Circuitos Lógicos
Aula 11

Aula passada

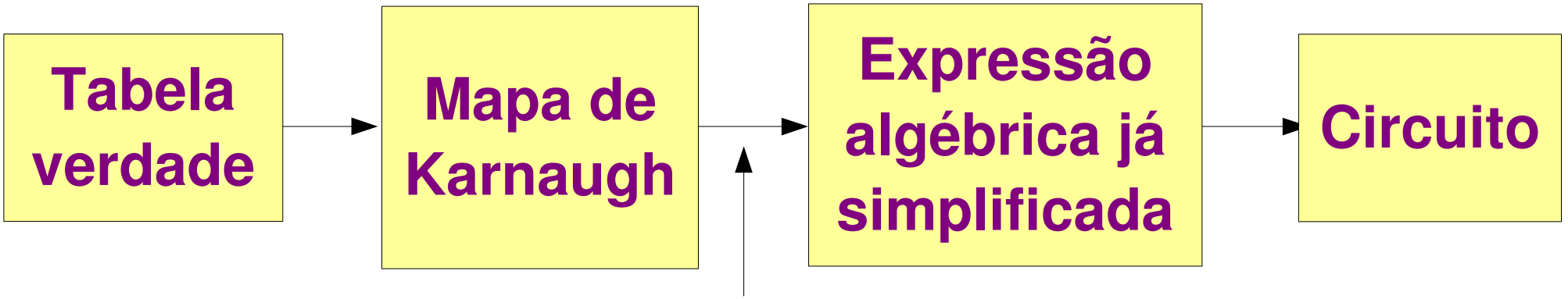
- Simplificação via Mapas de Karnaugh
- Mecânica
- Exemplos

Aula de hoje

- Condição de “don't care”
- Mais simplificação
- Exemplos
- Portas XOR, NXOR

Mapa de Karnaugh

- Representação gráfica conveniente da tabela verdade
- Permite obter expressão simplificada equivalente a tabela verdade



Condição de “Don't Care”

- Entrada da tabela verdade não ocorre ou seu resultado não é importante
  - valor da saída pode ser 0 ou 1
- Lógica do circuito não se importa com o valor

A	B	C	x
0	0	0	X ←
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	X ←

Por que esta condição é interessante?

↓

Permite maior simplificação

Explorando o “Don't Care”

Como explorar a condição de don't care?

A	B	C	x
0	0	0	X
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	X

- Construir mapa de Karnaugh
- Usar X para denotar posições de don't care
- Usar X como 0 ou 1 de acordo com a melhor simplificação
- Criar expressão mais simplificada

Exemplo de “Don't Care”

- Expressão simplificada?

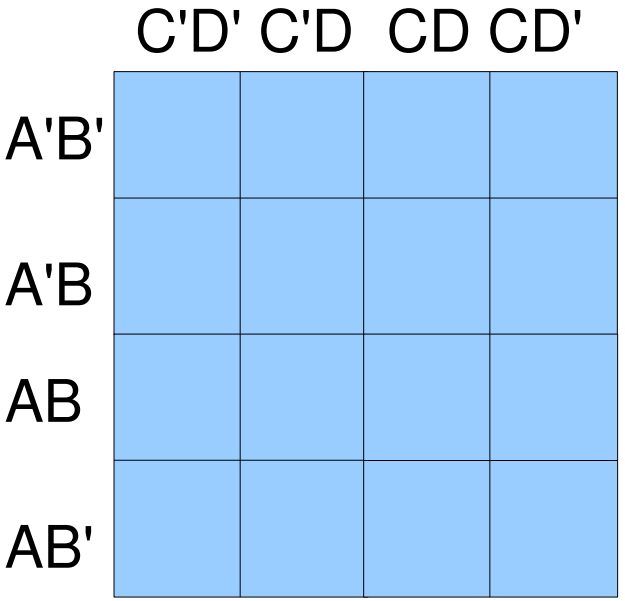
A	B	C	x
0	0	0	X
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	X

	C'	C
A'B'	**X**	**0**
A'B	**1**	**0**
AB	**0**	**X**
AB'	**0**	**1**

F = A'C' + AC

Exemplo de “Don't Care”

- Expressão simplificada?



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

Detalhes da Simplificação

- Um valor 1 pode ser circulado diversas vezes
- Um valor 1 pode participar de diferentes termos SOP
  - usamos isto na aula passada
- Regra vale sempre!

