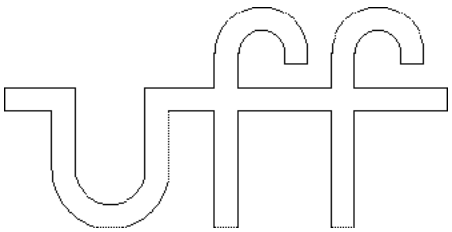
Apostila
de
Teoria
para
Técnicas Digitais I
(Versão 2k101018)



Universidade Federal Fluminense

Apostila
do
Departamento de Engenharia de Telecomunicações
da
Universidade Federal Fluminense
por
Alexandre Santos de la Vega
Outubro, 2010.

.

621.3192
mudar!
D278
mudar!
2010

de la Vega, Alexandre Santos

Apostila de Teoria para Técnicas Digitais I / Alexandre Santos de la Vega. – Niterói: UFF/TCE/TET, 2010.

123p. (atualizar...)

Apostila de Teoria – Graduação, Engenharia de Telecomunicações, UFF/TCE/TET, 2010.

1. Circuitos Digitais. 2. Técnicas Digitais. 3. Telecomunicações. I. Título.

Aos meus alunos.

Prefácio

O trabalho em questão cobre os tópicos abordados na disciplina Técnicas Digitais I.

A apostila foi escrita com o intuito de servir como uma referência rápida para os alunos do curso de graduação em Engenharia de Telecomunicações da Universidade Federal Fluminense (UFF).

O material básico utilizado foram as minhas notas de aula que, por sua vez, originaram-se em uma coletânea de livros sobre os assuntos abordados.

A motivação principal foi a de aumentar o dinamismo das aulas. Portanto, deve ficar bem claro que esta apostila não pretende substituir os livros textos ou outros livros de referência. Muito pelo contrário, ela deve ser utilizada apenas como ponto de partida para estudos mais aprofundados, utilizando-se a literatura existente.

Espero conseguir manter o presente texto em constante atualização e ampliação.

Correções e sugestões são sempre benvindas.

Rio de Janeiro, 04 de setembro de 2006.

Alexandre Santos de la Vega
UFF/TCE/TET

Agradecimentos

Aos alunos do Curso de Engenharia de Telecomunicações e aos professores do Departamento de Engenharia de Telecomunicações (TET), da Universidade Federal Fluminense (UFF), que colaboraram com críticas e sugestões bastante úteis à finalização deste trabalho. Em particular, à professora Carmen Maria Costa de Carvalho pela leitura meticulosa da versão original.

Aos funcionários do TET/UFF, Carmen Lúcia, Jussara, Arlei, Eduardo e Francisco, pelo apoio constante.

Aos meus alunos, que, além de servirem de motivação principal, obrigam-me sempre a tentar melhorar, em todos os sentidos.

Mais uma vez, e sempre, aos meus pais, por tudo.

Rio de Janeiro, 04 de setembro de 2006.
Alexandre Santos de la Vega
UFF/TCE/TET

Sumário

Prefácio **v**

Agradecimentos **vii**

1 Introdução **1**

2 Conceitos básicos **3**

3 Funções lógicas **7**
3.1 Introdução . . . 7
3.2 Conceitos básicos . . . 8
3.3 Variáveis lógicas . . . 8
3.4 Operadores lógicos . . . 9
3.5 Teoremas de DeMorgan . . . 10
3.6 Decomposição em funções básicas canônicas . . . 11
3.7 Conjunto funcionalmente completo de operadores . . . 12
3.8 Relações de implicação . . . 12
3.9 Blocos funcionais . . . 13
3.10 Manipulação algébrica de blocos . . . 13
3.11 Uso de bloco funcional como elemento de controle . . . 14
3.12 Exercícios propostos . . . 15

4 Álgebras de Boole **17**
4.1 Introdução . . . 17
4.2 Postulados de Huntington . . . 18
4.3 Dualidade . . . 18
4.4 Lemas e teoremas fundamentais . . . 19
4.5 Definição de uma álgebra de Boole concreta . . . 19
4.6 Outros exemplos de álgebras de Boole concretas . . . 20
4.7 Isomorfismo . . . 20
4.8 Simplificação algébrica de expressões lógicas . . . 20
4.9 Exemplos de manipulação algébrica não sistemática . . . 21
4.10 Manipulação através do Diagrama de Venn . . . 22
4.11 Resumo das relações algébricas . . . 23
4.12 Exercícios propostos . . . 25

5 Formas padrões para representação de expressões booleanas **27**
5.1 Introdução . . . 27
5.2 Definições . . . 28
5.3 Obtenção de formas SOP e POS padrões . . . 29

5.3.1 Manipulação algébrica 29
5.3.2 Utilização de tabela verdade 32
5.4 Conjuntos de formas padrões 34
5.5 Exercícios propostos 36

6 Simplificação algébrica de expressões booleanas 37
6.1 Operações básicas para a simplificação sistemática de expressões booleanas . . . 37
6.2 Simplificação de expressões booleanas a partir das formas padrões SOP e POS . 37
6.2.1 Uso da aglutinação 37
6.2.2 Uso da replicação 38
6.3 Eliminação sistemática de literais 39
6.4 Expressão mínima 42
6.5 Processo sistemático de minimização 42
6.6 Implicantes e implicados 43
6.7 Exercícios propostos 44

7 Mapa de Karnaugh 45
7.1 Introdução 45
7.2 Construção do mapa-K 46
7.2.1 Funções de 1 variável 46
7.2.2 Funções de 2 variáveis 47
7.2.3 Funções de 3 variáveis 48
7.2.4 Funções de 4 variáveis 49
7.3 Preenchimento do mapa-K 50
7.4 Mapa-K como forma de expressão de função booleana 50
7.5 Mapa-K na simplificação de expressões booleanas 52
7.5.1 Adjacência lógica, aglutinação e replicação 52
7.5.2 Seleção sistemática de termos (implicantes ou implicados) 53
7.5.3 Mapa-K de funções com múltiplos mínimos e mapa cíclico 53
7.5.4 Indeterminações: *don't-care* e *can't-happen* 54
7.6 Exercícios propostos 56

8 Sistemas de numeração 57
8.1 Introdução 57
8.2 Sistema de numeração posicional convencional 59
8.2.1 Representação de números inteiros positivos 59
8.2.2 Representação de números fracionários positivos 60
8.2.3 Representação de números inteiros negativos 61
8.2.4 Representação de números fracionários negativos 65
8.2.5 Adição e subtração em complemento-a-2 67
8.2.6 Tabelas de operações básicas entre dígitos 70
8.2.7 Escalamento por potência inteira da base 71
8.2.8 Conversão entre bases 72
8.2.9 Bases mais comuns em circuitos digitais 74
8.3 Quantização 76
8.4 Exercícios propostos 77

9 Circuitos combinacionais básicos **79**
9.1 Introdução . . . 79
9.2 Uso de portas lógicas como elementos de controle . . . 79
9.3 Multiplexadores e demultiplexadores . . . 79
9.4 Decodificadores de linha . . . 79
9.5 Somadores simples . . . 79
9.5.1 *Half-adder* . . . 79
9.5.2 *Full-adder* . . . 79
9.5.3 *Ripple-carry adder* . . . 79
9.6 Subtratores simples . . . 80
9.6.1 *Half-subtractor* . . . 80
9.6.2 *Full-subtractor* . . . 80
9.6.3 *Ripple-borrow subtractor* . . . 80
9.7 Complementadores . . . 80
9.7.1 Complementador-a-1 (*bitwise implementation*) . . . 80
9.7.2 Complementador-a-2 . . . 80
9.8 Comparadores . . . 80

A Tópicos sobre divisão de números inteiros **81**
A.1 Algoritmo de divisão inteira . . . 81
A.2 Quociente . . . 81
A.3 Resto ou resíduo . . . 81
A.4 Congruência . . . 81
A.5 Relações úteis . . . 82

Bibliografia **83**

Lista de Tabelas

3.1 Tabela de funções de uma variável. 9
3.2 Tabela de operadores de 1 variável. 9
3.3 Tabela de funções de duas variáveis. 9
3.4 Tabela de operadores de duas variáveis. 10
3.5 Tabela de funções básicas (mintermos e maxtermos) para duas variáveis. 11
3.6 Exemplo de decomposição em funções básicas. 11
3.7 Uso de bloco funcional como elemento de controle. 14

4.1 Tabela de mapeamento: Teoria de Conjuntos × Álgebra de Boole. 20
4.2 Tabela de mapeamento: Cálculo Proposicional × Álgebra de Boole. 20
4.3 Resumo dos postulados de Huntington para a álgebra de Boole abstrata. 23
4.4 Resumo dos lemas da álgebra de Boole abstrata. 23
4.5 Resumo dos teoremas da álgebra de Boole abstrata. 24
4.6 Resumo da definição de uma álgebra de Boole concreta. 24
4.7 Resumo das relações de isomorfismo. 24

5.1 Definição de mintermos para três variáveis (A,B,C). 29
5.2 Definição de maxtermos para três variáveis (A,B,C). 29
5.3 Exemplo de função e associação de mintermos. 32
5.4 Exemplo de função e definição de maxtermos. 33
5.5 Exemplo da obtenção do grupo AND-OR para a função XOR. 35
5.6 Exemplo da mudança de grupo para a função XOR. 35
5.7 Exemplo da obtenção do grupo OR-AND para a função XOR. 35

6.1 Tabela verdade para funções de 3 variáveis. 39

7.1 Tabela verdade para funções de 1 variável. 46
7.2 Tabela verdade para funções de 2 variáveis. 47
7.3 Tabela verdade para funções de 3 variáveis. 48
7.4 Tabela verdade para funções de 4 variáveis. 49
7.5 Tabela verdade relativa à Equação (7.1). 51
7.6 Tabela verdade de função incompletamente especificada. 54

8.1 Tabela de sinal-e-magnitude, para número inteiros, $b = 2$ e $N = 4$. 62
8.2 Tabela de complemento-a-1, para número inteiros, $b = 2$ e $N = 4$. 63
8.3 Tabela de complemento-a-2, para número inteiros, $b = 2$ e $N = 4$. 65
8.4 Tabela de sinal-e-magnitude, para números puramente fracionários, $b = 2$ e $N = 4$. 66
8.5 Tabela de complemento-a-1, para números puramente fracionários, $b = 2$ e $N = 4$. 66
8.6 Tabela de complemento-a-2, para números puramente fracionários, $b = 2$ e $N = 4$. 67
8.7 Forma 1 para detecção de *overflow* na adição em complemento-a-2. 68
8.8 Forma 2 para detecção de *overflow* na adição em complemento-a-2. 69

Lista de Figuras

3.1 Blocos funcionais padrões, associados aos operadores lógicos. 13
3.2 Exemplos de manipulação algébrica de blocos. 13
3.3 Uso de bloco funcional como elemento de controle. 14

6.1 Eliminações de 1 literal em combinações de 2 mintermos. 40
6.2 Eliminações de 2 literais em combinações 4 de mintermos. 40
6.3 Eliminações de 1 literal em combinações de 2 maxtermos. 41
6.4 Eliminações de 2 literais em combinações 4 de maxtermos. 41

7.1 Exemplos de mapas de Karnaugh para funções de 1 variável. 46
7.2 Exemplos de mapas de Karnaugh para funções de 2 variáveis. 47
7.3 Exemplos de mapas de Karnaugh para funções de 3 variáveis. 48
7.4 Exemplos de mapas de Karnaugh para funções de 4 variáveis. 50
7.5 Mapa de Karnaugh relativo à Equação (7.1). 51
7.6 Mapa de Karnaugh relativo à Equação (7.6). 52
7.7 Mapa de Karnaugh relativo à Equação (7.7). 52
7.8 Mapa de Karnaugh com múltiplas formas mínimas. 54
7.9 Mapa de Karnaugh com ciclo. 54
7.10 Mapa de Karnaugh da Tabela 7.6. 55
7.11 Mapa de Karnaugh dos mintermos da Tabela 7.6. 55
7.12 Mapa de Karnaugh dos maxtermos da Tabela 7.6. 55

8.1 Representação de quantidades $q < b$, para $b = 3$. 59
8.2 Representação de quantidades $q \geq b$, para $b = 3$, com ambigüidade. 59
8.3 Representação de quantidades $q \geq b$, para $b = 3$, com eliminação da ambigüidade através da justaposição dos dígitos. 60
8.4 Uso repetido da técnica de justaposição de dígitos para representação de quantidades $q \geq b$, para $b = 3$, sem ambigüidade. 60
8.5 Tabelas de operações entre dígitos para $b = 2$: (a) adição e (b) multiplicação. . . 70
8.6 Tabelas de operações entre dígitos para $b = 3$: (a) adição e (b) multiplicação. . . 70
8.7 Tabelas de operações entre dígitos para $b = 4$: (a) adição e (b) multiplicação. . . 70

Capítulo 1

Introdução

- Este é um documento em constante atualização.
- Ele consta de tópicos desenvolvidos em sala de aula.
- Na preparação das aulas são utilizados os seguintes livros:
  - Livros indicados pelo professor: [HP81], [Rhy73], [TWM07], [Uye02].
  - Livros indicados pela ementa da disciplina: [IC08], [Tau82].
- Este documento aborda os seguintes assuntos:
  - Conceitos básicos: que busca contextualizar a disciplina no âmbito do curso de graduação.
  - Funções lógicas: que define as bases para a representação de informações não numéricas em circuitos digitais.
  - Álgebras de Boole: que apresentam um formalismo matemático para a representação lógica em circuitos digitais.
  - Formas padrões para representação de expressões booleanas: que define formas de expressões booleanas adequadas ao processo de simplifição das mesmas.
  - Simplificação algébrica de expressões booleanas: que ilustra um processo algébrico para a simplificação de expressões booleanas.
  - Mapa de Karnaugh: que apresenta uma ferramenta sistemática para a simplificação de expressões booleanas.
  - Sistemas de numeração: que define as bases para a representação de quantidades numéricas em circuitos digitais.
  - Circuitos combinacionais básicos: que apresenta os blocos funcionais combinacionais básicos utilizados em sistemas digitais.

Capítulo 2

Conceitos básicos

- Telecomunicações: definição e exemplos.
- Sistemas de Telecomunicações: definição, anatomia e exemplos.
- Relação: Circuito X Sistema.
- Sinais: definição.
- Sistemas: definição, anatomia (variáveis, elementos e topologia), análise × projeto.
- Classificação de sistemas:
  - parâmetros concentrados × distribuídos.
  - parâmetros constantes no tempo, fixo ou invariante no tempo × parâmetros variáveis no tempo, variável ou variante no tempo.
  - linear × não linear.
  - instantâneo ou sem memória × dinâmico ou com memória.
  - contínuo × discreto (analógico/amostrado/quantizado/digital).
- Sistemas dinâmicos: estado e variáveis de estado.
- Medição e armazenamento: discretização.
- Discretização: amostragem × quantização.
- Tipos de discretização: uniforme × não uniforme.
- Tipos de aproximação na quantização: truncamento, arredondamento e truncamento em magnitude.
- Anatomia de sistemas digitais:
  - processamento digital dos sinais: digitais × analógicos.
  - processamento digital dos sinais digitais: sinais de dados, sinais de temporização, sinais de controle e alimentação.
  - processamento digital dos sinais analógicos: conversores A/D e D/A, filtro limitador de banda (*anti-aliasing filter*) e filtro de interpolação (*smoothing filter*).
- Hierarquia: em *hardware* e em *software*.

- Codificação:
  - Código: sintaxe (símbolos) × semântica (significado).
  - No caso geral: representação de idéias.
  - No caso de transmissão digital: fonte (compressão) e canal (redução de taxa de erro).
- Elementos codificados:
  - Informação (fatos, classificações) × Quantidade (números, contagem).
  - Informação pode ser codificada em quantidade.
  - Quantidade pode ser codificada como informação.
  - Ambos podem ser representados e manipulados como um único elemento.
  - Ambos são um único elemento.
  - Ambos significam idéias codificadas.
- Representação de informação:
  - Uma vez que a informação é multivalorada, pode-se utilizar, para representá-la, um único dispositivo com múltiplos estados ou vários dispositivos com um número reduzido de estados.
  - O número total de estados, ou de condições às quais os dispositivos podem ser ajustados, deve ser igual ou maior ao número de valores possíveis que a informação a ser representada pode assumir.
  - Na tentativa de minimizar o número de dispositivos e o número de diferentes estados em cada dispositivo, já foi demonstrado que o número de estados ótimo para cada dispositivo é o número $e = 2.718281828459$ [Ric56].
  - Embora a melhor aproximação seja um total de 3 estados por dispositivo, a disponibilidade de dispositivos eletro-eletrônicos que apresentam 2 estados de operação (chaves, relés, diodos, transistores), aliada à facilidade e à confiabilidade (estabilidade e robustez) de implementação, têm levado à escolha da representação através de mecanismos envolvendo 2 estados por dispositivo.
  - Assim, para informações envolvendo N estados, são necessários M dispositivos de 2 estados, tal que $2^M \geq N$.
- Representação de quantidade:
  - Representação numérica (quantizada) de uma informação analógica ou discreta.
  - Representação de informação analógica: amostragem e quantização.
  - Representação de informação discreta: quantização.
  - Fontes de erro:
    - Acurácia da medida.
    - Resolução da medida (precisão).
    - Limites da representação da medida (máximo e mínimo).
    - Erros de conversão A/D.
    - Capacidade de armazenamento da amostra.

  - Quantidades são naturalmente multivaloradas e representadas por símbolos.
  - Novamente, dois extremos são possíveis. Por um lado, pode-se usar um único símbolo variável, cujas variações representam todos os valores numéricos desejados. De outra forma, pode-se optar por uma combinação de símbolos de um conjunto, o qual é capaz de representar apenas uma faixa de valores.
  - Exemplos de representação numérica:
    * Um sistema de numeração com 4 dígitos e resolução de 0.001 pode representar números positivos de 0.000 até 9.999, num total de 10.000 valores diferentes.
    * Para representar 100 valores diferentes, pode-se utilizar 100 símbolos fixos diferentes ou X símbolos variáveis, com Y valores cada.

- Um sistema de numeração significa, para a representação de quantidades numéricas, o mesmo que símbolos lógicos significam para a representação de informações não numéricas.
- Utilizando-se uma simbologia que atenda a ambas as representações, pode-se implementar sistemas que manipulem quantidades numéricas e informações não numéricas sem distinção.
- Tais sistemas, e seus circuitos, são denominados sistemas e circuitos digitais.

Capítulo 3

Funções lógicas

3.1 Introdução

- A manipulação de informações pode ser dividida em três partes básicas: a obtenção dos dados, o processamento desses dados e a geração de novos dados.
- Toda ação envolve, de certa forma, tomadas de decisão.
- Compreender o raciocínio humano que rege as tomadas de decisão possibilita que o mecanismo seja implantado em sistemas artificiais.
- A lógica pode ser vista como um ramo de estudos da matemática que fornece elementos para a tentativa de modelagem do raciocínio humano.
- Argumentos são conjuntos de enunciados, dos quais um deles é definido como *conclusão* e os demais como *premissas*.
- Quanto à validade, os argumentos podem ser divididos em: dedutivos e indutivos.
- Em um argumento dedutivo, premissas verdadeiras conduzem a uma conclusão verdadeira.
- Nos argumentos indutivos, premissas verdadeiras não garantem uma conclusão verdadeira.
- A lógica formal fornece uma linguagem estruturada para a definição e a manipulação dos argumentos.
- A lógica dedutiva pode ser dividida em: clássica, complementar, e não-clássica.
- Atualmente, a lógica clássica é mais conhecida como Cálculo de Predicados de Primeira Ordem.
- Exemplos de lógica complementar são: modal, deôntica e epistêmica.
- Alterando-se os princípios da lógica clássica, surgem as lógicas não-clássicas. Alguns exemplos são: paracompletas, intuicionistas, não-aléticas, não-reflexivas, probabilísticas, polivalentes, *fuzzy*.

3.2 Conceitos básicos

- Valores fixos, variáveis e funções.
- Função: mapeamento entre variáveis.
- Representação de funções: tabelas, gráficos e equações.
- Variáveis (e funções) não necessitam ser numéricas, embora possam ser associados números aos possíveis valores (estados) das variáveis.

3.3 Variáveis lógicas

- Nesse texto, será abordado apenas um tipo de lógica: binária, bivalente ou clássica.
- Proposição: sentença afirmativa declarativa (ou afirmação declarativa ou assertiva ou *statement*), sobre a qual faça sentido se afirmar que a mesma é verdadeira ou falsa.
- Propriedade de variáveis lógicas binárias:
  - as variáveis representam assertivas.
  - só existem dois valores fixos que podem ser atribuídos a uma variável.
  - os dois valores possíveis devem ser, do ponto de vista lógico, mutuamente excludentes.
- Modelagem lógica de um problema real:
  - Envolve diversas representações ou codificações.
  - Problema real → um sistema formado por um conjunto de assertivas (*statements*), associadas por meio de conectivos (operadores ou funções).
  - Conectivo → elemento de conexão que é modelado por uma função lógica.
  - Assertiva → afirmação declarativa (*statements*) sobre algum elemento do problema, a qual é representada por uma variável de asserção do sistema.
  - Variável de asserção → variável do sistema associada a uma assertiva, à qual será atribuído um valor fixo lógico (*truth value*).
  - Valor fixo lógico → representação do estado de uma variável de asserção por meio de um símbolo com significado lógico Por exemplo: F/T, F/V, 0/1, 0/5 ou +12/-12.
- Uma vez que os dois valores possíveis são mutamente excludentes, surge, naturalmente, a idéia de negação (da associação de assertivas, do conectivo, da assertiva, da variável de asserção, do valor ou do símbolo).

3.4 Operadores lógicos

- Um operador lógico pode ser definido por uma função de variáveis lógicas (*truth function*).
- Formalização matemática das funções lógicas: cálculo de funções lógicas ou cálculo sentencial ou cálculo proposicional ou tautologia.
- Representação eficiente de funções lógicas: tabela verdade (*truth table*).
- Funções de 1 variável:
  - Cada uma das funções de uma variável é definida por sua Tabela Verdade na Tabela 3.1.

