Lógica e Sistemas Digitais - 2

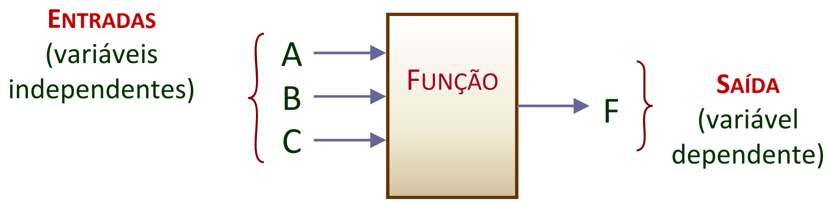
Representação e Minimização de Funções

ISEL
Departamento de Engenharia de Electrónica
e Telecomunicações e de Computadores
Lisboa

Mário Araújo

2016-1

Representação Tabular de uma Função



Representação de uma função F (A, B, C)
de 3 variáveis em diagrama de blocos.

Ordenação	C	B	A	F
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

Representação tabular da função F.

A representação mais elementar duma função lógica é a tabela de verdade, que indica o valor da função para cada combinação de entradas possível.

A tabela de verdade duma função de n-variáveis possui 2^n linhas.

Enumeram-se as linhas pelos equivalentes decimais da configuração de variáveis em cada linha.

A numeração só faz sentido depois de atribuídos pesos às variáveis. A variável **C** mais à esquerda na tabela é a que tem maior peso (4), e a variável **A** mais à direita a que tem o menor peso (1). É esta a regra seguida doravante.

A numeração das linhas não faz parte da tabela.

Directamente da tabela de verdade (sem introduzir simplificação), podem extrair-se dois tipos de expressões booleanas da função:

- extracção pelos **uns** (**1s**) para se construir uma soma de produtos canónica (**soma de mintermos**, forma **AND-OR**);
- extracção pelos **zeros** (**0s**) para se construir um produto de somas canónico (**produto de maxtermos**, forma **OR-AND**).

Representação Algébrica de uma Função em Soma de Mintermos

Na extracção da expressão algébrica pelos 1s constrói-se uma **soma dos mintermos** que correspondem às linhas da tabela de verdade para as quais a função é 1.

Um **mintermo** (produto canónico, implicante canónico ou termo minimal) representa uma combinação das variáveis binárias na tabela de verdade da função.

Cada mintermo é designado por m_i em que o índice **i** indica o número decimal equivalente à combinação binária por ele representada (número da respectiva linha da tabela). Uma função de n variáveis tem 2^n mintermos.

Num mintermo m_i todas as variáveis aparecem exactamente uma vez, complementadas ou não: uma variável é complementada se o bit correspondente na representação binária de **i** for 0, caso contrário a variável é não-complementada.

Por exemplo o mintermo m_3 corresponde ao 1 da quarta linha da tabela e é representado por ABC' .

A expressão algébrica de F na forma de **soma de mintermos** , também designada **forma canónica AND-OR** ou **disjuntiva**, é então:

$$F(C, B, A) = ABC' + AB'C + A'BC + ABC$$

$$= m_3 + m_5 + m_6 + m_7 =$$

$$= \Sigma m\,(3, 5, 6, 7) \qquad \text{ou simplesmente}$$

$$= \Sigma\,(3, 5, 6, 7)$$

Representação algébrica de F em soma de **mintermos** – **forma canónica AND-OR** ou disjuntiva.

Mintermo	C	B	A	F	Expressão algébrica do mintermo
m_0	0	0	0	0	
m_1	0	0	1	0	
m_2	0	1	0	0	
m_3	0	1	1	1	A B C'
m_4	1	0	0	0	
m_5	1	0	1	1	A B' C
m_6	1	1	0	1	A' B C
m_7	1	1	1	1	A B C

Representação tabular da função F assinalando-se na coluna da direita a expressão algébrica dos mintermos para os quais a função é 1.

A notação Σ m (3, 5, 6, 7) identifica uma **lista de mintermos** e representa a soma dos mintermos 3, 5, 6 e 7 envolvendo as variáveis A, B e C.

À lista de mintermos também se dá o nome de **ON-SET** da função lógica.

A forma canónica AND-OR é única para uma dada função F, mas a sua expressão algébrica não é a mais simples possível para a função.

Representação Algébrica de uma Função em Produto de Maxtermos

Na extracção pelos 0s constrói-se um **produto dos maxtermos** que correspondem às linhas da tabela de verdade para as quais a função é 0.

Um **maxtermo** (ou termo maximal) representa uma combinação das variáveis binárias na tabela de verdade da função.

Os maxtermos podem ser numerados à semelhança do que se fez para os mintermos: cada maxtermo é designado por $\mathbf{M_i}$ em que o índice **i** indica o número decimal equivalente à combinação binária por ele representada (número da respectiva linha da tabela). Uma função de n variáveis tem 2^n maxtermos.

Quando se 'lêem' somas nas tabelas de verdade para construir um produto de maxtermos as variáveis que se mantêm em 0 são lidas na forma não negada, e as que se mantêm em 1 são lidas na forma negada.

Isto é de esperar porque a leitura pelos 0 equivale à leitura pelos 1s da função invertida F'. A expressão obtida para F' deve portanto ser novamente invertida para ser apresentada na forma normal, o que levará, pelas leis de De Morgan, à expressão algébrica de F na forma de **soma de maxtermos** , também designada **forma canónica OR-AND** ou **conjuntiva**::

$$F(C, B, A) = (A + B + C)(A' + B + C)(A + B' + C)(A + B + C')$$

$$= M_0 \cdot M_1 \cdot M_2 \cdot M_4 =$$

$$= \Pi M (0, 1, 2, 4) \quad \text{ou simplesmente}$$

$$= \Pi (0, 1, 2, 4)$$

Representação algébrica de F em produto de maxtermos – forma canónica **OR-AND** ou conjuntiva.

Maxtermo	C	B	A	F	Expressão algébrica do maxtermo
M_0	0	0	0	0	A + B + C
M_1	0	0	1	0	A'+ B + C
M_2	0	1	0	0	A + B'+ C
M_3	0	1	1	1	
M_4	1	0	0	0	A+ B + C'
M_5	1	0	1	1	
M_6	1	1	0	1	
M_7	1	1	1	1	

Representação tabular da função F assinalando-se a expressão algébrica dos maxtermos para os quais a função é 0.

A notação Π M (0 , 1, 2, 4) identifica uma **lista de maxtermos** e representa o produto dos maxtermos 0, 1, 2 e 4 envolvendo as variáveis A, B e C.

À lista de maxtermos também se dá o nome de **Off-set** da função lógica.

A forma canónica OR-AND é única para uma dada função F, mas a sua expressão algébrica não é a mais simples possível para a função.

TERMINOLOGIA

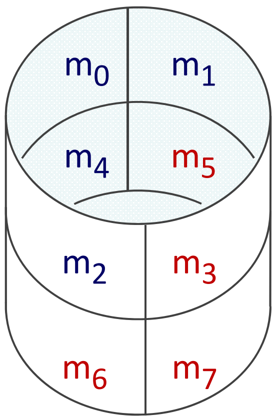
São utilizadas 5 representações possíveis de uma função lógica neste capítulo:

- Tabela de verdade.
- Mapa de Karnaugh.
- Soma canónica (forma canónica disjuntiva ou AND-OR).
- Lista de Mintermos.
- Produto canónico (forma canónica conjuntiva ou OR-AND e dual da soma canónica).
- Lista de Maxtermos (dual da Lista de Mintermos).

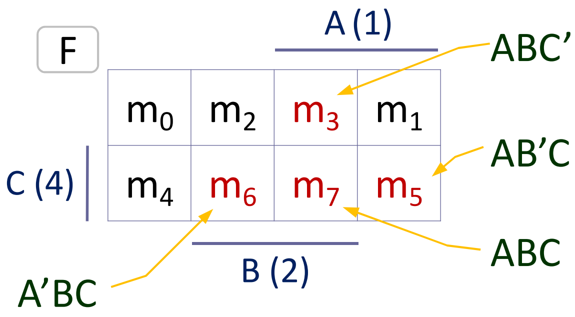
Na representação algébrica de funções salienta-se que uma função booleana tem: uma só forma canónica disjuntiva, uma só forma canónica conjuntiva, várias formas normais disjuntivas (somas de produto), várias formas normais conjuntivas (produtos de somas).