	C'	C
A'B'	0	1
A'B	0	0
AB	0	1
AB'	0	1

F = B'C + AC

	C'D'	C'D	CD	CD'
A'B'	1	1	0	1
A'B	1	1	0	1
AB	0	0	0	0
AB'	0	0	0	0

F = A'C' + A'D'

Processo de Simplificação

- 1) Construir mapa de Karnaugh a partir da tabela verdade
- 2) Circular os valores 1 que não são adjacentes a nenhum outro
- 3) Circular os valores 1 que aparecem em pares
- 4) Circular os valores 1 que aparecem em 4-tuplas (mesmo que já tenham sido circulados)
- 5) Circular os valores 1 que aparem em 8-tuplas (mesma que já tenham sido circulados
- 6) Escrever o SOP dos termos gerados por cada loop **(reusando valores 1)**

Exemplo

	C'D'	C'D	CD	CD'
A'B'	1	0	1	1
A'B	1	0	0	0
AB	1	1	1	1
AB'	0	0	1	1

	C'	C
A'B'	1	1
A'B	0	0
AB	0	1
AB'	1	1

F = AB + AC + B'C + A'C'D'

F = B' + ABC

F = B' + AC

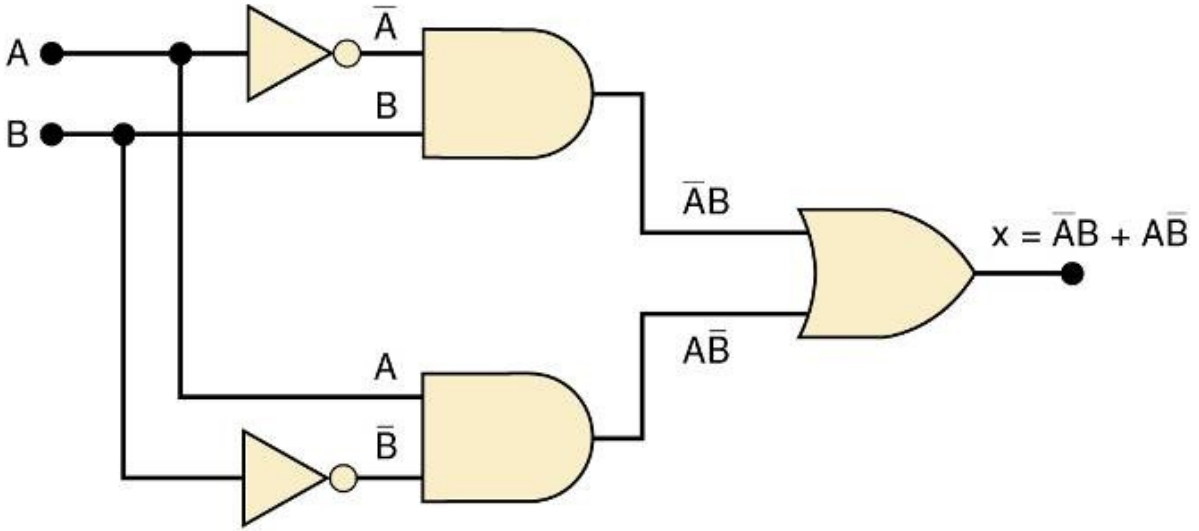
XOR

- Ou-exclusivo, conhecido como XOR (eXclusive OR)
- Porta lógica de ordem mais alta
  - como NAND e NOR
- Vale 1 apenas quando entradas são diferentes

Tabela Verdade XOR

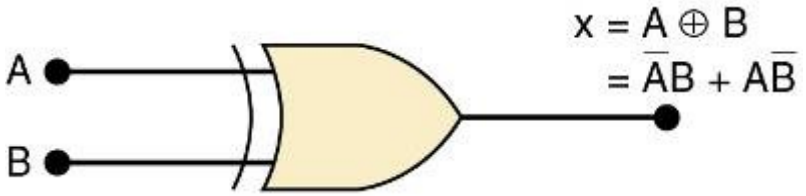
A	B	x
0	0	0
0	1	1
1	0	1
1	1	0

- Expressão booleana?
- A'B + AB'
- Circuito?

