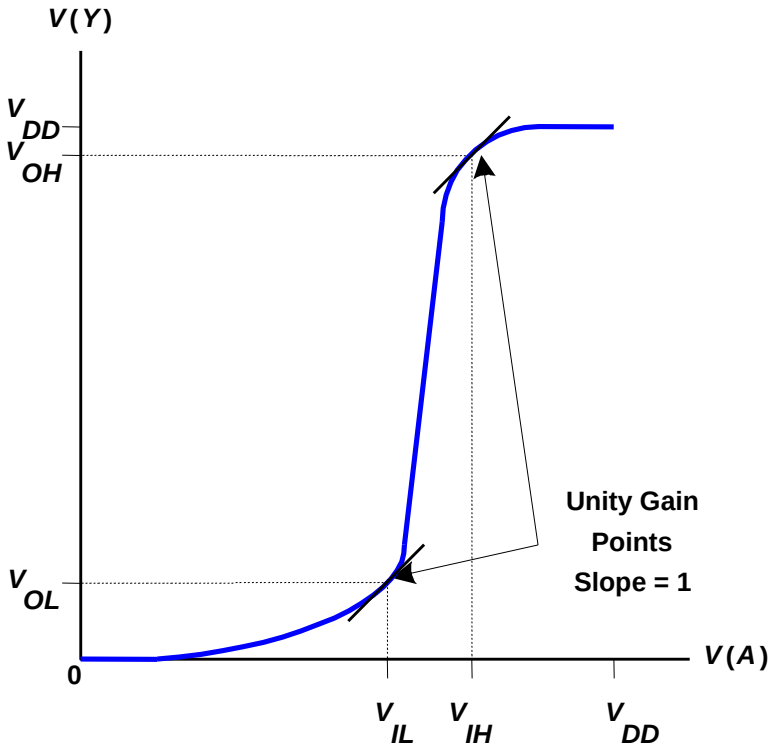
$$NM_H = V_{OH} - V_{IH}$$

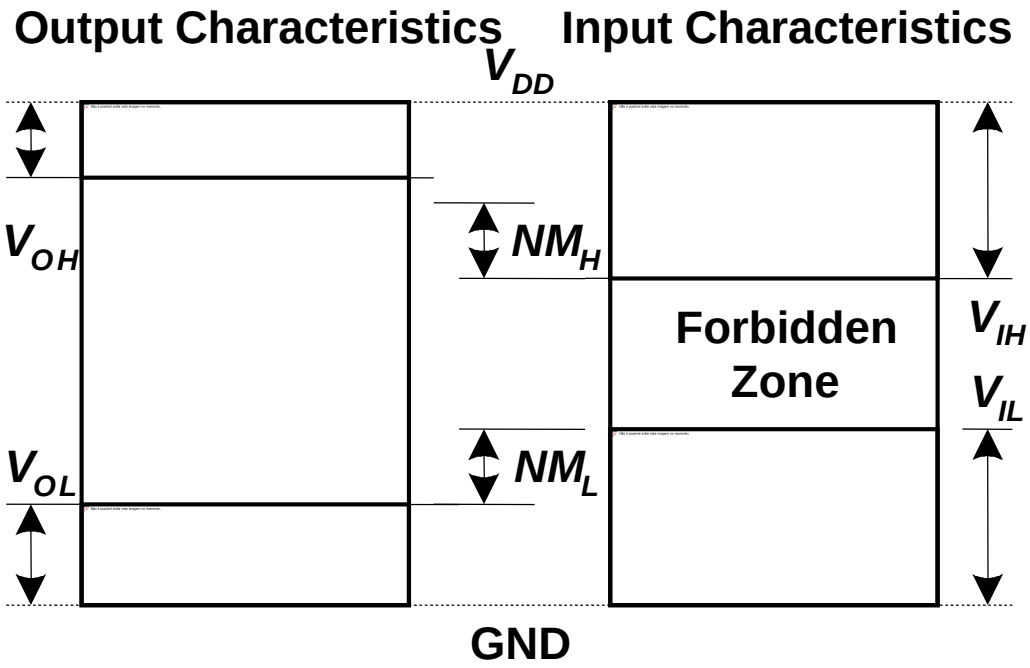
$$NM_L = V_{IL} - V_{OL}$$

Característica de Transferência DC

IC-UNICAMP

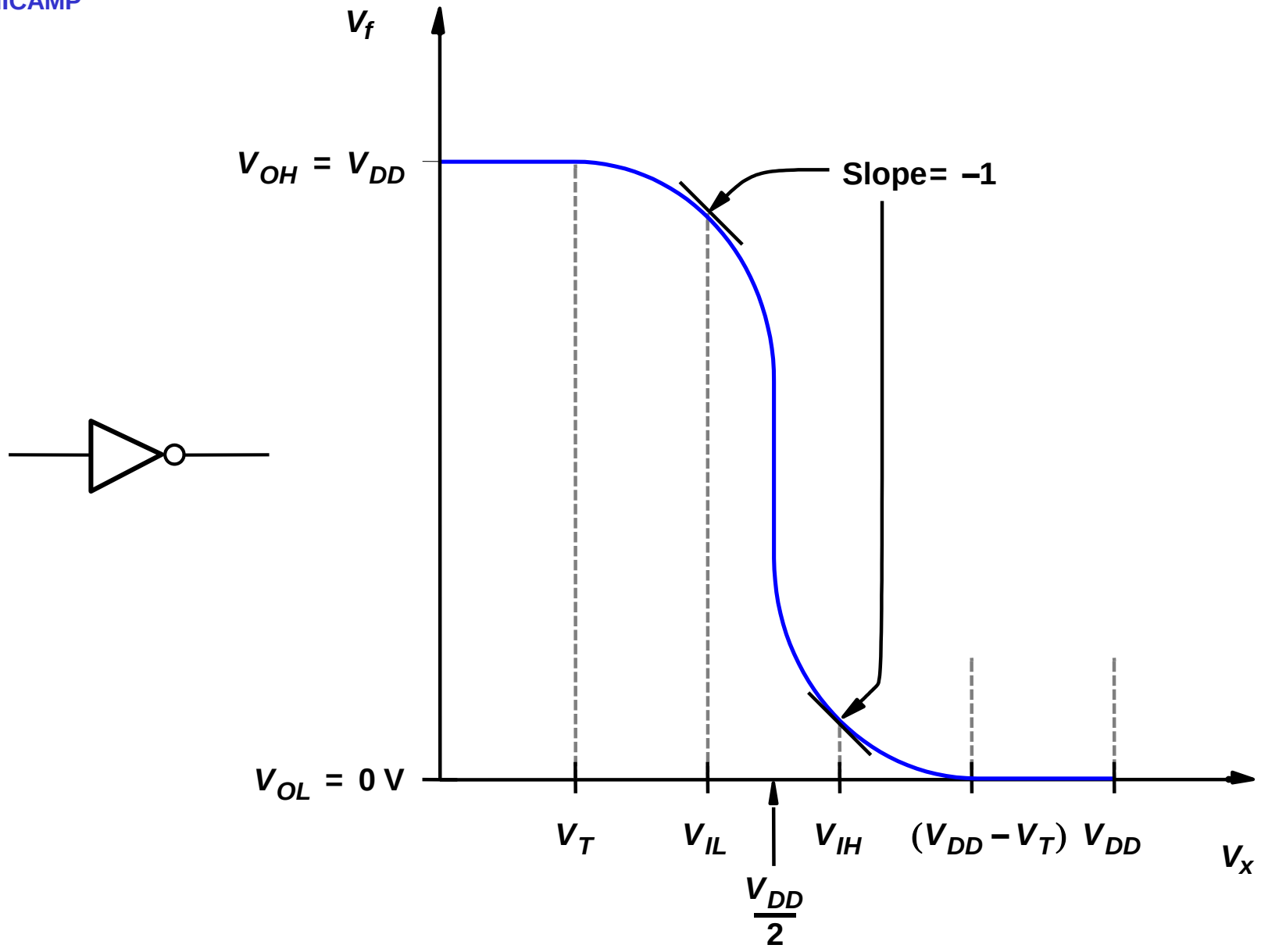




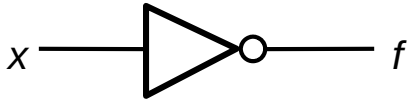


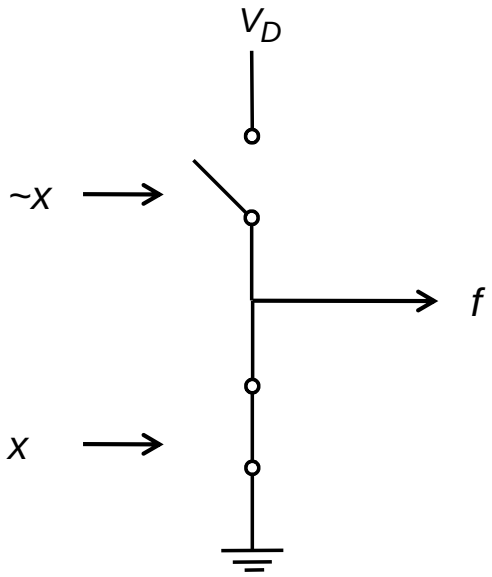
Característica de Transferência DC

IC-UNICAMP

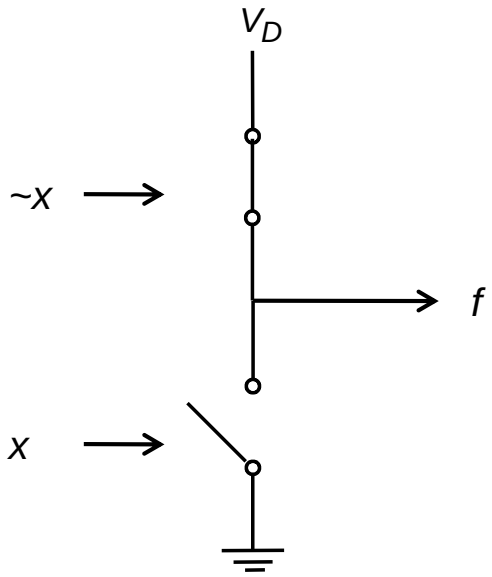


Saída ativa normal inversor CMOS



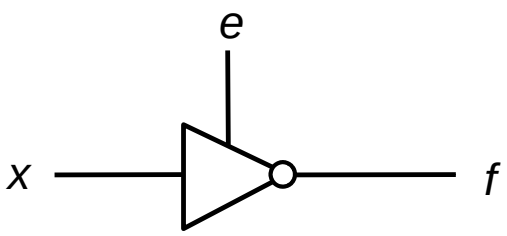


x = 1 → f = 0

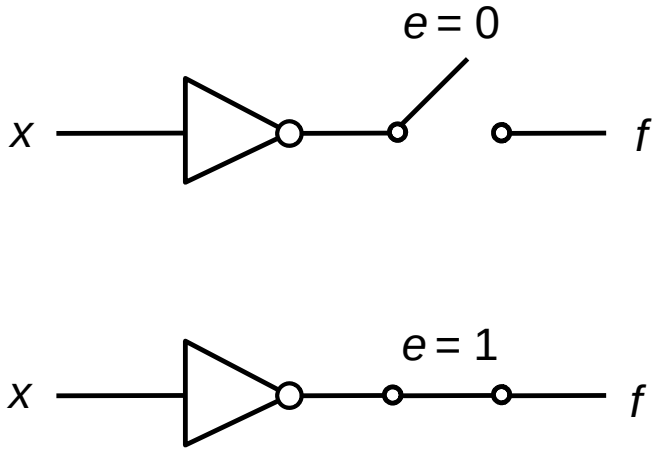


x = 0 → f = 1

Saída tri-state CMOS



Inversor tri-state

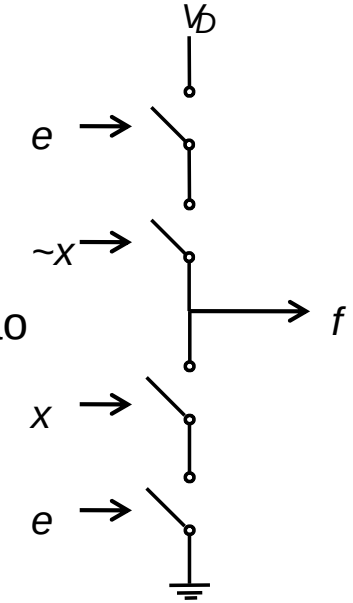


Circuito equivalente

e	x	f
0	0	Z
0	1	Z
1	0	1