Terminologia:

- Constante: 0 (F) ou 1 (T).
- **Literal**: uma variável ou o seu complemento (negação): A e A' são literais.
- **Termo Produto** (ou **Termo de Produto**): AND de um ou mais literais como A, B'C, A'BC', ABC.
- **Termo Soma** (ou **Termo de Soma**): OR de um ou mais literais como A, B'+ C, A'+ B + C', A + B + C.
- Termo Normal: um termo produto ou soma em que nenhuma variável aparece mais que uma vez: se aparece uma variável, não aparece o seu complemento e vice-versa.
- **Mintermo** (ou termo minimal): um termo produto normal que engloba todas as variáveis independentes (entradas primárias).
- **Maxtermo** (ou termo maximal): um termo soma normal que inclui todas as variáveis independentes
- **Soma canónica**: soma de Mintermos.
- **Produto canónico**: produto de Maxtermos.
- **Soma de Produtos** (SOP): uma expressão lógica com a estrutura de uma soma (OR) de termos produto (AND) – forma normal disjuntiva.
- **Produto de somas** (POS): uma expressão lógica com a estrutura de um produto (AND) de termos soma (OR) – forma normal conjuntiva.



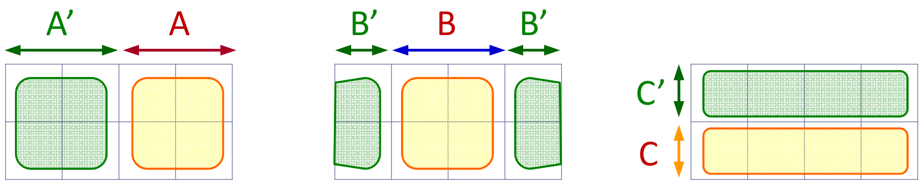
Visualização do mapa de Karnaugh a 3 dimensões.



Uma forma de desenhar o Mapa de Karnaugh genérico de uma função a 3 variáveis.

Maurice Karnaugh

1924 -
(New York – USA)



Os quadrados a amarelo situados por baixo do traço (a vermelho) de referência de cada variável correspondem às configurações em que ela assume o valor 1 na tabela de verdade. Os quadrados a verde correspondem às configurações em que ela variável assume o valor 0 na tabela de verdade.

O mapa de Karnaugh (já abordado no capítulo anterior), é uma reordenação da tabela de verdade de uma função em duas dimensões.

Cada quadrado corresponde a uma linha da tabela de verdade e inscreve-se nele o valor que a função toma para essa configuração: 0 ou 1. Os quadrados que só diferem numa variável ficam em quadrículas adjacentes.

Para uma função de 3 variáveis cada quadrado tem 3 quadrados **adjacentes**. Por exemplo: o quadrado correspondente ao termo m_5 tem como posições adjacentes os quadrados m_1, m_7 e m_4. Tudo se passa como se as posições laterais extremas no mapa estivessem encostadas, como mostra o desenho na perspectiva de 3 dimensões. Genericamente um termo com **n** literais tem **n** adjacentes possíveis.

*O diagrama de Veitch foi um embrião do que viria a ficar consagrado como mapa de Karnaugh (m-K) quando surgiu em 1952 num artigo de autoria do físico Edward Veitch intitulado **A Chart Method for Simplifying Truth Functions**.*

***Maurice Karnaugh**, um físico de formação que trabalhava nos laboratórios Bell, refinou o mapa de Veitch dando origem ao mapa de Karnaugh actual, revelando o método num artigo intitulado **The Map method for Synthesis of Combinational Logic Circuits** publicado nas **Transactions of the American Institute of Electrical Engineers** em Novembro de 1953.*

Simplificação de uma Função por Agrupamento de Mintermos em Mapa de Karnaugh

A representação de funções por 'mapas de Karnaugh' permite um método de simplificação de maior eficiência e sistematização. A reordenação das linhas da tabela facilita o agrupamento dos mintermos contendo os 1s (uns) da função e a extracção da expressão booleana com o menor número de variáveis nos termos, e com o menor número de termos.

Por exemplo, entre m_3 e m_7 só a variável C é que varia, o que permite pôr AB em evidência. De igual forma só B varia entre m_5 e m_7, e só A varia entre m_6 e m_7 o que permite as seguintes simplificações:

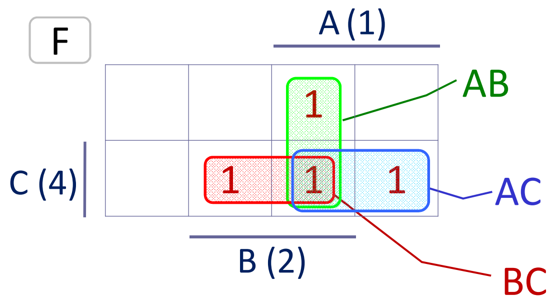
$$m_3 + m_7 = ABC' + ABC = AB\,(C' + C) = AB$$

$$m_5 + m_7 = AB'C + ABC = AC\,(B' + B) = AC$$

$$m_6 + m_7 = A'BC + ABC = BC\,(A' + A) = BC$$

$$
\begin{aligned}
F &= m_3 + m_5 + m_6 + m_7 = \\
&= \textstyle\sum m\,(3,5,6,7) = \\
&= \textstyle\sum (3, 5, 6, 7) = \\
&= ABC' + AB'C + A'BC + ABC \quad \leftarrow \text{forma canónica antes da minimização} \\
&= AB + AC + BC \quad \leftarrow \text{forma MSOP depois da minimização}
\end{aligned}
$$

Notações algébricas equivalentes de F.



$$F = AB + AC + BC$$

Simplificação da função F no mapa de Karnaugh por agrupamento de mintermos.

Resumindo: no mapa agrupam-se os 1s adjacentes, numa quantidade que seja potência inteira de 2, e na maior quantidade possível.

A expressão simplificada da função é obtida pela união dos grupos obtidos.

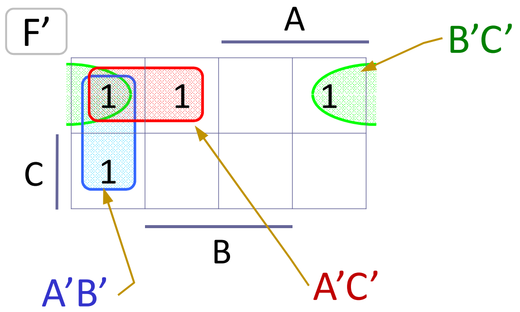
Um novo grupo pode utilizar 1s que já tenham servido noutros grupos.

A expressão obtida está na forma de soma de produtos minimizada (MSOP, de Minimized Sum of Products), ou forma AND-OR simplificada.

Simplificação de uma Função por Agrupamento de Maxtermos em Mapa de Karnaugh

Considera-se agora a função invertida de F, ou seja F'.

Fazendo a leitura de F' pelos uns (1s) do mapa de Karnaugh pelo método preconizado obtém-se a função F' minimizada:



$$F' = m_0 + m_1 + m_2 + m_4 =$$
$$= \sum m\,(0, 1, 2, 4) =$$
$$= A'B' + A'C' + B'C'$$

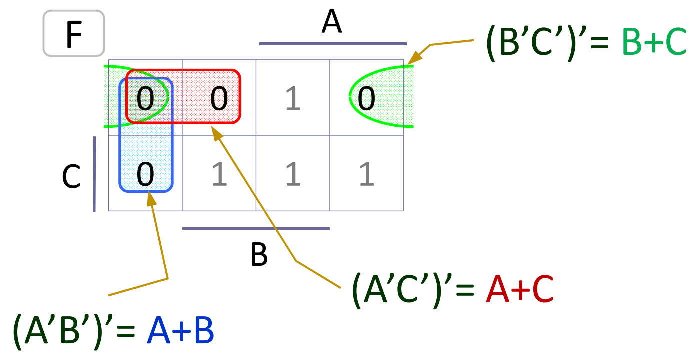
Minimização da função F' (complementar de F) no mapa de Karnaugh por agrupamento de mintermos.

Para a obtenção da função F a partir F' há que negar a sua expressão algébrica. Aplicando à expressão obtida as leis de Morgan obtém-se:

$$F = (F')' = \{ \Sigma\, m\,(0, 1, 2, 4) \}' = (\, m_0 + m_1 + m_2 + m_4\,)' =$$
$$= m_0' \cdot m_1' \cdot m_2' \cdot m_4' = M_0 \cdot M_1 \cdot M_2 \cdot M_4 = \prod M\,(0, 1, 2, 4)$$
$$= (A'B')'\,(A'C')'\,(B'C')' = (A+B)\,(A+C)\,(B+C)$$

A obtenção da função F' pelo agrupamento dos 1s corresponde à leitura e agrupamento dos 0s no mapa da função F.

