Circuitos Digitais

Prof. Ricardo Pedroni
ricardopedroni@utfpr.edu.br
www.rpedroni.com.br

Álgebra Booleana Técnicas de Redução

Prof. Ricardo Pedroni
ricardopedroni@utfpr.edu.br
www.rpedroni.com.br

Ementa

- Revisão – Álgebra booleana
- Representações e Termos de Equações booleanas
- Mapas de Karnaugh
- Algortimo de Quine-McCluskey

Representações de Equações Booleanas

Representações

- SOP (Soma de Produtos)
  - y = abc + a'b + bc'

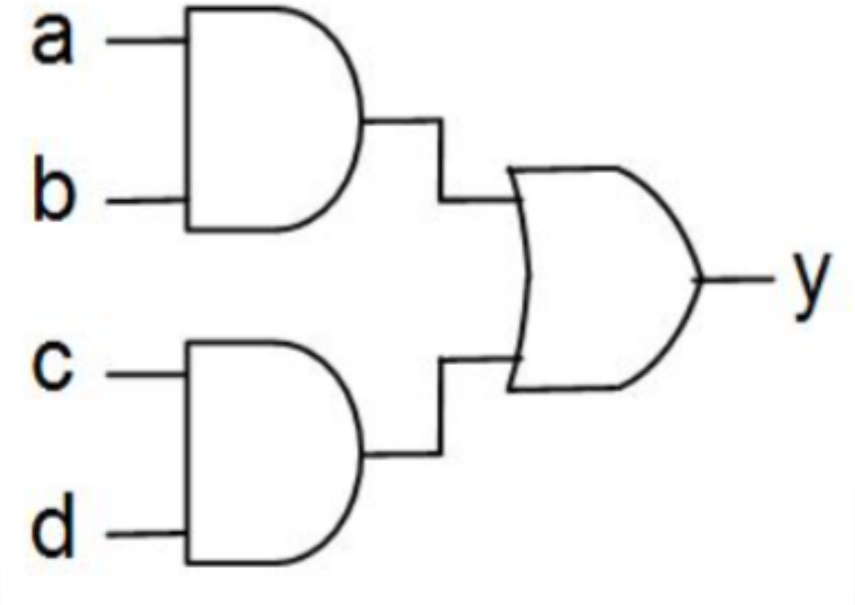
- POS (Produto de Somas)
  - y = (a + b')(a + b + c)(a' + c')

Representações

- Representação SOP
  - A representação SOP é mais utilizada
  - Toda SOP pode ser montada com uma camada **AND** seguida por uma camada **OR**

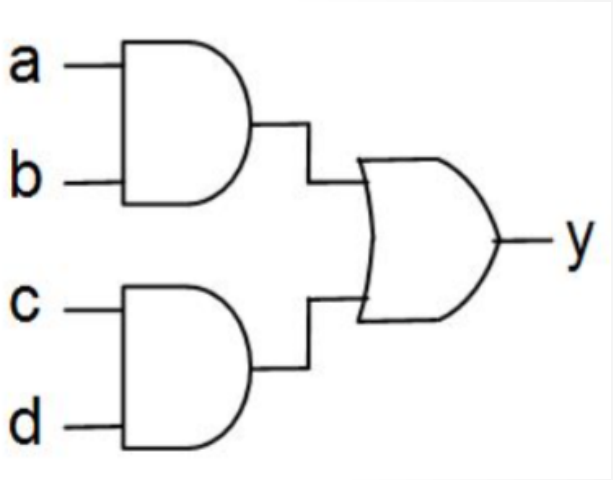
Representações

- SOP: AND → OR



Representações

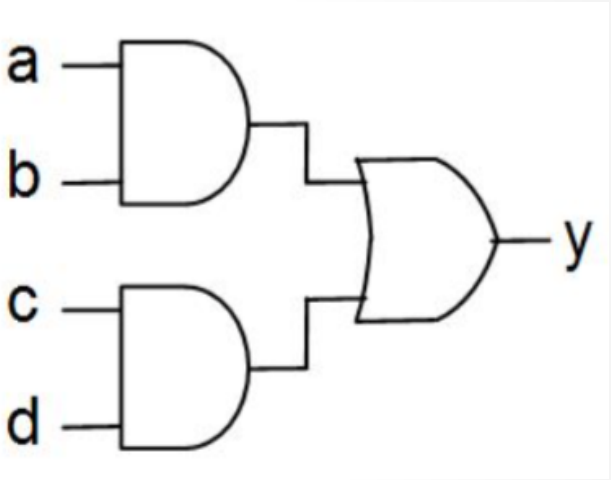
- SOP: AND → OR
  - Exemplo: y = ab + cd

