UNIVERSIDADE EDUARDO MONDLANE

FACULDADE DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA

Electrónica Digital

Engº. Albino B Cuinhane

AULA TEÓRICA 3
SUMÁRIO

Capítulo 3

FUNDAMENTOS DE CIRCUITOS LÓGICOS

3.1. Álgebra de Boole

3.1. Álgebra de Boole

- **Álgebra de Boole** é uma classe de elementos que podem tomar valores perfeitamente diferenciados, nomeadamente 0 e 1, e que se relacionam pelas operações binárias + (soma) e **.** (produto).

Definição 1.
Variável lógica

Suponha que alguém diz: "*está a chover e a estrada molha*". Nesta frase podemos encontrar duas afirmações a saber:
Primeira – "*está a chover*"
Segunda – "*a estrada molha*"

A cada uma destas duas afirmações se designa por ***proposição.*** A proposição poder ser Verdadeira ou Falsa. De facto se alguém diz "*está a chover*" pode ser verdade ou falso.

Do mesmo modo dizer "*a estrada molha*" pode ser verdade ou falso

3.1. Álgebra de Boole

Proposição – é uma afirmação que pode ser verdadeira ou pode ser falsa. Por outro lado, as proposições podem ser relacionadas por:
+ _ soma
. _ produto e
= _ igualdade.

Se A = "*está a chover*"
e B = "*a estrada molha*"

então A e B designam-se por ***Variáveis Lógicas*** ou ***Variáveis Binárias***. São variáveis uma vez que podem assumir pelo menos dois valores distintos, 0 ou 1. Assim,
Se A é verdadeira teremos A = 1
Se A é falsa teremos A = 0

3.1. Álgebra de Boole

Com efeito, se alguém diz que *"está a chover"* e de facto estiver a chover, A assume o valor lógico 1. Se entretanto não estiver a chover, A assume o valor lógico 0.

Matematicamente **Variável Lógica** é a variável *A* tal que

$$\begin{cases} A = 0, & sse \quad A \neq 1 \\ A = 1, & sse \quad A \neq 0 \end{cases} \qquad (3.1)$$

ou seja, A=1 sse Ā=0
em que Ā é o ***complementar*** de A

De (3.1) verificamos que o conjunto universo de todos os valores possíveis para a VVL contem apenas 2 elementos opostos. É daí que a negação dum deles é complementar do outro.

3.1. Álgebra de Boole

Definição 2.
Principio do terceiro excluído - Da definição de variável lógica (3.1) estabelece-se o principio do terceiro excluído que refere que uma proposição só pode ser verdadeira ou falsa e não existe terceira hipótese.

Definição 3.
Literal - é uma variável lógica ou seu complementar

Definição 4.
Termo Produto - é um grupo de literais relacionados pela operação . (produto). Por exemplo são termos produto os seguintes: A.B.C ou X_1.Ā.X_2.
No termo produto pode-se omitir o símbolo ".". Assim os termos anteriores ficariam ABC ou X_1ĀX_2 sem prejuízo algum

3.1. Álgebra de Boole

Definição 5.
Termo Soma - é um grupo de literais relacionados pela operação + (soma). Por exemplo são termos soma os seguintes: A+B+C ou X_1+Ā+X_2.

Definição 6
Termo Normal – é um termo produto ou termo soma em que nenhum literal surge mais do que uma vez. Por exemplo, não são termos normais os seguintes: A.C.A ou Ā+X_2+A.

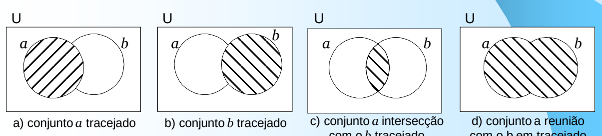
Definição 7
Função Lógica – é uma proposição que depende de outras proposições relacionadas por qualquer operação lógica . ou +. Portanto, pode assumir também dois valores.

3.2. Postulados e Teoremas

3.2.1. Postulados

Postulado 1.
Álgebra de Boole é um sistema algébrico contendo o conjunto K com dois ou mais elementos e duas operações +(OR) e .(AND) de tal modo que sejam dados dois elementos *a* e *b* tem-se:

$$a + b \in K \qquad (P1)$$
$$a.b \in K$$



a) conjunto *a* tracejado b) conjunto *b* tracejado c) conjunto *a* intersecção com o *b* tracejado d) conjunto a reunião com o b em tracejado

Fig. 3.1 Ilustração do postulado 1

3.2.1. Postulados

Postulado 2. "0" como elemento neutro da soma e "1" como elemento neutro do produto

a) Existe um elemento 0 tal que se

$$0 \in K \therefore \forall a \in K, \qquad a + 0 = a \qquad (p2a)$$

b) Existe um elemento 1 tal que se

$$1 \in K \therefore \forall a \in K, \qquad a.1 = a \qquad (p2b)$$

3.2.1. Postulados

Postulado 3. Comutatividade das operações + e .

As operações lógicas + e . gozam da propriedade comutativa

$$\forall a,b \in K,$$
$$a)\; a+b=b+a$$
$$b)\; a.b=b.a \qquad (p3)$$

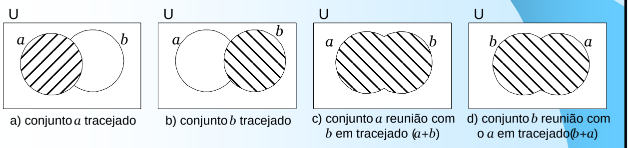


Fig. 3.2 Ilustração do postulado 3

3.2.1. Postulados

Postulado 4. Associatividade das operações + e .