Este método pode ser utilizado directamente no mapa como se mostra, e corresponde à minimização por agrupamento de maxtermos.



$$F = (A+B)\,(A+C)\,(B+C)$$

Minimização da função F no mapa de Karnaugh por agrupamento de maxtermos.

Representação Gráfica do Produto dos Maxtermos

F

C (4) \ A (1), B (2)				
	m_0	m_2	m_3	m_1
C (4)	m_4	m_6	m_7	m_5

$$F = \sum m\,(3, 5, 6, 7)$$

F'

C (4) \ A (1), B (2)				
	m_0	m_2	m_3	m_1
C (4)	m_4	m_6	m_7	m_5

$$F' = \sum m\,(0, 1, 2, 4)$$

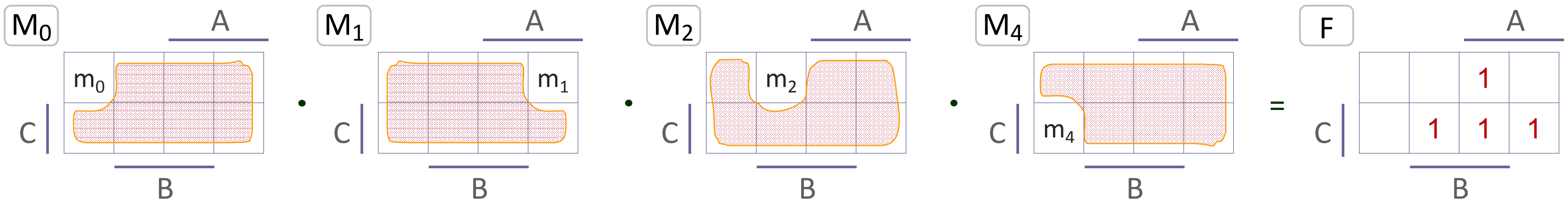
Um mintermo $\mathbf{m_i}$ corresponde a uma função ≠ 0 com o número mínimo de 1's na tabela de verdade.

Um maxtermo $\mathbf{M_i}$ corresponde a uma função ≠ 1 com o número máximo de 1's na tabela de verdade. Um mintermo e um maxtermo com o mesmo índice são complementos um do outro.

O produto lógico dos vários maxtermos permite obter o mapa de F como se mostra em baixo:

$$F = (F')' = (m_0 + m_1 + m_2 + m_4)' =$$

$$= m_0' \cdot m_1' \cdot m_2' \cdot m_4' = M_0 \cdot M_1 \cdot M_2 \cdot M_4 = \prod M\,(0, 1, 2, 4)$$



$M_0 = m_0' = (A'B'C')' = A + B + C$

$M_1 = m_1' = (AB'C')' = A' + B + C$

$M_2 = m_2' = (A'BC')' = A + B' + C$

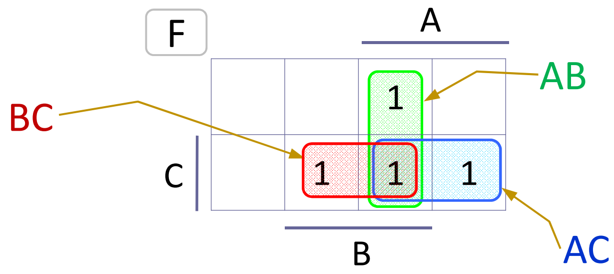
$M_4 = m_4' = (A'B'C)' = A + B + C'$

$F = M_0 \cdot M_1 \cdot M_2 \cdot M_4$

Representação do produto dos maxtermos intervenientes na forma OR-AND de F.

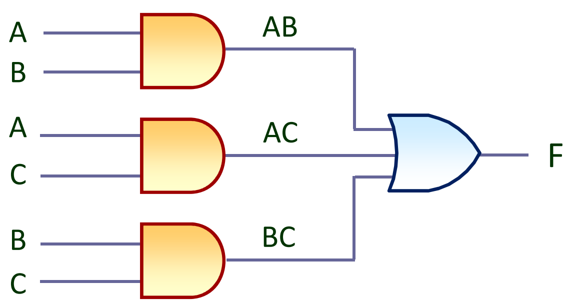
Minimização de Funções por Agrupamento de Mintermos e de Maxtermos

Representação da Função por Lista de Mintermos: $F = \sum m\,(3, 5, 6, 7)$



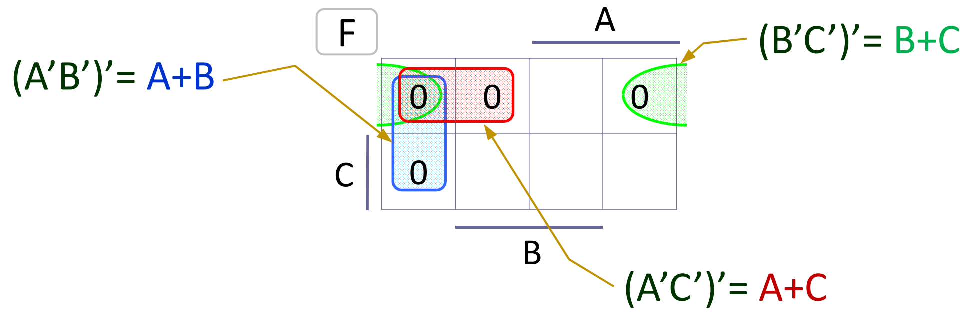
$F = AB + AC + BC$

Minimização da função F no mapa de Karnaugh por agrupamento de mintermos.



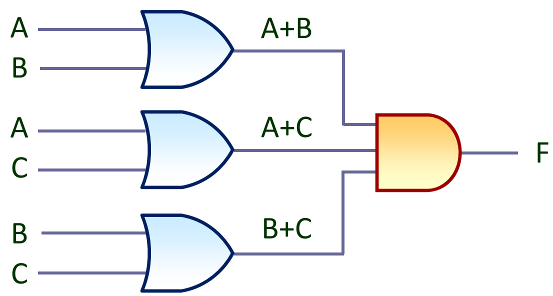
Implementação da função F por simplificação na forma AND-OR (**MSOP**).

Representação da Função por Lista de Maxtermos: $F = \prod M\,(0, 1, 2, 4)$



$F = (A+B)\,(A+C)\,(B+C)$

Minimização da função F no mapa de Karnaugh por agrupamento de maxtermos.



Esta forma é também designada por **MPOS**, do inglês **M**inimized **P**roduct-**O**f-**S**ums).

Implementação da função F por simplificação na forma OR-AND (**MPOS**).

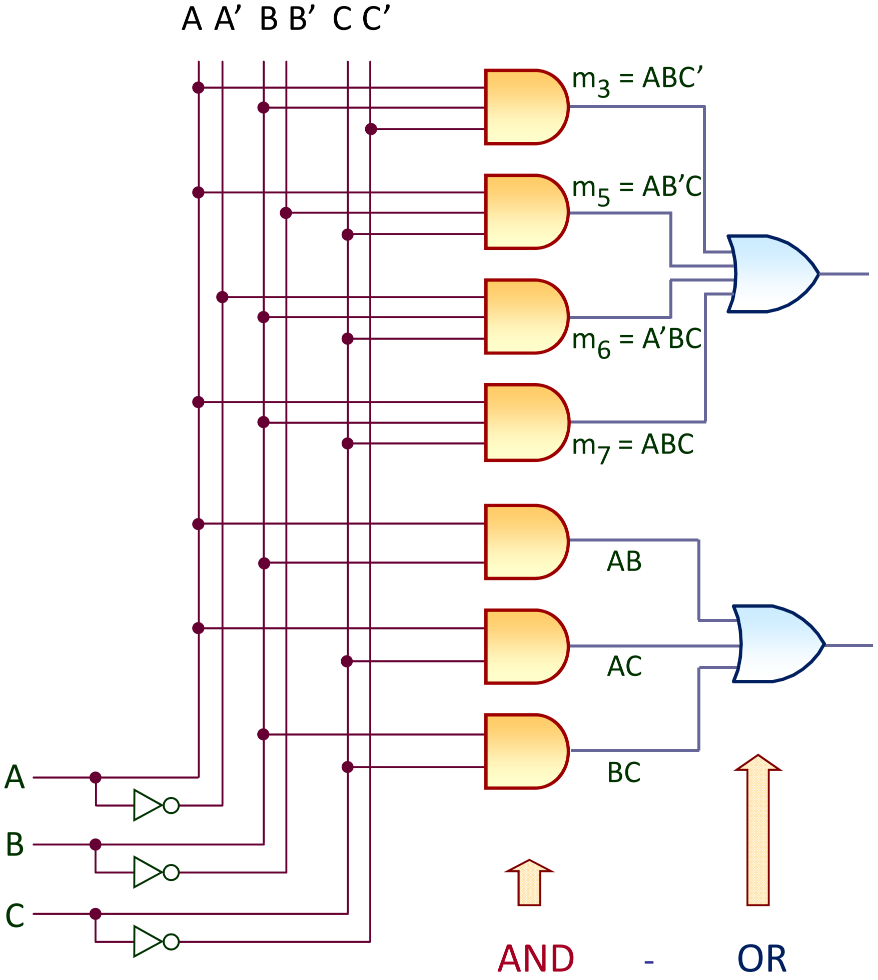
Formas AND-OR Canónica (SOP) e Minimizada (MSOP)