1	1	0

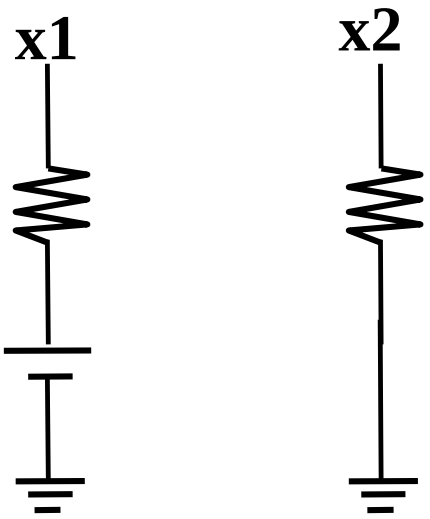
Tabela verdade

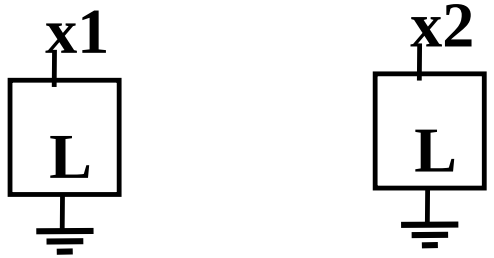
Implementação



Níveis lógicos fracos

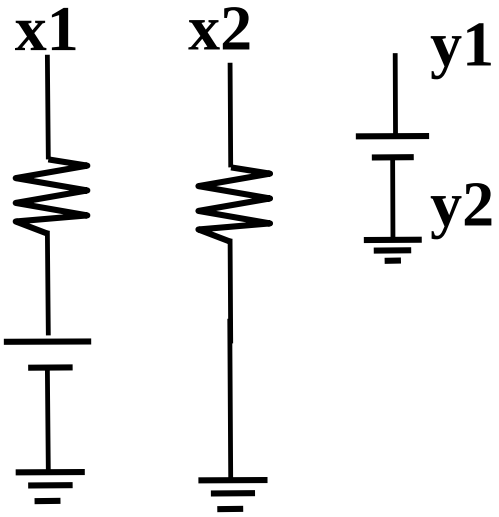
- Quais os valores de x1 e x2?
- As lâmpadas acendem?





Níveis lógicos fortes e fracos

- Quais os valores de x1, x2, y1 e y2

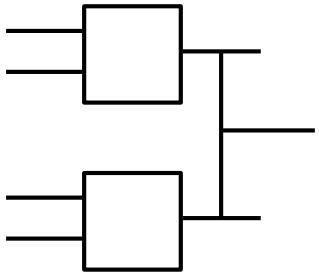


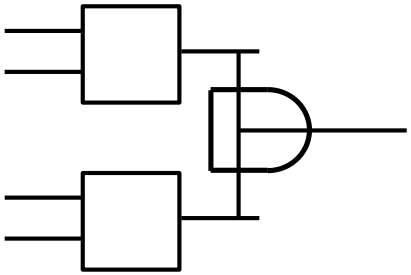
- O que acontece ao ligar

	x1	x2	y1	y2
x1	-			
x2		-		
y1			-	
y2				-

Wired AND, Wired OR

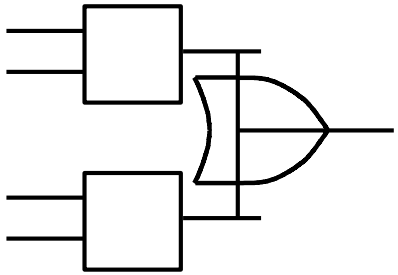
- O que acontece se ligarmos as saídas de circuitos lógicos em curto circuito?
- Dependendo da relação de força dos transitores pullup e pull down