As operações lógicas + e . gozam da propriedade associativa

$$\forall a,b,c \in K,$$
$$a)\; a+(b+c)=(a+b)+c$$
$$b)\; a.(b.c)=(a.b).c \qquad (p4)$$

Postulado 5. Distributividade da + sobre o . e do . sobre a +

As operações lógicas + e . gozam da propriedade distributiva

$$\forall a,b,c \in K,$$
$$a)\; a+(b.c)=(a+b).(a+c)$$
$$b)\; a.(b+c)=(a.b)+(a.c) \qquad (p5)$$

3.2.1. Postulados

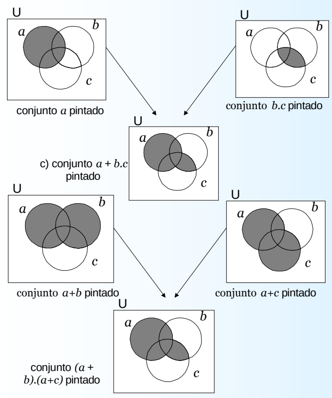


Fig. 3.3 Demonstração da distributividade (postulado 5, parte a))pelos diagramas de venn

3.2.1. Postulados

Postulado 6. Complementaridade

Para qualquer variável lógica existe o seu complementar que tem o valor oposto. Os dois juntos completam o conjunto universo K que contem todos os valores possíveis.

a € K, ā € K ∴ a.ā=0, a+ā=1 *(p6)*

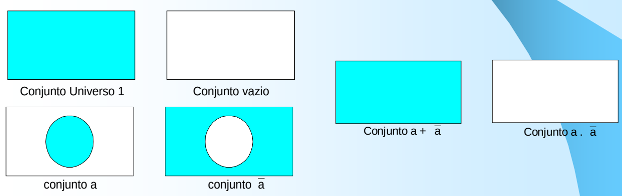


Fig. 3.4 Demonstração da complementaridade (postulado 6) pelos diagramas de venn

3.2.1. Postulados

Postulado 7. Igualdade

Existe um conjunto K com dois ou mais elementos sujeitos à equivalência = a qual satisfaz o princípio da substituição:

$$se \quad a=b \Rightarrow a+c=b+c \qquad (p7a)$$
$$se \quad a=b \Rightarrow a.c=b.c \qquad (p7b)$$

3.2.2. Teoremas

Teorema 1. Identidade ou Idepotencia

a+a = a *(T1a)*
a.a = a *(T1b)*

Teorema 2. Elementos absorventes

a+1 = 1 *(T2a)*
a.0 = 0 *(T2b)*

Em analogia à análise lógica podemos exemplificar assim:
a = kherl é homem(assuma-se que Kherl seja nome duma pessoa)
b = Os homens são mundanos
c = Os homens são extraterrestres

A proposição *a* pode assumir tanto o valor Verdadeiro como Falso posto que um nome pode ser dado a um homem ou a uma mulher. No entanto os homens são sempre mundanos, então *b* é sempre verdade. Por outro lado c é sempre falso

Parte a) Dizer *"Kherl é homem **ou** é mundano"* vai ser sempre verdade.
Parte b) Dizer *"Kherl é homem **e** é extraterrestre"* vai ser sempre falso.

3.2.2. Teoremas

Teorema 3. Dupla negação.

$a = \bar{\bar{a}}$ *(T3)*

Teorema 4. Absorção

a) $a + a.b = a$ (T4a)
b) $a (a + b) = a$ (T4b)

Teorema 5. *Irrelevancia*

a) $a + \bar{a}.b = a + b$ (T5a)

b) $a . (\bar{a} + b) = a.b$ (T5b)

3.2.2. Teoremas

Teorema 6.Adjacência

$a.b + \bar{a}.b = b$ *(T6a)*

$(a + b) . (\bar{a} + b) = b$ *(T6b)*

Teorema 7.

$ab + a.\bar{b}c = ab + ac$ *(T7a)*

$(a + b).(a + \bar{b} + c) = (a + b).(a + c)$ *(T7b)*

3.2.2. Teoremas

Teorema 8. Teorema de De Morgan

Para qualquer que sejam as variáveis lógicas *a, b, c, ...,*

$\overline{a+b+c+...} = \bar{a}.\bar{b}.\bar{c}. ...$ *(T8a)*

$\overline{a.b.c. ...} = \bar{a}+\bar{b}+\bar{c}+...$ *(T8b)*

Teorema 9. Consenso

a) $ab + \bar{a}c + bc = ab + \bar{a}c$ *(T9a)*

b) $(a + b).(\bar{a} + c).(b + c) = (a + b).(\bar{a} + c)$ *(T9b)*

3.2.2. Teoremas

Principio de Dualidade

O principio de dualidade estabelece que se uma expressão boolena é válida, é também válida a sua expressão dupla ou dual. Esta expressão é obtida pela substituição de todas as operações "+" por ".", todas as operações "." por "+", todos os 1's por 0's e todos os 0's por 1's.

Na verdade a parte b) de todos os lemas é a dupla da parte a)

EXEMPLO

Obter a expressão dual de *a+(b.c)=(a+b).(a+c).*

Com efeito, aplicando o principio de dualidade teremos que *a.(b+c)=(a.b)+(a.c).*

Atenção na colocação dos parênteses que não devem mudar de posição. Lembremos que diferentemente da álgebra linear, na álgebra de Boole a multiplicação não tem prioridade sobre a adição.

3.3. Funções e Portas Lógicas

3.3.1. Funções e Portas Lógicas Elementares

A teoria de comutação de circuitos, que assenta na álgebra de Boole, comporta algumas funções elementares, a partir das quais se desenvolve todas as outras.