$F = m_3 + m_5 + m_6 + m_7 = \Sigma\, m\,(3, 5, 6, 7) = \Sigma\,(3, 5, 6, 7) =$

$= A\,B\,C' + A\,B'C + A'B\,C + A\,B\,C =$

$= A\,B + A\,C + B\,C$

Notações algébricas equivalentes de F.



F = ABC' + AB'C + A'BC + ABC

Representação de F na **forma canónica AND-OR** (Soma de Mintermos).
Esta forma é também designada por **SOP**, do inglês **S**um-**O**f-**P**roducts.

F = AB + AC + BC

Representação algébrica de F na forma AND-OR minimizada, não canónica.
(Soma de Termos Produto).
Esta forma é também designada por **MSOP**, do inglês **M**inimized **S**um-**O**f-**P**roducts.

Diagrama lógico de F configurado em duas formas AND-OR, uma canónica, outra não-canónica.

Instituto Superior de Engenharia de Lisboa

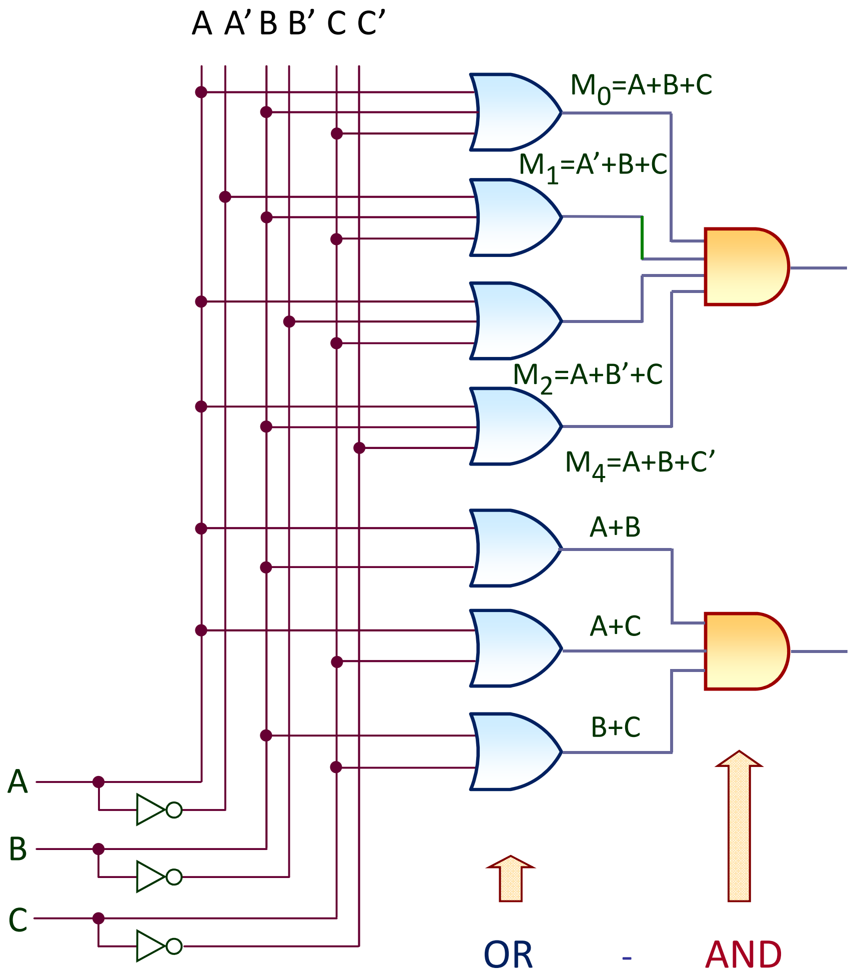
FORMAS OR-AND CANÓNICA (POS) E MINIMIZADA (MPOS)

$F = M_0 . M_1 . M_2 . M_4 = \Pi M(0, 1, 2, 4) = \Pi(0,1,2,4) =$

$= (A + B + C) \cdot (A' + B + C) \cdot (A + B' + C) \cdot (A + B + C') =$

$= (A + B) \cdot (A + C) \cdot (B + C)$

Notações algébricas equivalentes de F.



$F = (A + B + C) \cdot (A' + B + C) \cdot (A + B' + C) \cdot (A + B + C')$

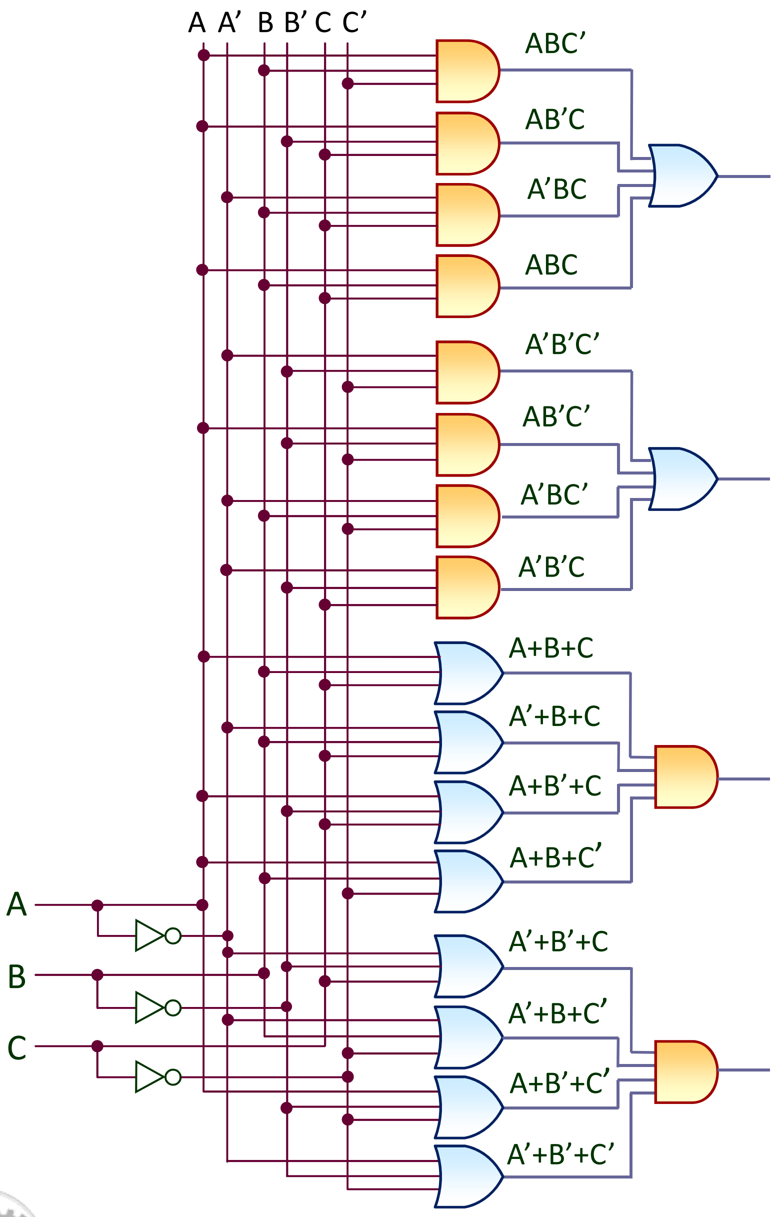
Representação de F na **forma canónica OR-AND** (PRODUTO DE MAXTERMOS).
Esta forma é também designada por **POS**, do inglês **P**RODUCT-**O**F-**S**UMS.