Wired AND

(saída = 0 se alguma entrada for =0)



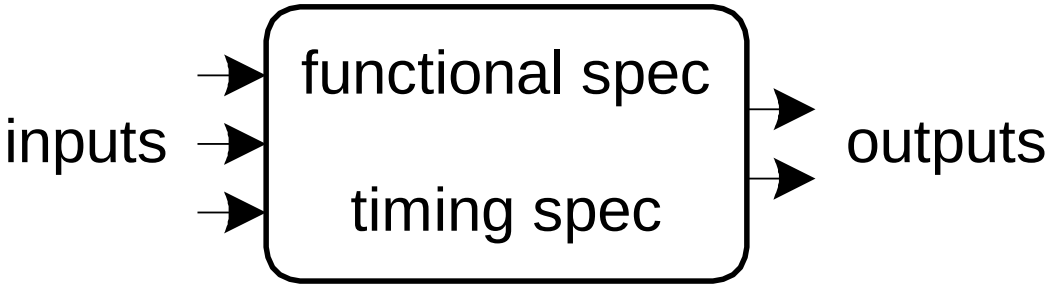
Wired OR

(saída = 1 se alguma entrada for =1)

Circuitos combinacionais – conceitos básicos

Tipos de Circuitos Lógicos

- Combinacional
  - Sem memória
  - As saídas são determinadas pelos valores correntes das entradas
- Seqüencial
  - Tem memória
  - As saídas são determinadas pelos valores anteriores e correntes das entradas



Forma Soma-de-Produtos (SOP)

- **Toda equação booleana pode ser descrita na forma SOP**
- **Cada linha da tabela verdade é associada a um mintermo**
- **Um mintermo é um produto (AND) de literais**
- **Cada mintermo é TRUE (1) para uma dada linha (e somente para essa linha)**
- **A função é formada pelo OR dos mintermos para os quais a saída é TRUE (1)**
- **Assim, a função é a soma (OR) de produtos (termos AND)**

A	B	Y	minterm
0	0	0	$\overline{A}\,\overline{B}$
0	1	1	$\overline{A}\,B$
1	0	0	$A\,\overline{B}$
1	1	1	$A\,B$

$$Y = F(A, B, C) = \overline{A}B + AB$$

IC-UNICAMP

Terminologia

- Literal - Uma variável complementada ou não em um termo produto (ou termo soma)
- Implicante – Um termo produto que implementa um ou mais 1´s da função. Exemplo: um míntermo é um implicante; um produto gerado pela simplificação de uma váriável de dois míntermos é um implicante.
- Implicante Principal – Um implicante que não pode ser simplificado em outro implicante com menos literais.
- Implicante Essencial – Implicante Principal que é imprescindivel na realização da função (existe pelo menos um “1” que só é coberto por ele).
- Cobertura – Uma coleção de implicantes que implementam a função (implementam todos os 1´s da função).
- Custo – número de portas + número de entradas de todas as portas (assumiremos que as entrads primárias estão disponíveis tanto na forma verdadeira quanto complementada).

Forma Produto-de-Somas (POS)

- Toda equação booleana pode ser descrita na forma POS
- Cada linha da tabela verdade é associada a um maxtermo
- Um maxtermo é uma soma (OR) de literais
- cada maxtermo é FALSE (0) para uma dada linha (e somente para essa linha)
- A função é formada pelo AND dos maxtermos para os quais a saída é False (0)
- Assim, a função é um produto (AND) de soma (termos OR)

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

$$Y = F(A, B, C) = (A + B)(\overline{A} + B)$$

Álgebra Booleana

- Conjunto de Axiomas e Teoremas: usados para simplificar equações Booleanas
- Similar à algebra regular, porém mais simples em muitos casos já que as variáveis só podem ter dois valores (1 ou 0)
- Axiomas e Teoremas obedecem aos principios da dualidade:
  - Trocando-se ANDs por Ors (e vice-versa) e 0's por 1's (e vice-versa)

Axiomas e Teoremas

	Axiom		Dual	Name
A1	$B = 0$ if $B \neq 1$	A1′	$B = 1$ if $B \neq 0$	Binary field
A2	$\overline{0} = 1$	A2′	$\overline{1} = 0$	NOT
A3	$0 \bullet 0 = 0$	A3′	$1 + 1 = 1$	AND/OR
A4	$1 \bullet 1 = 1$	A4′	$0 + 0 = 0$	AND/OR
A5	$0 \bullet 1 = 1 \bullet 0 = 0$	A5′	$1 + 0 = 0 + 1 = 1$	AND/OR

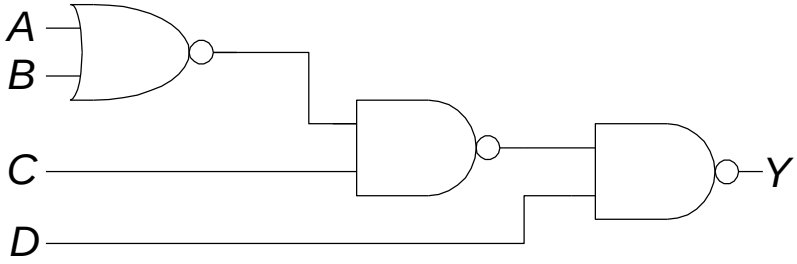
	Theorem		Dual	Name
T1	$B \bullet 1 = B$	T1′	$B + 0 = B$	Identity
T2	$B \bullet 0 = 0$	T2′	$B + 1 = 1$	Null Element
T3	$B \bullet B = B$	T3′	$B + B = B$	Idempotency
T4		$\overline{\overline{B}} = B$		Involution
T5	$B \bullet \overline{B} = 0$	T5′	$B + \overline{B} = 1$	Complements