$X = f(A)$				
A	X_0	X_1	X_2	X_3
F	F	F	T	T
T	F	T	F	T

Tabela 3.1: Tabela de funções de uma variável.

  - Operações: os operadores $X_i = f_i(A)$ são definidos na Tabela 3.2.

	$f(A)$	
X_i	Operação	Notação
X_0	(F)	(F)
X_1	(A)	(A)
X_2	$NOT(A)$	$\neg(A) \equiv \sim(A) \equiv \overline{(A)} \equiv (A)' \equiv (A)^* \equiv !(A)$
X_3	(T)	(T)

Tabela 3.2: Tabela de operadores de 1 variável.

- Funções de 2 variáveis:
  - Cada uma das funções de duas variáveis é definida por sua Tabela Verdade na Tabela 3.3.

$X = f(A,B)$																	
A	B	X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}
F	F	F	F	F	F	F	F	F	F	T	T	T	T	T	T	T	T
F	T	F	F	F	F	T	T	T	T	F	F	F	F	T	T	T	T
T	F	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T
T	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T

Tabela 3.3: Tabela de funções de duas variáveis.

– Operações: os operadores $X_i = f_i(A, B)$ são definidos na Tabela 3.4.

X_i	$f(A,B)$	
	Operação	Notação
X_0	(F)	(F)
X_1	$(A\ AND\ B)$	$(A \wedge B)$
X_2	$NOT(A\ IMPLICA\ B)$	$\neg(A \rightarrow B) \equiv \neg(A \supset B)$
X_3	(A)	(A)
X_4	$NOT(B\ IMPLICA\ A)$	$\neg(A \leftarrow B) \equiv \neg(A \subset B)$
X_5	(B)	(B)
X_6	$(A\ XOR\ B)$	$(A \veebar B)$
X_7	$(A\ OR\ B)$	$(A \vee B)$
X_8	$NOT(A\ OR\ B) \equiv (A\ NOR\ B)$	$\neg(A \vee B) \equiv (A \downarrow B)$
X_9	$NOT(A\ XOR\ B) \equiv (A\ XNOR\ B)$	$\neg(A \veebar B) \equiv (A \equiv B)$
X_{10}	$NOT(B)$	$\neg(B)$
X_{11}	$(B\ IMPLICA\ A)$	$(A \leftarrow B) \equiv (A \subset B)$
X_{12}	$NOT(A)$	$\neg(A)$
X_{13}	$(A\ IMPLICA\ B)$	$(A \rightarrow B) \equiv (A \supset B)$
X_{14}	$NOT(A\ AND\ B) \equiv (A\ NAND\ B)$	$\neg(A \wedge B) \equiv (A \uparrow B)$
X_{15}	(T)	(T)

Tabela 3.4: Tabela de operadores de duas variáveis.

- Com base nas operações identificadas nas Tabelas 3.1 e 3.3, podem ser definidos os operadores lógicos encontrados nas Tabelas 3.2 e 3.4.

  – Operador unário: NOT (negação lógica).

  – Operadores binários: AND (E lógico), OR (OU-inclusivo lógico), XOR (OU-eXclusivo), NAND (NOT-AND), NOR (NOT-OR), XNOR (NOT-XOR ou bi-implicação ou equivalência lógica) e IMPLICA (implicação lógica).

- Funções de $N > 2$ variáveis: definidas através da combinação das variáveis lógicas e das operações identificadas nas funções de 1 e 2 variáveis.

3.5 Teoremas de DeMorgan

- (A NAND B) ≡ NOT (A AND B) ≡ (NOT A) OR (NOT B)
  ou
  $(A \uparrow B) \equiv \neg(A \wedge B) \equiv (\neg A) \vee (\neg B)$

- (A NOR B) ≡ NOT (A OR B) ≡ (NOT A) AND (NOT B)
  ou
  $(A \downarrow B) \equiv \neg(A \vee B) \equiv (\neg A) \wedge (\neg B)$

3.6 Decomposição em funções básicas canônicas

- Uma função lógica genérica pode ser decomposta em funções básicas canônicas, dos tipos m e M.
- Em uma função canônica do tipo m, apenas um de seus valores é T, enquanto todos os demais são F. Por isso, ela pode ser chamada de mintermo.
- Em uma função canônica do tipo M, apenas um de seus valores é F, enquanto todos os demais são T. Por isso, ela pode ser chamada de maxtermo.
- Para funções de duas variáveis, os mintermos m_i são definidos por: $m_0(A,B) = (\neg A \wedge \neg B)$, $m_1(A,B) = (\neg A \wedge B)$, $m_2(A,B) = (A \wedge \neg B)$ e $m_3(A,B) = (A \wedge B)$.
- Para funções de duas variáveis, os maxtermos M_i são definidos por: $M_0(A,B) = (A \vee B)$, $M_1(A,B) = (A \vee \neg B)$, $M_2(A,B) = (\neg A \vee B)$ e $M_3(A,B) = (\neg A \vee \neg B)$.
- A Tabela 3.5 apresenta as funções básicas (mintermos e maxtermos) para duas variáveis.

A	B	m_0	m_1	m_2	m_3	M_0	M_1	M_2	M_3
F	F	T	F	F	F	F	T	T	T
F	T	F	T	F	F	T	F	T	T
T	F	F	F	T	F	T	T	F	T
T	T	F	F	F	T	T	T	T	F

Tabela 3.5: Tabela de funções básicas (mintermos e maxtermos) para duas variáveis.

- A título de exemplo, a Tabela 3.6 ilustra a decomposição da função $X(A,B) = (A \veebar B)$ de duas formas: $X(A,B) = (A \veebar B) = m_1 \vee m_2 = (\neg A \wedge B) \vee (A \wedge \neg B)$ e $X(A,B) = (A \veebar B) = M_0 \wedge M_3 = (A \vee B) \wedge (\neg A \vee \neg B)$.

A	B	$A \veebar B$	m_1	m_2	M_0	M_3
F	F	F	F	F	F	T
F	T	T	T	F	T	T
T	F	T	F	T	T	T
T	T	F	F	F	T	F

Tabela 3.6: Exemplo de decomposição em funções básicas.

- Deve-se notar que, para uma determinada função alvo $X(\cdot)$, cada função m_i (ou M_i) utilizada na sua decomposição é responsável por sintetizar um dos valores T (ou F) de $X(\cdot)$.
- Deve-se notar ainda que $\neg(m_i) = M_i$.
- Portanto, as relações $\neg m_3 = M3$ e $\neg M_0 = m_0$ representam, ao mesmo tempo, uma prova para os Teoremas de DeMorgan e uma outra forma de enunciá-los.

3.7 Conjunto funcionalmente completo de operadores

- Pode-se notar que alguns operadores (conectivos) binários podem ser descritos através da combinação de outros operadores.
- Questões que surgem naturalmente:
  - É possível descrever todos os demais operadores a partir de um determinado conjunto (conjunto completo) ?
  - Todos os operadores de um conjunto completo são absolutamente necessários (independentes)?
  - Existe um conjunto mínimo de operadores que forme um conjunto completo (conjunto completo mínimo) ?
- Soluções:
  - Tentativa 1: (AND) → Não!
  - Tentativa 2: (OR) → Não!
  - Tentativa 3: (AND + OR) → Não!
  - Tentativa 4: (AND + OR + NOT) → OK! → Conjunto completo, mas não mínimo...
  - Tentativa 5: (NOT + AND = NAND) → OK! → Conjunto completo e mínimo!
  - Tentativa 6: (NOT + OR = NOR) → OK! → Conjunto completo e mínimo!
- Os operadores NOT, AND e OR são naturalmente utilizados nas expressões lógicas elaboradas pelo ser humano.
- Os operadores NOT, NAND e NOR são facilmente implementados por dispositivos eletro-eletrônicos.
- Assim sendo, é comum que se definam as expressões lógicas utilizando os operadores do conjunto {NOT, AND, OR} e, em seguida, que elas sejam convertidas em expressões equivalentes, empregando os operadores do conjunto {NOT, NAND, NOR}.

3.8 Relações de implicação

- Podem-se definir três relações de implicação de uma assertiva precedente A para uma assertiva conseqüente B:
  - Condição necessária: "SOMENTE SE $(A = T)$ ENTÃO $(B = T)$"
    ou "$(B = T)$ SOMENTE SE $(A = T)$".
  - Condição suficiente: "SE (A = T) ENTÃO (B = T)" ou "(B = T) SE (A = T)".
  - Condição necessária e suficiente: "SE E SOMENTE SE (A = T) ENTÃO (B = T)" ou "(B = T) SE E SOMENTE SE (A = T)".
- Portanto, pode-se estabelecer a seguinte modelagem:
  - Condição necessária: $(A \leftarrow B)$.
  - Condição suficiente: $(A \rightarrow B)$.
  - Condição necessária e suficiente: $(A \rightarrow B) \wedge (A \leftarrow B) \equiv (A \equiv B)$.

3.9 Blocos funcionais

- Para cada operador lógico, pode-se definir um bloco funcional que realize a operação correspondente.
- Posteriormente, pode-se propor um sistema físico que implemente o bloco funcional desejado.
- Blocos funcionais padrões podem ser encontrados na Figura 3.1.

Figura 3.1: Blocos funcionais padrões, associados aos operadores lógicos.

3.10 Manipulação algébrica de blocos

- Os blocos funcionais são uma representação alternativa para os operadores lógicos.
- Circuitos envolvendo os blocos funcionais são uma representação alternativa para as equações envolvendo os operadores lógicos.
- Portanto, pode-se realizar uma manipulação sobre os blocos lógicos de um circuito, a qual será idêntica àquela realizada sobre os operadores lógicos de uma equação.
- Alguns exemplos de manipulação algébrica de blocos são mostrados na Figura 3.2.

Figura 3.2: Exemplos de manipulação algébrica de blocos.

3.11 Uso de bloco funcional como elemento de controle

- No projeto de circuitos digitais, é comum que se necessite de alguns elementos básicos de controle, os quais podem ser implementados através dos blocos funcionais.

- Considerando-se as variáveis de entrada E, de controle $CTRL$ e de saída S, e que elas assumam apenas os valores lógicos F e T, podem-se definir as ações de controle apresentadas na Tabela 3.7.

- Tais operações podem ser visualizadas na Figura 3.3.

Operador	Ação de controle
AND	$S = (CTRL \wedge E) = \begin{cases} F & , \ CTRL = F \\ E & , \ CTRL = T \end{cases}$
$NAND$	$S = (CTRL \uparrow E) = \begin{cases} T & , \ CTRL = F \\ \overline{E} & , \ CTRL = T \end{cases}$
OR	$S = (CTRL \vee E) = \begin{cases} E & , \ CTRL = F \\ T & , \ CTRL = T \end{cases}$
NOR	$S = (CTRL \downarrow E) = \begin{cases} \overline{E} & , \ CTRL = F \\ F & , \ CTRL = T \end{cases}$
XOR	$S = (CTRL \veebar E) = \begin{cases} E & , \ CTRL = F \\ \overline{E} & , \ CTRL = T \end{cases}$
$XNOR$	$S = (CTRL \equiv E) = \begin{cases} \overline{E} & , \ CTRL = F \\ E & , \ CTRL = T \end{cases}$
$IMPLICA$	$S = (CTRL \rightarrow E) = \begin{cases} T & , \ CTRL = F \\ E & , \ CTRL = T \end{cases}$
	$S = (CTRL \leftarrow E) = \begin{cases} \overline{E} & , \ CTRL = F \\ T & , \ CTRL = T \end{cases}$
NOT $IMPLICA$	$S = \neg(CTRL \rightarrow E) = \begin{cases} F & , \ CTRL = F \\ \overline{E} & , \ CTRL = T \end{cases}$
	$S = \neg(CTRL \leftarrow E) = \begin{cases} E & , \ CTRL = F \\ F & , \ CTRL = T \end{cases}$

Tabela 3.7: Uso de bloco funcional como elemento de controle.

Figura 3.3: Uso de bloco funcional como elemento de controle.

3.12 Exercícios propostos

1. Discutir a propriedade de comutatividade dos operandos para todos os operadores binários.
2. Provar os Teoremas de DeMorgan.
3. Decompor as funções de todos os conectivos binários como combinações das funções básicas m_i (mintermos), associadas pelo conectivo OR.
4. Decompor as funções de todos os conectivos binários como combinações das funções básicas M_i (maxtermos), associadas pelo conectivo AND.
5. Considerando que todas as funções lógicas sejam descritas pela combinação dos operadores unário e binários, provar que:
   (a) Os conjuntos $\{AND\}$, $\{OR\}$ e $\{AND, OR\}$, não são conjuntos completos.
   (b) O conjunto $\{AND, OR, NOT\}$ é um conjunto completo.
   (c) O conjunto $\{AND, OR, NOT\}$ não é um conjunto completo mínimo.
   (d) O conjunto $\{AND, NOT\} \equiv \{NAND\}$ é um conjunto completo mínimo.
   (e) O conjunto $\{OR, NOT\} \equiv \{NOR\}$ é um conjunto completo mínimo.
6. Escrever as funções de todos os conectivos binários utilizando apenas as seguintes funções básicas:
   (a) $\{AND, NOT\}$.
   (b) $\{OR, NOT\}$.
   (c) $NAND$.
   (d) NOR.
7. Para os exercícios listados abaixo, considerar as equações lógicas apresentadas em seguida.
   (a) Desenhar um Diagrama de Blocos Funcionais equivalente, para cada uma das equações lógicas fornecidas.
   (b) Decompor cada uma das equações lógicas fornecidas como combinação das funções básicas m_i (mintermos), associadas pelo conectivo OR.
   (c) Decompor cada uma das equações lógicas fornecidas como combinação das funções básicas M_i (maxtermos), associadas pelo conectivo AND.
   (d) Desenhar um Diagrama de Blocos Funcionais equivalente, para cada uma das decomposições pedidas anteriormente.

   Equações lógicas:
   i. $F(A, B) = (A \vee B) \wedge (A \vee \neg B) \wedge (\neg A \vee B)$.
   ii. $F(A, B) = (\neg A \vee B) \wedge (A \vee \neg B) \wedge (\neg A \vee \neg B)$.
   iii. $F(A, B) = (A \wedge \neg B) \vee (\neg A \wedge B) \vee (A \wedge B)$.
   iv. $F(A, B) = (\neg A \wedge \neg B) \vee (\neg A \wedge B) \vee (A \wedge \neg B)$.
   v. $F(A, B, C, D) = (A \vee B) \wedge (\neg(C \wedge D))$.

Capítulo 4

Álgebras de Boole

4.1 Introdução

- A implementação de um sistema digital apresenta um custo.
- Sempre é desejado o menor custo possível.
- Deve-se minimizar a implementação a fim de se reduzir o seu custo.
- O cálculo proposicional não apresenta ferramentas adequadas para encontrar a implementação mais econômica.
- É necessário definir uma representação lógica que forneça ferramentas para a minimização das funções lógicas.
- Tais ferramentas podem ser encontradas na álgebra abstrata.
- A álgebra é o ramo da matemática que estuda as generalizações dos conceitos e das operações da aritmética.
- Em álgebra abstrata definem-se estruturas abstratas que representam, de uma forma global, diversas estruturas encontradas na prática.
- Uma estrutura algébrica adequada para a formulação, a manipulação e a minimização de funções lógicas é a Álgebra de Boole, a qual será tratada neste capítulo.

4.2 Postulados de Huntington

- Na definição de estruturas algébricas abstratas, são apresentados axiomas que estabelecem:
  - um conjunto de elementos;
  - um determinado número de operações;
  - alguns elementos particulares;
  - algumas propriedades.
- Na associação de uma determinada estrutura abstrata com um determinado sistema existente são definidos:
  - os elementos dos conjuntos;
  - o funcionamento das operações;
  - os elementos particulares.
- Entre as diversas formas de abordar a estrutura proposta por Boole, uma das mais utilizadas são os Postulados de Huntington, apresentados a seguir.
- Postulados de Huntington
  1. Existe um conjunto $\boldsymbol{K}$ de objetos ou elementos, sujeito a uma relação de equivalência, denotada pelo símbolo "=", que satisfaz ao princípio da substituição.
  2. É definida uma operação, denotada por "+", tal que, dados a e $b \in \boldsymbol{K}$, $(a+b) \in \boldsymbol{K}$.
     É definida uma operação, denotada por "$\cdot$", tal que, dados a e $b \in \boldsymbol{K}$, $(a \cdot b) \in \boldsymbol{K}$.
  3. Existe um elemento $0 \in \boldsymbol{K}$, tal que, para cada $a \in \boldsymbol{K}$, $(a+0) = a$.
     Existe um elemento $1 \in \boldsymbol{K}$, tal que, para cada $a \in \boldsymbol{K}$, $(a \cdot 1) = a$.
  4. As seguintes relações de comutatividade são válidas:
     $(a+b) = (b+a)$
     $(a \cdot b) = (b \cdot a)$
  5. As seguintes relações de distributividade são válidas:
     $a + (b \cdot c) = (a+b) \cdot (a+c)$
     $a \cdot (b+c) = (a \cdot b) + (a \cdot c)$
  6. Para cada elemento $a \in \boldsymbol{K}$ existe um elemento $\overline{a} \in \boldsymbol{K}$, tal que
     $a + \overline{a} = 1$
     $a \cdot \overline{a} = 0$
  7. Deve haver, pelo menos, um total de dois elementos a e $b \in \boldsymbol{K}$, tal que $a \neq b$.

4.3 Dualidade

- Os postulados de Huntington são apresentados em pares.
- Em cada par, um postulado pode ser obtido através do outro, efetuando-se a troca das operações "+" e "$\cdot$", bem como dos elementos 0 e 1.
- Cada teorema relacionado à álgebra de Boole possui um teorema dual.
- Ao usar a dualidade sobre a prova de um teorema, pode-se provar o seu dual.

4.4 Lemas e teoremas fundamentais

- Nessa seção, são apresentados lemas e teoremas para a álgebra de Boole.
- Teoremas são ferramentas para resolução de problemas.
- Lemas são resultados intermediários das provas dos teoremas.
- Os lemas e os teoremas, apresentados a seguir, podem ser demonstrados a partir dos postulados definidos anteriormente.
- Lemas:
  1. Os elementos 0 e 1 são únicos.
  2. Para cada $a \in \boldsymbol{K}$, $(a + a) = a$ e $(a \cdot a) = a$.
  3. Para cada $a \in \boldsymbol{K}$, $(a + 1) = 1$ e $(a \cdot 0) = 0$.
  4. Os elementos 0 e 1 são distintos e $\overline{1} = 0$.
  5. Para cada par a e $b \in \boldsymbol{K}$, $a + (a \cdot b) = a$ e $a \cdot (a + b) = a$,
  6. O elemento $\overline{a}$, definido no Postulado 6, é único, para cada $a \in \boldsymbol{K}$.
  7. Para cada $a \in \boldsymbol{K}$, $a = \overline{(\overline{a})}$.
  8. Para quaisquer três elementos a, b e $c \in \boldsymbol{K}$, $a \cdot [(a + b) + c] = [(a + b) + c] \cdot a = a$.
- Teoremas:
  1. Para quaisquer três elementos a, b e $c \in \boldsymbol{K}$,
     $a + (b + c) = (a + b) + c$
     $a \cdot (b \cdot c) = (a \cdot b) \cdot c$
  2. Para cada par a e $b \in \boldsymbol{K}$,
     $a + (\overline{a} \cdot b) = (a + b)$
     $a \cdot (\overline{a} + b) = (a \cdot b)$
  3. Para cada par a e $b \in \boldsymbol{K}$,
     $\overline{a + b} = \overline{a} \cdot \overline{b}$
     $\overline{a \cdot b} = \overline{a} + \overline{b}$
  4. Para quaisquer três elementos a, b e $c \in \boldsymbol{K}$, $(a \cdot b) + (\overline{a} \cdot c) + (b \cdot c) = (a \cdot b) + (\overline{a} \cdot c)$

4.5 Definição de uma álgebra de Boole concreta

- $\boldsymbol{K} = \{0, 1\}$
- $0 = \overline{1}$ e $1 = \overline{0}$
- $(1 \cdot 1) = (1 + 1) = (1 + 0) = (0 + 1) = 1$
- $(0 + 0) = (0 \cdot 0) = (0 \cdot 1) = (1 \cdot 0) = 0$

4.6 Outros exemplos de álgebras de Boole concretas

- Teoria de conjuntos: a associação entre a teoria de conjuntos e a álgebra de Boole abstrata é apresentada na Tabela 4.1.
- Cálculo proposicional: a associação entre o cálculo proposicional e a álgebra de Boole abstrata é apresentada na Tabela 4.2.

Teoria de Conjuntos	Álgebra de Boole Abstrata
$\cap$	$\cdot$
$\cup$	$+$
S_Z	0
S_U	1
$C(S)$	$\overline{S}$

Tabela 4.1: Tabela de mapeamento: Teoria de Conjuntos $\times$ Álgebra de Boole.

Cálculo Proposicional	Álgebra de Boole Abstrata
$\wedge$	$\cdot$
$\vee$	$+$
F	0
T	1
$\neg(S)$	$\overline{S}$

Tabela 4.2: Tabela de mapeamento: Cálculo Proposicional $\times$ Álgebra de Boole.

4.7 Isomorfismo

- Sistemas que possuem estrutura definida pela mesma álgebra abstrata são ditos sistemas isomórficos.
- Sistemas isomórficos podem ser mapeados uns nos outros.
- As operações realizadas em cada sistema isomórfico são equivalentes e podem ser relacionadas entre si.

4.8 Simplificação algébrica de expressões lógicas

- Manipulação algébrica das expressões lógicas a fim de reduzí-las a formas mais simples e, conseqüentemente, reduzir o custo de sua implementação.
- As ferramentas utilizadas são os postulados, os lemas e os teoremas da álgebra abstrata com a qual se esteja trabalhando. Também podem ser empregadas as definições da álgebra concreta em questão.
- A manipulação algébrica não sistemática depende da habilidade do profissional e não é diretamente automatizável.