$F = (A + B) \cdot (A + C) \cdot (B + C)$

Representação algébrica de F na forma OR-AND minimizada**,** não canónica (PRODUTO DE TERMOS SOMA).
Esta forma é também designada por **MPOS**, do inglês **M**INIMIZED **P**RODUCT-**O**F-**S**UMS.

Diagrama lógico de F configurado em duas formas OR-AND, uma canónica, outra não-canónica.

Conversão entre Formas Canónicas



F = ABC' + AB'C + A'BC + ABC =

= $\sum$ m (3, 5, 6, 7)

F' = A'B'C' + AB'C' + A'BC' + A'B'C =

= $\sum$ m (0, 1, 2, 4)

Formas Canónicas AND-OR de F e F'. Soma de Mintermos.

F = (A+B+C)·(A'+B+C)·(A+B'+C)·(A+B+C') =

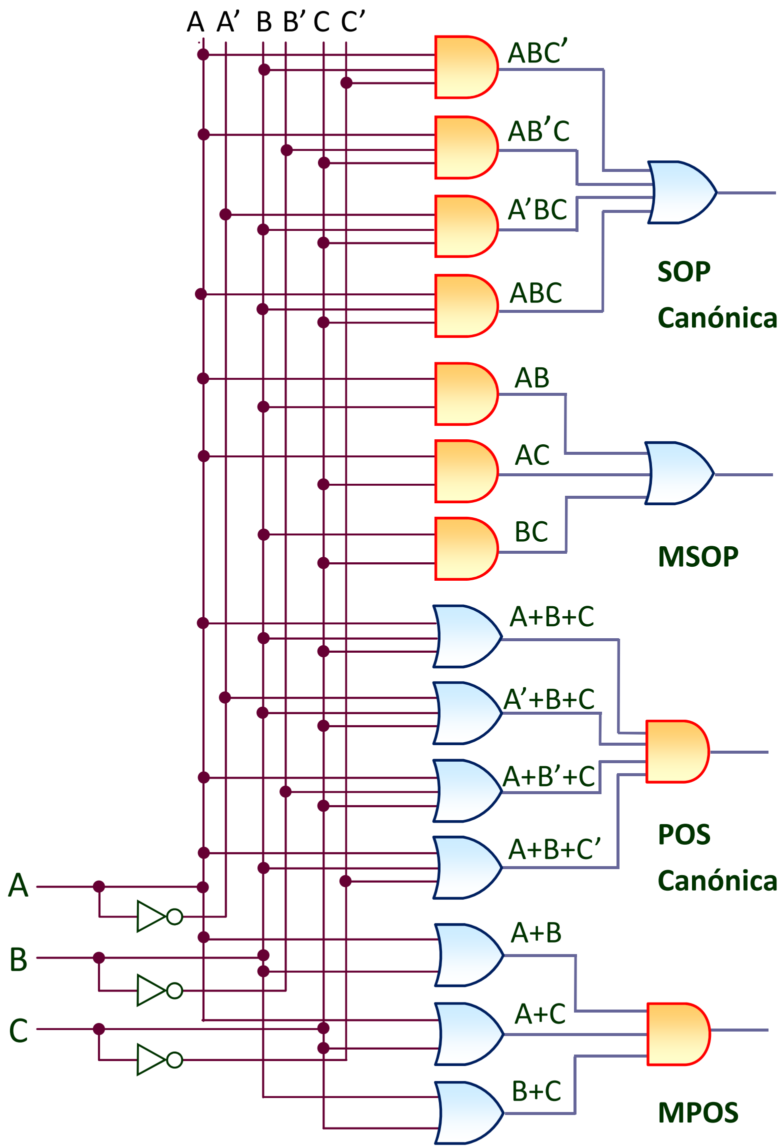
= Π M (0, 1, 2, 4)

F' =(A'+B'+C)·(A'+B+C')·(A+B'+C')·(A'+B'+C') =

= Π M (3, 5, 6, 7)

Formas Canónicas OR-AND de F e F'. Produtos de Maxtermos.

IMPLEMENTAÇÕES ALTERNATIVAS DE UMA FUNÇÃO



$F = ABC' + AB'C + A'BC + ABC =$

$= \sum m\,(3, 5, 6, 7)$

SOP – Forma Canónica AND-OR. SOMA DE PRODUTOS (MINTERMOS).

$F = AB + AC + BC$

MSOP – Soma de Produtos **Minimizada**.

$F = (A+B+C)\cdot(A'+B+C)\cdot(A+B'+C)\cdot(A+B+C') =$

$= \prod M\,(0, 1, 2, 4)$

POS – Forma Canónica OR-AND. PRODUTO DE SOMAS (MAXTERMOS).

$F = (A+B) \cdot (A+C) \cdot (B+C)$

MPOS – Produto de Somas **Minimizado**.

Diagrama lógico de 4 implementações equivalentes de uma função F: duas formas canónicas e duas formas minimizadas.

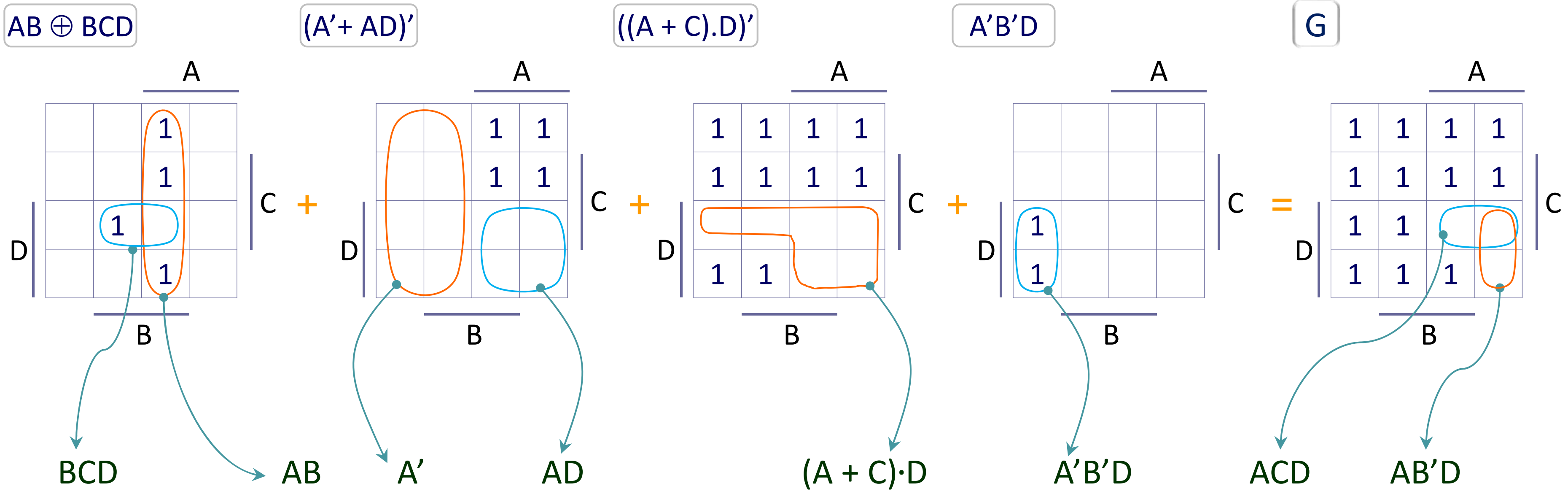
Simplificação Gráfica de uma Função na forma OR-AND (Ex. 2-1)

Exemplo 2-1

Objectivo

Obter a forma OR-AND simplificada da função G definida pela equação ao lado utilizando mapas de Karnaugh.

$$G = (AB \oplus BCD) + (A' + AD)' + ((A + C)\,D)' + A'B'D$$



Mapa de Karnaugh de G obtido pela soma de mapas-K parciais correspondentes aos termos da função G.

$$G' = ACD + AB'D$$

$$G = (ACD + AB'D)' = (A' + C' + D')\,(A' + B + D')$$

Obtenção de G por aplicação das Leis de De Morgan à equação algébrica de G'.

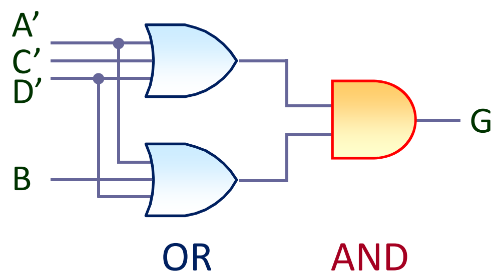


Diagrama lógico da função na forma OR-AND.