As funções lógicas elementares são AND, OR, NOT e alguns autores ainda consideram a função Identidade.
Outras funções que se obtêm da agregação das funções lógica elementares são NAND, NOR, XOR e XNOR. São também consideradas elementares e possuem símbolos próprios.

Os dispositivos que na prática realizam as funções lógicas elementares são chamados de ***Portas Lógicas***. As mais importantes sao as electronicas

3.3.1. Funções e Portas Lógicas Elementares

Uma função lógica depende de uma ou várias proposições, ou seja pode depender duma ou várias variáveis lógicas.

A função lógica pode comportar *n(≥1)* variáveis lógicas relacionadas pelas duas operações booleanas. Como cada uma das variáveis só pode assumir um dos dois valores lógicos, a função lógica assumirá igualmente um dos dois valores

Convencionemos que:

Proposição	VVL	Valor Lógico
Interruptor Fechado	K	1
Interruptor Aberto	K	0
Luz Acesa	L	1
Luz Apagada	L	0

3.3.1. Funções e Portas Lógicas Elementares

Independentemente de quantos interruptores e como forem montados, eles só podem estar ou Abertos(K=0) ou Fechados(K=1).
Por sua vez, por mais complexo que seja o circuito estabelecido pelos interruptores, a Lâmpada ou está Apagada(L=0) ou Acesa(L=1)

O Estado 0 representa uma situação física determinada e contrária à do Estado 1. Aberto, Apagado, Sem tensão e Vazio representam o Estado 0 enquanto Fechado, Aceso, Em tensão e Cheio representam o Estado 1

O Valor Lógico 0 representa o valor assumido pela Variável Lógica que representa uma situação física correspondente ao Estado 0.

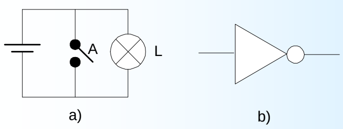
O Valor Lógico 1 representa o valor assumido pela Variável Lógica que representa uma situação física correspondente ao Estado 1.

3.3.1.1. Funções NOT ou NÃO

A função elementar mais simples é função NOT (ou NÃO) definida como:

$$L(A) = \bar{A} \qquad (3.1)$$

Ou seja, a função NOT realiza o complemento duma variável lógica. A função lógica NOT assume o valor lógico 1 sse o valor lógico da variável de entrada for igual a 0.



A	L
0	1
1	0

c)

Fig. 3.5. Função NOT. a) Representação esquemática. Se A está fechado, isto é A=1, L estará apagado, i.e., L=0 e vice versa. b) Símbolo convencional. c) Tabela de verdade

TABELA DE VERDADE

No esquema anterior foi introduzido um conceito muito importante: a Tabela de Verdade

Tabela de Verdade é um quadro no qual são colocadas, na coluna à esquerda, todas ou algumas combinações possíveis das variáveis independentes e na coluna à direita são colocados os resultados da função face à cada combinação.

Na linha do título são colocadas as variáveis lógicas tanto de entrada como de saída(funções lógicas). Por baixo de cada variável é colocado o valor lógico que ela pode assumir. Em cada linha é colocada uma combinação de todas as variáveis de entrada colocadas na linha do título. É colocado também o resultado da função

TABELA DE VERDADE

Variáveis de Entrada

Funções lógicas ou Variáveis de Saída

A B C D E F	f_1 f_2 f_3
X X X X X X	Y Y Y
X X X X X X	Y Y Y
X X X X X X	Y Y Y

Linha do título

Linhas de conteúdo

X e Y podem ser iguais a 0 ou a 1

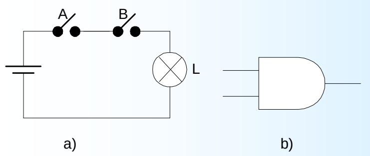
Fig. 3.6. Tabela de verdade

3.3.1.2. Função AND ou E

A função AND é definida como

$$L(A,B) = AB \qquad (3.3)$$

A função lógica AND assume o valor lógico 1 sse todas as variáveis de entrada assumirem o valor lógico 1.
Do esquema abaixo vê-se que a lâmpada só irá acender sse todas as chaves estiverem fechadas.



AB	L
00	0
01	0
10	0
11	1

c)

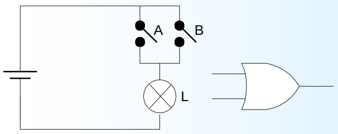
Fig. 3.7. Função AND. a) Representação esquemática. A lâmpada L estará acesa, i.e., L=1 sse as Chaves A e B estiverem fechados, i.e., A=1 e B=1. b) Símbolo convencional. c) Tabela de verdade

3.3.1.3. Função OR ou OU

A função OR é definida como

$$L(A,B) = A+B \qquad (3.4)$$

Para que a função lógica OR assuma o valor lógico 1 é bastante e suficiente que uma das variáveis de entrada assuma o valor lógico 1. De esquema vemos que basta qualquer uma das chaves fechar para haver passagem da corrente pela lâmpada.



AB	L
00	0
01	1
10	1
11	1

a) b) c)

Fig. 3.8 Função OR. a) Representação esquemática. A lâmpada L acende (L=1) bastará que qualquer uma das Chaves A ou B esteja fechada(A=1 ou B=1). b) Símbolo convencional. c) Tabela de verdade

3.3.2. Combinação de Funções Lógicas Elementares

Com as tres funções estudadas até aqui realizam-se virtualmente todos os restantes dispositivos digitais. No entanto foram definidas outras funções simples e que são de grande utilidade. Auxiliam na análise, síntese e implementação das funções booleanas (lógicas). Estas funções simples têm nomes específicos e símbolos próprios como as elementares.