4.9 Exemplos de manipulação algébrica não sistemática

- As Equações 4.1 e 4.3 ilustram possíveis manipulações algébricas, não sistemáticas, de uma mesma equação lógica, a fim de minimizá-la.

- É fácil perceber que, dependendo das escolhas realizadas, há uma grande diferença no esforço dispendido.

- Além disso, não há qualquer garantia de que a expressão final seja a expressão mínima, ou de que a expressão mínima será alcançada.

$$
\begin{aligned}
&(A+B)\cdot(A+\overline{B})\cdot(\overline{A}+B) \\
&\qquad\downarrow\ P5 \\
&([(A+B)\cdot(A)]+[(A+B)\cdot(\overline{B})])\cdot(\overline{A}+B) \\
&\qquad\downarrow\ P5 \\
&([(A\cdot A)+(B\cdot A)]+[(A\cdot\overline{B})+(B\cdot\overline{B})])\cdot(\overline{A}+B) \\
&\qquad\downarrow\ L2/P6 \\
&([A+(B\cdot A)]+[(A\cdot\overline{B})+0])\cdot(\overline{A}+B) \\
&\qquad\downarrow\ P4/P3 \\
&([A+(A\cdot B)]+[(A\cdot\overline{B})])\cdot(\overline{A}+B) \\
&\qquad\downarrow\ L5 \\
&\left[A+(A\cdot\overline{B})\right]\cdot(\overline{A}+B) \\
&\qquad\downarrow\ P5 \\
&\left[A\cdot(\overline{A}+B)\right]+\left[(A\cdot\overline{B})\cdot(\overline{A}+B)\right] \\
&\qquad\downarrow\ P5 \\
&(A\cdot\overline{A})+(A\cdot B)+(A\cdot\overline{B}\cdot\overline{A})+(A\cdot\overline{B}\cdot B) \\
&\qquad\downarrow\ P6/P4 \\
&0+(A\cdot B)+(A\cdot\overline{A}\cdot\overline{B})+(A\cdot B\cdot\overline{B}) \\
&\qquad\downarrow\ P6 \\
&0+(A\cdot B)+(0\cdot\overline{B})+(A\cdot 0) \\
&\qquad\downarrow\ P4/L3 \\
&0+(A\cdot B)+(\overline{B}\cdot 0)+0 \\
&\qquad\downarrow\ L3 \\
&0+(A\cdot B)+0+0 \\
&\qquad\downarrow\ P4 \\
&(A\cdot B)+0+0+0 \\
&\qquad\downarrow\ P3/P3/P3 \\
&(A\cdot B)
\end{aligned}
\tag{4.1}
$$

$$
\begin{aligned}
&(A+B)\cdot(A+\overline{B})\cdot(\overline{A}+B) \\
&\qquad\qquad \downarrow \quad P5 \\
&\big[A\cdot(B+\overline{B})\big]\cdot(\overline{A}+B) \\
&\qquad\qquad \downarrow \quad P6 \\
&(A\cdot 1)\cdot(\overline{A}+B) \\
&\qquad\qquad \downarrow \quad P3 \\
&A\cdot(\overline{A}+B) \\
&\qquad\qquad \downarrow \quad P5 \\
&(A\cdot\overline{A})+(A\cdot B) \\
&\qquad\qquad \downarrow \quad P6 \\
&0+(A\cdot B) \\
&\qquad\qquad \downarrow \quad P4 \\
&(A\cdot B)+0 \\
&\qquad\qquad \downarrow \quad P3 \\
&(A\cdot B)
\end{aligned}
\tag{4.2}
$$

$$
\begin{aligned}
&(A+B)\cdot(A+\overline{B})\cdot(\overline{A}+B) \\
&\qquad\qquad \downarrow \quad L2 \\
&(A+B)\cdot(A+B)\cdot(A+\overline{B})\cdot(\overline{A}+B) \\
&\qquad\qquad \downarrow \quad P4 \\
&(A+B)\cdot(A+\overline{B})\cdot(A+B)\cdot(\overline{A}+B) \\
&\qquad\qquad \downarrow \quad P5 \\
&\big[A\cdot(B+\overline{B})\big]\cdot\big[(A+\overline{A})\cdot B)\big] \\
&\qquad\qquad \downarrow \quad P6 \\
&A\cdot 1\cdot 1\cdot B \\
&\qquad\qquad \downarrow \quad P3/P3 \\
&(A\cdot B)
\end{aligned}
\tag{4.3}
$$

4.10 Manipulação através do Diagrama de Venn

- Na tentativa de sistematizar o processo de simplificação de uma expressão lógica, pode-se aproveitar o isomorfismo existente entre a Teoria de Conjuntos e o Cálculo Proposicional.
- Mapeando-se as operações e os elementos dos dois sistemas, o Diagrama de Venn pode ser usado na simplificação de expressões lógicas que envolvam 2 ou 3 variáveis.
- Ainda assim, embora seja um processo sistemático, a etapa final da simplificação através do Diagrama de Venn exige habilidade para encontrar a expressão mais simples.
- Além disso, para expressões que envolvam mais variáveis, o processo torna-se complexo e confuso.
- Portanto, devem ser utilizadas ferramentas mais eficientes, as quais serão tratadas a seguir.

4.11 Resumo das relações algébricas

As Tabelas 4.3 – 4.7 apresentam um resumo das relações algébricas abordadas neste capítulo: os postulados, os lemas, os teoremas, a definição de uma álgebra de Boole concreta e os isomorfismos.

Postulados de Huntington	
P3	$\begin{cases} a+0 &= a \\ a\cdot 1 &= a \end{cases}$
P4	$\begin{cases} (a+b) &= (b+a) \\ (a\cdot b) &= (b\cdot a) \end{cases}$
P5	$\begin{cases} a+(b\cdot c) &= (a+b)\cdot(a+c) \\ a\cdot(b+c) &= (a\cdot b)+(a\cdot c) \end{cases}$
P6	$\begin{cases} a+\overline{a} &= 1 \\ a\cdot\overline{a} &= 0 \end{cases}$

Tabela 4.3: Resumo dos postulados de Huntington para a álgebra de Boole abstrata.

Lemas	
L2	$\begin{cases} a+a &= a \\ a\cdot a &= a \end{cases}$
L3	$\begin{cases} a+1 &= 1 \\ a\cdot 0 &= 0 \end{cases}$
L4	$\begin{cases} 0 &= \overline{1} \\ \overline{0} &= 1 \end{cases}$
L5	$\begin{cases} a+(a\cdot b) &= a \\ a\cdot(a+b) &= a \end{cases}$
L7	$\{\ a = \overline{(\overline{a})}$
L8	$\{\ a\cdot[(a+b)+c] = [(a+b)+c]\cdot a = a$

Tabela 4.4: Resumo dos lemas da álgebra de Boole abstrata.

Teoremas	
T1	$\begin{cases} a+(b+c) = (a+b)+c \\ a\cdot(b\cdot c) = (a\cdot b)\cdot c \end{cases}$
T2	$\begin{cases} a+(\overline{a}\cdot b) = (a+b) \\ a\cdot(\overline{a}+b) = (a\cdot b) \end{cases}$
T3	$\begin{cases} \overline{(a+b)} = \overline{a}\cdot\overline{b} \\ \overline{(a\cdot b)} = \overline{a}+\overline{b} \end{cases}$
T4	$\begin{cases} (a\cdot b)+(\overline{a}\cdot c)+(b\cdot c) = (a\cdot b)+(\overline{a}\cdot c) \\ (a+b)\cdot(\overline{a}+c)\cdot(b+c) = (a+b)\cdot(\overline{a}+c) \end{cases}$

Tabela 4.5: Resumo dos teoremas da álgebra de Boole abstrata.

Álgebra de Boole Concreta	
Elementos	$\{K=\{0,1\}$
Complementos	$\begin{cases} 0 = \overline{1} \\ \overline{0} = 1 \end{cases}$
Identidades	$\begin{cases} 1\cdot 1 = 1+1 = 1+0 = 0+1 = 1 \\ 0+0 = 0\cdot 0 = 0\cdot 1 = 1\cdot 0 = 0 \end{cases}$

Tabela 4.6: Resumo da definição de uma álgebra de Boole concreta.

Isomorfismos		
Teoria de Conjuntos	Cálculo Proposicional	Álgebra de Boole Concreta
$\cup$	$\vee$	$+$
$\cap$	$\wedge$	$\cdot$
S_Z	F	0
S_U	T	1
$C(S)$	$\neg(S)$	$\overline{S}$

Tabela 4.7: Resumo das relações de isomorfismo.

4.12 Exercícios propostos

1. Considerando o isomorfismo do Cálculo Proposicional com a Álgebra de Boole Concreta Binária:

   (a) Provar que os postulados, os lemas e os teoremas, da Álgebra Abstrata de Boole, se aplicam para o Cálculo Proposicional.

   (b) Para cada um dos operadores lógicos (unário e binários), escrever sua Tabela Verdade usando a notação da Álgebra de Boole.

2. Para uma função genérica de duas variáveis, montar sua Tabela Verdade e identificar, em um Diagrama de Venn correspondente, cada uma das possibilidades da tabela.

3. Para uma função genérica de três variáveis, montar sua Tabela Verdade e identificar, em um Diagrama de Venn correspondente, cada uma das possibilidades da tabela.

4. Para os exercícios listados abaixo, considerar as equações lógicas apresentadas em seguida.

   (a) Escrever as equações booleanas referentes às equações lógicas fornecidas.

   (b) Montar a Tabela Verdade para cada uma das equações lógicas fornecidas, usando a notação da Álgebra de Boole.

   (c) Aplicando os postulados, os lemas e os teoremas, da Álgebra Abstrata de Boole, apresentar uma simplificação para as equações booleanas referentes às equações lógicas fornecidas.

   (d) Utilizando o Diagrama de Venn, apresentar uma simplificação para as equações booleanas referentes às equações lógicas fornecidas.

   Equações lógicas:

   i. $F(A, B) = (A \vee B) \wedge (A \vee \neg B) \wedge (\neg A \vee B)$.
   ii. $F(A, B) = (\neg A \vee B) \wedge (A \vee \neg B) \wedge (\neg A \vee \neg B)$.
   iii. $F(A, B) = (A \wedge \neg B) \vee (\neg A \wedge B) \vee (A \wedge B)$.
   iv. $F(A, B) = (\neg A \wedge \neg B) \vee (\neg A \wedge B) \vee (A \wedge \neg B)$.
   v. $F(A, B, C, D) = (A \vee B) \wedge (\neg(C \wedge D))$.

Capítulo 5

Formas padrões para representação de expressões booleanas

5.1 Introdução

- O projeto de sistemas digitais convencionais envolve a implementação de equações lógicas.
- Equações lógicas expressas por uma álgebra de Boole são denominadas equações booleanas.
- A minimização do custo de implementação de um projeto está associada à simplificação de suas equações booleanas.
- Um processo eficiente de simplificação deve ser simples de se entender, fácil de se operar, de rápida execução e completamente sistemático, a fim de permitir sua automatização.
- Os processos sistemáticos de simplificação que serão apresentados adiante trabalham sobre uma função expressa em formas padrões.
- Assim, para que se possa usar tais ferramentas de projeto, expressões booleanas genéricas devem ser inicialmente expandidas para tais formas.
- As formas padrões básicas são as decomposições em mintermos e maxtermos.
- As demais formas padrões surgem como resultado de manipulações das formas padrões básicas.
- Conjuntos de formas padrões:
  - Grupo AND-OR → AND-OR , NAND-NAND, OR-NAND, NOR-OR.
  - Grupo OR-AND → AND-NOR, NAND-AND , OR-AND , NOR-NOR.

5.2 Definições

- Literal: variável lógica booleana ou seu complemento.
- Termo produto: série de literais relacionados pelo operador AND.
- Termo soma: série de literais relacionados pelo operador OR.
- Termo normal: termo produto ou termo soma onde nenhum literal aparece mais de uma vez.
  - Uma vez que $(A \cdot A) = (A + A) = A$, $(A + \overline{A}) = 1$ e $(A \cdot \overline{A}) = 0$, conclui-se que múltiplas ocorrências de um literal em um termo soma ou em um termo produto acarretam: i) redundância ou ii) funções triviais.
  - Portanto, pode-se dizer que, para fins de simplificação, a forma normal é a melhor forma de representação.
- Soma de produtos (SOP): combinação de termos produto através do operador OR.
- Produto de somas (POS): combinação de termos soma através do operador AND.
- Soma de produtos normal: SOP onde os termos produto são termos normais.
- Produto de somas normal: POS onde os termos soma são termos normais.
- Forma normal expandida: forma normal (SOP ou POS) onde cada termo contém todos os literais envolvidos na expressão booleana.
  - Em uma forma SOP normal expandida, os termos produto são chamados de: produtos padrões, produtos canônicos ou mintermos.
  - A forma SOP normal expandida recebe as seguintes denominações: SOP padrão, SOP canônica, soma de mintermos, decomposição em mintermos ou forma normal disjuntiva completa.
  - Em uma forma POS normal expandida, os termos soma são chamados de: somas padrões, somas canônicas ou maxtermos.
  - A forma POS normal expandida recebe as seguintes denominações: POS padrão, POS canônica, produto de maxtermos, decomposição em maxtermos ou forma normal conjuntiva completa.

- Exemplos da definição de mintermos e maxtermos, para três variáveis, são apresentados na Tabela 5.1 e na Tabela 5.2, respectivamente.

Linha	*A*	*B*	*C*	*Produto*	*Mintermo*
0	0	0	0	$\left(\overline{A} \cdot \overline{B} \cdot \overline{C}\right)$	m_0
1	0	0	1	$\left(\overline{A} \cdot \overline{B} \cdot C\right)$	m_1
2	0	1	0	$\left(\overline{A} \cdot B \cdot \overline{C}\right)$	m_2
3	0	1	1	$\left(\overline{A} \cdot B \cdot C\right)$	m_3
4	1	0	0	$\left(A \cdot \overline{B} \cdot \overline{C}\right)$	m_4
5	1	0	1	$\left(A \cdot \overline{B} \cdot C\right)$	m_5
6	1	1	0	$\left(A \cdot B \cdot \overline{C}\right)$	m_6
7	1	1	1	$(A \cdot B \cdot C)$	m_7

Tabela 5.1: Definição de mintermos para três variáveis (A,B,C).

Linha	*A*	*B*	*C*	*Soma*	*Maxtermo*
0	0	0	0	$(A + B + C)$	M_0 $(ou\ M_7)$
1	0	0	1	$\left(A + B + \overline{C}\right)$	M_1 $(ou\ M_6)$
2	0	1	0	$\left(A + \overline{B} + C\right)$	M_2 $(ou\ M_5)$
3	0	1	1	$\left(A + \overline{B} + \overline{C}\right)$	M_3 $(ou\ M_4)$
4	1	0	0	$\left(\overline{A} + B + C\right)$	M_4 $(ou\ M_3)$
5	1	0	1	$\left(\overline{A} + B + \overline{C}\right)$	M_5 $(ou\ M_2)$
6	1	1	0	$\left(\overline{A} + \overline{B} + C\right)$	M_6 $(ou\ M_1)$
7	1	1	1	$\left(\overline{A} + \overline{B} + \overline{C}\right)$	M_7 $(ou\ M_0)$

Tabela 5.2: Definição de maxtermos para três variáveis (A,B,C).

5.3 Obtenção de formas SOP e POS padrões

Dada uma expressão booleana qualquer, pode-se obter uma forma padrão (SOP ou POS) através dos seguintes procedimentos: manipulação algébrica e utilização de tabela verdade. Ambos são abordados a seguir.

5.3.1 Manipulação algébrica

- Para se obter uma forma normal:
  - Inicialmente, se houver negação de algum termo que não seja um literal, deve-se aplicar o teorema de DeMorgan.
  - Quando houver negação apenas de literais, deve-se aplicar, repetidamente, as regras de distributividade.
  - Finalmente, deve-se eliminar literais e/ou termos redundantes ou triviais.

- Para se obter a forma normal expandida:
  - Primeiro, deve-se inserir os literais faltosos nos termos normais. Isso é feito aplicando-se os postulados, os lemas e os teoremas da álgebra de Boole sobre a forma normal.
  - Em seguida, deve-se eliminar literais e/ou termos redundantes ou triviais.
- Um exemplo do procedimento para a obtenção de uma forma POS padrão é apresentado na Equação (5.1), para $F = f(A, B, C, D)$.

$$
\begin{aligned}
F &= (A+B)\cdot\overline{(C\cdot D)} \\
&= (A+B)\cdot(\overline{C}+\overline{D}) \\
&= (A+B+0)\cdot(0+\overline{C}+\overline{D}) \\
&= [A+B+(C\cdot\overline{C})]\cdot[(B\cdot\overline{B})+\overline{C}+\overline{D}] \\
&= (A+B+C)\cdot(A+B+\overline{C})\cdot(B+\overline{C}+\overline{D})\cdot(\overline{B}+\overline{C}+\overline{D}) \\
&= (A+B+C+0)\cdot(A+B+\overline{C}+0)\cdot(0+B+\overline{C}+\overline{D})\cdot(0+\overline{B}+\overline{C}+\overline{D}) \\
&= [A+B+C+(D\cdot\overline{D})]\cdot[A+B+\overline{C}+(D\cdot\overline{D})]\cdot \\
&\quad [(A\cdot\overline{A})+B+\overline{C}+\overline{D}]\cdot[(A\cdot\overline{A})+\overline{B}+\overline{C}+\overline{D}] \\
&= (A+B+C+D)\cdot(A+B+C+\overline{D})\cdot(A+B+\overline{C}+D)\cdot \\
&\quad (A+B+\overline{C}+\overline{D})\cdot(A+B+\overline{C}+\overline{D})\cdot(\overline{A}+B+\overline{C}+\overline{D})\cdot \\
&\quad (A+\overline{B}+\overline{C}+\overline{D})(\overline{A}+\overline{B}+\overline{C}+\overline{D}) \\
&= (A+B+C+D)\cdot(A+B+C+\overline{D})\cdot(A+B+\overline{C}+D)\cdot \\
&\quad (A+B+\overline{C}+\overline{D})\cdot(\overline{A}+B+\overline{C}+\overline{D})\cdot \\
&\quad (A+\overline{B}+\overline{C}+\overline{D})(\overline{A}+\overline{B}+\overline{C}+\overline{D}) \\
&= \prod M(0,1,2,3,11,7,15) \qquad (5.1)
\end{aligned}
$$

- Um exemplo do procedimento para a obtenção de uma forma SOP padrão é apresentado na Equação (5.2), para $F = f(A, B, C, D)$.

$$
\begin{aligned}
F &= (A+B)\cdot\overline{(C\cdot D)} \\
&= (A+B)\cdot(\overline{C}+\overline{D}) \\
&= [(A+B)\cdot\overline{C}]+[(A+B)\cdot\overline{D}] \\
&= [(A\cdot\overline{C})+(B\cdot\overline{C})]+[(A\cdot\overline{D})+(B\cdot\overline{D})] \\
&= (A\cdot\overline{C})+(B\cdot\overline{C})+(A\cdot\overline{D})+(B\cdot\overline{D}) \\
&= (A\cdot 1\cdot\overline{C})+(1\cdot B\cdot\overline{C})+(A\cdot 1\cdot\overline{D})+(1\cdot B\cdot\overline{D}) \\
&= [A\cdot(B+\overline{B})\cdot\overline{C}]+[(A+\overline{A})\cdot B\cdot\overline{C}]+ \\
&\quad [A\cdot(B+\overline{B})\cdot\overline{D}]+[(A+\overline{A})\cdot B\cdot\overline{D}] \\
&= (A\cdot B\cdot\overline{C})+(A\cdot\overline{B}\cdot\overline{C})+(A\cdot B\cdot\overline{C})+(\overline{A}\cdot B\cdot\overline{C})+ \\
&\quad (A\cdot B\cdot\overline{D})+(A\cdot\overline{B}\cdot\overline{D})+(A\cdot B\cdot\overline{D})+(\overline{A}\cdot B\cdot\overline{D}) \\
&= (A\cdot B\cdot\overline{C})+(A\cdot\overline{B}\cdot\overline{C})+(\overline{A}\cdot B\cdot\overline{C})+ \\
&\quad (A\cdot B\cdot\overline{D})+(A\cdot\overline{B}\cdot\overline{D})+(\overline{A}\cdot B\cdot\overline{D}) \\
&= (A\cdot B\cdot\overline{C}\cdot 1)+(A\cdot\overline{B}\cdot\overline{C}\cdot 1)+(\overline{A}\cdot B\cdot\overline{C}\cdot 1)+ \\
&\quad (A\cdot B\cdot 1\cdot\overline{D})+(A\cdot\overline{B}\cdot 1\cdot\overline{D})+(\overline{A}\cdot B\cdot 1\cdot\overline{D}) \\
&= [A\cdot B\cdot\overline{C}\cdot(D+\overline{D})]+[A\cdot\overline{B}\cdot\overline{C}\cdot(D+\overline{D})]+[\overline{A}\cdot B\cdot\overline{C}\cdot(D+\overline{D})]+ \\
&\quad [A\cdot B\cdot(C+\overline{C})\cdot\overline{D}]+[A\cdot\overline{B}\cdot(C+\overline{C})\cdot\overline{D}]+[\overline{A}\cdot B\cdot(C+\overline{C})\cdot\overline{D}] \\
&= (A\cdot B\cdot\overline{C}\cdot D)+(A\cdot B\cdot\overline{C}\cdot\overline{D})+(A\cdot\overline{B}\cdot\overline{C}\cdot D)+ \\
&\quad (A\cdot\overline{B}\cdot\overline{C}\cdot\overline{D})+(\overline{A}\cdot B\cdot\overline{C}\cdot D)+(\overline{A}\cdot B\cdot\overline{C}\cdot\overline{D})+ \\
&\quad (A\cdot B\cdot C\cdot\overline{D})+(A\cdot B\cdot\overline{C}\cdot\overline{D})+(A\cdot\overline{B}\cdot C\cdot\overline{D})+ \\
&\quad (A\cdot\overline{B}\cdot\overline{C}\cdot\overline{D})+(\overline{A}\cdot B\cdot C\cdot\overline{D})+(\overline{A}\cdot B\cdot\overline{C}\cdot\overline{D}) \\
&= (A\cdot B\cdot\overline{C}\cdot D)+(A\cdot B\cdot\overline{C}\cdot\overline{D})+(A\cdot\overline{B}\cdot\overline{C}\cdot D)+(A\cdot\overline{B}\cdot\overline{C}\cdot\overline{D})+ \\
&\quad (\overline{A}\cdot B\cdot\overline{C}\cdot D)+(\overline{A}\cdot B\cdot\overline{C}\cdot\overline{D})+(A\cdot B\cdot C\cdot\overline{D})+(A\cdot\overline{B}\cdot C\cdot\overline{D})+ \\
&\quad (\overline{A}\cdot B\cdot C\cdot\overline{D}) \\
&= \sum m(13,12,9,8,5,4,14,10,6) \qquad (5.2)
\end{aligned}
$$

5.3.2 Utilização de tabela verdade

A partir de uma tabela verdade, pode-se obter diretamente as formas padrões na forma de decomposição em mintermos ou maxtermos. Ambas as formas são discutidas a seguir.

