CIRCUITOS DIGITAIS
MAPA DE KARNAUGH

Marco A. Zanata Alves

Slides baseados nos slides de Rodrigo Hausen - CMCC – UFABC (2013)
http://compscinet.org/hausen/courses/circuitos/

SIMPLIFICAÇÃO DE LÓGICAS

Teorema: Toda função lógica pode ser escrita como disjunção de **mintermos** (também chamada “soma de produtos” – SOP).

Portanto, toda função lógica possui uma expressão que a define.

A forma de soma de produtos é uma forma padrão de representação de expressões booleanas. Outra forma padrão é o **produto de somas (maxtermos)**.

SIMPLIFICAÇÃO DE LÓGICAS

Como podemos simplificar tal expressão?

$$\overline{V}FU\overline{N} + \overline{V}FUN + VFU\overline{N} + VFUN$$

SIMPLIFICAÇÃO DE LÓGICAS

Como podemos simplificar tal expressão?

$$\overline{V}FU\overline{N} + \overline{V}FUN + VFU\overline{N} + VFUN$$

$$FU(\overline{VN} + \overline{V}N + V\overline{N} + VN)$$

$$FU\left(\overline{V}\left(\overline{N} + N\right) + V\left(\overline{N} + N\right)\right)$$

$$FU\left(\overline{V} + V\right)$$

$$FU$$

SIMPLIFICAÇÃO DE LÓGICAS

Observe que, quando temos algo do tipo:

$$\ldots + A\overline{B} + AB + \cdots$$

em uma expressão na forma soma-de-produtos podemos colocar A em evidência:

$$\ldots + A(\overline{B} + B) + \cdots$$

e simplificar por:

$$\ldots + A + \cdots$$

SIMPLIFICAÇÃO DE LÓGICAS

Observe que, quando temos algo do tipo:

$$\ldots + A\overline{B} + AB + \cdots$$

em uma expressão na forma soma-de-produtos podemos colocar A em evidência:

$$\ldots + A(\overline{B} + B) + \cdots$$

e simplificar por:

$$\ldots + A + \cdots$$

O problema sempre foi: Como encontrar dois mintermos idênticos a menos de uma mesma variável B, que aparece como B e $\overline{B}$?

Solução: expresse a tabela verdade de forma que isso seja fácil de encontrar!

MAPAS DE KARNAUGH

MAPAS DE KARNAUGH

Considerando a seguinte tabela verdade

Como gerar a lógica que implementa a função f(A,B,C) ?

Como implementar essa lógica de maneira otimizada?

A	B	C	f	
0	0	0	0	
0	0	1	0	
0	1	0	1	<= !A · B · !C
0	1	1	1	<= !A · B · C
1	0	0	1	<= A · !B · !C
1	0	1	0	
1	1	0	0	
1	1	1	1	<= A · B · C

R1. Regras de álgebra booleana

R2. **Mapas de Karnaugh**

MAPAS DE KARNAUGH...

Os mapas de Karnaugh permite simplificar funções utilizando **técnicas de mapeamento visual**

Para iniciar, precisamos criar uma grade para mapear as entradas (ex. A,B,C)

A	B	C	f	
0	0	0	0	
0	0	1	0	
0	1	0	1	<= !A · B · !C
0	1	1	1	<= !A · B · C
1	0	0	1	<= A · !B · !C
1	0	1	0	
1	1	0	0	
1	1	1	1	<= A · B · C

MAPAS DE KARNAUGH...

Representação em matriz para a tabela verdade, onde **em linhas ou colunas adjacentes apenas uma variável muda de 1 para 0 ou vice-versa.**

A\BC	00	01	11	10
0	0	0	1	1
1	1	0	1	0

A	B	C	f	
0	0	0	0	
0	0	1	0	
0	1	0	1	<= !A · B · !C
0	1	1	1	<= !A · B · C
1	0	0	1	<= A · !B · !C
1	0	1	0	
1	1	0	0	
1	1	1	1	<= A · B · C

MAPAS DE KARNAUGH...