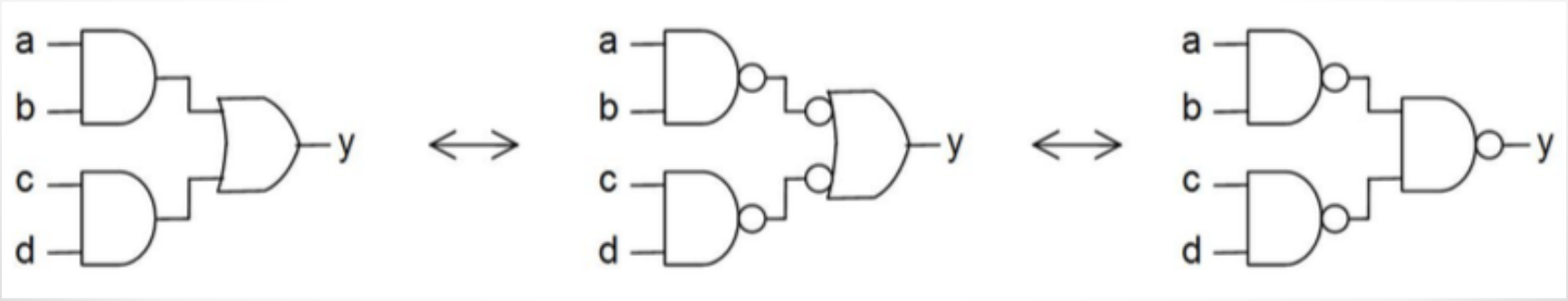


  - Pode ser convertido em duas camadas NAND (economiza transistores)

Representações

- SOP: AND → OR
  - Exemplo: y = ab + cd





A Grande Pergunta

- Dada uma equação
  f(a, b, c) = ac + a'b + abc'

  - **Como saber se essa equação está otimizada?**
  - **Teremos algum dia certeza?**
  - **Vou me formar um dia?**
  - **PAN PAN PAN**

Termos de Equações Booleanas

Termos

- SOP: AND → OR
  - **Literal:** é cada elemento (bit) de uma função booleana
  - Forma direta (a) ou complementada/indireta (a')
  - Logo, uma função de **N** variáveis pode ter até **2N** literais

Termos

- SOP: AND → OR
  - **Exemplo:**
    - A função f(a,b) pode ter até 4 literais: a, b, a', b'

Termos

- SOP: AND → OR
  - **Minterm:** termo da equação booleana com **N** literais
  - **Exemplo:**
    - f(a,b,c) = abc' + bc
    - Como **N = 3**, abc' é um minterm mas bc não é

Termos

- SOP: AND → OR
  - **Tabela Verdade:** lista de todos os **minterms** de uma função, com o respectivo valor produzido na **saída** para cada **minterm** aplicado como entrada
  - Basicamente: Existe ou não aquele minterm naquela função/circuito?

Termos

- SOP: AND → OR
  - Para uma função de **N** variáveis, quantos minterms existem na tabela verdade?

    → 2^N

  - Por que que nunca consideramos os termos com mais de N literais?
    - Ex: abb'c

Termos

- SOP: AND → OR
  - **Exemplo:**
    - y = a'b'c' + a'bc' + abc

Símbolo	Minterm	a b c	y
m_0	a'.b'.c'	0 0 0	1
m_1	a'.b'.c	0 0 1	0
m_2	a'.b.c'	0 1 0	1
m_3	a'.b.c	0 1 1	0
m_4	a.b'.c'	1 0 0	0
m_5	a.b'.c	1 0 1	0
m_6	a.b.c'	1 1 0	0
m_7	a.b.c	1 1 1	1

Termos

- SOP: AND → OR
  - **Implicante:** qualquer termo da função booleana que produz 1 na saída
  - Ou seja, qualquer termo da SOP