As funções que iremos ver a seguir são também consideradas elementares, não obstante serem obtidas pela combinação das funções elementares vistos antes.

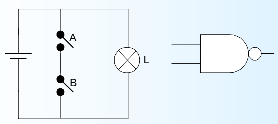
Esta posição encontra sustento no facto de que tecnicamente a porta mais elementar é a NAND da qual se obtém as restantes.
Algumas famílias de circuitos lógicos tem a NOR como porta elementar

3.3.2.1.Função NAND ou NE

A função NAND é uma agregação das funções AND e NOT. Nesta função realiza-se primeiro a operação AND e a seguir a NOT:

$$L(A,B) = \overline{AB} \qquad (3.5)$$

A função NAND inverte a função AND. Se nesta última é necessário que todas as chaves estejam fechadas para que a lâmpada acenda, na função NAND se as duas chaves estiverem fechadas a lâmpada apaga-se.



AB	L
00	1
01	1
10	1
11	0

a) b) c)

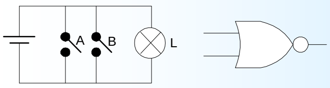
Fig. 3.9 Função NAND. a) Representação esquemática. A lâmpada L estará acesa, i.e., L=1 se qualquer uma das Chaves A ou B estiverem aberta, i.e., A=0 ou B=0. b) Símbolo convencional. c) Tabela de verdade .

3.3.2.2. Função NOR ou NOU

A função NOR é uma agregação das funções OR e NOT. Nesta função realiza-se primeiro a operação OR e de seguida NOT:

$$L(A,B) = \overline{A+B} \qquad (3.6)$$

A seguir mostra-se a implementação técnica da função NOR que no fundo inverte a função OR. Se nesta última bastava que pelo menos uma das chaves estivesse fechada para acender a lâmpada, desta vez a lâmpada acende sse ambas as chaves estiverem abertas.



AB	L
00	1
01	0
10	0
11	0

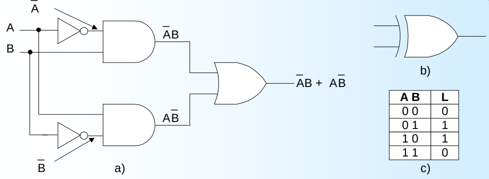
a) b) c)

Fig. 3.10 Função NOR. a) Representação esquemática. A lâmpada L estará acesa, i.e., L=1 sse as Chaves A e B estiverem aberta, i.e., A=0 e B=0. b) Símbolo convencional. c) Tabela de verdade

3.3.2.3. Função Exclusive-OR, XOR ou OU EXCLUSIVO

Por vezes precisamos de comparar informações. A forma mais simples é comparar bit a bit essa informação. A função XOR e XNOR são as mais indicadas. A função XOR detecta a diferença entre duas variáveis à entrada. Então a função XOR assume o valor lógico 1 sempre que forem diferentes as variáveis à entrada e é definida como:

$$L(A,B) = \overline{A}B + A\overline{B} = A \oplus B \qquad (3.7)$$



A B	L
0 0	0
0 1	1
1 0	1
1 1	0

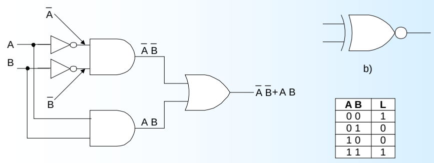
a) b) c)

Fig. 3.11 Função XOR. a) Implementação. A saída do circuito é igual a 1 se as variáveis A e B forem diferentes. b) Símbolo convencional. c) Tabela de verdade

3.3.2.4. Função Exclusive-NOR, XNOR ou COINCIDÊNCIA

A última função elementar que vamos estudar é a função XNOR ou Coincidência. Esta função detecta a igualdade entre as variáveis de entrada. Realiza uma operação inversa da função XOR. Pode ser obtida pela simples inversão da saída da porta XOR. A função XNOR é definida como:

$$L(A,B) = \overline{A}.\overline{B} + A.B = A \otimes B \qquad (3.8)$$



A B	L
0 0	1
0 1	0
1 0	0
1 1	1

a) b) c)

Fig. 3.12 Função XNOR. a) Implementação. A saída do circuito é igual a 1 se as variáveis A e B forem iguais. b) Símbolo convencional. c) Tabela de verdade

3.3.3. Funções Lógicas Generalizadas

Vimos nas secções 3.3.1 e 2 as funções de uma e de duas variáveis. Nestas últimas as variáveis lógicas estão relacionada por uma única operação . ou +. As funções XOR e XNOR já são combinações das funções elementares relacionadas pelos dois operadores lógicos.

No estudo das funções lógicas na forma generalizada encontraremos funções que dependem de mais de duas variáveis a relacionarem-se simultaneamente pelas duas operações lógicas.

Suponha que uma função é definida pela combinação de três variáveis lógicas A, B e C de tal modo que ela assume o valor lógico 1 em certas combinações e 0 noutras combinações. Diz-se que esta função existe nas combinações em que é igual a 1 e não existe nas outras.

Generalizemos ainda mais, fazendo com que em algumas combinações (que representamos por x) não exista a obrigatoriedade de assumir o valor lógico 0 ou 1. Construímos a seguir a tabela de verdade para esta função.

3.3.3. Funções Lógicas Generalizadas

As combinações em que colocamos x no resultado da função são chamados "***don't care***", isto é, não importa o valor assumido pela função uma vez que isso não afecta o resultado esperado.