Representação em matriz para a tabela verdade, onde **em linhas ou colunas adjacentes apenas uma variável muda de 1 para 0 ou vice-versa.**

A\BC	00	01	11	10
0	0	0	1	1
1	1	0	1	0

A · !B · !C

A	B	C	f	
0	0	0	0	
0	0	1	0	
0	1	0	1	<= !A · B · !C
0	1	1	1	<= !A · B · C
1	0	0	1	**<= A · !B · !C**
1	0	1	0	
1	1	0	0	
1	1	1	1	<= A · B · C

MAPAS DE KARNAUGH...

A\BC	$\overline{B}\,\overline{C}$ =00	$\overline{B}C$ =01	BC =11	$B\overline{C}$ =10
$\overline{A}$=0	0	0	1	1
A=1	1	0	1	0

A · !B · !C

MAPAS DE KARNAUGH...

A\BC	$\overline{B}\,\overline{C}$ 00	$\overline{B}C$ 01	BC 11	$B\overline{C}$ 10
$\overline{A}$=0	0	0	1	1
A=1	1	0	1	0

!A · B (C + !C) = **!A · B**

MAPAS DE KARNAUGH

A\BC	$\overline{B}\,\overline{C}$ 00	$\overline{B}C$ 01	BC 11	$B\overline{C}$ 10
$\overline{A}$=0	0	0	1	1
A=1	1	0	1	0

!A · B (C + !C) = **!A · B**

A · !B · !C

A · !A (B · C) = **B · C**

$$\boldsymbol{A\,\overline{B}\,\overline{C} + \overline{A}B + BC}$$

MAPAS DE KARNAUGH

Simplifique:

$$C = \overline{V}\,F\,\overline{U}\,N + V\,\overline{F}\,\overline{U}\,N + \overline{V}\,\overline{F}\,UN + \overline{V}FUN + VFUN + \overline{V}\,FU\,\overline{N} + VFU\overline{N} + V\,\overline{F}\,U\,\overline{N}$$

MAPAS DE KARNAUGH

\UN VF\	00	01	11	10
00	0	0	1	0
01	0	1	1	1
11	0	0	1	1
10	0	1	0	1

$$C = FU + \overline{V}FN + \overline{V}UN + VUN + V\overline{F}\,\overline{U}N$$

MAPAS DE KARNAUGH

Simplifique:

$$C = \overline{A}\,\overline{B}\,\overline{C} + A\,\overline{B}\,\overline{C} + \overline{A}\,B\,\overline{C} + AB\overline{C}$$

MAPAS DE KARNAUGH - EXEMPLOS

A\BC	00	01	11	10
0	1	0	0	1
1	1	0	0	1

$\Rightarrow$ **(!B · !C) + (B · !C)**

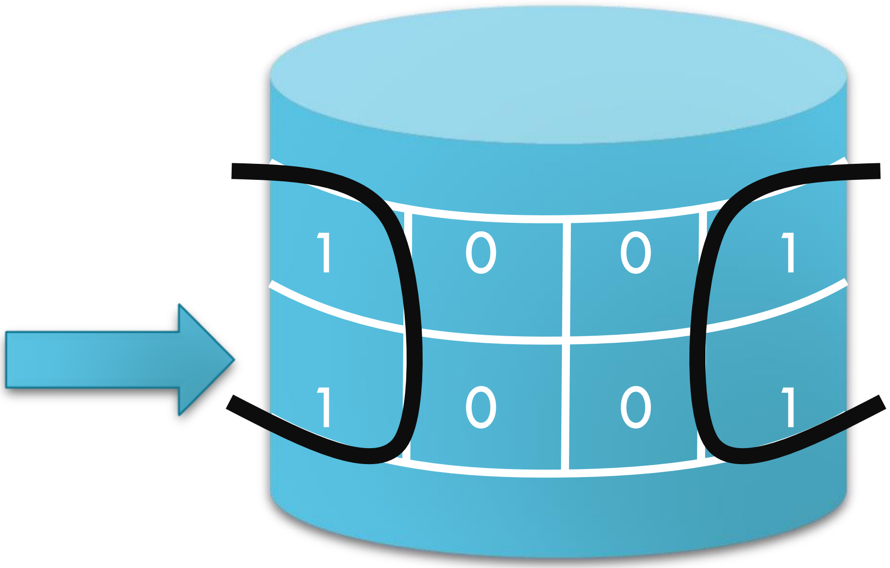
$\Rightarrow$ **!C(!B+B)**

$\Rightarrow$ **!C**

Será que poderíamos inferir isso pelo mapa?