Termos

- SOP: AND → OR
  - **Implicante Primo:** qualquer termo implicante **irredutível** da função booleana
  - **Exemplo:**
    - **f(a,b,c) = abc' + abc + b'c'**
    - Todos os termos são implicantes (produzem '1' na saída)
    - Os dois primeiros termos não são implicantes primos pois podem ser reduzidos (prove!)
    - A função resultante **f(a,b,c) = ab + b'c'** tem dois implicantes, ambos **primos**, e não tem nenhum **minterm**

Termos

- SOP: AND → OR
  - **Legal, p'ssor!** Mas eu não me importo com termos e palavras dificil, mim escolheu faser engenaria. Pra que servir isso?
  - **Técnicas de redução**

Técnicas de Redução
Mapas de Karnaugh

Mapas de Karnaugh

- **Mapa de Karnaugh**
  - É um mapa com os **valores dos minterms** da função
  - Utiliza código **Gray** (pois nele palavras adjacentes diferem em apenas um bit)

  - Curiosidade
    - Para implementar o código Gray à mão , basta inverter o bit mais à direita que produz uma nova palavra
    - Exemplo: 000 → 001 → 011 → 010 → 110 → ...

Mapas de Karnaugh

- Exemplo: Karnaugh com 4 variáveis

Código Gray ⟶

Cód. Gray ↓

c d \ a b	00	01	11	10
00	0000	0100	1100	1000
01	0001	0101	1101	1001
11	0011	0111	1111	1011
10	0010	0110	1110	1010

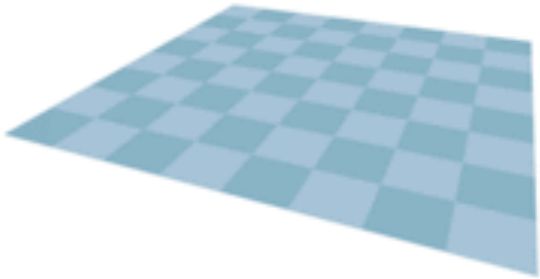
c d \ a b	00	01	11	10
00	m_0	m_4	m_{12}	m_8
01	m_1	m_5	m_{13}	m_9
11	m_3	m_7	m_{15}	m_{11}
10	m_2	m_6	m_{14}	m_{10}

Adjacentes

Adjacentes

Mapas de Karnaugh

- Os mapas são como toróides
- Todos os cantos se “encostam” e são adjacentes



Mapas de Karnaugh

- **Passo a Passo do Karnaugh**

1) Defina o **tamanho** do mapa a ser usado (2x2, 2x4, 4x4...)
2) Liste as **variáveis** em cada um dos lados do mapa, usando **código Gray**
3) Coloque os valores de **saída** (da tabela verdade) nas posições respectivas

Mapas de Karnaugh

- **Passo a Passo do Karnaugh**

4) Circule todos os **retângulos** de '1's adjacentes, lembrando que os grupos precisam ter um números de '1's que seja uma **potência de 2** (1, 2, 4...)
5) Verifique se o **número de grupos** é o **menor** possível e que cada grupo seja do **maior tamanho** possível
6) Para cada grupo, verifique quais variáveis **mudaram de valor** ao longo desse grupo (i.e. 0 → 1 ou 1 → 0)

Mapas de Karnaugh

- **Passo a Passo do Karnaugh**

7) O termo resultante para esse grupo será formado apenas pelas variáveis de **valor constante** (ou seja, as que não mudaram de valor)
8) Faça isso para todos os grupos existentes. Juntando todos os termos, você terá a equação booleana reduzida
9) (opcional) Abrace você mesmo ou
- alguém na adjacência, você merece.

Mapas de Karnaugh

- Exemplo N = 2
  - Dada a tabela verdade, obter a equação ótima de y

a b	y
0 0	0
0 0	0
0 1	0
0 1	1

	a	
b	0	1
0	0	0
1	0	1

Mapas de Karnaugh

- Exemplo N = 3
  - Dada a tabela verdade, obter a equação ótima de y

a b c	y
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	1
1 0 1	0
1 1 0	0
1 1 1	0

→

	a b			
c	00	01	11	10
0	0	1	0	1
1	0	1	0	0

$$y = a'b + ab'c'$$

Mapas de Karnaugh

- **Exemplo N = 4**
  - **Dada a tabela verdade, obter a equação ótima de y**

a b c d	y
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	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	1
1 1 1 0	0
1 1 1 1	1