Decomposição em mintermos

- Dada uma expressão booleana, pode-se montar uma tabela verdade que a represente, como demonstrado na Tabela 5.3, para uma função $X = f(A, B, C)$.

A	B	C	X	Y_1	Y_2	Y_3	*Linha*	*Lógica*	*Mintermo*
0	0	0	0	0	0	0	0	$\left(\overline{A} \cdot \overline{B} \cdot \overline{C}\right)$	m_0
0	0	1	1	1	0	0	1	$\left(\overline{A} \cdot \overline{B} \cdot C\right)$	m_1
0	1	0	0	0	0	0	2	$\left(\overline{A} \cdot B \cdot \overline{C}\right)$	m_2
0	1	1	0	0	0	0	3	$\left(\overline{A} \cdot B \cdot C\right)$	m_3
1	0	0	1	0	1	0	4	$\left(A \cdot \overline{B} \cdot \overline{C}\right)$	m_4
1	0	1	0	0	0	0	5	$\left(A \cdot \overline{B} \cdot C\right)$	m_5
1	1	0	1	0	0	1	6	$\left(A \cdot B \cdot \overline{C}\right)$	m_6
1	1	1	0	0	0	0	7	$(A \cdot B \cdot C)$	m_7

Tabela 5.3: Exemplo de função e associação de mintermos.

- Da Tabela 5.3, pode-se escrever que

$$\begin{array}{rcrcrcl} Y_1 & = & f_1(A,B,C) & = & \left(\overline{A} \cdot \overline{B} \cdot C\right) & = & m_1 \\ Y_2 & = & f_2(A,B,C) & = & \left(A \cdot \overline{B} \cdot \overline{C}\right) & = & m_4 \\ Y_3 & = & f_3(A,B,C) & = & \left(A \cdot B \cdot \overline{C}\right) & = & m_6 \end{array}$$

  e que $X = f(A, B, C) = (Y_1) + (Y_2) + (Y_3)$.

- Pelas definições apresentadas, as funções auxiliares Y_1–Y_3 são mintermos e a função X pode ser descrita pela forma SOP padrão $X = m_1 + m_4 + m_6 = \sum m(1, 4, 6)$.

- Analisando-se as funções auxiliares Y, pode-se observar que, para cada combinação das variáveis, apenas um dos termos produto apresenta um valor lógico 1, enquanto todos os outros assumem o valor lógico 0. Essa é a razão pela qual tais termos são denominados produtos canônicos ou mintermos.

- Uma vez que toda expressão booleana é completamente representada por uma tabela verdade e, a partir da tabela verdade, sempre é possível se obter uma forma SOP padrão, pode-se enunciar o teorema a seguir.

- **Teorema**: Qualquer expressão booleana de N variáveis, $y = f(x_1, x_2, \cdots, x_N)$, pode ser expressa por uma forma SOP padrão.

Decomposição em maxtermos

- Dada uma expressão booleana, pode-se montar uma tabela verdade que a represente, como demonstrado na Tabela 5.4, para uma função $X = f(A, B, C)$.

A	B	C	X	Z_1	Z_2	Z_3	Z_4	Z_5	*Linha*	*Lógica*	$Maxtermo$
0	0	0	0	0	1	1	1	1	0	$(A+B+C)$	M_0 (*ou* M_7)
0	0	1	1	1	1	1	1	1	1	$\left(A+B+\overline{C}\right)$	M_1 (*ou* M_6)
0	1	0	0	1	0	1	1	1	2	$\left(A+\overline{B}+C\right)$	M_2 (*ou* M_5)
0	1	1	0	1	1	0	1	1	3	$\left(A+\overline{B}+\overline{C}\right)$	M_3 (*ou* M_4)
1	0	0	1	1	1	1	1	1	4	$\left(\overline{A}+B+C\right)$	M_4 (*ou* M_3)
1	0	1	0	1	1	1	0	1	5	$\left(\overline{A}+B+\overline{C}\right)$	M_5 (*ou* M_2)
1	1	0	1	1	1	1	1	1	6	$\left(\overline{A}+\overline{B}+C\right)$	M_6 (*ou* M_1)
1	1	1	0	1	1	1	1	0	7	$\left(\overline{A}+\overline{B}+\overline{C}\right)$	M_7 (*ou* M_0)

Tabela 5.4: Exemplo de função e definição de maxtermos.

- Da Tabela 5.4, pode-se escrever que

$$\begin{array}{rcccccc} Z_1 & = & f_1(A,B,C) & = & (A+B+C) & = & M_0 \\ Z_2 & = & f_2(A,B,C) & = & \left(A+\overline{B}+C\right) & = & M_2 \\ Z_3 & = & f_3(A,B,C) & = & \left(A+\overline{B}+\overline{C}\right) & = & M_3 \\ Z_4 & = & f_4(A,B,C) & = & \left(\overline{A}+B+\overline{C}\right) & = & M_5 \\ Z_5 & = & f_5(A,B,C) & = & \left(\overline{A}+\overline{B}+\overline{C}\right) & = & M_7 \end{array}$$

  e que $X = f(A, B, C) = (Z_1) \cdot (Z_2) \cdot (Z_3) \cdot (Z_4) \cdot (Z_5)$.

- Pelas definições apresentadas, as funções auxiliares Z_1–Z_5 são maxtermos e a função X pode ser descrita pela forma POS padrão $X = M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_7 = \prod M(0, 2, 3, 5, 7)$.

- Analisando-se as funções auxiliares Z, pode-se observar que, para cada combinação das variáveis, apenas um dos termos soma apresenta um valor lógico 0, enquanto todos os outros assumem o valor lógico 1. Essa é a razão pela qual tais termos são denominados somas canônicas ou maxtermos.

- Uma vez que toda expressão booleana é completamente representada por uma tabela verdade e, a partir da tabela verdade, sempre é possível se obter uma forma POS padrão, pode-se enunciar o teorema a seguir.

- **Teorema**: Qualquer expressão booleana de N variáveis, $y = f(x_1, x_2, \cdots, x_N)$, pode ser expressa por uma forma POS padrão.

5.4 Conjuntos de formas padrões

- Uma expressão booleana pode ser representada por um total de oito formas padrões.
- Uma soma de mintermos é identificada como uma forma AND-OR.
- Um produto de maxtermos é identificado como uma forma OR-AND.
- A partir da forma AND-OR, pode-se obter o seguinte grupo de formas padrões: AND-OR, NAND-NAND, OR-NAND, NOR-OR.
- A partir da forma OR-AND, pode-se obter o seguinte grupo de formas padrões: OR-AND, NOR-NOR, AND-NOR, NAND-AND.
- Dentro de um mesmo grupo, as formas podem ser obtidas através da aplicação sucessiva dos Teoremas de DeMorgan.
- A mudança de grupo pode ser realizada aplicando-se a regra de distributividade entre as formas AND-OR e OR-AND.
- As Tabelas (5.5)–(5.7), exemplificam, para a função XOR, a obtenção do grupo AND-OR, a mudança de grupo e a obtenção do grupo OR-AND, respectivamente.
- Partindo-se da tabela verdade, pode-se obter diretamente algumas formas:
  - A soma de mintermos da função fornece a forma AND-OR.
  - O produto de maxtermos da função, fornece a forma OR-AND.
  - A soma de mintermos da função complementar, fornece a forma AND-NOR (grupo OR-AND).
  - O produto de maxtermos da função complementar, fornece a forma OR-NAND (grupo AND-OR).
- Assim, uma outra técnica para mudança de grupo é montar a tabela verdade da função, a partir de uma dada forma. De posse da tabela verdade, pode-se obter uma forma do outro grupo.
- O projeto e a análise de circuitos digitais convencionais são baseados na formação, na manipulação e na implementação de funções lógicas booleanas. Por dois motivos básicos, as formas AND-OR e OR-AND são as formas mais utilizadas na representação de tais funções. Primeiro, ela são diretamente obtidas no processo de especificação do problema. Segundo, elas são mais próximas da forma como se processa o pensamento (expressão lógica) do ser humano.
- Por outro lado, as formas NAND-NAND e NOR-NOR são as formas básicas de operação dos circuitos eletro-eletrônicos usados para implementar as funções lógicas booleanas.
- Portanto, transformações entre tais formas são freqüentemente realizadas.
- As formas padrões possuem dois grandes atrativos. Por um lado, e de uma forma geral, elas apresentam o menor retardo de operação, uma vez que são compostas apenas por dois planos de lógica. Além disso, elas são o ponto de partida para um processo de simplificação sistemático e eficiente, conforme será abordado em seguida.

Expressão boolena					Forma padrão
$F(A,B)$	$=$	$A \oplus B$			
	$=$	$\big(\overline{A} \cdot B\big) + \big(A \cdot \overline{B}\big)$			AND-OR
	$=$	$\overline{\overline{\big(\overline{A} \cdot B\big) + \big(A \cdot \overline{B}\big)}}$			
	$=$	$\overline{\overline{\big(\overline{A} \cdot B\big)} \cdot \overline{\big(A \cdot \overline{B}\big)}}$	$=$	$(\overline{A} \uparrow B) \uparrow (A \uparrow \overline{B})$	NAND-NAND
	$=$	$\overline{\big(A + \overline{B}\big) \cdot \big(\overline{A} + B\big)}$	$=$	$\big(A + \overline{B}\big) \uparrow \big(\overline{A} + B\big)$	OR-NAND
	$=$	$\overline{\big(A + \overline{B}\big) + \big(\overline{A} + B\big)}$	$=$	$\big(A \downarrow \overline{B}\big) + \big(\overline{A} \downarrow B\big)$	NOR-OR
	$=$	$\big(\overline{A} \cdot B\big) + \big(A \cdot \overline{B}\big)$			AND-OR

Tabela 5.5: Exemplo da obtenção do grupo AND-OR para a função XOR.

$F(A,B)$	$=$	$A \oplus B$
	$=$	$\big(\overline{A} \cdot B\big) + \big(A \cdot \overline{B}\big)$
	$=$	$\big[\big(\overline{A} \cdot B\big) + A\big] \cdot \big[\big(\overline{A} \cdot B\big) + \overline{B}\big]$
	$=$	$\big[(\overline{A} + A) \cdot (B + A)\big] \cdot \big[\big(\overline{A} + \overline{B}\big) \cdot \big(B + \overline{B}\big)\big]$
	$=$	$\big[(1) \cdot (B + A)\big] \cdot \big[\big(\overline{A} + \overline{B}\big) \cdot (1)\big]$
	$=$	$(B + A) \cdot \big(\overline{A} + \overline{B}\big)$
	$=$	$\big(\overline{A} + \overline{B}\big) \cdot (A + B)$

Tabela 5.6: Exemplo da mudança de grupo para a função XOR.

Expressão booleana					Forma padrão
$F(A,B)$	$=$	$A \oplus B$			
	$=$	$\big(\overline{A} + \overline{B}\big) \cdot (A + B)$			OR-AND
	$=$	$\overline{\overline{\big(\overline{A} + \overline{B}\big) \cdot (A + B)}}$			
	$=$	$\overline{\overline{\big(\overline{A} + \overline{B}\big)} + \overline{(A + B)}}$	$=$	$(\overline{A} \downarrow \overline{B}) \downarrow (A \downarrow B)$	NOR-NOR
	$=$	$\overline{(A \cdot B) + \big(\overline{A} \cdot \overline{B}\big)}$	$=$	$(A \cdot B) \downarrow \big(\overline{A} \cdot \overline{B}\big)$	AND-NOR
	$=$	$\overline{(A \cdot B)} \cdot \overline{\big(\overline{A} \cdot \overline{B}\big)}$	$=$	$(A \uparrow B) \cdot \big(\overline{A} \uparrow \overline{B}\big)$	NAND-AND
	$=$	$\big(\overline{A} + \overline{B}\big) \cdot (A + B)$			OR-AND

Tabela 5.7: Exemplo da obtenção do grupo OR-AND para a função XOR.

5.5 Exercícios propostos

1. Para os exercícios listados abaixo, considerar as equações booleanas apresentadas em seguida.

   (a) Algebricamente, obter a forma SOP normal da equação fornecida.

   (b) Algebricamente, obter a forma SOP padrão da equação fornecida.

   (c) Expressar a função por uma lista de mintermos.

   (d) A partir da SOP padrão, obter as demais formas do seu grupo.

   (e) Algebricamente, obter a forma POS normal da equação fornecida.

   (f) Algebricamente, obter a forma POS padrão da equação fornecida.

   (g) Expressar a função por uma lista de maxtermos.

   (h) A partir da POS padrão, obter as demais formas do seu grupo.

   (i) Expressar a função por uma tabela verdade, com notação booleana.

   Equações boolenas:

   i. $F(A,B,C) = \left\{B \cdot \left[\left(\overline{A} \cdot \overline{C}\right) + (A \cdot C)\right]\right\} + \overline{\left\{B + \left[\left(\overline{A} + \overline{C}\right) \cdot (A + C)\right]\right\}}$.

   ii. $F(A,B,C) = \overline{\left\{\left[\overline{(A+B)} + C\right] \cdot \left[\overline{A} + \overline{\left(B + \overline{C}\right)}\right]\right\}}$.

Capítulo 6

Simplificação algébrica de expressões booleanas

6.1 Operações básicas para a simplificação sistemática de expressões booleanas

No processo sistemático de simplificação algébrica de expressões booleanas, duas operações são fundamentais: a aglutinação e a replicação, as quais são definidas nas Equações (6.1) e (6.2), respectivamente. Da Álgebra de Boole, a aglutinação utiliza os Postulados 5, 6 e 3, enquanto a replicação emprega o Lema 2.

$$\text{Aglutinação} \begin{cases} (A \cdot B) + (A \cdot \overline{B}) = A \cdot (B + \overline{B}) = A \cdot 1 = A \\ (A + B) \cdot (A + \overline{B}) = A + (B \cdot \overline{B}) = A + 0 = A \end{cases} \tag{6.1}$$

$$\text{Replicação} \begin{cases} A = A + A + A + \ldots \\ A = A \cdot A \cdot A \cdot \ldots \end{cases} \tag{6.2}$$

6.2 Simplificação de expressões booleanas a partir das formas padrões SOP e POS

As formas AND-OR (SOP) e OR-AND (POS) apresentam duas grandes vantagens: a facilidade de descrição do problema e a quantidade de planos lógicos utilizados na sua implementação. Porém, nem sempre tais formas estão em sua expressão mais simples, pois, nestes casos, apresentam redundâncias.

A aplicação das operações de aglutinação e de replicação em funções lógicas expressas nas formas padrões SOP e POS é a base para um processo sistemático de simplificação.

6.2.1 Uso da aglutinação

A propriedade de distributividade mostra que, se dois termos diferem de apenas um literal (A e $\overline{A}$), eles podem ser fatorados. Surgem, desse modo, combinações do literal com seu complemento. Tais combinações geram valores lógicos 0 ou 1, os quais podem ser eliminados da expressão. Isto é exemplificado nas Equações (6.1) , (6.3) e (6.4).

$$\begin{aligned} F(A,B,C) &= \sum m(4,6) \\ &= m_4 + m_6 \\ &= (A \cdot \overline{B} \cdot \overline{C}) + (A \cdot B \cdot \overline{C}) \\ &= (A \cdot \overline{C} \cdot \overline{B}) + (A \cdot \overline{C} \cdot B) \\ &= [(A \cdot \overline{C}) \cdot \overline{B})] + [(A \cdot \overline{C}) \cdot B)] \\ &= (A \cdot \overline{C}) \cdot (\overline{B} + B) \\ &= (\overline{A} \cdot \overline{C}) \cdot (1) \\ &= (A \cdot \overline{C}) \end{aligned} \tag{6.3}$$

$$\begin{aligned} F(A,B,C) &= \prod (0,4) \\ &= M_0 \cdot M_4 \\ &= (A + B + C) \cdot (\overline{A} + B + C) \\ &= [A + (B + C)] \cdot [\overline{A} + (B + C)] \\ &= (A \cdot \overline{A}) + (B + C) \\ &= (0) + (B + C) \\ &= (B + C) \end{aligned} \tag{6.4}$$

6.2.2 Uso da replicação

A operação de replicação possibilita que um mesmo termo seja combinado com diversos outros, para que se possa obter simplificações através da aglutinação. Isto é exemplificado na Equação (6.5).

$$\begin{aligned} F(A,B,C) &= \sum m(0,1,2) \\ &= m_0 + m_1 + m_2 \\ &= (\overline{A} \cdot \overline{B} \cdot \overline{C}) + (\overline{A} \cdot \overline{B} \cdot C) + (\overline{A} \cdot B \cdot \overline{C}) \\ &= m_0 + m_1 + m_2 + m_0 \\ &= (\overline{A} \cdot \overline{B} \cdot \overline{C}) + (\overline{A} \cdot \overline{B} \cdot C) + (\overline{A} \cdot B \cdot \overline{C}) + (\overline{A} \cdot \overline{B} \cdot \overline{C}) \\ &= [(\overline{A} \cdot \overline{B} \cdot \overline{C}) + (\overline{A} \cdot \overline{B} \cdot C)] + [(\overline{A} \cdot B \cdot \overline{C}) + (\overline{A} \cdot \overline{B} \cdot \overline{C})] \\ &= [(\overline{A} \cdot \overline{B}) \cdot (\overline{C} + C)] + [(\overline{A} \cdot \overline{C}) \cdot (\overline{B} + B)] \\ &= (\overline{A} \cdot \overline{B}) + (\overline{A} \cdot \overline{C}) \\ &= A \cdot (\overline{B} + \overline{C}) \end{aligned} \tag{6.5}$$

6.3 Eliminação sistemática de literais

- Aplicando-se as operações de aglutinação e de replicação às formas padrões SOP e POS, vários literais podem ser eliminados.

- A quantidade de literais eliminados depende do número de termos combinados e da configuração de literais em cada termo.

- Eliminação de 1 literal: exceto 1 literal, o qual será eliminado, todos os demais literais são idênticos em uma combinação de 2 termos normais.

- Eliminação de 2 literais: exceto 2 literais, os quais serão eliminados, todos os demais literais são idênticos em uma combinação de 4 termos normais.

- Eliminação de 3 literais: exceto 3 literais, os quais serão eliminados, todos os demais literais são idênticos em uma combinação de 8 termos normais.

- Eliminação de N literais: exceto N literais, os quais serão eliminados, todos os demais literais são idênticos em uma combinação de 2^N termos normais.

- A título de exemplo, a Tabela 6.1 apresenta a tabela verdade para funções de 3 variáveis.

- Para tais funções, as Figuras 6.1 e 6.2 ilustram as possibilidades de eliminação de 1 e 2 literais em combinações de 2 e 4 mintermos, respectivamente.

- Por sua vez, as Figuras 6.3 e 6.4 ilustram as possibilidades de eliminação de 1 e 2 literais em combinações de 2 e 4 maxtermos, respectivamente.