MAPAS DE KARNAUGH - EXEMPLOS

A\BC	00	01	11	10
0	1	0	0	1
1	1	0	0	1



=> !C

MAPAS DE KARNAUGH - EXEMPLOS

A\BC	00	01	11	10
0	0	0	0	0
1	1	0	0	1

1 Mintermo (SDP):
(A · !C)

2x Maxtermos (PDS):
(A) · (!C)

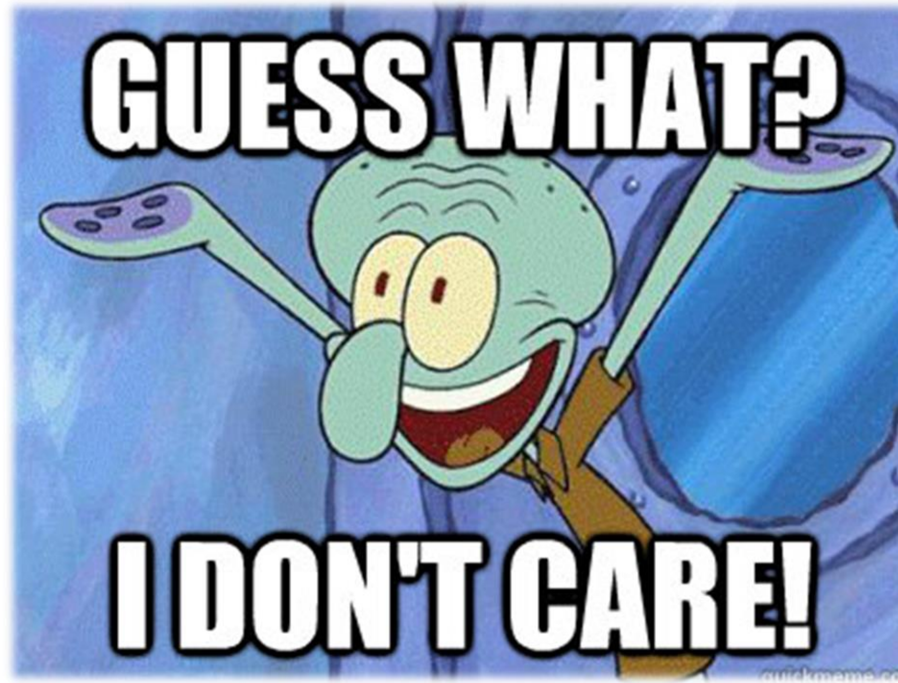
BITS DE DON'T-CARE

BITS DON'T-CARE (SEM IMPORTÂNCIA)

Bit de Don't-Care (X) é uma sequência de entrada para qual a saída não importa.

Uma entrada a qual é conhecida por nunca ocorrer é chamada de **Can't-Happen**.

Os dois casos são tratado de mesma forma no projeto de lógica, e são referidos de modo genérico por **don't-care**.



BITS DON'T-CARE (SEM IMPORTÂNCIA)

Bit de Don't-Care (X) é uma sequência de entrada para qual a saída não importa.

Uma entrada a qual é conhecida por nunca ocorrer é chamada de **Can't-Happen**.

Os dois casos são tratado de mesma forma no projeto de lógica, e são referidos de modo genérico por **don't-care**.

O projetista não precisa se importar com a saída gerada para esses termos, logo essa saída pode ser tratada da forma que for mais conveniente (gerar menor circuito).

Ex. Ativar a seta para direita e para esquerda ao mesmo tempo num carro.