Resumindo, a MINIMIZAÇÃO LÓGICA significa:

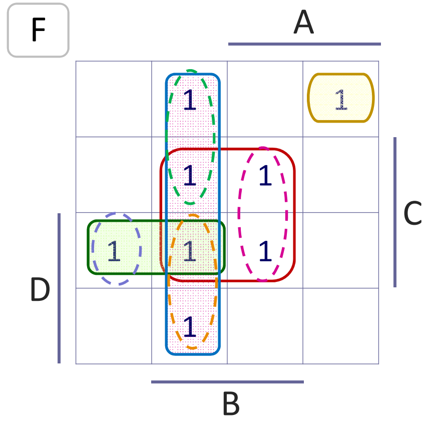
- Na forma AND-OR (SOP):

  a minimização do número de termos produto (redução do número de portas lógicas) e a minimização do número de literais (redução do número de entrada das portas lógicas).

- Na forma OR-AND (POS):

  a minimização do número de termos soma (redução do número de portas lógicas) e a minimização do número de literais (redução do número de entrada das portas lógicas).

Para aprofundar o método de minimização em SOP e POS pelo mapa de Karnaugh e delinear o algoritmo adequado ver-se-ão primeiro algumas definições.



Implicantes:
AB’; BC; A’CD; AB’C’D’; A’BC;
ABC; A’B’C’D’; etc.

Alguns implicantes de uma função F agrupados: cada 1 ou grupo de 1s (uns) que podem ser combinados são implicantes.

$$F = \sum m\,(1, 2, 6, 7, 10, 12, 14, 15) =$$
$$= A'B + BC + A'CD + A'B'C'D'$$

Para funções das mesmas variáveis, diz-se que uma função booleana simples X implica outra função Y (X⇒Y) se, para todos os valores de entrada em que a função X vale 1, a função Y também vale 1.

Quando X é um produto, diz-se ser um IMPLICANTE da função Y.

Um termo produto diz-se IMPLICANTE de uma função sse essa função assume 1 para todos os mintermos que constituem esse termo produto.

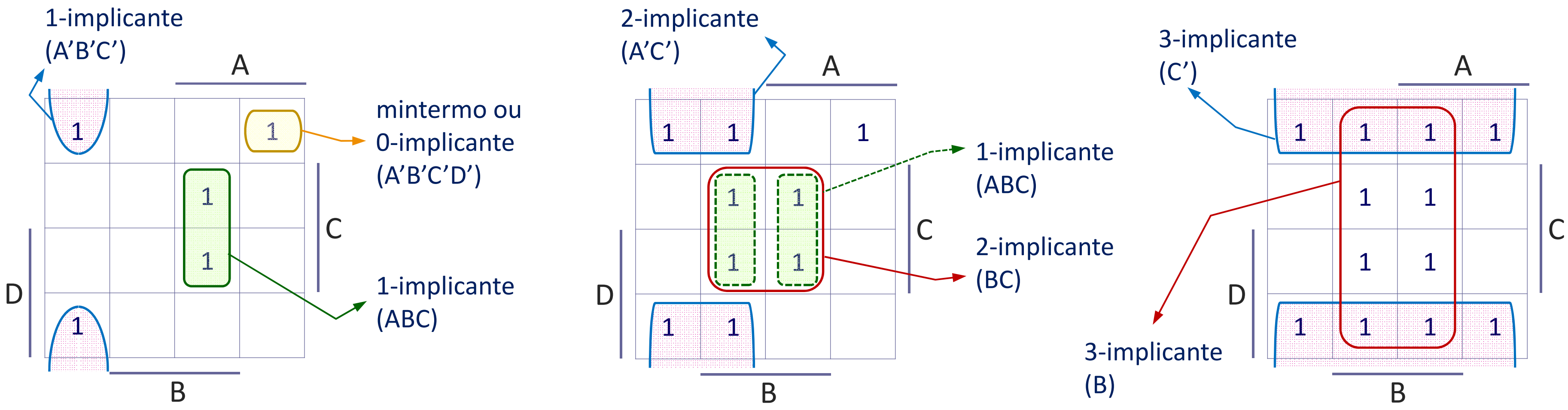
Diz-se que o termo produto de menor dimensão é IMPLICANTE de um termo produto de maior dimensão se todos os seus mintermos estão incluídos no termo produto maior.

Todos os mintermos de uma função F são IMPLICANTES dessa função.

Nos quadros de Karnaugh, os IMPLICANTES correspondem a associações válidas de 1s (cada 1 ou grupo de 1s que podem ser combinados).

No mapa indicam-se alguns implicantes da função F de 4 variáveis.

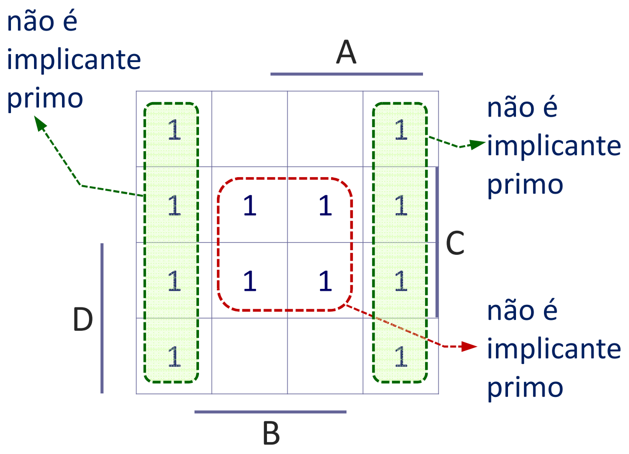
TERMO PRODUTO OU IMPLICANTE



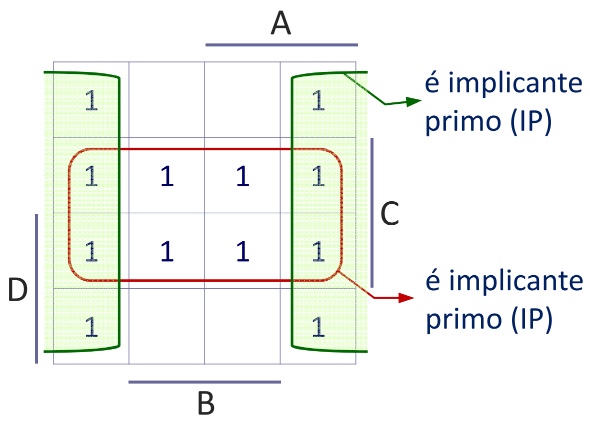
Vários tipos de implicantes de uma função de 4 variáveis: 0-implicantes, 1-implicantes, 2-implicantes, 3-implicantes. Implicantes maiores correspondentes ao menor produto.

- IMPLICANTES de uma função são quaisquer elementos do ON-SET (cada 1 ou grupos de 1s que podem ser combinados). Graficamente são quaisquer círculos válidos (não necessariamente os maiores possível).
- Um mintermo é um caso especial de implicante que tem presentes todas as variáveis da função (é o produto de maior dimensão). Um mintermo é também designado de 0-IMPLICANTE (também designado por 0-cubo).
- Todos os mintermos são implicantes, mas nem todos os implicantes são mintermos.
- Um 1-IMPLICANTE (ou um 1-cubo) é um produto com uma variável eliminada (obtido pela combinação de dois 0 implicantes adjacentes).
- A remoção de uma variável de um termo é conhecida como a expansão do termo, e corresponde a expandir o tamanho de um círculo no mapa de Karnaugh.

IMPLICANTE PRIMO E IMPLICANTE NÃO-PRIMO



Implicantes de 4 variáveis não primos.



Implicantes primos (IP) de 4 variáveis.