ABC	F(A,B,C)
000	X
001	0
010	0
x11	1
100	0
101	1

Tab. 3.3 Tabela de verdade duma função qualquer de três variáveis A, B e C

A situação de *don't care* pode suceder tanto na saída como na entrada.

1ª situação:

Suponha uma instalação eléctrica em que há corte de energia. Se a lâmpada L é função do interruptor K, não importa a posição do interruptor. L está sempre apagado. **É *don´t care* à entrada**

2ª situação:

É preciso luz para ler um livro. De dia e no jardim, não importa se lâmpada L (função de K) está acesa ou não. **É *don´t care* à saída**

3.3.3. Funções Lógicas Generalizadas

A tabela de verdade mostra em cada linha uma das 2^n combinações binárias possíveis das *n* variáveis de entrada. Indica-se na coluna correspondente a F o valor lógico que a função assume quando as variáveis de entrada assumem os valores lógicos de cada linha.

Numa tabela de verdade cada linha representa o estado em que as variáveis de entrada assumem em simultâneo determinados valores. Decorre disto que a operação lógica que as relaciona é o produto lógico ou disjunção (Λ).

Por exemplo a função F(A,B,C) em análise assume o valor lógico 1 quando B=1ΛC=1 sem importar A.

3.3.3. Funções Lógicas Generalizadas

Mas esta não é a única combinação em que a função existe. Ela existe nesta OU noutra combinações mais abaixo. Então a operação lógica que unirá todas as combinações em que a função existe será a soma lógica ou conjunção (V). Assim:

$$F(A,B,C)= B\ C + C\ \bar{B}\ A \qquad (3.9)$$

A expressão final acima indica que a função existe nos casos:

B=1ΛC=1 V A=1ΛB=0ΛC=1.

Para que a função assuma o valor lógico 1, as variáveis devem entrar com o valor lógico 0 nas posições em que estão negadas. Com efeito, se B entrar com 1 no segundo termo produto, o resultado será 0 pois o Teorema 3 mostra claramente que a inversão de 1 resulta em 0 e o Teorema 2 mostra que 0 é absorvente na multiplicação.

3.3.3. Funções Lógicas Generalizadas

EXEMPLO

O circuito a seguir comporta três interruptores A, B e C. Assume-se que o interruptor pressionado assume o valor lógico 1 e de contrário o 0.

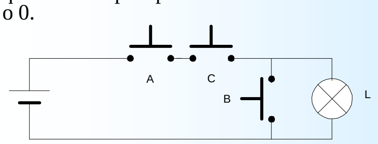


Fig. 3.13 Função generalizada L(A,B,C)

Fica evidente que para acender a lâmpada teremos: B não pressionado(B=0) ^ A ^ C terão que ser pressionados (A=C=1).

3.3.4. Funções Lógicas na Forma Canónica

A expressão da função em (3.9) embora represente correctamente a situação real, diz-se não canónica ou não normalizada.

Uma função está na forma canónica se for representada através de termos *Normalizados* do mesmo tipo.

Termo Normalizado é um termo normal em que aparecem todas as variáveis da função e na mesma ordem que no argumento desta.

Para normalizar a função (3.9) comecemos por organizar o último termo. Uma vez que o produto lógico goza da comutativa, nada nos impede de escrever o termo assim:

$$A\ \bar{B}\ C$$

Resolvemos o problema da desordem dos literais. Mas ainda temos um termo em que lhe falta um literal.

3.3.4. Funções Lógicas na Forma Canónica

Vamos recorrer aos postulados para incorporarmos o literal A sem mudar o valor da função usando transformações idênticas.

Se multiplicarmos o termo BC por 1 nada muda – postulado 2

$$BC = BC.1$$

Para obter 1 usando o literal A vamos fazer A + Ā – postulado 6

$$BC = BC.1 = BC.(A + \bar{A}) =$$
$$= BCA + BC\bar{A} \text{, postulado 5}$$

Agora resta organizar os termos para $ABC + \bar{A}BC$ e levando para (3.9) temos F(A,B,C) na forma canónica:

$$F(A,B,C) = ABC + \bar{A}BC + A\bar{B}C \qquad (3.10)$$

Os termos da função em (3.10) estão normalizados.
Um termo produto normalizado chama-se ***Termo mínimo.***
Um termo soma normalizado chama-se ***Termo Máximo.***

3.3.4. Funções Lógicas na Forma Canónica
Soma de Produtos Canónicos

A função (3.10), que foi representada através de soma de termos mínimos, diz-se que é uma ***soma de produtos canónicos***

Para simplificar a anotação, os termos mínimos são representados por números binários de *n* bits, onde *n* é a quantidade de variáveis no argumento da função.

Cada bit assume o valor lógico 0 ou 1 conforme se a variável correspondente está na forma negada ou na afirmativa. Quer isto dizer que o termo mínimo é sempre igual a 1

Termo mínimo	Código binário	Representação
$\bar{A}BC$	011	m_3
$A\bar{B}C$	101	m_5
ABC	111	m_7