MAPAS DE KARNAUGH - EXEMPLOS

Don´t cares (X) podem ser utilizados se necessário para aumentar a cobertura e assim simplificar a lógica

A\BC	00	01	11	10
0	0	X	0	0
1	X	X	1	1

=> A · B

MAPAS DE KARNAUGH - EXEMPLOS

Don´t cares (X) podem ser utilizados se necessário para aumentar a cobertura e assim simplificar a lógica

A\BC	00	01	11	10
0	0	X	0	0
1	X	X	1	1

A\BC	00	01	11	10
0	0	X	0	0
1	X	X	1	1

=> A · B

=> A

MAPAS DE KARNAUGH

Note que nosso mapa de Karnaugh não seguiu a ordem crescente para os termos

Isso se deve a um requisito dos mapas de Karnaugh, entre células adjacentes, deve ter apenas 1 bit mudando por vez

Podemos gerar uma lista dessas utilizando **codificação gray**

A\BC	00	01	11	10
0	0	0	1	1
1	1	0	1	0

GRAY CODING
"CODIFICAÇÃO CINZA"

GRAY CODING

Trata-se de um algoritmo para geração de uma sequência de números onde apenas um bit muda por vez

GRAY CODING

Para criar uma codificação gray, começamos com 1 dígito (0 ou 1)

Para cada novo dígito a ser adicionado uma função de espelho é aplicada

E em cada parte do espelho adiciona-se 0s ou 1s

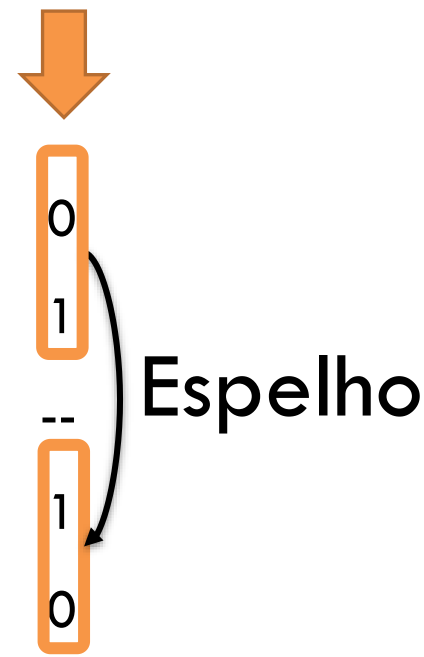
0
1

GRAY CODING

Para criar uma codificação gray, começamos com 1 dígito (0 ou 1)

Para cada novo dígito a ser adicionado uma função de espelho é aplicada

E em cada parte do espelho adiciona-se 0s ou 1s

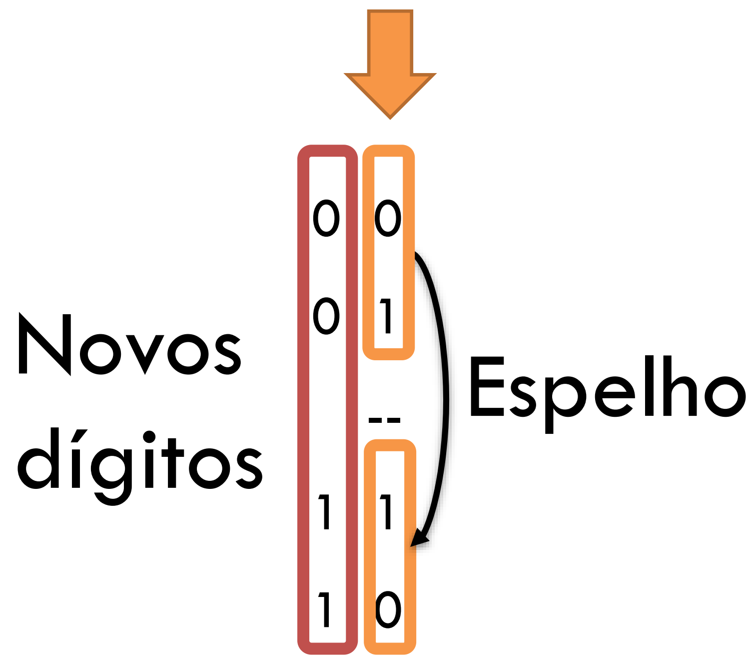


GRAY CODING

Para criar uma codificação gray, começamos com 1 dígito (0 ou 1)

Para cada novo dígito a ser adicionado uma função de espelho é aplicada

E em cada parte do espelho adiciona-se 0s ou 1s



ALGORITMO PARA O MAPA DE KARNAUGH

ALGORITMO DE MAPAS DE KARNAUGH PARA ATÉ 4 VARIÁVEIS

1) Expresse a tabela verdade como uma matriz, com no máximo duas variáveis para as linhas/colunas. Em linhas ou colunas adjacentes, apenas uma das variáveis muda (use gray coding).