Linha	A	B	C	$F(A,B,C)$
0	0	0	0	F_0
1	0	0	1	F_1
2	0	1	0	F_2
3	0	1	1	F_3
4	1	0	0	F_4
5	1	0	1	F_5
6	1	1	0	F_6
7	1	1	1	F_7

Tabela 6.1: Tabela verdade para funções de 3 variáveis.

$$
\begin{aligned}
m_0 + m_1 &= (\overline{A} \cdot \overline{B} \cdot \overline{C}) + (\overline{A} \cdot \overline{B} \cdot C) = (\overline{A} \cdot \overline{B}) \\
m_2 + m_3 &= (\overline{A} \cdot B \cdot \overline{C}) + (\overline{A} \cdot B \cdot C) = (\overline{A} \cdot B) \\
m_4 + m_5 &= (A \cdot \overline{B} \cdot \overline{C}) + (A \cdot \overline{B} \cdot C) = (A \cdot \overline{B}) \\
m_6 + m_7 &= (A \cdot B \cdot \overline{C}) + (A \cdot B \cdot C) = (A \cdot B)
\end{aligned}
$$

$$
\begin{aligned}
m_0 + m_2 &= (\overline{A} \cdot \overline{B} \cdot \overline{C}) + (\overline{A} \cdot B \cdot \overline{C}) = (\overline{A} \cdot \overline{C}) \\
m_1 + m_3 &= (\overline{A} \cdot \overline{B} \cdot C) + (\overline{A} \cdot B \cdot C) = (\overline{A} \cdot C) \\
m_4 + m_6 &= (A \cdot \overline{B} \cdot \overline{C}) + (A \cdot B \cdot \overline{C}) = (A \cdot \overline{C}) \\
m_5 + m_7 &= (A \cdot \overline{B} \cdot C) + (A \cdot B \cdot C) = (A \cdot C)
\end{aligned}
$$

$$
\begin{aligned}
m_0 + m_4 &= (\overline{A} \cdot \overline{B} \cdot \overline{C}) + (A \cdot \overline{B} \cdot \overline{C}) = (\overline{B} \cdot \overline{C}) \\
m_1 + m_5 &= (\overline{A} \cdot \overline{B} \cdot C) + (A \cdot \overline{B} \cdot C) = (\overline{B} \cdot C) \\
m_2 + m_6 &= (\overline{A} \cdot B \cdot \overline{C}) + (A \cdot B \cdot \overline{C}) = (B \cdot \overline{C}) \\
m_3 + m_7 &= (\overline{A} \cdot B \cdot C) + (A \cdot B \cdot C) = (B \cdot C)
\end{aligned}
$$

Figura 6.1: Eliminações de 1 literal em combinações de 2 mintermos.

$$
\begin{aligned}
m_0 + m_1 + m_2 + m_3 &= (\overline{A} \cdot \overline{B} \cdot \overline{C}) + (\overline{A} \cdot \overline{B} \cdot C) + (\overline{A} \cdot B \cdot \overline{C}) + (\overline{A} \cdot B \cdot C) = (\overline{A}) \\
m_4 + m_5 + m_6 + m_7 &= (A \cdot \overline{B} \cdot \overline{C}) + (A \cdot \overline{B} \cdot C) + (A \cdot B \cdot \overline{C}) + (A \cdot B \cdot C) = (A)
\end{aligned}
$$

$$
\begin{aligned}
m_0 + m_1 + m_4 + m_5 &= (\overline{A} \cdot \overline{B} \cdot \overline{C}) + (\overline{A} \cdot \overline{B} \cdot C) + (A \cdot \overline{B} \cdot \overline{C}) + (A \cdot \overline{B} \cdot C) = (\overline{B}) \\
m_2 + m_3 + m_6 + m_7 &= (\overline{A} \cdot B \cdot \overline{C}) + (\overline{A} \cdot B \cdot C) + (A \cdot B \cdot \overline{C}) + (A \cdot B \cdot C) = (B)
\end{aligned}
$$

$$
\begin{aligned}
m_0 + m_2 + m_4 + m_6 &= (\overline{A} \cdot \overline{B} \cdot \overline{C}) + (\overline{A} \cdot B \cdot \overline{C}) + (A \cdot \overline{B} \cdot \overline{C}) + (A \cdot B \cdot \overline{C}) = (\overline{C}) \\
m_1 + m_3 + m_5 + m_7 &= (\overline{A} \cdot B \cdot \overline{C}) + (\overline{A} \cdot \overline{B} \cdot C) + (A \cdot \overline{B} \cdot C) + (A \cdot B \cdot C) = (C)
\end{aligned}
$$

Figura 6.2: Eliminações de 2 literais em combinações 4 de mintermos.

$$
\begin{aligned}
&M_0 \cdot M_1 = (A+B+C)\cdot(A+B+\overline{C}) = (A+B)\\
&M_2 \cdot M_3 = (A+\overline{B}+C)\cdot(A+\overline{B}+\overline{C}) = (A+\overline{B})\\
&M_4 \cdot M_5 = (\overline{A}+B+C)\cdot(\overline{A}+B+\overline{C}) = (\overline{A}+B)\\
&M_6 \cdot M_7 = (\overline{A}+\overline{B}+C)\cdot(\overline{A}+\overline{B}+\overline{C}) = (\overline{A}+\overline{B})\\
&\\
&M_0 \cdot M_2 = (A+B+C)\cdot(A+\overline{B}+C) = (A+C)\\
&M_1 \cdot M_3 = (A+B+\overline{C})\cdot(A+\overline{B}+\overline{C}) = (A+\overline{C})\\
&M_4 \cdot M_6 = (\overline{A}+B+C)\cdot(\overline{A}+\overline{B}+C) = (\overline{A}+C)\\
&M_5 \cdot M_7 = (\overline{A}+B+\overline{C})\cdot(\overline{A}+\overline{B}+\overline{C}) = (\overline{A}+\overline{C})\\
&\\
&M_0 \cdot M_4 = (A+B+C)\cdot(\overline{A}+B+C) = (B+C)\\
&M_1 \cdot M_5 = (A+B+\overline{C})\cdot(\overline{A}+B+\overline{C}) = (B+\overline{C})\\
&M_2 \cdot M_6 = (A+\overline{B}+C)\cdot(\overline{A}+\overline{B}+C) = (\overline{B}+C)\\
&M_3 \cdot M_7 = (A+\overline{B}+\overline{C})\cdot(\overline{A}+\overline{B}+\overline{C}) = (\overline{B}+\overline{C})
\end{aligned}
$$

Figura 6.3: Eliminações de 1 literal em combinações de 2 maxtermos.

$$
\begin{aligned}
&M_0 \cdot M_1 \cdot M_2 \cdot M_3 = (A+B+C)\cdot(A+B+\overline{C})\cdot(A+\overline{B}+C)\cdot(A+\overline{B}+\overline{C}) = (A)\\
&M_4 \cdot M_5 \cdot M_6 \cdot M_7 = (\overline{A}+B+C)\cdot(\overline{A}+B+\overline{C})\cdot(\overline{A}+\overline{B}+C)\cdot(\overline{A}+\overline{B}+\overline{C}) = (\overline{A})\\
&\\
&M_0 \cdot M_1 \cdot M_4 \cdot M_5 = (A+B+C)\cdot(A+B+\overline{C})\cdot(\overline{A}+B+C)\cdot(\overline{A}+B+\overline{C}) = (B)\\
&M_2 \cdot M_3 \cdot M_6 \cdot M_7 = (A+\overline{B}+C)\cdot(A+\overline{B}+\overline{C})\cdot(\overline{A}+\overline{B}+C)\cdot(\overline{A}+\overline{B}+\overline{C}) = (\overline{B})\\
&\\
&M_0 \cdot M_2 \cdot M_4 \cdot M_6 = (A+B+C)\cdot(A+\overline{B}+C)\cdot(\overline{A}+B+C)\cdot(\overline{A}+\overline{B}+C) = (C)\\
&M_1 \cdot M_3 \cdot M_5 \cdot M_7 = (A+B+\overline{C})\cdot(A+\overline{B}+\overline{C})\cdot(\overline{A}+B+\overline{C})\cdot(\overline{A}+\overline{B}+\overline{C}) = (\overline{C})
\end{aligned}
$$

Figura 6.4: Eliminações de 2 literais em combinações 4 de maxtermos.

6.4 Expressão mínima

Por dois motivos básicos, serão consideradas apenas as expressões com dois planos de lógica (AND-OR ou OR-AND): i) tempo de resposta e ii) existência de procedimentos sistemáticos de minimização para tais expressões.

Uma expressão com dois planos de lógica será considerada mínima se não existir outra expressão: i) com um número menor de termos e ii) não existir outra expressão com mesmo número de termos, porém com um número menor de literais.

Assim sendo, embora a expressão da Equação (6.6) seja menor do que aquela da Equação (6.7), esta última será considerada a expressão mínima para o processo de minimização descrito a seguir.

$$\begin{aligned} F(A,B,C,D) &= \sum m(5,6,9,10,13,14) \\ &= (A+B)\cdot(C\cdot\overline{D}+\overline{C}\cdot D) \end{aligned} \tag{6.6}$$

$$\begin{aligned} F(A,B,C,D) &= \sum m(5,6,9,10,13,14) \\ &= (A\cdot\overline{C}\cdot D)+(A\cdot C\cdot\overline{D})+(B\cdot\overline{C}\cdot D)+(B\cdot C\cdot\overline{D}) \end{aligned} \tag{6.7}$$

Com base nessa definição de forma mínima, uma determinada função lógica pode ter diversas expressões mínimas equivalentes.

Além disso, não se pode garantir que, para uma dada função, a expressão mínima seja da forma SOP ou da forma POS. É necessário minimizar ambas as formas e escolher a menor delas.

6.5 Processo sistemático de minimização

O ponto de partida do processo é expressar a função lógica na sua forma máxima (ou mais completa ou mais extensa): a forma SOP padrão ou a forma POS padrão.

Em seguida, a operação de aglutinação é aplicada sucessivamente. Sempre que possível, a operação de replicação deve ser empregada, para maximizar a simplificação das expressão.

Quando mais nenhuma aglutinação puder ser efetuada, a expressão restante será, naturalmente, a expressão definida anteriormente como mínima.

No caso da existência de diversas formas mínimas equivalentes, deve-se aplicar algum critério extra para a escolha final.

O processo em questão ainda apresenta um certo grau de subjetividade: a escolha dos termos a serem replicados e a escolha dos termos a serem aglutinados.

A fim de tornar o processo de minimização ainda menos subjetivo, pode-se realizá-lo não diretamente sobre as equações, mas, alternativamente, sobre uma forma pictórica de representação ou através de um procedimento computacional. As alternativas comumente empregadas são o Mapa de Karnaugh e o Algoritmo de Quine-McCluskey.

6.6 Implicantes e implicados

Quando uma função é expressa na forma AND-OR, cada termo produto é denominado de implicante (*implicant*). O nome se deve ao fato de que, caso o termo produto (implicante) assuma o valor lógico 1, isso implicará em um valor lógico 1 para a função.

No caso de uma SOP padrão, os implicantes são os próprios mintermos. Caso contrário, eles são o resultado de simplificações provenientes de combinações de mintermos.

A Equação (6.8) apresenta um exemplo de implicantes. Na primeira expressão, ela apresenta 3 implicantes, que são os mintermos responsáveis pelas 3 combinações lógicas de literais que fazem a função assumir o valor lógico 1. A segunda expressão apresenta 2 implicantes. O primeiro deles, sendo uma combinação de 2 mintermos, representa 2 combinações lógicas de literais capazes de produzir um valor lógico 1 para a função. O segundo deles, sendo um dos mintermos, representa a terceira combinação lógica de literais capaz de produzir um valor lógico 1 para a função.

$$\begin{aligned} F(A,B,C) &= \sum m(0,1,7) \\ &= (\overline{A}\cdot\overline{B}\cdot\overline{C}) + (\overline{A}\cdot\overline{B}\cdot C) + (A\cdot B\cdot C) \\ &= (\overline{A}\cdot B) + (A\cdot B\cdot C) \end{aligned} \tag{6.8}$$

Quando uma função é expressa na forma OR-AND, cada termo soma é denominado de implicado (*implicate*). O nome se deve ao fato de que, caso o termo soma (implicado) assuma o valor lógico 0, isso implicará em um valor lógico 0 para a função.

No caso de um POS padrão, os implicados são os próprios maxtermos. Caso contrário, eles são o resultado de simplificações provenientes de combinações de maxtermos.

A Equação (6.9) apresenta um exemplo de implicados. Na primeira expressão, ela apresenta 3 implicados, que são os maxtermos responsáveis pelas 3 combinações lógicas de literais que fazem a função assumir o valor lógico 0. A segunda expressão apresenta 2 implicados. O primeiro deles, sendo uma combinação de 2 maxtermos, representa 2 combinações lógicas de literais capazes de produzir um valor lógico 0 para a função. O segundo deles, sendo um dos maxtermos, representa a terceira combinação lógica de literais capaz de produzir um valor lógico 0 para a função.

$$\begin{aligned} F(A,B,C) &= \prod M(2,3,5) \\ &= (A+\overline{B}+C)\cdot(A+\overline{B}+\overline{C})\cdot(\overline{A}+B+\overline{C}) \\ &= (A+\overline{B})\cdot(\overline{A}+B+\overline{C}) \end{aligned} \tag{6.9}$$

6.7 Exercícios propostos

1. Para os exercícios listados abaixo, considerar as equações booleanas apresentadas em seguida.

   (a) Algebricamente, obter a forma SOP padrão da equação fornecida.

   (b) Algebricamente, obter a forma SOP mínima, a partir da SOP padrão.

   (c) Algebricamente, obter a forma POS padrão da equação fornecida.

   (d) Algebricamente, obter a forma POS mínima, a partir da POS padrão.

   (e) Apresentar a expressão mínima para função.

   Equações boolenas:

   i. $F(A,B,C) = \left\{B \cdot \left[\left(\overline{A} \cdot \overline{C}\right) + (A \cdot C)\right]\right\} + \overline{\left\{B + \left[\left(\overline{A} + \overline{C}\right) \cdot (A + C)\right]\right\}}$.

   ii. $F(A,B,C) = \overline{\left\{\left[\overline{(A+B)} + C\right] \cdot \left[\overline{A} + \overline{\left(B + \overline{C}\right)}\right]\right\}}$.

Capítulo 7

Mapa de Karnaugh

7.1 Introdução

- O mapa de Karnaugh (mapa-K) é mais uma das possíveis expressões de uma função lógica, além de uma equação lógica genérica, uma equação booleana genérica, uma forma do grupo SOP, uma forma do grupo POS, uma forma padrão do grupo SOP, uma forma padrão do grupo POS, uma lista de mintermos, uma lista de maxtermos e uma tabela verdade.

- Além de representar uma simples expressão para uma função lógica, o mapa-K pode ser usado como ferramenta para a minimização da equação que a define.

- Ele pode ser interpretado como uma tabela verdade rearranjada ou como uma representação análoga ao Diagrama de Venn.

- Para cada linha da tabela verdade de uma função lógica booleana é associada uma posição no mapa.

- Uma vez que cada linha da tabela verdade é associada a um mintermo ou a um maxtermo, a cada um deles também é associada uma posição do mapa.

- A fim de que o mapa seja empregado no processo de simplificação de funções lógicas booleanas, ele deve ser arranjado da seguinte forma:

  - Deve existir uma localização única no mapa para cada combinação das variáveis das quais a função lógica é dependente.
  - As localizações devem ser arranjadas de tal forma que grupos de mintermos/maxtermos possam ser facilmente combinados em formas reduzidas.

- Devido a uma limitação prática, são construídos mapas-K para funções lógicas de até 6 variáveis.

- Para funções lógicas com um número superior a 6 variáveis, pode-se utilizar um algoritmo de minimização, tal como o algoritmo tabular de Quine-McCluskey.

7.2 Construção do mapa-K

7.2.1 Funções de 1 variável

Linha	A	$F(A)$
0	0	F_0
1	1	F_1

Tabela 7.1: Tabela verdade para funções de 1 variável.

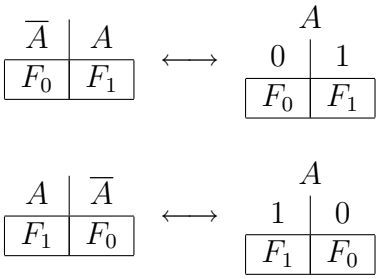


Figura 7.1: Exemplos de mapas de Karnaugh para funções de 1 variável.

7.2.2 Funções de 2 variáveis

Linha	A	B	$F(A,B)$
0	0	0	F_0
1	0	1	F_1
2	1	0	F_2
3	1	1	F_3

Tabela 7.2: Tabela verdade para funções de 2 variáveis.

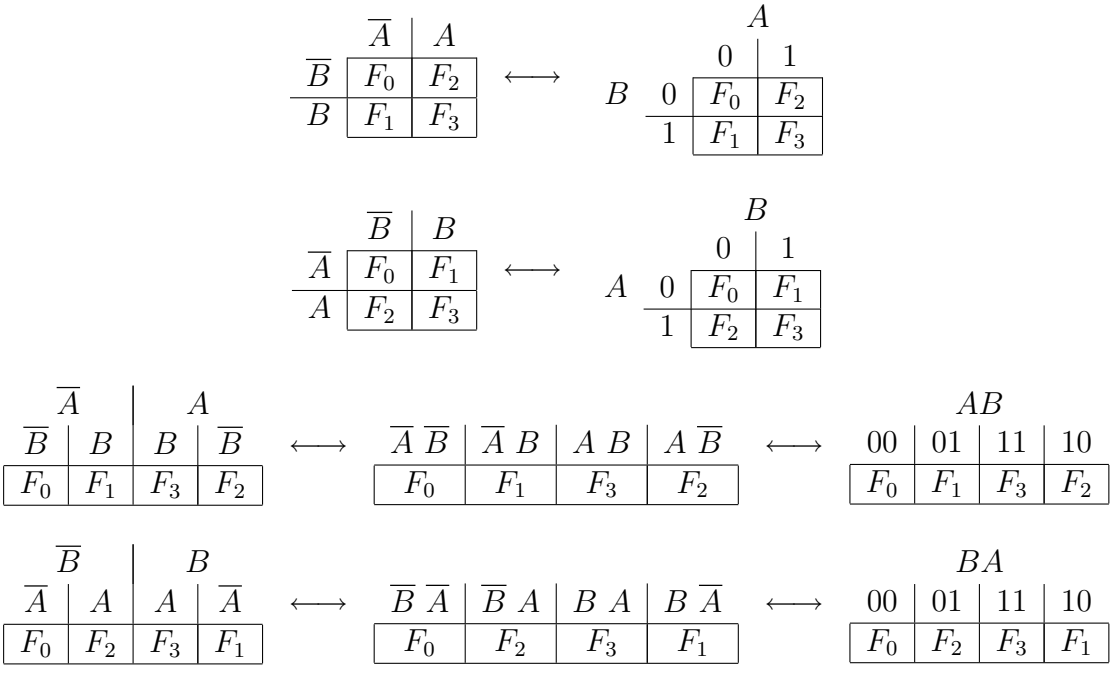


Figura 7.2: Exemplos de mapas de Karnaugh para funções de 2 variáveis.

7.2.3 Funções de 3 variáveis

Linha	A	B	C	$F(A,B,C)$
0	0	0	0	F_0
1	0	0	1	F_1
2	0	1	0	F_2
3	0	1	1	F_3
4	1	0	0	F_4
5	1	0	1	F_5
6	1	1	0	F_6
7	1	1	1	F_7

Tabela 7.3: Tabela verdade para funções de 3 variáveis.

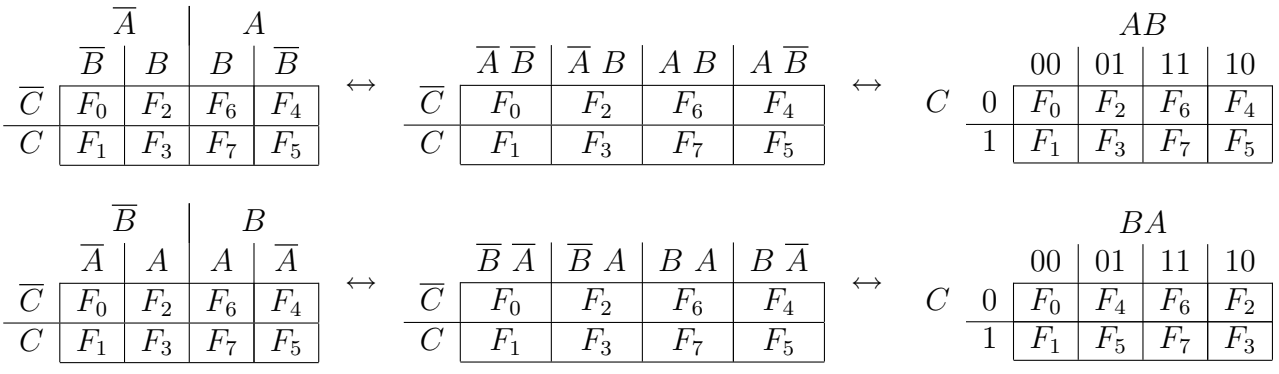


Figura 7.3: Exemplos de mapas de Karnaugh para funções de 3 variáveis.

7.2.4 Funções de 4 variáveis

Linha	A	B	C	D	$F(A,B,C,D)$
0	0	0	0	0	F_0
1	0	0	0	1	F_1
2	0	0	1	0	F_2
3	0	0	1	1	F_3
4	0	1	0	0	F_4
5	0	1	0	1	F_5
6	0	1	1	0	F_6
7	0	1	1	1	F_7
8	1	0	0	0	F_8
9	1	0	0	1	F_9
10	1	0	1	0	F_{10}
11	1	0	1	1	F_{11}
12	1	1	0	0	F_{12}
13	1	1	0	1	F_{13}
14	1	1	1	0	F_{14}
15	1	1	1	1	F_{15}

Tabela 7.4: Tabela verdade para funções de 4 variáveis.

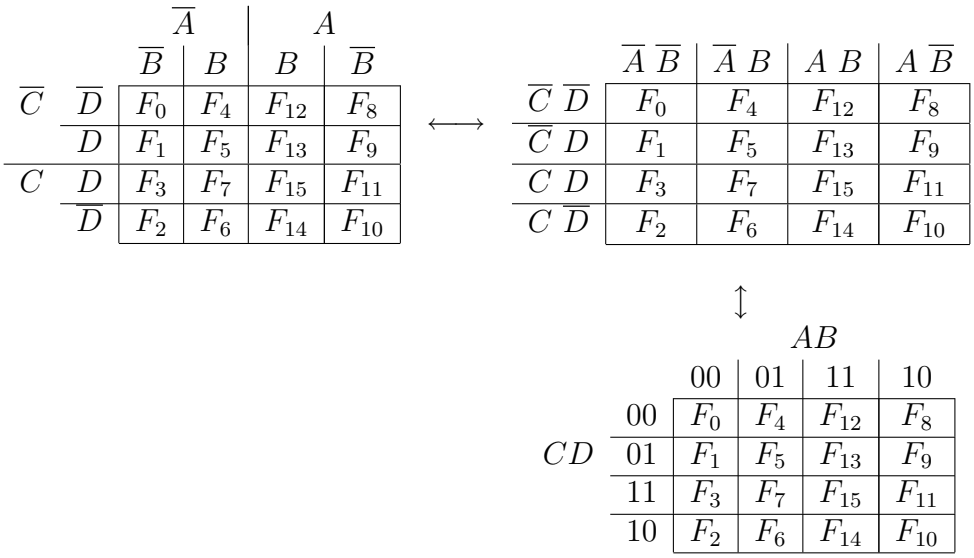


Figura 7.4: Exemplos de mapas de Karnaugh para funções de 4 variáveis.

7.3 Preenchimento do mapa-K

- Cada uma das localizações do mapa é associada a cada uma das combinações das variáveis das quais a função é dependente.
- Isso equivale a dizer que cada uma das localizações do mapa é associada a uma linha da tabela verdade da função.
- Logo, cada uma das localizações do mapa é preenchida com o respectivo valor lógico da função (0 ou 1).
- Para montar e simplificar uma forma SOP, deve-se manter os valores lógicos 1 (mintermos) no mapa e ignorar os valores lógicos 0 (maxtermos).
- Para montar e simplificar uma forma POS, deve-se manter os valores lógicos 0 (maxtermos) no mapa e ignorar os valores lógicos 1 (mintermos).