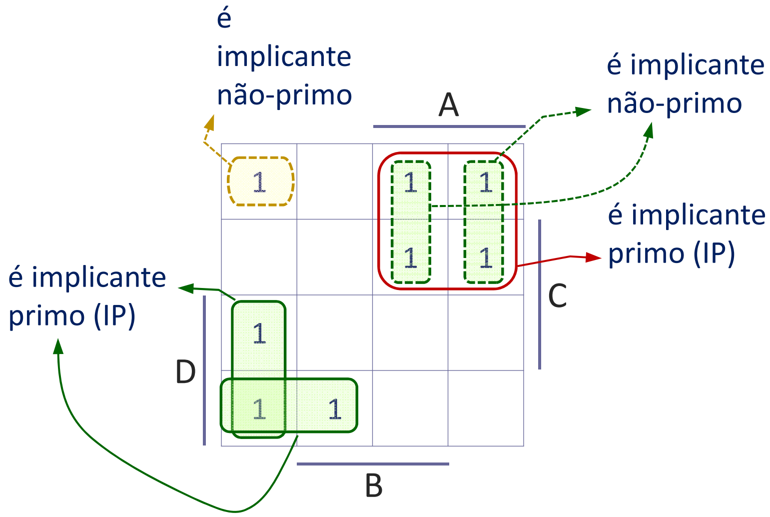
Graficamente, um IMPLICANTE PRIMO de uma função corresponde aos círculos que são os maiores possíveis.

Se uma variável for eliminada deste implicante, o resultado será um termo produto que não é implicante da função, pois cobrirá um termo que não pertence ao ON-SET da função.

Um implicante designa-se de **IMPLICANTE PRIMO** quando não pode ser combinado com mais nenhum outro implicante para eliminar outra variável (primo vem do inglês 'prime' e neste contexto significa principal). Um IMPLICANTE PRIMO de uma função booleana não implica nenhum outro implicante.

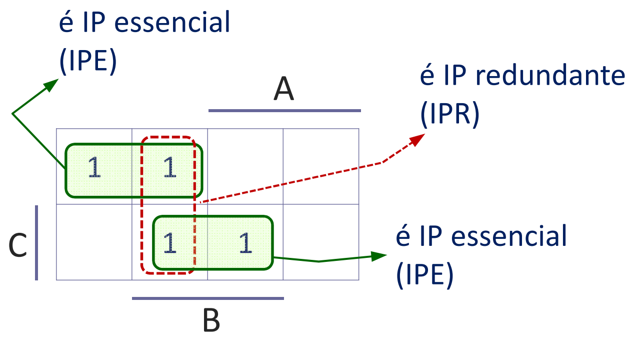
A importância dos implicantes primos decorre de eles corresponderem, no mapa de Karnaugh, aos grupos maiores, que não podem ser expandidos.

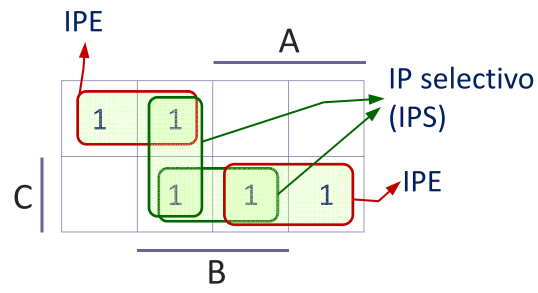
Esses são exactamente os grupos que interessam para as minimizações, porque possuem as expressões mais simples, com menos literais.



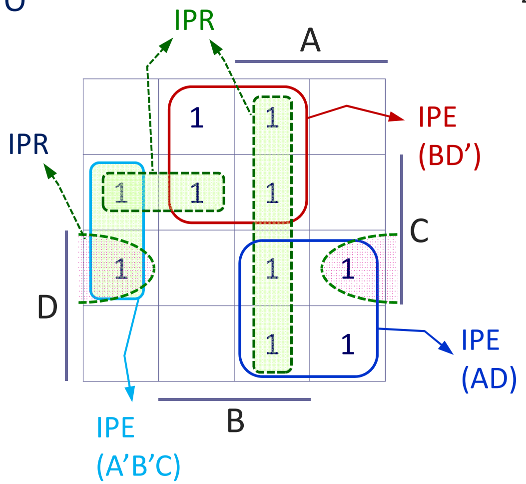
Implicantes primos e não primos de 4 variáveis.

IMPLICANTE PRIMO ESSENCIAL, IMPLICANTE PRIMO REDUNDANTE E IMPLICANTE PRIMO SELECTIVO





Implicantes primos essenciais (IPE) e implicantes primos redundantes.



Mapa com 6 implicantes primos, 3 essenciais (IPE) que cobrem todos os 1s do mapa, e 3 redundantes (IPR).

Um implicante primo de uma função diz-se IMPLICANTE PRIMO ESSENCIAL (IPE) se contém pelo menos um mintermo não contido em nenhum outro implicante primo.

Os IMPLICANTES PRIMO ESSENCIAIS associam pois mintermos que não podem ser associados em implicantes primos de outra forma.

Um implicante primo não essencial (redundante) é um implicante primo em que todos os 1s (uns) estão também incluídos em implicantes primos essenciais.

Nem todos os implicantes primos da função serão usados na expressão minimizada, mas todos os implicantes primos essenciais têm de estar presentes na expressão mínima na forma SOP ou POS.

Um implicante primo que não é IMPLICANTE PRIMO ESSENCIAL (IPE) nem IMPLICANTE PRIMO REDUNDANTE (IPR) é designado de IMPLICANTE PRIMO SELECTIVO (IPS). Os IPS ocorrem em pares.

Como nem todos os implicantes primos são utilizados para originar a expressão minimizada, a escolha é feita de acordo com o seguinte critério.

ALGORITMO DE MINIMIZAÇÃO

O procedimento sistemático para a obtenção da expressão simplificada (MSOP ou MPOS) de uma função representada num mapa de Karnaugh, corresponde à execução dos seguintes passos:

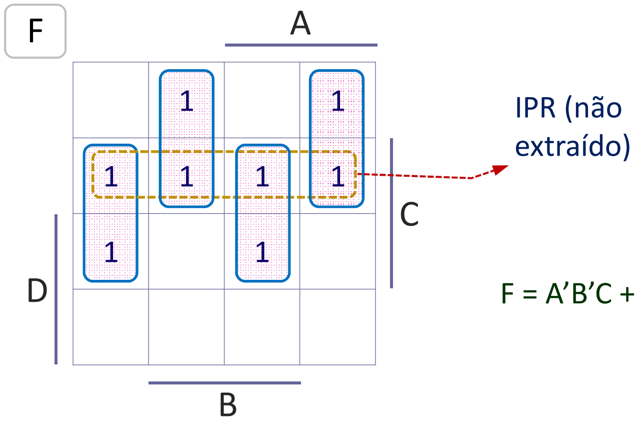
Passo 1: Agrupar todos os 1s (cobrir todos os mintermos). Considerar as eventuais indiferenças e adjacências das linhas e colunas dos extremos, e dos cantos.

Passo 2: Usar só implicantes primos essenciais (IPE).

Passo 2: Caso não estejam cobertos todos os 1s do quadro, determinar o menor conjunto de implicantes primos selectivos que cobrem os 1s remanescentes.

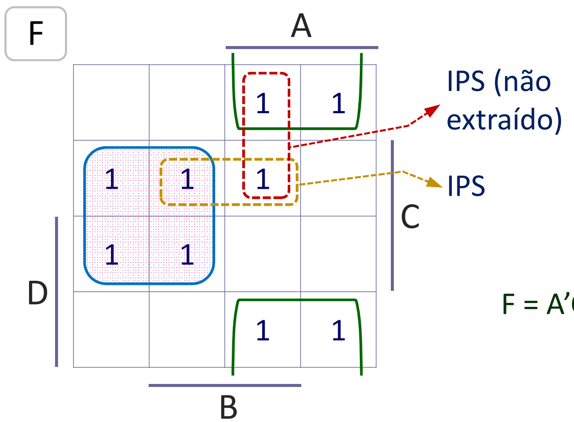
Passo 3: Não usar implicantes primos redundantes (IPR).

Passo 3: É boa sugestão começar-se pelos IPE em ordem ascendente: primeiro os 0-implicantes, depois os 1-implicantes, 2-implicantes, etc.. Extrair a expressão simplificada como soma/produto de todos os termos produto/soma seleccionados nos passos anteriores.



F = A'B'C + A'BD' + ABC + AB'D'

Extração tomando em conta só os implicantes primos essenciais e descartando os redundantes.



F = A'C + AC' + ABC'

Extração tomando em conta só os implicantes primos essenciais e selectivos.

Mapas de Karnaugh a Uma, Duas, Três, Quatro e Cinco variáveis

A

A’	A

m-K a 1 variável.

A

	A’B’	AB’
B	A’B	AB

m-K a 2 variáveis.

A

	A’B’C’	A’BC’	ABC’	AB’C’
C	A’B’C	A’BC	ABC	AB’C

B

m-K a 3 variáveis.