→

	a b			
c d	00	01	11	10
00	1	0	0	1
01	0	1	1	0
11	0	1	1	0
10	0	0	0	0

$$y = bd + b'c'd'$$

Mapas de Karnaugh

- **Exemplo N > 4**
  - Mapas de Karnaugh em 3D

Mapas de Karnaugh

- **Exemplo N > 4**
  - Lembrar da propriedade
    - f(a,b,c,…) = a'.f(0,b,c,...) + a.f(1,b,c,...)
  - Se N = 5, queremos f(a,b,c,d,e) ótima
  - Fazer a = 0 e obter a equação ótima para f(0,b,c,d,e)
  - Fazer a = 1 e obter a equação ótima para f(1,b,c,d,e)
  - Usar a propriedade acima para obter a equação final
    - f(a,b,c,d,e) = a'.f(0,b,c,d,e) + a.f(1,b,c,d,e)

Mapas de Karnaugh

- **Computacionalmente falando..**
  - Para equações a serem resolvidas computacionalmente, os **Mapas-K** não são recomendados
  - Para equações com mais de 4 literais (mas não muito mais que isso), usa-se o algoritmo de **Quine-McCluskey**
  - Para equações grandes (> 32 literais), o padrão é utilizar o minimizador heurístico **Espresso**

Técnicas de Redução

Algoritmo de Quine-McCluskey

Quine-McCluskey

- **Computacionalmente falando..**
  - Humanos são execelentes em detectar padrões. Computadores… nem tanto.
  - Mapas-K são baseado bastante em visão e nossa capacidade de encontrar grupos visualmente
  - Computadores são excelentes em comparar valores e seguir passo-a-passos

Quine-McCluskey

- **O Algortimo**
  - Análogo a Mapas-K, o algoritmo de Quine-McCluskey procura valores "adjacentes" que sejam diferentes por apenas 1 bit por vez
  - A cada passo do algortimo, faz essa comparação e produz uma nova coluna com os novos valores reduzidos

Quine-McCluskey

- **Passo a Passo**
  - 1) Liste todos os minterms da função
    - Σm(0, 1, 3, 8, 9, 10, 13, 15)
  - 2) Liste todos os minterms numa coluna, separados em grupos divididos pela quantidade de bits '1' no minterm
  - 3) Para cada valor dentro de um grupo, compare com os valores do próximo grupo. Caso houver apenas 1 bit de diferença, coloque um – no lugar e adicione esse termo na próxima coluna. Para todo termo que for possível fazer a combinação, marque que esse termo foi utilizado

Quine-McCluskey

- **Passo a Passo**
  - 4) Faça essa comparação para todos os grupos e crie novas colunas quando houver termos que possam ser combinados
  - 5) Quando não houver mais termo que possa ser combinado, acabou a etapa da comparação
  - 6) Junte todos os termos finais numa tabela, listando os termos finais com os minterms que eles cobrem
    - Ex: “0-01” cobre “0001” e “0101”
    - Ex: “101--” cobre “10100”, “10101”, “10110” e “10111”

Quine-McCluskey

- **Passo a Passo**
  - 7) Verifique quais minterms são cobertos por apenas 1 termo. Caso isso acontecer, esse termo faz parte da resposta final. Pode riscar todos os termos que são cobertos por esse termo
  - 8) Caso sobrar minterms que são cobertos por mais que um termo, selecione o menor número possível de termos que cubram todos os minterms restantes
  - 9) Esses termos selecionados são a resposta da redução
  - 10) (opcional, quase virando obrigatório) Abrace a si mesmo.

Quine-McCluskey

- **Exemplo**
  - Σm(0, 4, 5, 7, 8, 11, 12, 15)
  - Vamos para o quadro!
  - (me lembrem de atualizar isso aqui com fotos. Sérião)

Quine-McCluskey

- **Vantagens**
  - Funções com N > 4 não são tão mais complexas
  - Algortimo computacional mais fácil de implementar
  - Podemos pedir circuitos com N > 4 em prova sem peso na consciência ☺

Conclusão

Conclusão

- Representações SOP e POS
- Termos da álgebra booleana
- Mapas de Karnaugh
- Algortimo de Quine-McCluskey

O Fim.

Perguntas? Não?
Então palmas para o professor.

Desafio do dia:
Fósforo do Deserto