7.4 Mapa-K como forma de expressão de função booleana

Uma função de variáveis booleanas pode ser expressa por uma equação genérica, por uma forma do grupo SOP, por uma forma do grupo POS, por uma forma padrão do grupo SOP, por uma forma padrão do grupo POS por uma lista de mintermos, por uma lista de maxtermos e por uma tabela verdade.

Além de ser usado como ferramenta de minimização, o mapa-K pode ser visto como mais uma alternativa de representação para funções booleanas.

As transformações entre: i) uma equação genérica, ii) uma forma dos grupos SOP ou POS e iii) uma forma padrão dos grupos SOP ou POS, envolvem manipulação algébrica das equações.

Por outro lado, as transformações realizadas entre uma lista de mintermos ou maxtermos, uma tabela verdade, um mapa-K e as demais representações, envolvem catalogação direta.

Portanto, partindo-se de uma dada forma de representação, pode-se facilmente obter todas as demais, independentemente do tipo de mapeamento utilizado.

Um exemplo de tais relacionamentos pode ser obtido a partir da função dada por

$$F(A,B,C) = \overline{\left(\overline{A} + \overline{(B+C)}\right)} \ . \tag{7.1}$$

Após alguma manipulação algébrica, a Equação (7.1) pode gerar a forma POS

$$F(A,B,C) = (A) \cdot (B+C) \tag{7.2}$$

e a forma SOP

$$F(A,B,C) = (A \cdot B) + (A \cdot C) \ . \tag{7.3}$$

Expandindo-se os termos das Equações (7.2) e (7.3), obtém-se, respectivamente, a forma padrão POS

$$\begin{aligned} F(A,B,C) &= (A+B+C) \cdot (A+B+\overline{C}) \cdot (A+\overline{B}+C) \cdot (A+\overline{B}+\overline{C}) \cdot (\overline{A}+B+C) \\ &= \prod M(0,1,2,3,4) \end{aligned} \tag{7.4}$$

e a forma padrão SOP

$$\begin{aligned} F(A,B,C) &= (A \cdot \overline{B} \cdot C) + (A \cdot B \cdot \overline{C}) + (A \cdot B \cdot C) \\ &= \sum m(5,6,7) \ . \end{aligned} \tag{7.5}$$

Por sua vez, a tabela verdade referente à Equação (7.1) é apresentada na Tabela 7.5.

Linha	A	B	C	$F(A,B,C)$
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

Tabela 7.5: Tabela verdade relativa à Equação (7.1).

Finalmente, o mapa-K da função é mostrado na Figura 7.5.

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

Figura 7.5: Mapa de Karnaugh relativo à Equação (7.1).

7.5 Mapa-K na simplificação de expressões booleanas

7.5.1 Adjacência lógica, aglutinação e replicação

A simplificação algébrica de expressões booleanas baseia-se na utilização de duas operações: a aglutinação e a replicação.

Se dois termos diferem de apenas um literal (A e $\overline{A}$), a aplicação da aglutinação permite simplificá-los em um único termo, sem o literal em questão. Tais termos são ditos logicamente adjacentes. Isso pode ser exemplificado por

$$\begin{aligned} F(A,B,C) &= (\overline{A}\cdot\overline{B}\cdot\overline{C}) + (A\cdot\overline{B}\cdot\overline{C}) + (\overline{A}\cdot B\cdot C) + (A\cdot B\cdot C) \\ &= (\overline{B}\cdot\overline{C}) + (B\cdot C)\ . \end{aligned} \tag{7.6}$$

Por sua vez, a replicação permite que um mesmo termo seja utilizado em simplificações envolvendo diversos outros termos. Um exemplo de replicação é dado por

$$\begin{aligned} F(A,B,C) &= (\overline{A}\cdot\overline{B}\cdot\overline{C}) + (\overline{A}\cdot B\cdot\overline{C}) + (A\cdot\overline{B}\cdot\overline{C}) \\ &= (\overline{A}\cdot\overline{B}\cdot\overline{C}) + (\overline{A}\cdot B\cdot\overline{C}) + (A\cdot\overline{B}\cdot\overline{C}) + (\overline{A}\cdot\overline{B}\cdot\overline{C}) \\ &= (\overline{A}\cdot\overline{C}) + (\overline{B}\cdot\overline{C})\ . \end{aligned} \tag{7.7}$$

Os mapas de Karnaugh são construídos de tal forma que as adjacências geométricas do mapa são equivalentes às adjacências lógicas dos termos das equações. Portanto, a combinação algébrica dos termos de uma equação é equivalente à combinação de termos adjacentes do mapa. Assim sendo, a equação pode ser simplificada através da leitura direta da informação do mapa. O mapa da Figura 7.6 exemplifica a Equação (7.6), onde são realizadas as combinações m_0+m_4 e m_3+m_7.

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

Figura 7.6: Mapa de Karnaugh relativo à Equação (7.6).

No mapa, a replicação é interpretada como a combinação de um termo com os demais geometricamente adjacentes. O mapa da Figura 7.7 exemplifica a Equação (7.7), onde o mintermo m_0 é replicado para as combinações m_0+m_2 e m_0+m_4.

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

Figura 7.7: Mapa de Karnaugh relativo à Equação (7.7).

7.5.2 Seleção sistemática de termos (implicantes ou implicados)

Para as equações que possuem uma forma mínima única, duas definições são de grande auxílio na escolha de termos (implicantes ou implicados) a serem agrupados para simplificação: termo essencial e termo primo.

Quando um termo original é coberto por um único agrupamento possível, o termo resultante do agrupamento é denominado de termo essencial. Isso indica que ele deve ser incluído na expressão mínima que expressa a função desejada.

Um termo que não tenha sido coberto por qualquer agrupamento anterior deve ser incluído em um agrupamento máximo, o qual será denominado de termo primo.

Pode-se concluir que todo termo essencial deve ser primo (máximo), mas nem todo termo primo (máximo) é essencial.

Assim, uma forma sistemática de escolha de termos é:

S1 - Identificar todas as possibilidades de agrupamento, através dos maiores grupos possíveis.

S2 - Marcar todos os termos originais cobertos por apenas 1 agrupamento. Tais agrupamentos formam os termos essenciais.

S3 - Listar todos os termos essenciais.

S4 - Usar os maiores agrupamentos possíveis (termos primos) para cobrir os termos originais não cobertos pelos termos essenciais.

S5 - Listar apenas tais termos primos.

S6 - Montar a expressão mínima, a partir das duas listas.

Dada uma função, e suas formas SOP e POS, nada se pode garantir em relação a qual das duas conduzirá à expressão mais simples. Assim, é necessário encontrar a forma mínima de ambas e decidir qual delas é a mais simples.

7.5.3 Mapa-K de funções com múltiplos mínimos e mapa cíclico

Algumas equações booleanas não possuem uma forma mínima única. Isso acontece porque, em um conjunto de termos da expressão, cada um deles é coberto por mais de um agrupamento de termos logicamente equivalentes. Assim sendo, não é possível selecionar um conjunto único de termos essenciais e/ou primos.

Em tais casos, o que se deve fazer é avaliar as possíveis soluções e escolher a de menor custo.

Caso ainda existam opções logicamente equivalentes, todas de mesmo custo, deve-se adotar algum critério extra de escolha.

A Figura 7.8 apresenta um mapa com múltiplas formas mínimas, envolvendo o termo m_2, que possui duas soluções de mesmo custo: i) $(m_0 + m_4), (m_3 + m_7), (m_0 + m_2)$ e ii) $(m_0 + m_4), (m_3 + m_7), (m_3 + m_2)$.

Em alguns casos particulares, todos os termos de um subconjunto dos termos da função são cobertos por mais de um agrupamento, todos de mesmo custo. Tal subconjunto de termos forma um ciclo. Mapas de funções com tal característica são denominados de mapas cíclicos.

Nesses casos, deve-se adotar algum critério extra de escolha para quebrar o ciclo.

A Figura 7.9 apresenta um mapa com ciclo, que possui duas soluções de mesmo custo: i) $(m_0 + m_2), (m_3 + m_7), (m_4 + m_5)$ e ii) $(m_0 + m_4), (m_2 + m_3), (m_5 + m_7)$.

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

Figura 7.8: Mapa de Karnaugh com múltiplas formas mínimas.

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

Figura 7.9: Mapa de Karnaugh com ciclo.

7.5.4 Indeterminações: *don't-care* e *can't-happen*

Em alguns problemas, as funções booleanas podem não ser completamente especificadas. Nesses casos, duas situações podem ocorrer. Na primeira delas, para uma dada combinação de valores dos literais, o valor da função não é relevante (*don't-care*). Por outro lado, pode acontecer que uma determinada combinação de literais nunca ocorra (*can't-happen*). Em ambas as situações, pode-se especificar livremente qualquer um dos valores lógicos para a função. Na realidade, atribui-se um valor lógico indeterminado X, caracterizando-se o aspecto indeterminado da sua especificação.

Os valores indeterminados podem ser utilizados no processo de simplificação de formas padrões contendo mintermos ou maxtermos.

A Tabela 7.6 exemplifica uma função incompletamente especificada, a qual também pode ser expressa por

$$F(A,B,C) = \sum m(0,3,4) + \sum d(2,7) = \prod M(1,5,6) \cdot \prod d(2,7)\ . \tag{7.8}$$

As Figuras 7.10–7.12 ilustram os mapas de Karnaugh da função, de seus mintermos e de seus maxtermos, respectivamente.

Linha	A	B	C	$F(A,B,C)$
0	0	0	0	1
1	0	0	1	0
2	0	1	0	X
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	X

Tabela 7.6: Tabela verdade de função incompletamente especificada.

		AB			
		00	01	11	10
C	0	1	X	0	1
	1	0	1	X	0

Figura 7.10: Mapa de Karnaugh da Tabela 7.6.

		AB			
		00	01	11	10
C	0	1	X		1
	1		1	X	

Figura 7.11: Mapa de Karnaugh dos mintermos da Tabela 7.6.

		AB			
		00	01	11	10
C	0		X	0	
	1	0		X	0

Figura 7.12: Mapa de Karnaugh dos maxtermos da Tabela 7.6.

Da configuração de mintermos apresentada no mapa da Figura 7.11, pode-se escrever que

$$\begin{aligned} F(A,B,C) &= \sum m(0,3,4) + \sum d(2,7) \\ &= (m_0 + m_4) + (m_3 + d_2) \;=\; (\overline{B} \cdot \overline{C}) + (\overline{A} \cdot B) \\ &= (m_0 + m_4) + (m_3 + d_7) \;=\; (\overline{B} \cdot \overline{C}) + (B \cdot C)\ . \end{aligned} \tag{7.9}$$

Da configuração de maxtermos apresentada no mapa da Figura 7.12, pode-se escrever que

$$\begin{aligned} F(A,B,C) &= \prod M(1,5,6) \cdot \prod d(2,7) \\ &= (M_1 \cdot M_5) \cdot (M_6 \cdot d_2) \;=\; (B + \overline{C}) \cdot (\overline{B} + C) \\ &= (M_1 \cdot M_5) \cdot (M_6 \cdot d_7) \;=\; (B + \overline{C}) \cdot (\overline{A} + \overline{B})\ . \end{aligned} \tag{7.10}$$

As Equações 7.9 e 7.10 mostram que os valores lógicos indeterminados podem ser usados, ou não, no processo de simplificação. Elas ilustram ainda o papel relevante dos valores indeterminados na simplificação de funções booleanas.

Deve ser ressaltado que, uma vez escolhidos como "0" ou como "1", os valores indeterminados "X", bem como a função original, perdem a sua característica de indeterminação na expressão mínima. Assim sendo, a função minimizada final passa a ser completamente especificada.

7.6 Exercícios propostos

1. Para os exercícios listados abaixo, considerar as equações booleanas apresentadas em seguida.

   (a) Obter a forma SOP padrão da equação fornecida.
   (b) Obter a forma SOP mínima, utilizando o mapa de Karnaugh correspondente.
   (c) Obter a forma POS padrão da equação fornecida.
   (d) Obter a forma POS mínima, utilizando o mapa de Karnaugh correspondente.
   (e) Apresentar a expressão mínima para função.

   Equações boolenas:

   i. $F(A,B,C) = \left\{B \cdot \left[\left(\overline{A} \cdot \overline{C}\right) + (A \cdot C)\right]\right\} + \overline{\left\{B + \left[\left(\overline{A} + \overline{C}\right) \cdot (A + C)\right]\right\}}$.
   ii. $F(A,B,C) = \overline{\left\{\left[\overline{(A+B)} + C\right] \cdot \left[\overline{A} + \overline{\left(B + \overline{C}\right)}\right]\right\}}$.

2. Para os exercícios listados abaixo, considerar as equações booleanas apresentadas em seguida.

   (a) Obter a forma SOP mínima, utilizando o mapa de Karnaugh correspondente.
   (b) Obter a forma POS mínima, utilizando o mapa de Karnaugh correspondente.
   (c) Apresentar a expressão mínima para função.

   Equações boolenas:

   i. $F(A,B,C) = \sum m(0,2,3,5,7)$
   ii. $F(A,B,C) = \sum m(0,1,3,4,5)$
   iii. $F(A,B,C) = \sum m(0,1,2,4,6,7)$
   iv. $F(A,B,C,D) = \sum m(4,5,11,13,15)$
   v. $F(A,B,C,D) = \sum m(0,1,5,6,7,14)$
   vi. $F(A,B,C,D) = \sum m(0,1,2,6,7,8,9,10,14)$
   vii. $F(A,B,C,D) = \sum m(0,1,2,5,6,7,8,9,10,14)$
   viii. $F(A,B,C,D) = \sum m(0,1,2,6,7,8,9,10,14,15)$
   ix. $F(A,B,C,D) = \sum m(0,1,2,5,6,7,8,9,10,14,15)$
   x. $F(A,B,C,D) = \sum m(0,1,2,5,6,7,8,9,10,13,14,15)$
   xi. $F(A,B,C,D) = \sum m(0,2,6,7,8,9,10,12,13)$
   xii. $F(A,B,C,D) = \sum m(0,2,6,7,8,9,10,12,13,15)$
   xiii. $F(A,B,C,D) = \sum m(0,1,2,3,5,7,15)$
   xiv. $F(A,B,C,D) = \sum m(0,1,2,3,5,7,14,15)$
   xv. $F(A,B,C,D) = \sum m(0,1,2,3,5,7,11,15)$
   xvi. $F(A,B,C,D) = \sum m(0,1,2,3,5,7,13,15)$
   xvii. $F(A,B,C,D) = \sum m(0,2,5,6,8,10,13)$
   xviii. $F(A,B,C,D) = \sum m(0,2,5,7,8,10,11,13,15)$
   xix. $F(A,B,C,D) = \sum m(1,5,6,7,11,12,13,15)$
   xx. $F(A,B,C,D) = \sum m(2,3,4,5,6,7,10,12,13,15)$
   xxi. $F(A,B,C,D) = \sum m(1,2,4,6,7,9,11,12,13,14,15)$

Capítulo 8

Sistemas de numeração

8.1 Introdução

- Sistema numérico (*number system*) × sistema de numeração (*numeral system*).
- Sistemas numéricos: $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$, etc..
- Sistemas de numeração: grupos de numerais (símbolos) que representam quantidades.
- Máquinas digitais possuem capacidade de armazenamento finito. Um registro só pode armazenar uma quantidade finita de elementos básicos de informação e a máquina só pode armazenar uma quantidade finita de registros. Portanto, toda quantidade armazenada será uma aproximação da quantidade original. O sistema de numeração utilizado pela máquina tem influência direta na qualidade dessa aproximação.
- Da mesma forma, a eficiência de uma determinada implementação para as operações aritméticas básicas (adição, subtração, multiplicação e divisão) também é influenciada pelo sistema de numeração utilizado pela máquina. Deve ser ressaltado que a eficiência é, geralmente, medida em relação ao tempo necessário para a realização da operação, à quantidade de elementos constituintes utilizados, aos tipos de tais elementos e ao consumo de energia.
- Assim, através da escolha adequada entre as diversas alternativas matemáticas para a representação de quantidades, bem como da sua implementação (máquina e linguagem de programação), procura-se reduzir o erro das aproximações e/ou tornar as operações aritméticas mais eficientes.
- Sistemas comumente usados em máquinas digitais:
  - Posicional.
  - Resíduos (ou resto).
  - Racional.
  - Logarítmico.

- Sistemas de numeração posicional
  - Sistema vetorial posicional.
  - É definido um conjunto básico de dígitos ou símbolos $\boldsymbol{S} = \{s_1, s_2, \cdots, s_M\}$.
  - Os números x são representados por grupos de dígitos (vetores) pertencentes a $\boldsymbol{S}$: $\boldsymbol{x} = [d_N, \cdots, d_2, d_1]_S$, onde $d_i \in S$.
  - A posição de cada dígito no vetor tem significado.
  - A cada posição i é associado um peso numérico w_i, o qual é multiplicado pelo dígito d_i correspondente: $\boldsymbol{w} = [w_N, \cdots, w_2, w_1]$.
  - Os dígitos d_i representam números inteiros, podendo ser positivos e/ou negativos.
  - Os pesos podem ser os mais diversos possíveis.

- Sistemas de numeração de resíduos (ou restos)
  - Sistema vetorial não posicional.
  - É definido um vetor de elementos primos entre si dois a dois: $\boldsymbol{m} = [m_1, m_2, \cdots, m_N]$.
  - São calculados os resíduos (restos) r_i da divisão de um número inteiro x por cada elemento m_i.
  - Os números x são representados por um vetor contendo os resíduos: $\boldsymbol{x} = [r_1, r_2, \cdots, r_N]_m$.
  - Nas operações aritméticas, os resíduos podem ser tratados independentemente, acelerando o processo de cálculo.

- Sistemas de numeração racional
  - Representação de números através de frações.
  - Numerador e denominador da fração são representados por números inteiros.
  - As operações aritméticas são realizadas sem erro, mesmo em uma máquina com precisão finita.

- Sistemas de numeração logarítmico
  - Um número real $\mu > 1$ é definido como base.
  - É gerado um conjunto de números reais $\boldsymbol{L_\mu} = \{x \mid |x| = \mu^i, i \in \mathbb{Z}\}\ U\ \{0\}$.
  - É objetivada uma melhoria de precisão na representação dos números, conseguida através de arredondamento geométrico.

8.2 Sistema de numeração posicional convencional

Nos itens que se seguem, são abordados diversos aspectos do sistema de numeração posicional convencional:

- Representação de números inteiros e fracionários.
- Representação de números positivos e negativos.
- Tabelas de operações básicas entre dígitos.
- Escalamento por potência inteira da base.
- Conversão entre bases.
- Bases mais comuns em circuitos digitais.

8.2.1 Representação de números inteiros positivos

Para representar quantidades numéricas inteiras, positivas e ordenadas, o sistema de numeração posicional convencional utiliza um conjunto positivo e ordenado de símbolos simples (dígitos) $d_i \in \boldsymbol{S} = \{s_1, s_2, \cdots, s_M\} = \{0, 1, 2, \cdots, (b-1)\}$, juntamente com uma técnica de poderação ou escala. O número de elementos de $\boldsymbol{S}$, $M = b$, é denominado **base** ou **radical** (*radix*) do sistema de numeração. Os pesos ou fatores de escala utilizados são potências inteiras da base $w_i \in \boldsymbol{W} = \{w_1, w_2, w_3, \cdots\} = \{b^0, b^1, b^2, \cdots\}$.

Uma visão geométrica modular do processo de representação pode ser encontrada nas Figuras 8.1 – 8.4, para $b = 3$. Para representar quantidades $q < b$, é utilizado apenas um dos elementos de $\boldsymbol{S}$, como na Figura 8.1. Para quantidades $q \geq b$, como não existem símbolos disponíveis, repetem-se os elementos de $\boldsymbol{S}$, em módulos de comprimento b, como exemplificado na Figura 8.2. Porém, isso gera ambigüidade na representação, a qual é resolvida através da combinação de símbolos, como ilustrado na Figura 8.3. A cada módulo de b símbolos, são justapostos os elementos de $\boldsymbol{S}$, formando um novo nível de representação. Essa técnica é aplicada, sucessivamente, cada vez que o número de possibilidades de representação se esgota e uma nova ambigüidade é gerada pela repetição de símbolos, como na Figura 8.4.

Em cada nível da representação existe um módulo formado pelos símbolos de $\boldsymbol{S}$. Em cada nível, os módulos são versões escaladas daqueles presentes nos demais níveis. Os fatores de escala são as potências inteiras da base $\boldsymbol{W} = \{b^0, b^1, b^2, \cdots\}$. As mudanças de símbolos, dentro de cada nível, são reguladas pelo fator de escala do nível. Dessa forma, dentro de cada nível $L = 0, 1, 2, ..., (N-1)$, ocorre uma mudança de símbolos a cada b^L unidades da quantidade representada.

0	1	2

Figura 8.1: Representação de quantidades $q < b$, para $b = 3$.

0	1	2	0	1	2	0	1	2

Figura 8.2: Representação de quantidades $q \geq b$, para $b = 3$, com ambigüidade.

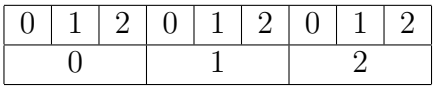


Figura 8.3: Representação de quantidades $q \geq b$, para $b = 3$, com eliminação da ambigüidade através da justaposição dos dígitos.

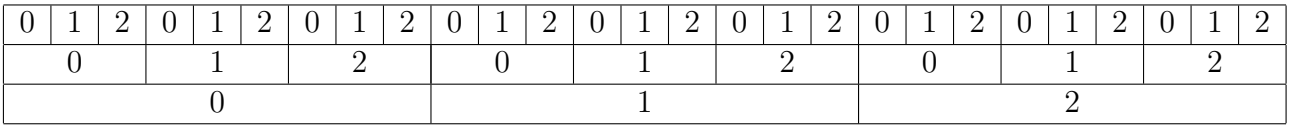


Figura 8.4: Uso repetido da técnica de justaposição de dígitos para representação de quantidades $q \geq b$, para $b = 3$, sem ambigüidade.