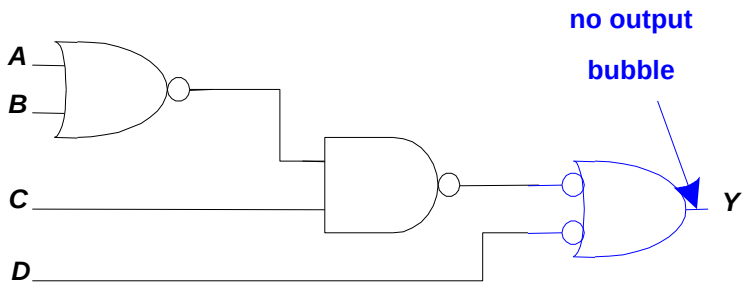
IC-UNICAMP

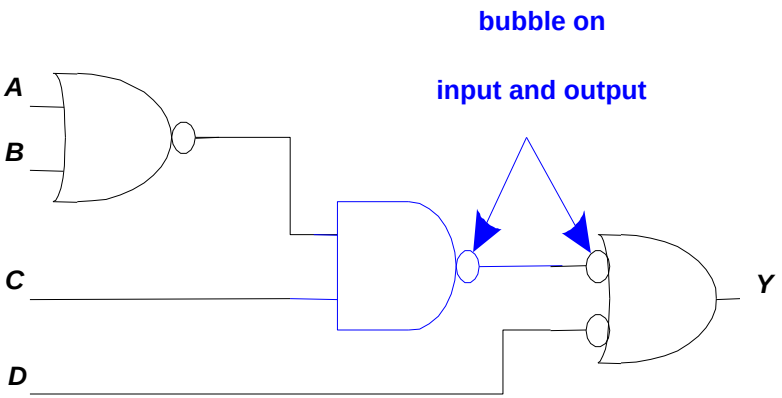
Teoremas

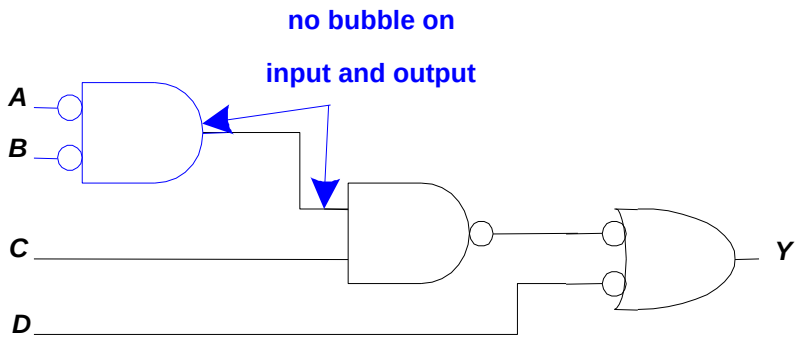
	Theorem		Dual	Name
T6	$B \bullet C = C \bullet B$	T6′	$B + C = C + B$	Commutativity
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	T7′	$(B + C) + D = B + (C + D)$	Associativity
T8	$(B \bullet C) + B \bullet D = B \bullet (C + D)$	T8′	$(B + C) \bullet (B + D) = B + (C \bullet D)$	Distributivity
T9	$B \bullet (B + C) = B$	T9′	$B + (B \bullet C) = B$	Covering
T10	$(B \bullet C) + (B \bullet \overline{C}) = B$	T10′	$(B + C) \bullet (B + \overline{C}) = B$	Combining
T11	$(B \bullet C) + (\overline{B} \bullet D) + (C \bullet D)$ $= B \bullet C + \overline{B} \bullet D$	T11′	$(B + C) \bullet (\overline{B} + D) \bullet (C + D)$ $= (B + C) \bullet (\overline{B} + D)$	Consensus
T12	$\overline{B_0 \bullet B_1 \bullet B_2 \ldots}$ $= (\overline{B_0} + \overline{B_1} + \overline{B_2} \ldots)$	T12′	$\overline{B_0 + B_1 + B_2 \ldots}$ $= (\overline{B_0} \bullet \overline{B_1} \bullet \overline{B_2})$	De Morgan's Theorem

Técnica Bubble Pushing









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

Tabela verdade

- AND e OR de 3 entradas
  - 2^3 combinações ou mintermos

x_1	x_2	x_3	$x_1 \cdot x_2 \cdot x_3$	$x_1 + x_2 + x_3$
0	0	0	0	0
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	1

Tabela verdade, mintermos

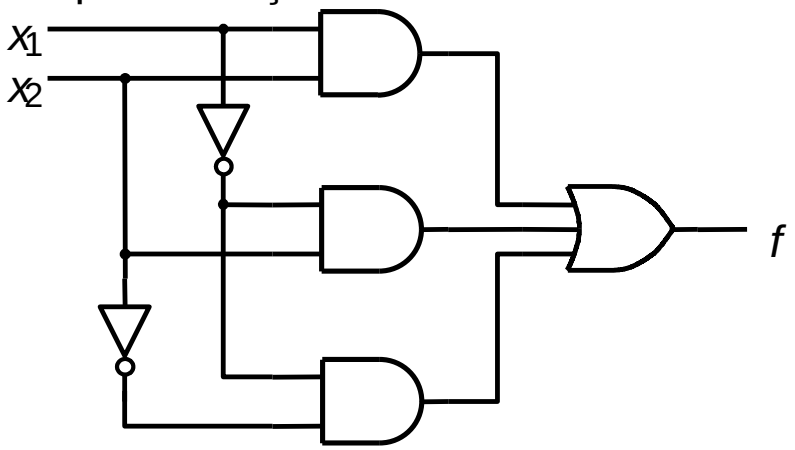
Tabela verdade

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

Soma de produtos canônica

f = Σ (0, 1, 3)

Implementação direta dos mintermos



Implementação de custo mínimo

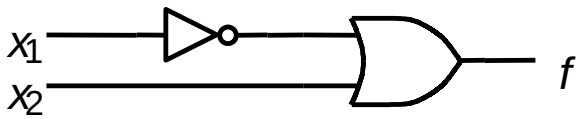
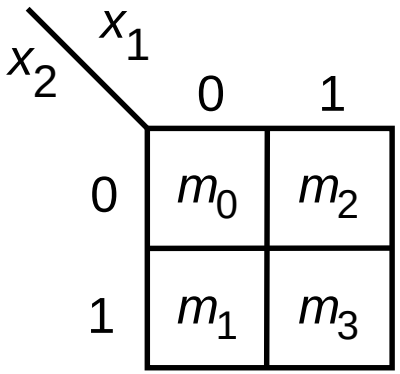


Tabela verdade e Mapa de Karnaugh

x_1	x_2	
0	0	m_0
0	1	m_1
1	0	m_2
1	1	m_3

(a) Truth table



(b) Karnaugh map

IC-UNICAMP

Procedimento para minimização

- Tabela verdade → mintermos
- Mapa de Karnaugh
- Identificar os implicantes principais para cobertura de todos os mintermos
- Identificar quais são essenciais e selecioná-los
- Verificar quais mintermos não foram cobertos pelos implicantes essenciais
- Selecionar implicantes principais para cobrir esses mintermos não cobertos

Mapa de Karnaugh e 3 variáveis

x_1	x_2	x_3	
0	0	0	m_0
0	0	1	m_1
0	1	0	m_2
0	1	1	m_3
1	0	0	m_4
1	0	1	m_5
1	1	0	m_6
1	1	1	m_7