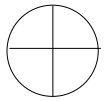


XOR

- XOR tem muitas aplicações
- Símbolo para porta XOR
  - “macro” para circuito anterior



- Símbolo para expressão booleana



- Exemplo: A ⊕ B = A XOR B

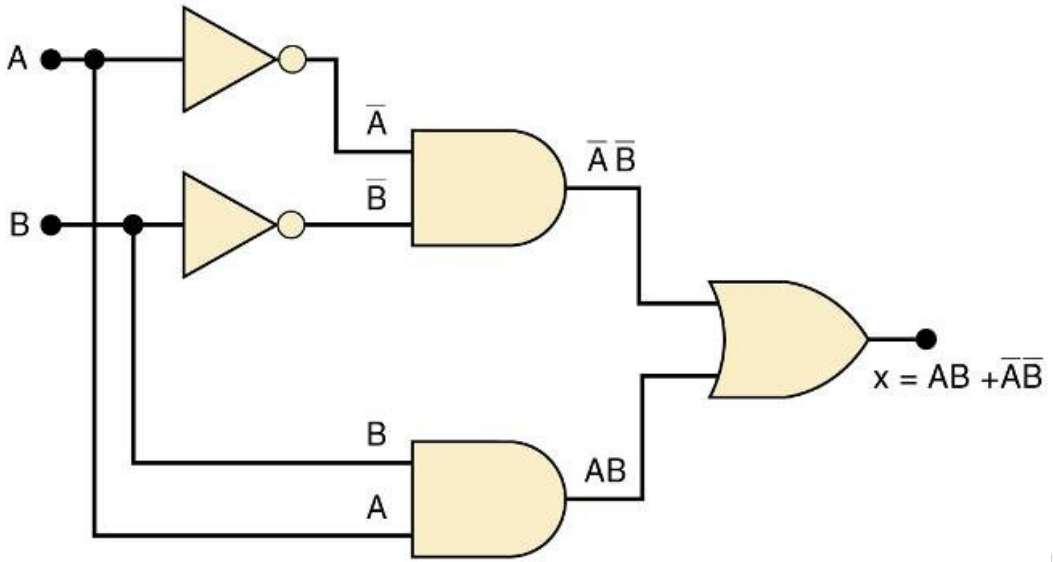
NXOR

- NOR-exclusivo, conhecido como NXOR (Not eXclusive OR)
- Vale 1 apenas quando entradas são iguais

Tabela Verdade NXOR

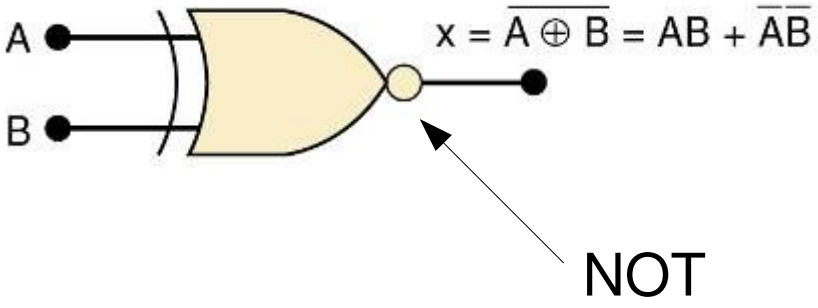
A	B	x
0	0	1
0	1	0
1	0	0
1	1	1

- Expressão booleana?
- A'B' + AB
- Circuito?



NXOR

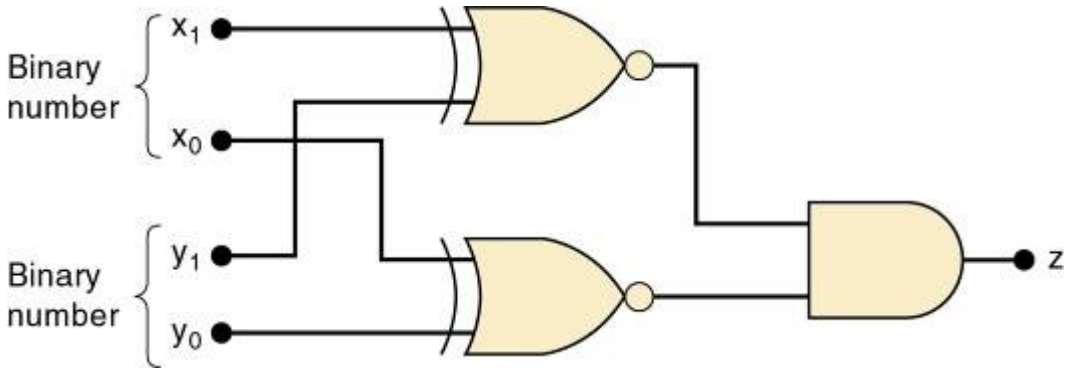
- NXOR tem muitas aplicações
- Símbolo para porta NXOR
  - “macro” para circuito anterior



- Expressão boolena: A NXOR B = $\overline{A \oplus B}$

Números iguais

- Problema: determinar se dois números binários de dois bits são iguais
- Entrada: x1, x0, y1, y0
- Idéia do circuito usando XOR e NXOR?
- x1 == y1 e y0 == y0



x_1	x_0	y_1	y_0	z (Output)
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
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