Tab. 3.4 Codificação dos termos mínimos

3.3.4. Funções Lógicas na Forma Canónica
Soma de Produtos Canónicos

Com base na codificação apresentada na tabela 3.4, a função (3.10) pode ser escrita assim:

$$f(A,B,C) = m_3 + m_5 + m_7 \qquad (3.11)$$

A tabela de verdade de F mostrou que o primeiro termo resulta em X à saída. Os termos mínimo para os quais a função é *don´t care* são representados por d_i e podem ser incorporados na soma em (3.11) resultando:

$$f(A,B,C) = m_3 + m_5 + m_7 + d_0 = \sum m(3,5,7) + d(0) \qquad (3.12)$$

CONCLUSÃO:
Duma forma geral qualquer função lógica F(A,B,C,...) pode ser expressa como soma de termos mínimos ou Soma de Produtos Canónicos(SPC):

$$f(A,B,C,...) = \sum m_i + d_j \qquad (3.13)$$

3.3.4. Funções Lógicas na Forma Canónica
Soma de Produtos Canónicos

IMPORTANTE:
1. Os *don´t care* à entrada não entram na função se ela não assume o valor 1 neles. Por outro lado, se entram, são considerado $\boldsymbol{m_i}$ e não $\boldsymbol{d_i}$. É que a expressão de F quer dizer que ela assume o valor lógico 1 quando as variáveis de entrada são combinadas como apresentadas no lado direito do sinal de igualdade.

2. A ordem das variáveis nos termos mínimos é importante, uma vez que a codificação em binário nos diz apenas que a primeira variável assume o valor 0/1, a segunda o 0/1, a terceira idem e assim por diante. Decorre disto que para a função F(A,B,C), o termo mínimo m_2 quer dizer a primeira variável do argumento (A) é igual a 0; a segunda (B) é igual a 1 e a última (C) é igual a 0

3.3.4. Funções Lógicas na Forma Canónica
Produto de Somas Canónicas

Analogamente podemos apresentar a função (3.9) na forma de produto de somas canónicas.

Primeiro temos que converter os termos produtos em termos somas normalizados que são designados por ***Termo Máximo.***

Para criarmos termos somas relacionados por produto, aplicamos a propriedade distributiva da soma em relação ao produto

$$F(A,B,C) = BC + C\bar{B}A = (BC + C)(BC + \bar{B})(BC + A)$$
$$= C(\bar{B} + C)(B + A)(C + A)$$

Ordenando conforme o argumento

$$= C(\bar{B} + C)(A + B)(A + C)$$

3.3.4. Funções Lógicas na Forma Canónica
Produto de Somas Canónicas

Já temos termos somas unidos por produto. Mas não são normalizados ainda. Para normalizá-los temos que incorporar, sem afectar a função, os literais em falta em cada um. No primeiro falta A e B, no segundo falta A, no terceiro falta C e no último falta B.

Como o 0 é neutro na soma, vamos ardilosamente inseri-lo em cada so$\underline{m}$a.

$$=C(\bar{B}+C)(A+B)(A+C)=$$
$$=(\bar{A}A+C)(\bar{A}A+\bar{B}+C)(A+B+\bar{C}C)(A+\bar{B}B+C)=$$
$$=(\bar{A}+C)(A+C)(\bar{A}+\bar{B}+C)(A+\bar{B}+C)(A+B+\bar{C})(A+B+C)\cancel{(A+\bar{B}+C)}\cancel{(A+B+C)}=$$
$$=(\bar{A}+\bar{B}B+C)(A+\bar{B}B+C)(\bar{A}+\bar{B}+C)(A+\bar{B}+C)(A+B+\bar{C})(A+B+C)=$$
$$=(\bar{A}+\bar{B}+C)(\bar{A}+B+C)(A+\bar{B}+C)(A+B+C)\cancel{(\bar{A}+\bar{B}+C)}\cancel{(A+\bar{B}+C)}(A+B+\bar{C})\cancel{(A+B+C)}=$$
$$=(\bar{A}+\bar{B}+C)(\bar{A}+B+C)(A+\bar{B}+C)(A+B+C)(A+B+\bar{C})$$

3.3.4. Funções Lógicas na Forma Canónica

Produto de Somas Canónicas

Está convencionado que o termo máximo é igual a 0. Quer isto dizer que o literal na forma negada entra no termo com o valor lógico 1 e na forma afirmativa entra com o valor lógico 0

$$F(A,B,C)=(\bar{A}+\bar{B}+C)\ (\bar{A}+B+C)\ (A+\bar{B}+C)\ (A+B+C)(A+B+\bar{C}) \qquad (3.14)$$

(1 1 0)(1 0 0)(0 1 0)(0 0 0)(0 0 1)

Termo Máximo	Código binário	Representação
$A+B+C$	0 0 0	M_0
$A+B+\bar{C}$	0 0 1	M_1
$A+\bar{B}+C$	0 1 0	M_2
$\bar{A}+B+C$	1 0 0	M_4
$\bar{A}+\bar{B}+C$	1 1 0	M_6

Tab. 3.5 Codificação dos termos máximos

3.3.4. Funções Lógicas na Forma Canónica

Produto de Somas Canónicas

Com base na codificação apresentada na Tab. 3.5, a função (3.14) pode ser escrita assim:

$$F(A,B,C) = M_0 \,.\, M_1 \,.\, M_2 . M_4 . M_6 = \prod M(0,1,2,4,6) \qquad (3.15)$$

CONCLUSÃO:
Duma forma geral qualquer função lógica F(A,B,C,...) pode ser expressa como produto de termos máximos ou Produto de Somas Canónicas(PSC):

$$F(A,B,C,...) = \prod M_i * D_j \qquad (3.16)$$

3.3.4. Funções Lógicas na Forma Canónica

Observe-se que, por exemplo,

$$\overline{m1} = \overline{\bar{A}\,\bar{B}\,C} = A+B+\bar{C}=M1$$

Ou seja $\overline{mi} = Mi$

Então, $\bar{f} \equiv F$ *(3.17)*

A fórmula (3.17) mostra que representar uma função como SPC ou PSC é equivalente. Em qualquer dos casos a função é igual a 1 onde ela existe.