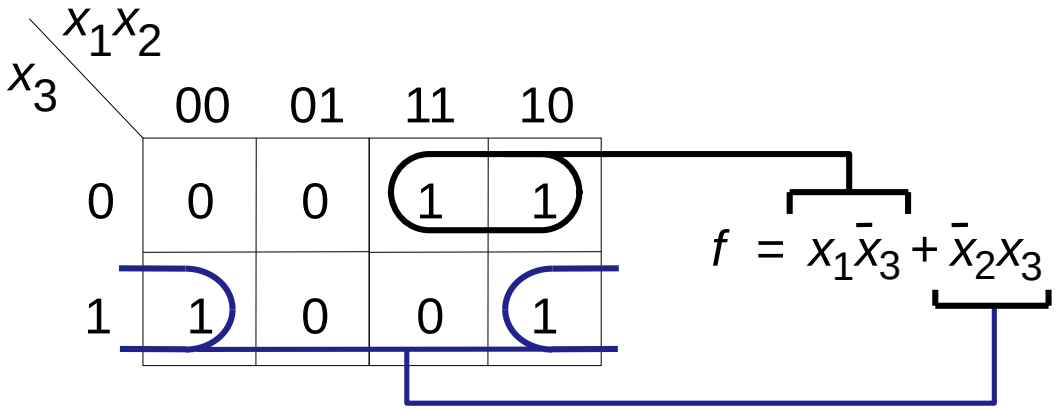
(a) Truth table

x_3 \ x_1x_2	00	01	11	10
0	m_0	m_2	m_6	m_4
1	m_1	m_3	m_7	m_5

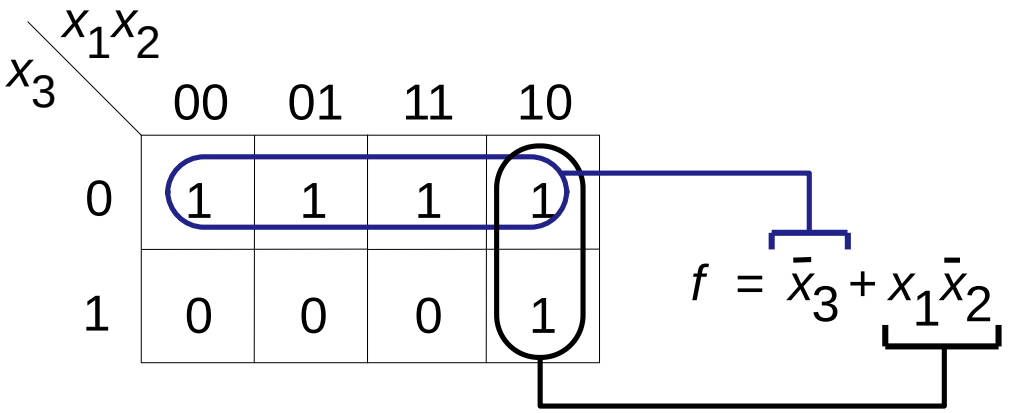
(b) Karnaugh map

Exemplos de funções de 3 variáveis

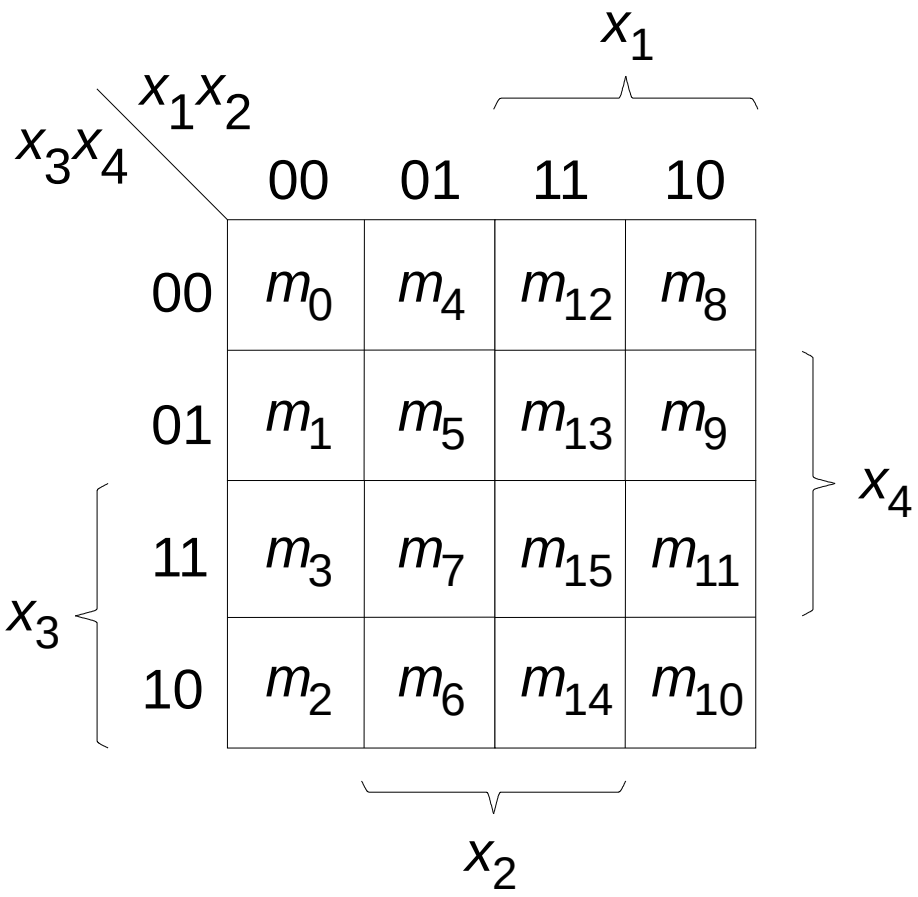
Função da Fig. 2.18



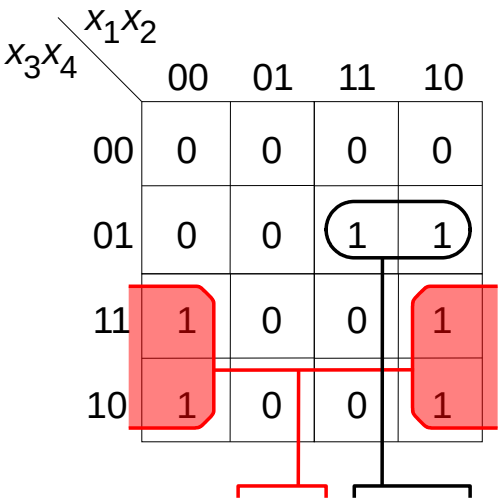
Função da Fig. 4.1



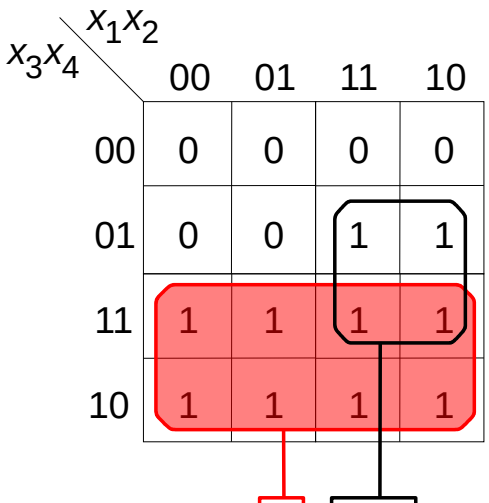
Karnaugh: 4 variáveis



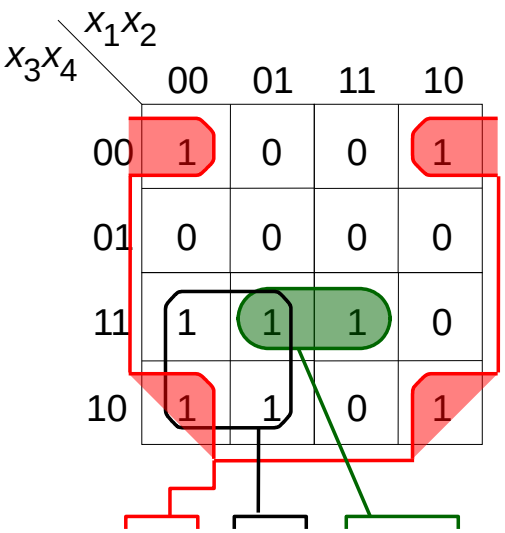
Exemplos de M.K. de 4 variáveis



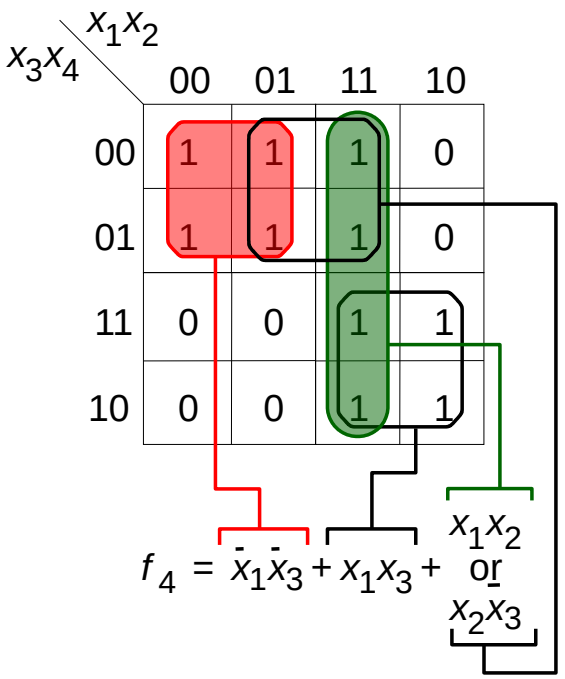
$f_1 = \bar{x}_2x_3 + x_1\bar{x}_3x_4$



$f_2 = x_3 + x_1x_4$

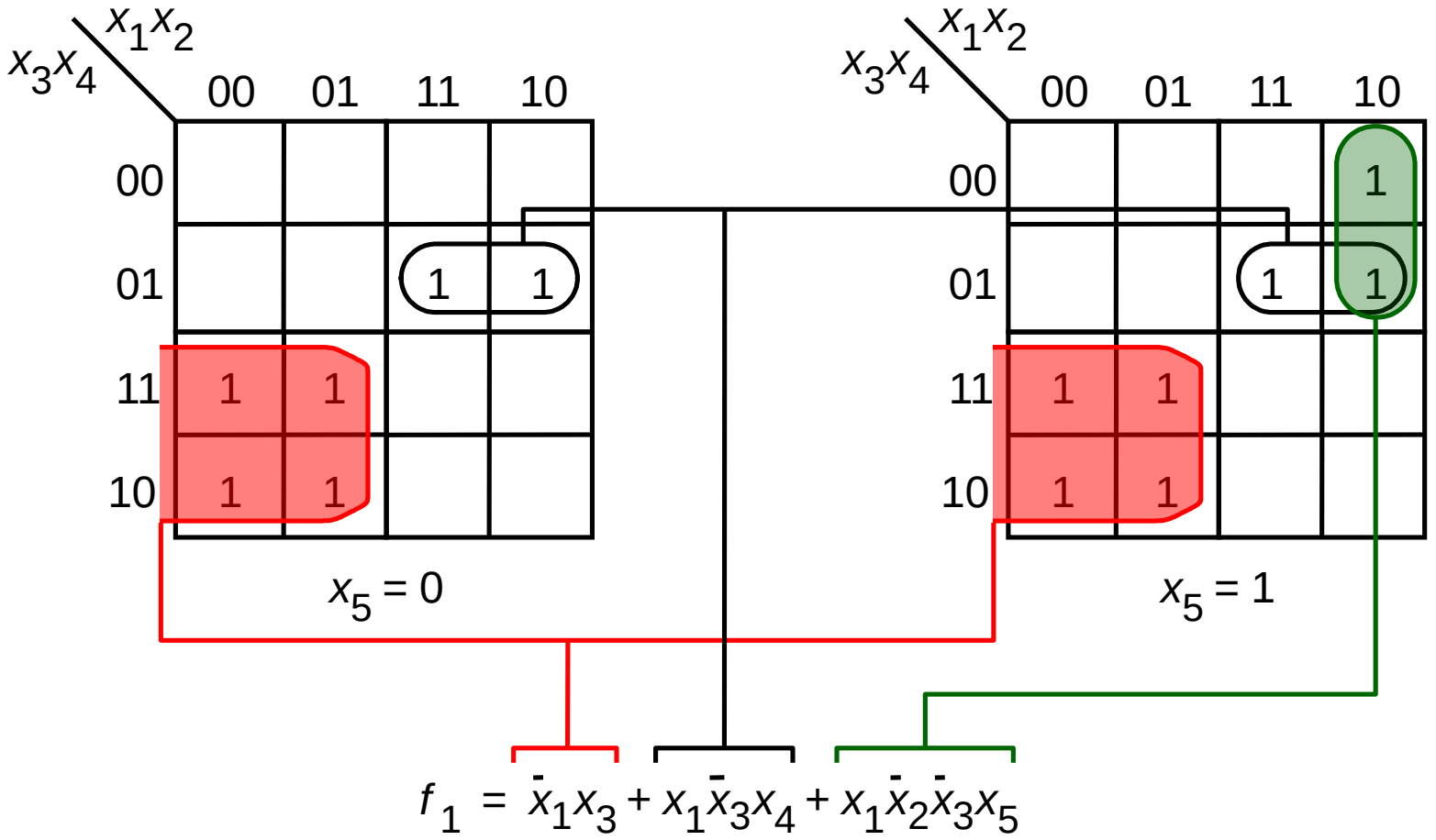


$f_3 = \bar{x}_2\bar{x}_4 + \bar{x}_1x_3 + x_2x_3x_4$