Algebricamente, a idéia geométrica modular de uma combinação de níveis pode ser expressa por uma soma de níveis de valores, onde o valor numérico de cada nível k é expresso por um dígito $d_k \in \boldsymbol{S}$, ponderado por um fator $w_k \in \boldsymbol{W}$, conforme a Equação (8.1). A notação pode ser simplificada através da justaposição dos dígitos, acompanhada da especificação da base, como ilustrado na Equação (8.2). Nos casos onde se opera sempre com a mesma base, a sua indicação pode ser omitida, como na Equação (8.3).

$$(q_I)_b = (d_N \times b^N) + \cdots + (d_2 \times b^2) + (d_1 \times b^1) + (d_0 \times b^0) = \sum_{k=0}^{N} d_k b^k \ . \tag{8.1}$$

$$(q_I)_b = \sum_{k=0}^{N} d_k b^k = [d_N \cdots d_2 d_1 d_0]_b \ . \tag{8.2}$$

$$q_I = \sum_{k=0}^{N} d_k b^k = [d_N \cdots d_2 d_1 d_0] \ . \tag{8.3}$$

8.2.2 Representação de números fracionários positivos

Para representar quantidades numéricas puramente fracionárias, positivas e ordenadas, o sistema de numeração posicional convencional utiliza o mesmo mecanismo empregado com números inteiros. Nesse caso, os pesos w_i, usados para ponderar os dígitos d_i, são potências inteiras e negativas da base b, conforme a Equação (8.4).

$$(q_F)_b = (d_{-1} \times b^{-1}) + (d_{-2} \times b^{-2}) + \cdots + (d_{-N} \times b^{-N}) = \sum_{k=-1}^{-N} d_k b^k = [d_{-1} d_{-2} \cdots d_{-N}]_b \ . \tag{8.4}$$

Na representação simplificada por um vetor de dígitos, emprega-se um símbolo extra para diferenciar as representações de números fracionários, puramente inteiros e puramente fracionários. Normalmente é utilizado um ponto ou uma vírgula, como é ilustrado na Equação (8.5), para números puramente fracionários, e na Equação (8.6), para números fracionários.

$$(q_F)_b = \sum_{k=-1}^{-N} d_k b^k = [\cdot d_{-1} d_{-2} \cdots d_{-N}]_b = [0 \cdot d_{-1} d_{-2} \cdots d_{-N}]_b \ . \tag{8.5}$$

$$(q)_b = (q_I)_b + (q_F)_b = \sum_{k=N_I}^{-N_F} d_k b^k = [d_{N_I} \cdots d_2 d_1 d_0 \cdot d_{-1} d_{-2} \cdots d_{-N_F}]_b \ . \tag{8.6}$$

Na representação utilizada nos circuitos digitais o símbolo extra não é utilizado, uma vez que o conhecimento de quantos dígitos são empregados para as partes inteira e fracionária transformam-no em uma informação redundante e, portanto, dispensável.

Por outro lado, para o uso humano, a redundância é útil para facilitar a visualização das partes inteira e fracionária, bem como a sua manipulação.

8.2.3 Representação de números inteiros negativos

- Na representação matemática para uso humano, números positivos e negativos são diferenciados através de um símbolo extra. Números positivos e negativos são precedidos pelos símbolos "+" e "−", respectivamente. Tais símbolos também podem ser interpretados como operadores unários. Assim sendo, a menos que seja necessário resolver alguma ambigüidade, o símbolo "+" é dispensado, uma vez que não realiza qualquer modificação sobre a quantidade original.

- Na representação utilizada nos circuitos digitais, é necessário empregar um dos próprios símbolos utilizados na codificação de quantidades para diferenciar quantidades positivas e negativas.

- Representação numérica
  - Base b.
  - Dígitos $d_i \in S = \{0, 1, 2, \cdots, (b-1)\}$.
  - N dígitos.

- Significado dos N dígitos
  - Dígito mais significativo representa o sinal: $d_{N-1} = s_{N-1}$.
  - Os restantes $(N-1)$ dígitos representam a quantidade numérica.

- Números positivos
  - Dígito mais significativo: $d_{N-1} = s_{N-1} = 0$.
  - Representação: $(q_{I+})_b = [s_{N-1} d_{N-2} \cdots d_2 d_1 d_0]_b = [0 \ d_{N-2} \cdots d_2 d_1 d_0]_b$.
  - Codificação: sinal-e-magnitude.

- Números negativos
  - Dígito mais significativo: $d'_{N-1} = s'_{N-1} = (b-1)$.
  - Representação: $(q_{I-})_b = [s'_{N-1} d'_{N-2} \cdots d'_2 d'_1 d'_0]_b = [(b-1) \ d'_{N-2} \cdots d'_2 d'_1 d'_0]_b$.
  - Codificações:
    - Sinal-e-magnitude.
    - Sinal-e-complemento:
      - Complemento à base.
      - Complemento à base diminuída.

- A seguir, são abordadas as codificações de números negativos para $b = 2$.

Sinal-e-magnitude

- Dígito de sinal sem peso numérico (w_{N-1}) na representação, indicando apenas o valor do sinal.
- O dígito de sinal pode ser associado a um fator multiplicativo igual a -1.
- Demais dígitos representam um valor numérico positivo.
- Apresenta dupla representação para o valor numérico nulo: $+(0)$ e $-(0)$.
- A Equação (8.7) apresenta uma interpretação numérica da representação, para $b = 2$.
- Um exemplo é apresentado na Tabela 8.1, para $b = 2$ e $N = 4$.

$$\begin{aligned}(q_-)_2 &= [s'_{N-1}d'_{N-2}\cdots d'_2d'_1d'_0]_2 \\ &= (-1)^{s'_{N-1}} \times \left[(d'_{N-2}\times 2^{N-2}) + \cdots + (d'_2\times 2^2) + (d'_1\times 2^1) + (d'_0\times 2^0)\right] \\ &= (-1)^{s'_{N-1}} \times \left(\sum_{k=0}^{N-2} d'_k 2^k\right) . \end{aligned} \tag{8.7}$$

Binário				Decimal	Interpretação
0	1	1	1	7	$(+1)\cdot(7)$
0	1	1	0	6	$(+1)\cdot(6)$
0	1	0	1	5	$(+1)\cdot(5)$
0	1	0	0	4	$(+1)\cdot(4)$
0	0	1	1	3	$(+1)\cdot(3)$
0	0	1	0	2	$(+1)\cdot(2)$
0	0	0	1	1	$(+1)\cdot(1)$
0	0	0	0	0	$(+1)\cdot(0)$
1	0	0	0	0	$(-1)\cdot(0)$
1	0	0	1	-1	$(-1)\cdot(1)$
1	0	1	0	-2	$(-1)\cdot(2)$
1	0	1	1	-3	$(-1)\cdot(3)$
1	1	0	0	-4	$(-1)\cdot(4)$
1	1	0	1	-5	$(-1)\cdot(5)$
1	1	1	0	-6	$(-1)\cdot(6)$
1	1	1	1	-7	$(-1)\cdot(7)$

Tabela 8.1: Tabela de sinal-e-magnitude, para número inteiros, $b = 2$ e $N = 4$.

Complemento à base diminuída

- Para base $b = 2$, tem-se uma representação em complemento-a-1.
- Dígito de sinal tem peso negativo: $w_{N-1} = [-(2^{N-1} - 1)]$.
- Demais dígitos representam um valor numérico positivo que, somado ao valor negativo do dígito de sinal, fornece o valor negativo desejado.
- Apresenta dupla representação para o valor numérico nulo: $+(0)$ e $-(0)$.
- A Equação (8.8) apresenta uma interpretação numérica da representação, para $b = 2$.
- Um exemplo é apresentado na Tabela 8.2, para $b = 2$ e $N = 4$.
- Pode-se definir o seguinte algoritmo para a conversão entre as representações de quantidades positivas e negativas, em complemento-a-1:
  - Dada uma representação numérica, em complemento-a-1, para se obter sua representação complementar basta que se troque os numerais 0 por 1 e que se troque os numerais 1 por 0.

$$\begin{aligned}(q_-)_2 &= [s'_{N-1}d'_{N-2}\cdots d'_2d'_1d'_0]_2 \\ &= [-(2^{N-1}-1)] + [(d'_{N-2}\times 2^{N-2}) + \cdots + (d'_2\times 2^2) + (d'_1\times 2^1) + (d'_0\times 2^0)] \\ &= [-(2^{N-1}-1)] + \left(\sum_{k=0}^{N-2} d'_k 2^k\right) \ . \end{aligned} \tag{8.8}$$

Binário				Decimal	Interpretação
0	1	1	1	7	$(0)+(7)$
0	1	1	0	6	$(0)+(6)$
0	1	0	1	5	$(0)+(5)$
0	1	0	0	4	$(0)+(4)$
0	0	1	1	3	$(0)+(3)$
0	0	1	0	2	$(0)+(2)$
0	0	0	1	1	$(0)+(1)$
0	0	0	0	0	$(0)+(0)$
1	1	1	1	0	$(-7)+(7)$
1	1	1	0	-1	$(-7)+(6)$
1	1	0	1	-2	$(-7)+(5)$
1	1	0	0	-3	$(-7)+(4)$
1	0	1	1	-4	$(-7)+(3)$
1	0	1	0	-5	$(-7)+(2)$
1	0	0	1	-6	$(-7)+(1)$
1	0	0	0	-7	$(-7)+(0)$

Tabela 8.2: Tabela de complemento-a-1, para número inteiros, $b = 2$ e $N = 4$.

Complemento à base

- Para base $b = 2$, tem-se uma representação em complemento-a-2.
- Dígito de sinal tem peso negativo: $w_{N-1} = (-2^{N-1})$.
- Demais dígitos representam um valor numérico positivo que, somado ao valor negativo do dígito de sinal, fornece o valor negativo desejado.
- Apresenta representação única para o valor numérico nulo: 0.
- A Equação (8.9) apresenta uma interpretação numérica da representação, para $b = 2$.
- Um exemplo é apresentado na Tabela 8.3, para $b = 2$ e $N = 4$.
- Podem-se definir os seguintes algoritmos para a conversão entre as representações de quantidades positivas e negativas, em complemento-a-2:
  - Dada uma representação numérica, em complemento-a-2, para se obter sua representação complementar basta: i) que se troque os numerais 0 por 1 e que se troque os numerais 1 por 0 (complemento-a-1) e, em seguida, ii) que seja adicionado 1 ao dígito menos significativo.
  - Dada uma representação numérica, em complemento-a-2, para se obter sua representação complementar deve-se realizar uma busca a partir do dígito menos significativo. Durante a busca, os dígitos não serão modificados até que seja encontrado o primeiro numeral 1, que também não será modificado. A partir deste ponto, basta que se troque os numerais 0 por 1 e que se troque os numerais 1 por 0.
- O primeiro algoritmo é mais adequado em processos de adição, quando já se dispõe de um circuito somador.
- O segundo algoritmo é mais recomendado quando se deseja a simples complementação.

$$\begin{aligned}(q_-)_2 &= [s'_{N-1}d'_{N-2}\cdots d'_2d'_1d'_0]_2 \\ &= (-2^{N-1}) + [(d'_{N-2}\times 2^{N-2}) + \cdots + (d'_2\times 2^2) + (d'_1\times 2^1) + (d'_0\times 2^0)] \\ &= (-2^{N-1}) + \left(\sum_{k=0}^{N-2} d'_k 2^k\right) \ . \end{aligned} \tag{8.9}$$

Binário				Decimal	Interpretação
0	1	1	1	7	$(0)+(7)$
0	1	1	0	6	$(0)+(6)$
0	1	0	1	5	$(0)+(5)$
0	1	0	0	4	$(0)+(4)$
0	0	1	1	3	$(0)+(3)$
0	0	1	0	2	$(0)+(2)$
0	0	0	1	1	$(0)+(1)$
0	0	0	0	0	$(0)+(0)$
1	1	1	1	-1	$(-8)+(7)$
1	1	1	0	-2	$(-8)+(6)$
1	1	0	1	-3	$(-8)+(5)$
1	1	0	0	-4	$(-8)+(4)$
1	0	1	1	-5	$(-8)+(3)$
1	0	1	0	-6	$(-8)+(2)$
1	0	0	1	-7	$(-8)+(1)$
1	0	0	0	-8	$(-8)+(0)$

Tabela 8.3: Tabela de complemento-a-2, para número inteiros, $b = 2$ e $N = 4$.

8.2.4 Representação de números fracionários negativos

Equacionamento escalado

- O equacionamento utilizado para a representação de números inteiros negativos, na base $b = 2$, pode ser aproveitado para números negativos puramente fracionários.
- Uma quantidade puramente fracionária x_F pode ser obtida através da multiplicação de uma quantidade inteira x_I por um fator de escala FE adequado ($x_F = FE \cdot x_I$).
- Assim, para aproveitar o equacionamento anterior, basta utilizar um escalamento.
- A título de exemplo, as Tabelas 8.1 - 8.3, que representam números inteiros, são transformadas nas Tabelas 8.4 - 8.6, para números fracionários, através do fator de escala $FE = 2^{-3} = 8^{-1}$.

Equacionamento alternativo

- Um equacionamento alternativo pode ser proposto para números puramente fracionários, negativos e ordenados, codificados em complemento-a-2.
- Suponha-se que x é uma quantidade numérica puramente fracionária, representada na base $b = 2$, tal que $0 < x < 1$.
- Utilizando-se a codificação em complemento-a-2, a representação da quantidade negativa $(-x)$ pode ser equacionada por $(-x) = (x_{C2}) = 2 - (x)$.

Binário				Decimal	Interpretação
0	1	1	1	0.875	$(+1) \cdot (0.875)$
0	1	1	0	0.750	$(+1) \cdot (0.750)$
0	1	0	1	0.625	$(+1) \cdot (0.625)$
0	1	0	0	0.500	$(+1) \cdot (0.500)$
0	0	1	1	0.375	$(+1) \cdot (0.375)$
0	0	1	0	0.250	$(+1) \cdot (0.250)$
0	0	0	1	0.125	$(+1) \cdot (0.125)$
0	0	0	0	0.000	$(+1) \cdot (0.000)$
1	0	0	0	0.000	$(-1) \cdot (0.000)$
1	0	0	1	-0.125	$(-1) \cdot (0.125)$
1	0	1	0	-0.250	$(-1) \cdot (0.250)$
1	0	1	1	-0.375	$(-1) \cdot (0.375)$
1	1	0	0	-0.500	$(-1) \cdot (0.500)$
1	1	0	1	-0.625	$(-1) \cdot (0.625)$
1	1	1	0	-0.750	$(-1) \cdot (0.750)$
1	1	1	1	-0.875	$(-1) \cdot (0.875)$

Tabela 8.4: Tabela de sinal-e-magnitude, para números puramente fracionários, $b = 2$ e $N = 4$.

Binário				Decimal	Interpretação
0	1	1	1	0.875	$(0.000) + (0.875)$
0	1	1	0	0.750	$(0.000) + (0.750)$
0	1	0	1	0.625	$(0.000) + (0.625)$
0	1	0	0	0.500	$(0.000) + (0.500)$
0	0	1	1	0.375	$(0.000) + (0.375)$
0	0	1	0	0.250	$(0.000) + (0.250)$
0	0	0	1	0.125	$(0.000) + (0.125)$
0	0	0	0	0.000	$(0.000) + (0.000)$
1	1	1	1	0.000	$(-0.875) + (0.875)$
1	1	1	0	-0.125	$(-0.875) + (0.750)$
1	1	0	1	-0.250	$(-0.875) + (0.625)$
1	1	0	0	-0.375	$(-0.875) + (0.500)$
1	0	1	1	-0.500	$(-0.875) + (0.375)$
1	0	1	0	-0.625	$(-0.875) + (0.250)$
1	0	0	1	-0.750	$(-0.875) + (0.125)$
1	0	0	0	-0.875	$(-0.875) + (0.000)$

Tabela 8.5: Tabela de complemento-a-1, para números puramente fracionários, $b = 2$ e $N = 4$.

Binário				Decimal	Interpretação
0	1	1	1	0.875	$(0) + (0.875)$
0	1	1	0	0.750	$(0) + (0.750)$
0	1	0	1	0.625	$(0) + (0.625)$
0	1	0	0	0.500	$(0) + (0.500)$
0	0	1	1	0.375	$(0) + (0.375)$
0	0	1	0	0.250	$(0) + (0.250)$
0	0	0	1	0.125	$(0) + (0.125)$
0	0	0	0	0.000	$(0) + (0.000)$
1	1	1	1	-0.125	$(-1) + (0.875)$
1	1	1	0	-0.250	$(-1) + (0.750)$
1	1	0	1	-0.375	$(-1) + (0.625)$
1	1	0	0	-0.500	$(-1) + (0.500)$
1	0	1	1	-0.625	$(-1) + (0.375)$
1	0	1	0	-0.750	$(-1) + (0.250)$
1	0	0	1	-0.875	$(-1) + (0.125)$
1	0	0	0	-1.000	$(-1) + (0.000)$

Tabela 8.6: Tabela de complemento-a-2, para números puramente fracionários, $b = 2$ e $N = 4$.

8.2.5 Adição e subtração em complemento-a-2

- A codificação em complemento-a-2 apresenta, entre outras, a grande vantagem de transformar o processo de subtração em pura adição: $x_1 - x_2 = x_1 + (-x_2) = x_1 + (x_2)_{C2}$.

- Assim, um único bloco somador pode ser usado para realizar as operações de adição e subtração de números codificados em complemento-a-2.

- A adição de dois números puramente fracionários pode produzir um número com parte inteira.

- Na representação de números puramente fracionários, com ponto fixo, não são utilizados dígitos para valores inteiros.

- Portanto, um resultado contendo parte inteira é considerado uma situação de *overflow*.

Análise de *overflow* na adição em complemento-a-2

- Considerando-se um bloco somador, operando com dados puramente fracionários, codificados em complemento-a-2, o sinal de saída *carry-out* representa uma parte inteira de valor $v_I = 2$.

- Caso 1: adição de números positivos.
  $0 \leq x_1 < 1$, $0 \leq x_2 < 1$ e $x_A = x_1 + x_2$.
  Logo: $0 \leq x_A < 2$.
  Se $0 \leq x_A < 1$: adição sem *overflow*.
  Se $1 \leq x_A < 2$: adição com *overflow*.

- Caso 2: subtração de números positivos.
  $0 \leq x_1 < 1$, $-1 < x_2 < 0$, $(-|x_2|)_{C2} = 2 - |x_2|$ e
  $x_A = (x_1 + x_2) = x_1 - |x_2| = x_1 + (x_2)_{C2} = 2 + (x_1 - |x_2|)$.
  Logo: $-1 < x_A < 1$.
  Portanto, nesse caso, não haverá ocorrência de *overflow*.
  Se $x_1 \geq |x_2|$: resultado positivo, bastando ignorar o sinal de *carry-out* ($x_P = x_A - 2$).
  Se $x_1 < |x_2|$: resultado negativo já codificado ($x_N = x_A$).

- Caso 3: adição de números negativos.
  $-1 < x_2 < 0$, $-1 < x_2 < 0$, $(-|x_1|)_{C2} = 2 - |x_1|$, $(-|x_2|)_{C2} = 2 - |x_2|$ e
  $(x_{C2})_A = x_1 + x_2 = (-|x_1|)_{C2} + (-|x_2|)_{C2} = (2 - |x_1|) + (2 - |x_2|) = 2 + [2 - (|x_1| + |x_2|)]$.
  Logo: $0 < |x_1| + |x_2| < 2 \rightarrow 2 < (x_{C2})_A < 4$.
  Se $2 < (x_{C2})_A \leq 3$: adição com *overflow*.
  Se $3 < (x_{C2})_A < 4$: adição sem *overflow*.
  Se o resultado for sem *overflow*, o mesmo já estará codificado, bastando ignorar o sinal de *carry-out* ($(x_A)_{C2} = (x_{C2})_A - 2$).

Detecção e tratamento de *overflow* na adição em complemento-a-2

- Pelos resultados da análise de *overflow* na adição em complemento-a-2, não é difícil encontrar um mecanismo que indique sua ocorrência.

- A detecção de *overflow* pode ser feita através da análise dos *bits* de sinal dos operandos e do resultado, bem como do sinal de *carry-out* do bloco somador.

- As Tabelas 8.7 e 8.8 apresentam duas formas de detecção *overflow* na adição em complemento-a-2, onde $OF = 0$ e $OF = 1$ indicam a ausência e a presença de *overflow*, respectivamente.

- O tratamento de *overflow* mais comumente empregado é a saturação no valor máximo representável (positivo ou negativo).

Caso	d_{s1}	d_{s2}	d_{sA}	c_O	OF
Adição de positivos	0	0	0	X	0
	0	0	1	X	1
Subtração de positivos	0	1	0	X	0
	1	0	0	X	0
Adição de negativos	1	1	0	X	1
	1	1	1	X	0

$X = don\acute{}t\ care$

$OF = 0 \rightarrow$ sem *overflow*
$OF = 1 \rightarrow$ com *overflow*

Tabela 8.7: Forma 1 para detecção de *overflow* na adição em complemento-a-2.

Caso	d_{s1}	d_{s2}	d_{sA}	c_O	OF
Adição de positivos	0	0	0	0	0
				1	X
			1	0	1
				1	X
Subtração de positivos	0	1	0	0	X
				1	0
			1	0	0
				1	X
	1	0	0	0	X
				1	0
			1	0	0
				1	X
Adição de negativos	1	1	0	0	X
				1	1
			1	0	X
				1	0

$X = \textit{can´t happen}$

$OF = 0 \rightarrow$ sem *overflow*
$OF = 1 \rightarrow$ com *overflow*

Tabela 8.8: Forma 2 para detecção de *overflow* na adição em complemento-a-2.

8.2.6 Tabelas de operações básicas entre dígitos

- Para uma determinada base, as operações de adição e multiplicação entre dígitos podem ser facilmente definidas por meio de tabelas.