2) Enquanto houver uma célula contendo 1 que não tiver sido agrupada, agrupe nesta ordem:

- Retângulos com 16 uns (Obs.: se houver, então F = 1)
- Retângulos com 8 uns (2x4 ou 4x2)
- Retângulos com 4 uns (1x4, 4x1 ou 2x2)
- Retângulos com 2 uns (1x2 ou 2x1)

Importante: a última linha/coluna é adjacente à primeira linha/coluna.

3) Elimine grupos redundantes (se puder)

4) Para cada grupo, escreva uma soma de produtos onde apenas as variáveis que não mudaram são representadas.
Importante: Se, no grupo, uma variável W é mantida em 0, então escreva $\overline{W}$.

EXEMPLO

Simplifique $F(A, B, C, D)$, cuja tabela verdade é dada pelo mapa de Karnaugh ao lado.

ab \ cd	00	01	11	10
00	1	1	0	0
01	1	1	1	1
11	0	0	0	0
10	0	1	1	0

EXEMPLO

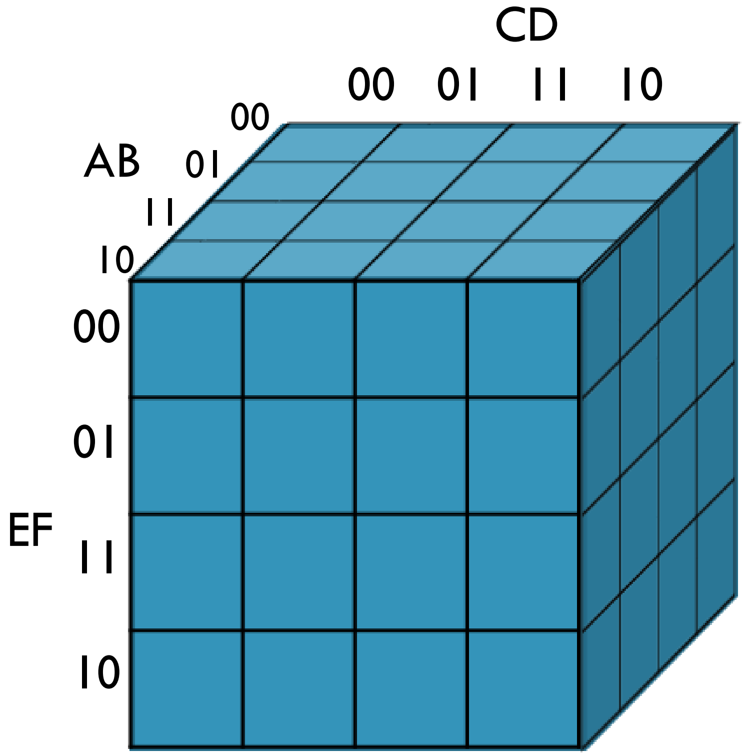
Simplifique $F(A,B,C,D)$, cuja tabela verdade é dada pelo mapa de Karnaugh ao lado.

$$F = \overline{A}\,\overline{C} + \overline{A}B + A\overline{B}D$$

ab \ cd	00	01	11	10
00	1	1	0	0
01	1	1	1	1
11	0	0	0	0
10	0	1	1	0

MAPAS DE KARNAUGH PARA MAIS DE 4 VARIÁVEIS

É possível construir mapas de Karnaugh para mais de 4 variáveis, mas eles se tornam difíceis de representar pois o mapa torna-se um cubo:



Entre 4 e 30 (aprox.) variáveis, é possível executar o método de Quine-McCluskey, que é exato mas possui complexidade exponencial.

Acima de 30 variáveis, há o minimizador Espresso, baseado em métodos heurísticos (não exato).

EXERCÍCIOS:

Simplifique:

1) $A'B'C'+A'B'C+ABC'+AB'C'$

2) $A'B'C+A'BC'+ABC'+ABC$

3) $ABCD+ABC'D+ABC'D'+AB'CD+A'BCD+A'BCD'+A'BC'D+A'B'C'D$