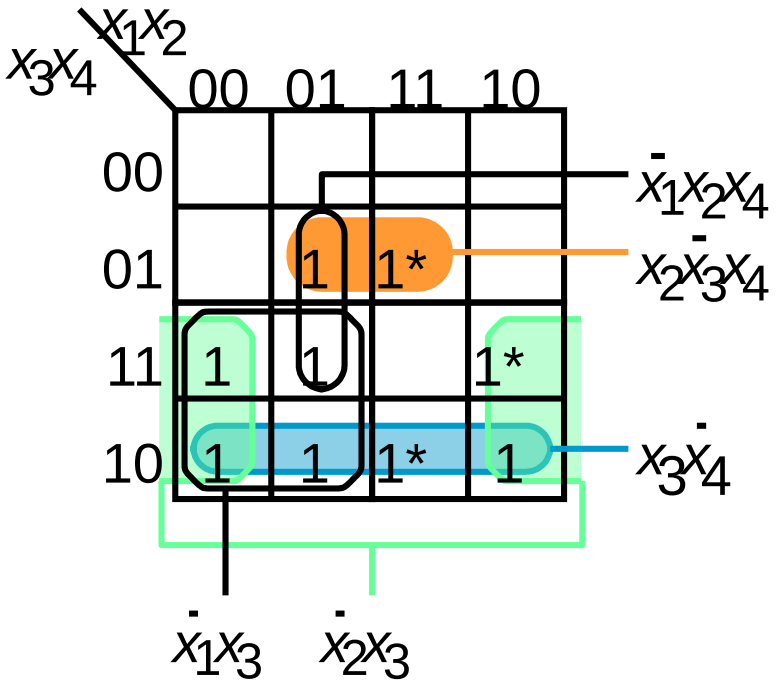
$f_4 = \bar{x}_1\bar{x}_3 + x_1x_3 +$ or x_2x_3

Mapa de Karnaugh de 5 variáveis

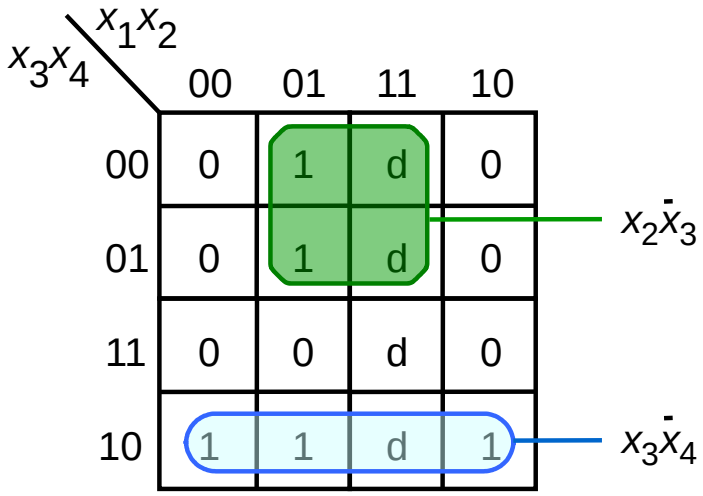


Implicante principal essencial

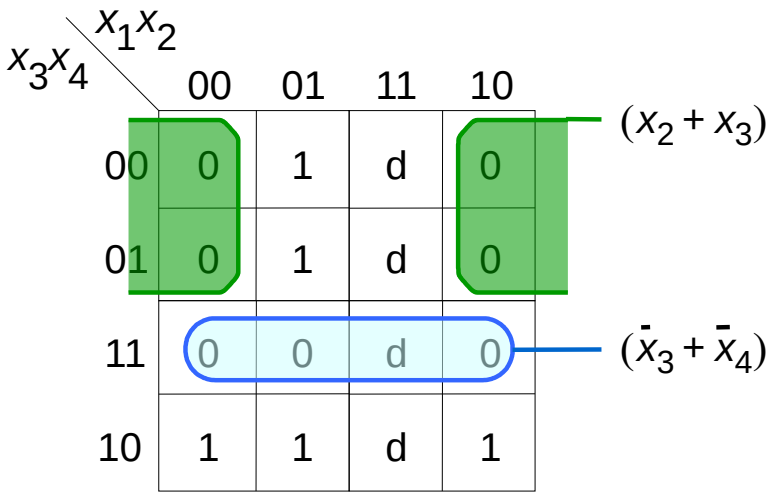
- Implicante principal é essencial se for o único a cobrir algum mintermo
- Exemplo: f = Σ m(2, 3, 5, 6, 7, 10, 11, 13, 14)
  - 5 implicantes principais
  - somente 3 são essenciais*
    - $\bar{x}_2 x_3$ devido a m11
    - $x_3 \bar{x}_4$ devido a m14
    - $x_2 \bar{x}_3 x_4$ devido a m13
  - faltou somente cobrir m7, e há 2 impl princ → escolher menor custo
  - f= $\bar{x}_2 x_3 + x_3 \bar{x}_4 + x_2 \bar{x}_3 x_4 + \bar{x}_1 x_3$