Se uma função é representada apenas como soma de termos mínimos ou apenas como produto de termos máximos, ela diz-se estar ***normalizada.***

3.3.4. Funções Lógicas na Forma Canónica

A função apresentada como Soma de Termos Mínimos assume o valor lógico 1 quando ocorre a combinação das variáveis de entrada tal que o Termo Mínimo seja igual a 1.

A função apresentada como Produto de Termos Soma assume o valor lógico 0 quando ocorre a combinação das variáveis de entrada tal que o Termo Máximo seja igual a 0.

Para que isso ocorra é necessário que a variável lógica entre no termo com o valor lógico segundo a tabela 3.6

Forma do Literal	Valor lógico com que entra	
	No termo mínimo	No Termo Máximo
Afirmativa – A	1	0
Negada - $\bar{A}$	0	1

Tab. 3.6 Relação entre a as literais e seus valores lógicos nos termos mínimos e termos máximos

3.4 Minimização ou simplificação de funções lógicas

3.4. Minimização de funcoes lógicas

Utilizando os conceitos de álgebra de Boole podemos analisar e implementar funções lógicas. É preciso notar que a cada função lógica corresponde um circuito lógico que consome portas lógicas e, consequentemente, espaço, potência e outros recursos. Sabendo disto, é evidente que é necessário encontrar mecanismos de torna os circuitos mais simples. Existem vários métodos para atingir esse fim, a saber: .

- Método algébrico
- Método de diagramas
- Método de tabelas

A sua aplicação depende da complexidade das situações como veremos a seguir

3.4.1 Minimização pelo método algébrico

Quando estamos em presença de funções de 2 a 3 variáveis a aplicação conveniente dos postulados e lemas pode ajudar a encontrar as expressões mais simples. Atente-se ao teorema 6 e aos restantes. Verifica-se que o membro direito é sempre mais simples que o esquerdo. Assim, aplicando-se sistematicamente estes lemas podemos chegar à expressões mais simples

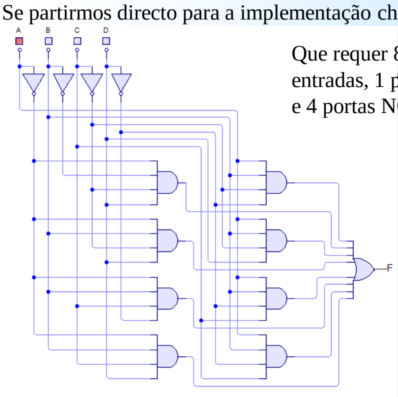
EXEMPLO

Seja dada a função lógica:

$$F(A,B,C,D)=\overline{A}\,\overline{B}\,\overline{C}D+\overline{A}B\overline{C}D+\overline{A}BC\overline{D}+\overline{A}BCD+AB\overline{C}\,\overline{D}+AB\overline{C}D+ABC\overline{D}+ABCD$$

3.4.1 Minimização pelo método algébrico

Se partirmos directo para a implementação chegamos ao circuito:



Que requer 8 portas AND de 4 entradas, 1 porta OR de 8 entradas e 4 portas NOT

3.4.1 Minimização pelo método algébrico

Podemos reorganizar os termos e, aplicando sistematicamente o teorema 6, computarmos como se segue:

$$F(A,B,C,D)=\overline{A}\,\overline{C}D\left(B+\overline{B}\right)+\overline{A}BC\left(D+\overline{D}\right)+AB\overline{C}\left(D+\overline{D}\right)+ABC\left(D+\overline{D}\right)$$

Usando a complementaridade temos:

$$=\overline{A}\,\overline{C}D+\overline{A}BC+AB\overline{C}+ABC$$

Na expressão acima vemos que há 2 termos que diferem de termo ABC apenas num literal. Podemos usar o teorema 6 de novo. Mas se o fizermos linearmente, um dos termos fica isolado. Com recurso a identidade(ou idepotencia), podemos escrever a expressao assim:

$$=\overline{A}\,\overline{C}D+\overline{A}BC+ABC+AB\overline{C}+ABC=$$

3.4.1 Minimização pelo método algébrico

E aplicando sistematicamente o teorema da adjacência segue:

$$=\overline{A}\,\overline{C}D+(\overline{A}+A)BC+AB(\overline{C}+C)=$$

$$=\overline{A}\,\overline{C}D+BC+AB \qquad (3.18)$$

Que é uma expressão bem mais simples que a inicial. Para implementarmos esta expressão precisaremos de:

3 portas AND para realizarem cada parcela;
1 porta OR para somar as parcelas e
2 portas NOT para realizarem o complemento de A e C.

A Figura seguinte ilustra a implementação do circuito da expressão (3.18)

3.4.1 Minimização pelo método algébrico

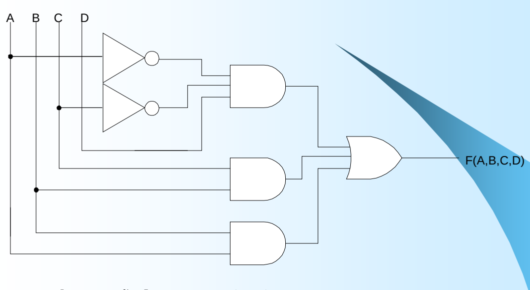


Fig. 3.14. Implementação da expressao (3.18)

3.4.1 Minimização pelo método algébrico

Vantagens do método:

1.Permite obter expressões mais simples de implementar

2.Rapidez de execução para poucas variáveis

Desvantagens do método:

1.Nem sempre é evidente a possível simplificação. Requer destreza na identificação do lema a aplicar em cada situação

2.Para um número de variáveis superior a 3 e expressões que contenham muitos termos, não é fácil computar algebricamente.