- As Figuras 8.5 - 8.7 apresentam as tabelas para as bases $b = 2$, $b = 3$ e $b = 4$, respectivamente.

- A partir de tais tabelas, definidas para dígitos, podem ser definidos algoritmos e implementações para uma operação envolvendo quantidades genéricas, expressas na base em questão.

+	0	1
0	0	1
1	1	10

(a)

×	0	1
0	0	0
1	0	1

(b)

Figura 8.5: Tabelas de operações entre dígitos para $b = 2$: (a) adição e (b) multiplicação.

+	0	1	2
0	0	1	2
1	1	2	10
2	2	10	11

(a)

×	0	1	2
0	0	0	0
1	0	1	2
2	0	2	11

(b)

Figura 8.6: Tabelas de operações entre dígitos para $b = 3$: (a) adição e (b) multiplicação.

+	0	1	2	3
0	0	1	2	3
1	1	2	3	10
2	2	3	10	11
3	3	10	11	12

(a)

×	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	10	12
3	0	3	12	101

(b)

Figura 8.7: Tabelas de operações entre dígitos para $b = 4$: (a) adição e (b) multiplicação.

8.2.7 Escalamento por potência inteira da base

- Um multiplicador é um circuito com relativa complexidade de implementação. Conseqüentemente, possui relevantes medidas de custo (espaço ocupado, energia consumida e tempo de operação).

- Por outro lado, o escalamento por potência inteira da base é uma operação simples, com baixa complexidade de implementação.

- O escalamento pode ser de dois tipos, dependendo do valor da potência inteira da base: multiplicação (valor positivo) ou divisão (valor negativo).

- A Equação (8.10) apresenta uma quantidade genérica q, representada na base b.

- A multiplicação de q pela base b é definida nas Equações (8.11) e (8.12).

- A divisão de q pela base b é definida nas Equações (8.13) e (8.14).

- De acordo com as Equações (8.10) e (8.14), a implementação do escalamento pode ser obtida através do simples deslocamento dos dígitos da representação.

$$(q)_b = (q_I)_b + (q_F)_b = \sum_{k=N_I}^{-N_F} d_k b^k = [d_{N_I} \cdots d_2 d_1 d_0 \cdot d_{-1} d_{-2} \cdots d_{-N_F}]_b \ . \tag{8.10}$$

$$\begin{aligned}
(q\prime)_b &= (q)_b \times b \\
&= \left(\sum_{k=N_I}^{-N_F} d_k b^k \right) \times b = \sum_{k=N_I}^{-N_F} d_k b^{k+1} = \sum_{k=N_I+1}^{-N_F+1} d_{k-1} b^k = \sum_{k=N_I+1}^{-N_F+1} d'_k b^k \\
&= [d_{N_I} \cdots d_2 d_1 d_0 d_{-1} \cdot d_{-2} \cdots d_{-N_F}]_b \\
&= [d'_{N_I+1} \cdots d'_3 d'_2 d'_1 d'_0 \cdot d'_{-1} d'_{-2} \cdots d'_{-N_F+1}]_b \ .
\end{aligned} \tag{8.11}$$

$$d'_k = d_{k-1} \ . \tag{8.12}$$

$$\begin{aligned}
(q\prime)_b &= (q)_b \times b^{-1} \\
&= \left(\sum_{k=N_I}^{-N_F} d_k b^k \right) \times b^{-1} = \sum_{k=N_I}^{-N_F} d_k b^{k-1} = \sum_{k=N_I-1}^{-N_F-1} d_{k+1} b^k = \sum_{k=N_I-1}^{-N_F-1} d'_k b^k \\
&= [d_{N_I} \cdots d_2 d_1 \cdot d_0 d_{-1} d_{-2} \cdots d_{-N_F}]_b \\
&= [d'_{N_I-1} \cdots d'_3 d'_2 d'_1 d'_0 \cdot d'_{-1} d'_{-2} \cdots d'_{-N_F-1}]_b \ .
\end{aligned} \tag{8.13}$$

$$d'_k = d_{k+1} \ . \tag{8.14}$$

8.2.8 Conversão entre bases

- A seguir, são consideradas as conversões de números positivos (inteiros e puramente fracionários).

Números positivos e inteiros

- A conversão da base s para a base t significa que, conhecendo-se os dígitos d'_i da Equação 8.18, deseja-se encontrar os dígitos d_i da Equação 8.19.

- Considerando-se todas as quantidades expressas na base s, podem-se definir as relações expressas na Equação 8.17.

- Assim, para que se encontrem os dígitos d_i, basta que se realizem divisões sucessivas do dividendo N_i pelo divisor t, gerando-se o quociente N_{i+1} e o resto d_i, e que, no final, os restos sejam posicionados na ordem adequada.

- Uma vez que o número de dígitos d_i é finito, é garantido que o algoritmo terá um número finito de passos.

$$(q)_s = [d'_J \cdots d'_1 d'_0]_s = (N_0)_s \ . \tag{8.15}$$

$$(q)_t = [d_K \cdots d_1 d_0]_t \ . \tag{8.16}$$

$$\begin{aligned}
N_0 &= \left(d_K \times t^K + \cdots + d_2 \times t^2 + d_1 \times t^1 + d_0 \times t^0\right) \\
&= \left(d_K \times t^{K-1} + \cdots + d_2 \times t^1 + d_1 \times t^0\right) \times t + \left(d_0 \times t^0\right) \\
&= N_1 \times t + d_0 \\
\\
N_1 &= \left(d_K \times t^{K-1} + \cdots + d_2 \times t^1 + d_1 \times t^0\right) \\
&= \left(d_K \times t^{K-2} + \cdots + d_2 \times t^0\right) \times t + \left(d_1 \times t^0\right) \\
&= N_2 \times t + d_1 \\
\\
&\vdots \\
\\
N_{K-1} &= \left(d_K \times t^1 + d_{K-1} \times t^0\right) \\
&= (d_K) \times t + \left(d_{K-1} \times t^0\right) \\
&= N_K \times t + d_{K-1} \\
\\
N_K &= d_K \ .
\end{aligned} \tag{8.17}$$

Números positivos e puramente fracionários

- A conversão da base s para a base t significa que, conhecendo-se os dígitos d'_i da Equação 8.18, deseja-se encontrar os dígitos d_i da Equação 8.19.

- Considerando-se todas as quantidades expressas na base s, podem-se definir as relações expressas na Equação 8.20.

- Assim, para que se encontrem os dígitos d_i, basta que se realizem multiplicações sucessivas do multiplicando puramente fracionário N_i pelo multiplicador t, gerando-se o resultado N_{i-1}, que contém d_i como parte inteira, e que, no final, os restos sejam posicionados na ordem adequada.

- Uma vez que não se pode garantir que o número de dígitos d_i será finito, deve-se estabelecer um número máximo de passos para garantir que o algoritmo terá um término.

$$(q)_s = [d'_{-1}d'_{-2}\cdots d'_{-J}]_s = (N_{-1})_s \ . \tag{8.18}$$

$$(q)_t = [d_{-1}d_{-2}\cdots d_{-K}]_t \ . \tag{8.19}$$

$$\begin{aligned}
N_{-1}\times t &= \left(d_{-1}\times t^{-1} + d_{-2}\times t^{-2} + d_{-3}\times t^{-3} + \cdots + d_{-K}\times t^{-K}\right)\times t \\
&= \left(d_{-1}\times t^{0}\right) + \left(d_{-2}\times t^{-1} + d_{-3}\times t^{-2} + \cdots + d_{K}\times t^{-K+1}\right) \\
&= d_{-1} + N_{-2} \\
\\
N_2\times t &= \left(d_{-2}\times t^{-1} + d_{-3}\times t^{-2} + \cdots + d_{-K}\times t^{-K+1}\right)\times t \\
&= \left(d_{-2}\times t^{0}\right) + \left(d_{-3}\times t^{-1} + \cdots + d_{K}\times t^{-K+2}\right) \\
&= d_{-2} + N_{-3} \\
\\
&\vdots
\end{aligned} \tag{8.20}$$

8.2.9 Bases mais comuns em circuitos digitais

- A notação em base $b = 2$ é mais adequada para a implementação de circuitos digitais.
- Para uma base de valor reduzido, a representação terá um número elevado de dígitos.
- Para o uso humano, quanto maior é o número de dígitos, mais trabalhoso é a sua interpretação e a sua manipulação.
- Para simplificar a representação, duas bases são muito utilizadas: octal e hexadecimal.
- A base octal emprega $b = 8$ e dígitos $d_i \in \boldsymbol{S} = \{0, 1, 2, \cdots, 7\}$.
- A base hexadecimal emprega $b = 16$ e dígitos $d_i \in \boldsymbol{S} = \{0, 1, 2, \cdots, 9, A, B, ..., F\}$.
- Supondo-se números positivos e inteiros, as Equações (8.21) – (8.23) ilustram as notações nas três bases.

$$(q_I)_2 = (d_J \times 2^J) + \cdots + (d_2 \times 2^2) + (d_1 \times 2^1) + (d_0 \times 2^0) \ . \tag{8.21}$$

$$(q_I)_8 = (d'_K \times 8^K) + \cdots + (d'_2 \times 8^2) + (d'_1 \times 8^1) + (d'_0 \times 8^0) \ . \tag{8.22}$$

$$(q_I)_{16} = (d''_L \times 16^L) + \cdots + (d''_2 \times 16^2) + (d''_1 \times 16^1) + (d''_0 \times 16^0) \ . \tag{8.23}$$

- As bases binária, octal e hexadecimal são comumente utilizadas em conjunto, devido à facilidade de conversão entre as três bases.
- As Equações (8.24) – (8.28) ilustram a relação entre as bases binária e octal.

$$\begin{aligned}
(q_I)_2 &= (d_J \times 2^J) + (d_{J-1} \times 2^{J-1}) + (d_{J-2} \times 2^{J-2}) + \cdots + \\
&\quad (d_5 \times 2^5) + (d_4 \times 2^4) + (d_3 \times 2^3) + \\
&\quad (d_2 \times 2^2) + (d_1 \times 2^1) + (d_0 \times 2^0) \\
&= \left[(d_J \times 2^2) + (d_{J-1} \times 2^1) + (d_{J-2} \times 2^0)\right] \times 2^{J-2} + \cdots + \\
&\quad \left[(d_5 \times 2^2) + (d_4 \times 2^1) + (d_3 \times 2^0)\right] \times 2^3 + \\
&\quad \left[(d_2 \times 2^2) + (d_1 \times 2^1) + (d_0 \times 2^0)\right] \times 2^0 \\
&= (d'_K \times 8^K) + \cdots + (d'_1 \times 8^1) + (d'_0 \times 8^0) \\
&= (q_I)_8 \ .
\end{aligned} \tag{8.24}$$

$$J - 2 = 3K \ . \tag{8.25}$$

$$[d_2 d_1 d_0]_2 = [d'_0]_8 \ . \tag{8.26}$$

$$[d_5 d_4 d_3]_2 = [d'_1]_8 \ . \tag{8.27}$$

$$[d_J d_{J-1} d_{J-2}]_2 = [d'_K]_8 \ . \tag{8.28}$$

- As Equações (8.29) – (8.33) ilustram a relação entre as bases binária e hexadecimal.

$$\begin{aligned}
(q_I)_2 &= (d_J \times 2^J) + (d_{J-1} \times 2^{J-1}) + (d_{J-2} \times 2^{J-2}) + (d_{J-3} \times 2^{J-3}) + \cdots + \\
&\quad (d_7 \times 2^7) + (d_6 \times 2^6) + (d_5 \times 2^5) + (d_4 \times 2^4) + \\
&\quad (d_3 \times 2^3) + (d_2 \times 2^2) + (d_1 \times 2^1) + (d_0 \times 2^0) \\
&= \left[(d_J \times 2^3) + (d_{J-1} \times 2^2) + (d_{J-2} \times 2^1) + (d_{J-3} \times 2^0)\right] \times 2^{J-3} + \cdots + \\
&\quad \left[(d_7 \times 2^3) + (d_6 \times 2^2) + (d_5 \times 2^1) + (d_4 \times 2^0)\right] \times 2^4 + \\
&\quad \left[(d_3 \times 2^3) + (d_2 \times 2^2) + (d_1 \times 2^1) + (d_0 \times 2^0)\right] \times 2^0 \\
&= (d''_L \times 16^L) + \cdots + (d''_1 \times 16^1) + (d''_0 \times 16^0) \\
&= (q_I)_{16} \; . \qquad (8.29)
\end{aligned}$$

$$J - 3 = 4L \; . \qquad (8.30)$$

$$[d_3 d_2 d_1 d_0]_2 = [d''_0]_{16} \; . \qquad (8.31)$$

$$[d_7 d_6 d_5 d_4]_2 = [d''_1]_{16} \; . \qquad (8.32)$$

$$[d_J d_{J-1} d_{J-2} d_{J-3}]_2 = [d''_L]_{16} \; . \qquad (8.33)$$

- As Equações (8.34) – (8.38) ilustram a relação entre as bases octal e hexadecimal.

$$\begin{aligned}
(q_I)_8 &= (d'_K \times 8^K) + (d'_{K-1} \times 8^{K-1}) + \cdots + \\
&\quad (d'_3 \times 8^3) + (d'_2 \times 8^2) + \\
&\quad (d'_1 \times 8^1) + (d'_0 \times 8^0) \\
&= \left[(d'_K \times 8^1) + (d'_{K-1} \times 8^0)\right] \times 8^{K-1} + \cdots + \\
&\quad \left[(d'_3 \times 8^1) + (d'_2 \times 8^0) +\right] \times 8^2 + \\
&\quad \left[(d'_1 \times 8^1) + (d'_0 \times 8^0)\right] \times 8^0 \\
&= (d''_L \times 16^L) + \cdots + (d''_1 \times 16^1) + (d''_0 \times 16^0) \\
&= (q_I)_{16} \; . \qquad (8.34)
\end{aligned}$$

$$K - 1 = 2L \; . \qquad (8.35)$$

$$[d'_1 d'_0]_8 = [d''_0]_{16} \; . \qquad (8.36)$$

$$[d'_3 d'_2]_8 = [d''_1]_{16} \; . \qquad (8.37)$$

$$[d'_K d'_{K-1}]_2 = [d''_L]_{16} \; . \qquad (8.38)$$

- Embora todas as equações tenham sido definidas para números positivos e inteiros, não é difícil mostrar que as relações se mantêm para números positivos e fracionários.

8.3 Quantização

- Quantizar significa representar, através de uma aproximação, uma faixa contínua de valores originais por uma faixa de discreta de valores correspondentes.

- Todo sistema de medição possui um intervalo mínimo de medida (resolução da medida).

- Por outro lado, todo sistema de numeração possui um intervalo mínimo de representação das quantidades numéricas (resolução da representação).

- Portanto, toda medida, bem como a sua respectiva representação, possuem um grau intrínseco de aproximação.

- Dependendo do parâmetros considerados, a quantização pode assumir diversas clasificações.

- Quanto à regularidade da discretização efetuada, a quantização pode ser classificada como: uniforme e não uniforme.

- Na quantização uniforme é utilizado um intervalo único de discretização.

- Na quantização não uniforme são empregados diversos intervalos de discretização diferentes.

- Quanto à aproximação adotada para o valor numérico, podem-se destacar três tipos de quantização: truncamento, arredondamento e truncamento em magnitude.

- O truncamento assume o simples abandono dos dígitos menos significativos. Assim sendo, não se pode garantir que o valor final seja mais próximo do valor original. Além disso, dependendo do código utilizado para representar a quantidade numérica, o módulo do valor original pode diminuir ou aumentar.

- No arredondamento, é realizada uma análise dos dígitos menos significativos, de forma que o valor final seja mais próximo do valor original.

- Em alguns sistemas digitais, é desejado que o módulo dos valores quantizados nunca seja aumentado. Dessa forma, realiza-se o denominado truncamento em magnitude. Para alguns códigos, isso significa o simples truncamento do valor original. Para outros, deve-se efetuar uma análise do valor original, de forma a garantir que não ocorra um aumento no seu módulo.

8.4 Exercícios propostos

1. Considerando o SNPC, para cada uma das bases listadas abaixo, obter as respectivas representações para as quantidades apresentadas em seguida.

   (a) Base $b = 2$.

   (b) Base $b = 3$.

   (c) Base $b = 16$.

   Quantidades numéricas:

   i. $q = (17)_{10}$.
   ii. $q = (24)_{10}$.
   iii. $q = (32)_{10}$.
   iv. $q = (48)_{10}$.
   v. $q = (80)_{10}$.
   vi. $q = (144)_{10}$.
   vii. $q = (272)_{10}$.
   viii. $q = (528)_{10}$.

2. Considerando o SNPC, com base $b = 2$, para cada uma das codificações listadas abaixo, obter as respectivas representações para as quantidades apresentadas em seguida.

   (a) Sinal-e-magnitude.

   (b) Complemento-a-1.

   (c) Complemento-a-2.

   Quantidades numéricas:

   i. $q = (-17)_{10}$.
   ii. $q = (-24)_{10}$.
   iii. $q = (-32)_{10}$.
   iv. $q = (-48)_{10}$.
   v. $q = (-80)_{10}$.
   vi. $q = (-144)_{10}$.
   vii. $q = (-272)_{10}$.
   viii. $q = (-528)_{10}$.

3. Considerando o SNPC, com base $b = 2$, com codificação em complemento-a-2, analise o resultado das seguintes operações:

   (a) $(00100) + (01001)$.

   (b) $(01100) + (01101)$.

   (c) $(00100) + (10111)$.

   (d) $(10100) + (01101)$.

   (e) $(11100) + (10111)$.

   (f) $(10100) + (10011)$.

4. Considerando o SNPC, com base $b = 2$, com codificação em complemento-a-2, com um total 5 dígitos, para cada uma das quantizações listadas abaixo, obter as respectivas representações para as quantidades apresentadas em seguida.

   (a) Truncamento.

   (b) Arredondento.

   (c) Truncamento em magnitude.

   Quantidades numéricas:

   i. (0010000).
   ii. (0010001).
   iii. (0010010).
   iv. (0010011).
   v. (0010100).
   vi. (0010101).
   vii. (0010110).
   viii. (0010111).
   ix. (1110000).
   x. (1101111).
   xi. (1101110).
   xii. (1101101).
   xiii. (1101100).
   xiv. (1101011).
   xv. (1101010).
   xvi. (1101001).

Capítulo 9

Circuitos combinacionais básicos

9.1 Introdução

Escrever...

9.2 Uso de portas lógicas como elementos de controle

Escrever...

9.3 Multiplexadores e demultiplexadores

Escrever...

9.4 Decodificadores de linha

Escrever...

9.5 Somadores simples

Escrever...

9.5.1 *Half-adder*

Escrever...

9.5.2 *Full-adder*

Escrever...

9.5.3 *Ripple-carry adder*

Escrever...

9.6 Subtratores simples

Escrever...

9.6.1 *Half-subtractor*

Escrever...

9.6.2 *Full-subtractor*

Escrever...

9.6.3 *Ripple-borrow subtractor*

Escrever...

9.7 Complementadores

Escrever...

9.7.1 Complementador-a-1 (*bitwise implementation*)

Escrever...

9.7.2 Complementador-a-2

Complementador-a-1 + somador

Escrever...

Complementador-a-2 puro (*bit-scanning implementation*)

Escrever...

9.8 Comparadores

Escrever...

Apêndice A

Tópicos sobre divisão de números inteiros

A.1 Algoritmo de divisão inteira

Teorema (Divisão com resto): Para cada inteiro c (dividendo) e cada inteiro positivo d (divisor), existe um único par de inteiros Q (quociente) e r (resto), tal que $c = d \cdot Q + r$, onde $0 \leq r < d$.

A.2 Quociente

O quociente pode ser descrito por

$$Q = \left\lfloor \frac{c}{d} \right\rfloor \ ,$$

onde $\lfloor(\cdot)\rfloor$ representa o maior inteiro menor que $(\cdot)$.

A.3 Resto ou resíduo

O resto da divisão de c por d pode ser descrito por

$$r = R_d[c] = ((c)) = c \qquad (\text{mod} \quad d) \ ,$$

podendo ainda ser denominado de resíduo de c, módulo d.

A.4 Congruência

Dois números inteiros c_1 e c_2 que, divididos por um terceiro inteiro positivo d, apresentam o mesmo resto (ou resíduo) r são ditos congruentes, módulo d, e são representados por

$$c_1 \equiv c_2 \qquad (\text{mod} \quad d) \ ,$$

onde $\equiv$ denota uma relação de equivalência.

A.5 Relações úteis

Teorema: Para um mesmo número inteiro positivo d,

(i) $R_d[a+b] = R_d[R_d[a] + R_d[b]]$

(ii) $R_d[a \cdot b] = R_d[R_d[a] \cdot R_d[b]]$

onde $+$ e $\cdot$ denotam, respectivamente, as operações de adição e multiplicação entre números inteiros.

Referências Bibliográficas

[HP81] F. J. Hill and G. R. Peterson. *Introduction to Switching Theory and Logical Design*. John Wiley, New York, NY, 3rd edition, 1981.

[IC08] I. V. Idoeta e F. G. Capuano. *Elementos de Eletrônica Digital*. Editora Érica, 40.ª edição, 2008.

[Rhy73] V. T. Rhyne. *Fundamentals of Digital Systems Design*. Prentice-Hall, Englewood Cliffs, NJ, 1973.

[Ric56] R. K. Richards. *Arithmetic Operations in Digital Computers*. Van Nostrand Reinhold, New York, NY, 1956.

[Tau82] H. Taub. *Digital Circuits and Microprocessors*. McGraw-Hill, New York, NY, 1982. Em português: McGraw-Hill, Rio de Janeiro, 1984.

[TWM07] R. J. Tocci, N. S. Widmer, e G. L. Moss. *Sistemas Digitais: Princípios e Aplicações*. Prentice Hall, Pearson Education, 10.ª edição, 2007.

[Uye02] J. P. Uyemura. *Sistemas Digitais: Uma abordagem integrada*. Thomson Pioneira, São Paulo, SP, 2002.