Uso de don´t care



(a) SOP implementation

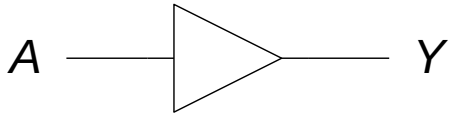


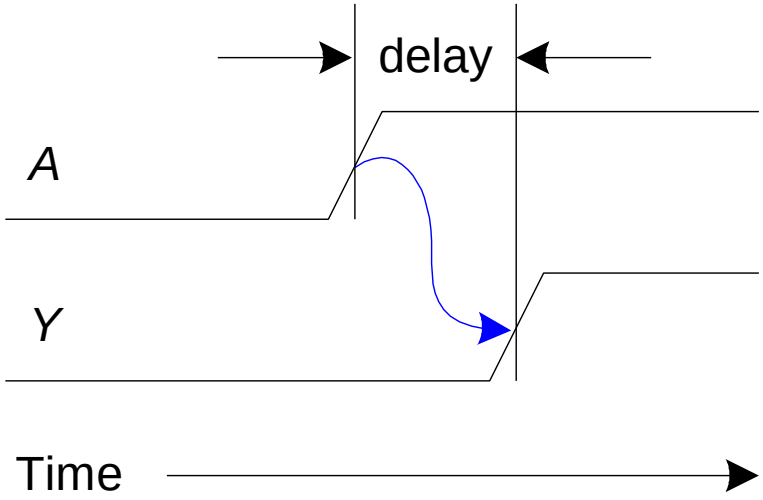
(b) POS implementation

$$f = \Sigma\, m(2, 4, 5, 6, 10) + D(12, 13, 14, 15)$$

Timing

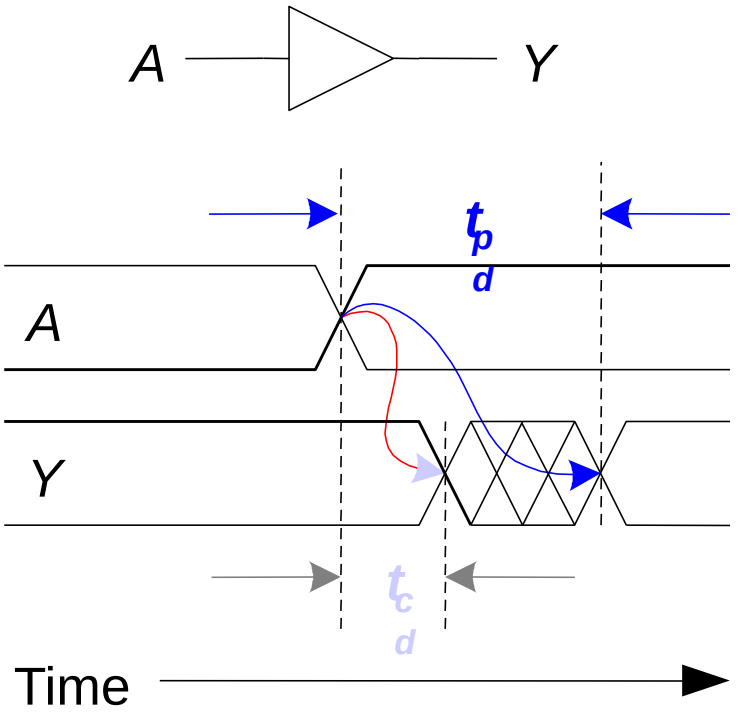
- Delay: atraso entre a mudança na entrada e na saída
- Um dos maiores desafios em projeto de circuitos: tornar o circuito mais rápido





Delay: Propagação e Contaminação

- Propagation delay: t_{pd} = max delay da entrada à saída
- Contamination delay: t_{cd} = min delay da entrada à saída

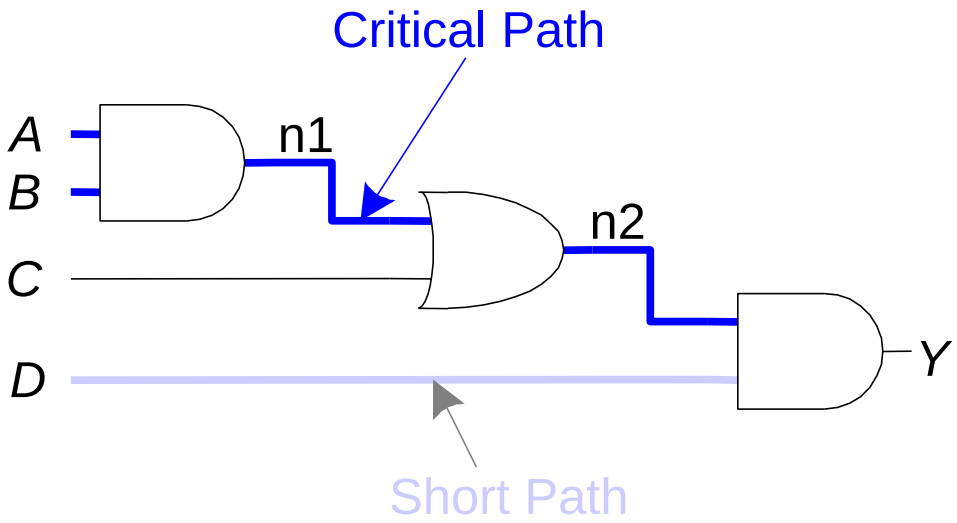


Delay: Propagação e Contaminação

- Os atrasos são causados por
  - Capacitância e
  - Resistências no circuito

- Razões porque t_{pd} and t_{cd} podem ser diferentes:
  - Diferentes tempos de subida (*rising*) e de descida (*falling*)
  - Múltiplas entradas e saídas, algumas podem ser mais rápidas do que as outras
  - Circuito mais lento quando quente e mais rápido quando frio

Caminhos: Críticos e Curtos



Critical (Long) Path: $t_{pd} = 2t_{pd\_AND} + t_{pd\_OR}$

Short Path: $t_{cd} = t_{cd\_AND}$

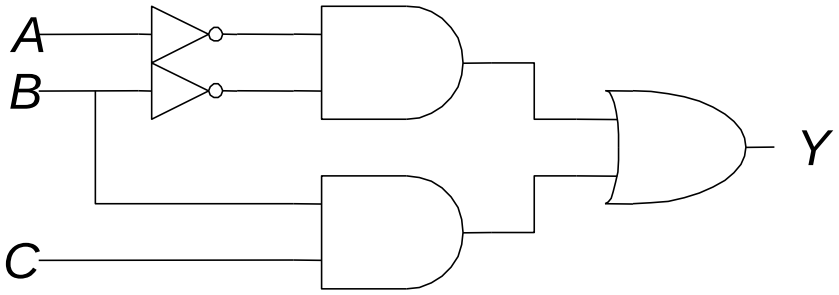
Glitches

- Um glitch ocorre quando uma mudança em uma entrada causa múltiplas mudanças na saída

- Glitches não causam problemas se seguirmos as convenções de projetos síncronos

- É importante reconhecer um glitch quando se vê um em uma simulação ou em um osciloscópio