3.Consumo elevado de tempo no processo de simplificação para muitas variáveis

3.4.2 Minimização pelo método gráfico

Na minimização pelo método algébrico a dificuldade aumenta à medida que a quantidade de variáveis e de termos aumenta. Vimos nas desvantagens do método que requer alguma destreza além de tempo.

Para colmatar essas desvantagens existem vários métodos que usam diagrama. O mais difundido é o dos Diagrama de Veitch-Karnaugh .

3.4.2.1. MAPA DE VEITCH-KARNAUGHT

Diagrama de V-K é um mapa composto por várias linhas e várias colunas que se intersectam criando células. Para um conjunto de *n* variáveis, o mapa cria 2^n células. Cada célula corresponde uma das 2^n combinações possíveis .

Existem mapas de 1, 2 3, ... variáveis. A regra na criação dos diagramas é que na passagem duma célula para outra adjacente apenas uma variável muda de estado. A Fig 3.15 mostra mapas de 1 a 5 variáveis.

3.4.2.1 Mapas de Veitch-Karnaught

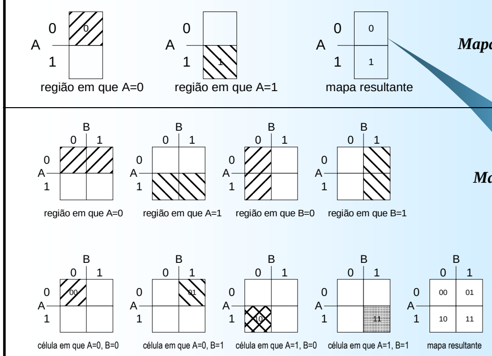


Fig. 3.15. Formação dos Mapas de V-K de varias variaveis

3.4.2.1 Mapas de Veitch-Karnaught

Mapa de 3 variaveis

A \ BC	00	01	11	10
0	000	001	011	010
1	100	101	111	110

Mapa de 4 variaveis

AB \ CD	00	01	11	10
00	0000	0001	0011	0010
01	0100	0101	0111	0110
11	1100	1101	1111	1110
10	1000	1001	1011	1010

Fig. 3.15a. Formação dos Mapas de V-K de varias variaveis

3.4.2.1 Mapas de Veitch-Karnaught

Mapa de 5 variaveis

ABC \ DE	00	01	11	10
000	00000	00001	00011	00010
001	00100	00101	00111	00110
011	01100	01101	01111	01110
010	01000	01001	01011	01010
110	11000	11001	11011	11010
111	11100	11101	11111	11110
101	10100	10101	10111	10110
100	10000	10001	10011	10010

Fig. 3.15a. Formação dos Mapas de V-K de varias variaveis

3.4.2.1 Mapas de Veitch-Karnaught

O Mapa de V-K, tal como a tbv, representa completamente uma função lógica

Lembremos que cada célula representa uma das combinações possiveis das variáveis. Nesta célula colocamos o valor que a função assume nessa combinação:

a) Valor 1 para representar um termo mínimo
b) Valor 0 para representar um termo máximo

EXEMPLO
Preencher o mapa de V-K para a função (3.11)

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

Isto é, Se A=B=C=1, F(A,B,C)=1!

Fig. 3.16

3.4.2.2 Minimização pelos Mapas de V-K

A simplificação pelos diagramas de V-K faz-se por agrupamento dum número par de células adjacentes com 1's se a função lógica original estiver expressa em soma de termos produto. Nesta reunião não se inclui 0.

Faz-se por agrupamento um número par de células adjacentes com 0's se a função lógica original estiver expressa como produto de termos soma. Nesta reunião não se inclui 1

Por cada agrupamento que se realiza resulta um novo termo produto(soma) no qual é eliminado a variável que mudou de estado, na passagem duma célula para outra, o que na verdade corresponde a aplicação do teorema 6

Uma célula pode ser usada em vários agrupamentos. O que corresponde à idepontência.

3.4.2.2 Minimização pelos Mapas de V-K

EXEMPLO:
Simplificar a função (3.11) pelo MV-K

Para isso vamos preencher o MV-K correpondente. Depois agrupamos todos os pares de 1´s adjacentes. Reutilizamos o termo ABC para agrupá-lo ao termo superior e ao do seu lado esquerdo.

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

Agrupamento 1

Agrupamento 2

Fig. 3.17

O agrupamento 1 gera o termo BC porque A mudou de valor
e
o agrupamento 2 gera o termo AC porque B mudou de valor

3.4.2.2 Minimização pelos Mapas de V-K

Alguns exemplos de agrupamentos

AB \ CD	00	01	11	10
00	1	0	1	0
01	0	1	0	0
11	1	0	0	1
10	1	0	0	1

a) Não permitido

$= \overline{A}\overline{B}\overline{D}$

AB \ CD	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

b) Permitido

$= A$

Fig. 3.18

3.4.2.2 Minimização pelos Mapas de V-K

Alguns exemplos de agrupamentos

AB \ CD	00	01	11	10
00	1	1	1	0
01	0	1	0	0
11	1	0	0	1
10	1	0	0	1

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

AB \ CD	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	0	0	0	0
10	1	0	0	1

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

Fig. 3.18a

3.4.2.2 Minimização pelos Mapas de V-K

Vantagens
a) É rápido obter as expressões mais simples de implementar
b) É conclusivo

Desvantagens
a) Para um número de variáveis superior a 4 os mapas são grandes e complexos
b) É apenas prático para computação manual.