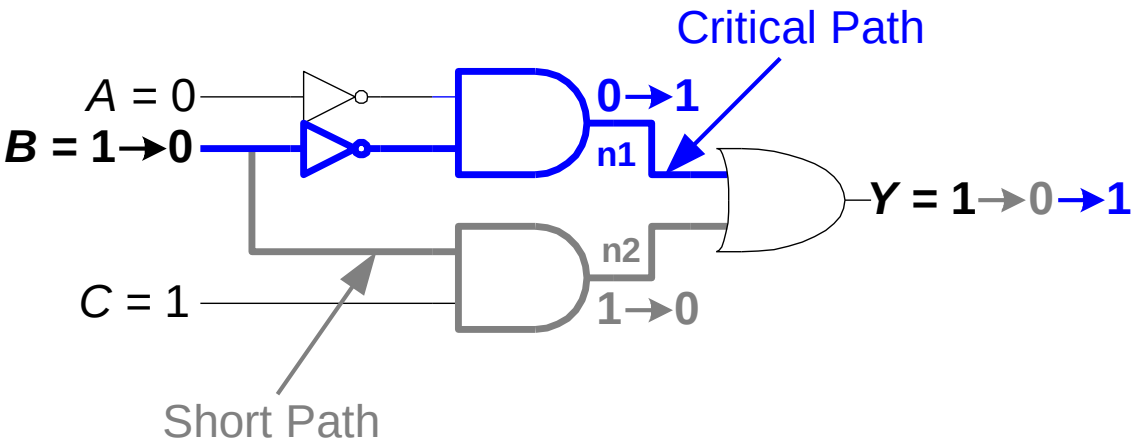
Exemplo de Glitch

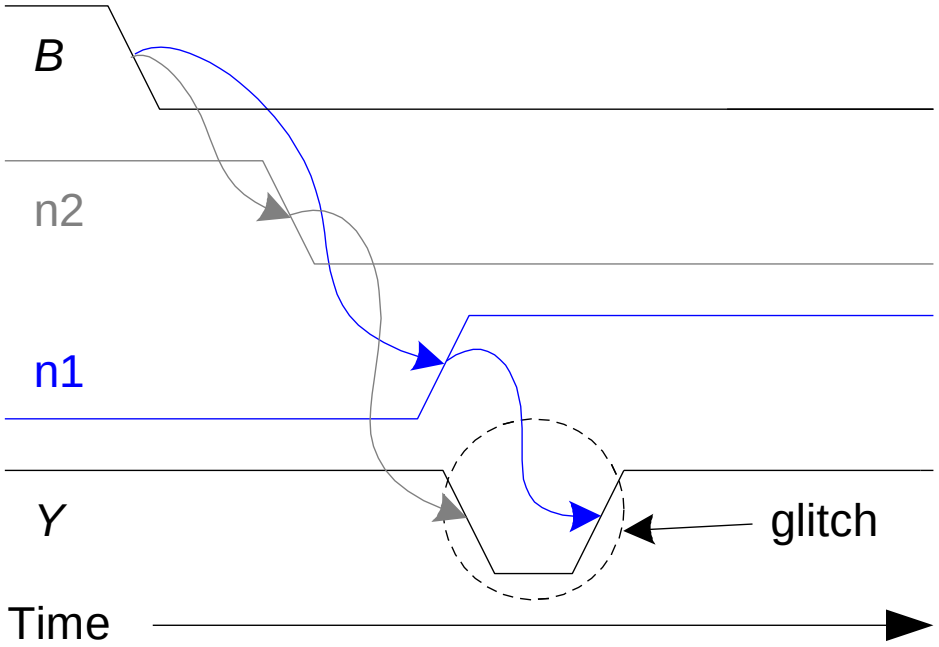


Y	AB = 00	01	11	10
C = 0	1	0	0	0
C = 1	1	1	1	0

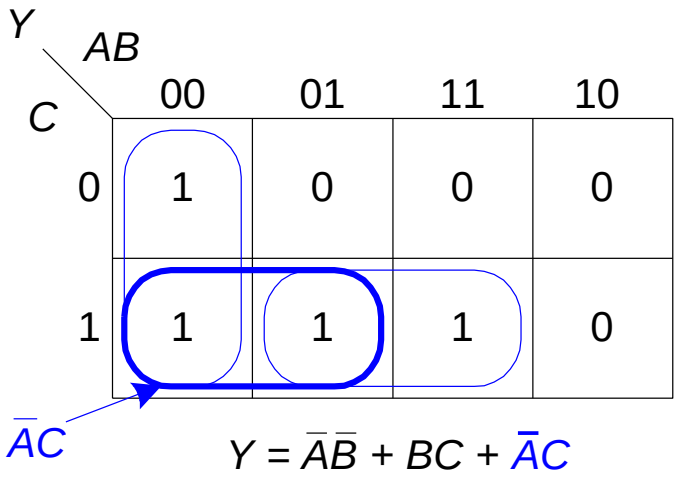
$$Y = \bar{A}\bar{B} + BC$$

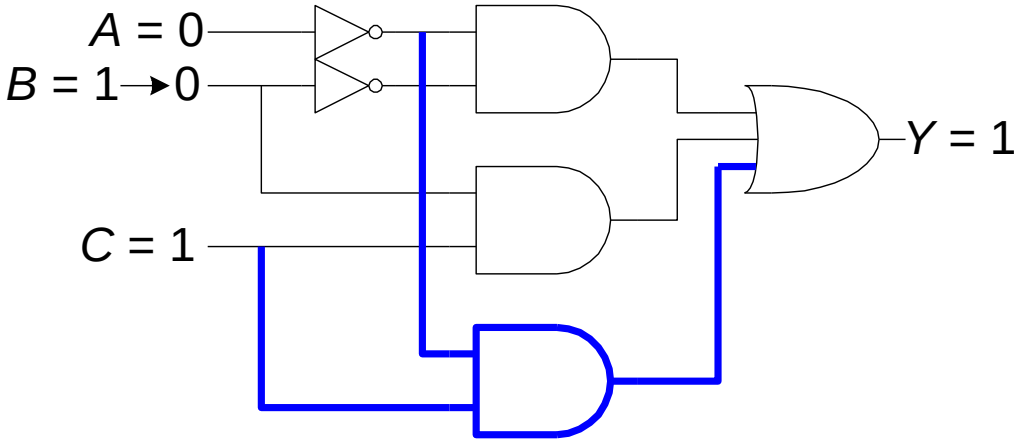
Exemplo de Glitch (cont.)





Exemplo de Glitch (cont.)



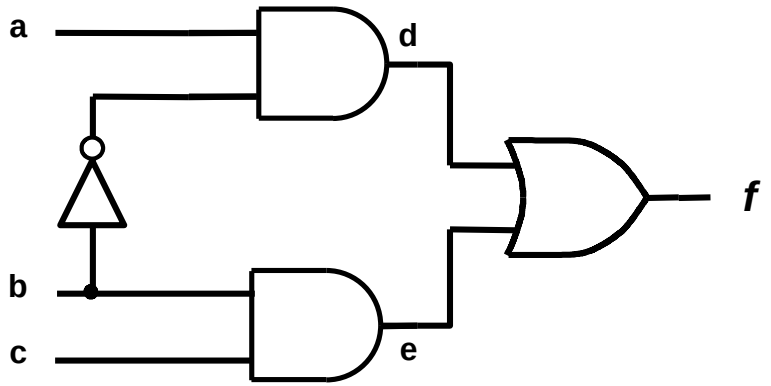


IC-UNICAMP

VHDL: introdução

- Linguagem de descrição de hardware: suporte para simulação e síntese (padrão IEEE)
- Como representar circuito combinacional simples?

Modelo completo de um circuito




```
Library IEEE;
use IEEE.std_logic_1164.all;

Entity exemplo IS
   Port (a, b, c : IN std_logic;
         f : OUT std_logic);
End exemplo;

Architecture estrutural OF exemplo IS
  signal d, e : std_logic;
Begin
   f  <=  d or e;
   d   <=  a and not(b);
   e   <=  b and c;
End estrutural
```

IC-UNICAMP

Principais blocos

```
Library IEEE;
use IEEE.std_logic_1164.all;
```

Cabeçalho:
- bibliotecas em uso

```
Entity exemplo IS
   Port (a, b, c : IN std_logic;
         f : OUT std_logic);
End exemplo;
```

Entity:
- Define o nome
- Define as interfaces
- Ports Inputs/Outputs
- Tipos de sinal

```
Architecture estrutural OF exemplo IS
  signal d, e : std_logic;
Begin
   f  <=  d or e;
   d   <=  a and (not b);
   e   <=  b and c;
End estrutural
```

Architecture:
- Descreve conteúdo funcional do componente
- Possívei mais de uma
- Definição de sinais internos
- Atribuição de sinais
- Ordem importa??

Conceitos básicos

- Sinais (no exemplo são os sinais: a, b, c, d, e, f)
  - Representam os “fios” do circuito
- Alguns tipos dos sinais

-- inicia um comentário

```
type bit is (‘0’, ‘1’);
type std_logic is (
            ‘U’, -- não iniciado (unitialized)
            ‘X’, -- desconhecido (unknow) forte
            ‘0’, -- zero forte
            ‘1’, -- um forte
            ‘Z’, -- alta impedância (tri-state)
            ‘W’, -- desconhecido fraco
            ‘L’, -- zero fraco
            ‘H’, -- um fraco
            ‘-’); -- indiferente (don’t care)
```

IC-UNICAMP

Construções de VHDL vistas nesta aula

- Cabeçalho e bibliotecas
- Entity: significado, ports, tipo de sinais
- Architecture
- Definição de sinais internos (não fazem parte da interface)
- Atribuição de sinais
- Alguns operadores booleanos
- Comandos concorrentes
- Tipos de sinal: bit e std_logic
- Convenção para comentário (--)