A

	A’B’C’D’	A’BC’D’	ABC’D’	AB’C’D’	
	A’B’CD’	A’BCD’	ABCD’	AB’CD’	C
D	A’B’CD	A’BCD	ABCD	AB’CD	C
D	A’B’C’D	A’BC’D	ABC’D	AB’C’D	

B

m-K a 4 variáveis.

E

A A

	A’B’C’D’E’	A’BC’D’E’	ABC’D’E’	AB’C’D’E’	A’B’C’D’E	A’BC’D’E	ABC’D’E	AB’C’D’E	
	A’B’CD’E’	A’BCD’E’	ABCD’E’	AB’CD’E’	A’B’CD’E	A’BCD’E	ABCD’E	AB’CD’E	C
D	A’B’CDE’	A’BCDE’	ABCDE’	AB’CDE’	A’B’CDE	A’BCDE	ABCDE	AB’CDE	C
D	A’B’C’DE’	A’BC’DE’	ABC’DE’	AB’C’DE’	A’B’C’DE	A’BC’DE	ABC’DE	AB’C’DE	

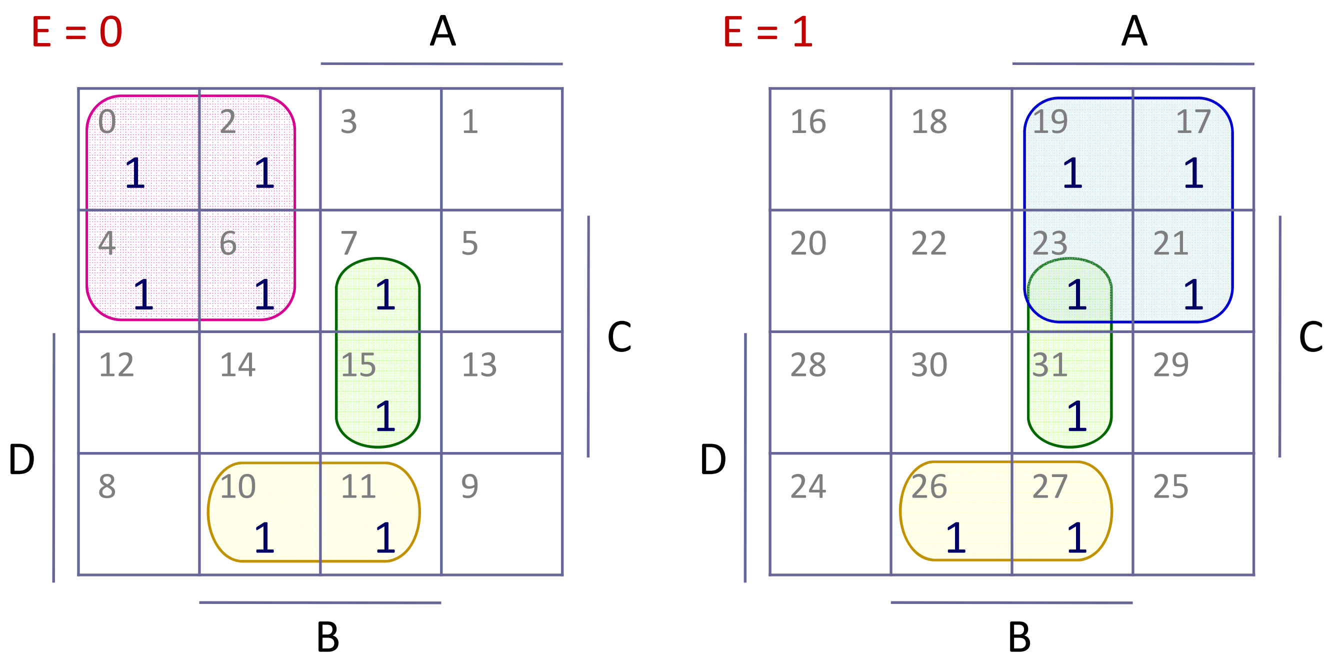
B B

m-K a 5 variáveis.

O mapa de Karnaugh (m-K) pode ser usado com qualquer número de varáveis. Mapas de Karnaugh com mais de 4 variáveis são pouco usuais por não ser simples a sua utilização. Na Fig. estão representados mapas de Karnaugh com diferentes número de variáveis. Assinalam-se a vermelho as quadrículas para as quais a variável A vale 1.

$F(A, B, C, D, E) = \sum m\,(0, 2, 4, 6, 7, 10, 11, 15, 17, 19, 21, 23, 26, 27, 31)$

Função de 5 variáveis da qual se pretende obter a expressão algébrica simplificada.



Dentro de cada quadrado do mapa e a vermelho estão indicadas as posições dos índices dos mintermos. A escuro estão marcados os 1s para os quadrados correspondentes aos mintermos em que a função é 1.

A forma mais simples de visualização de um mapa de Karnaugh de 5 variáveis é a que se obtém pela **duplicação** de um mapa de 4 variáveis.

Assume-se **E** como variável de maior peso. Do lado **esquerdo** está o mapa correspondente a **E=0**, do lado **direito** o de **E=1**.

Cada mapa apresenta uma leitura individual. Se a leitura de um dos mapas for sobreposta à do outro as duas leituras correspondem a um único termo – caso de ABC ou BC'D.

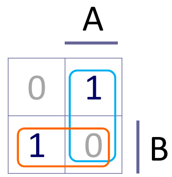
$F = A'D'E' + AD'E + ABC + BC'D$

Equação algébrica após simplificação.

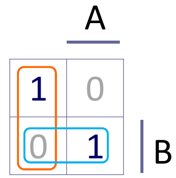
Mapa de Karnaugh: Padrões envolvendo XOR (1)

Através dos mapas de Karnaugh é possível obter expressões simplificadas envolvendo a operação XOR.

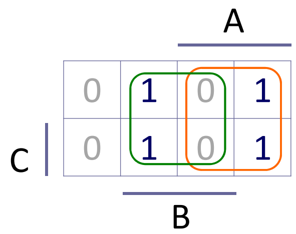
O XOR toma o valor 1 para as configurações que pertencem a um número **ímpar** de agrupamentos e o valor 0 para as configurações que pertencem a um número **par** de agrupamentos (ou que não pertencem a nenhum).



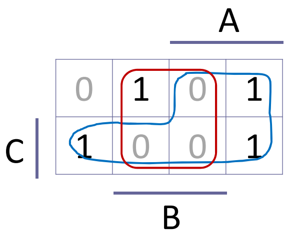
$F = AB' + A'B =$
$= A \oplus B$



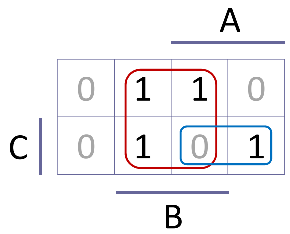
$F = A'B' + AB =$
$= (A' \oplus B) = (A \oplus B')$
$= (A \oplus B)'$



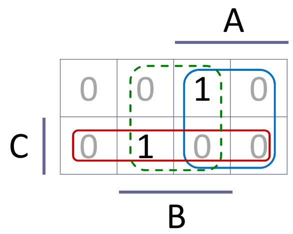
$F = A \oplus B$



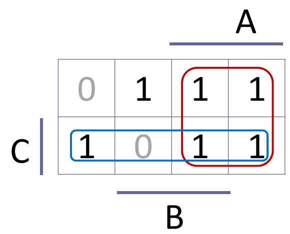
$F = B \oplus (A + C)$



$F = B \oplus (AC)$



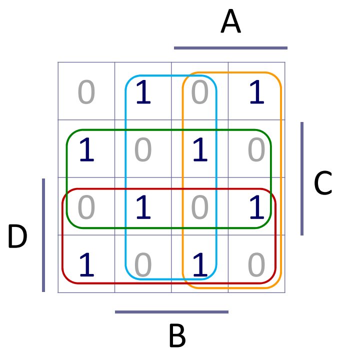
$F = B\,(A \oplus C)$



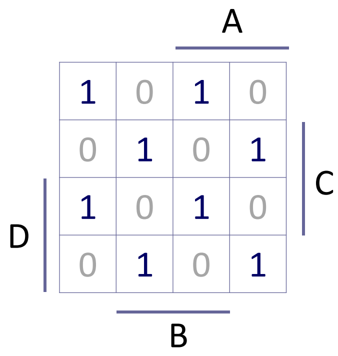
$F = A + (B \oplus C)$

Exemplos de funções extraídas do mapa de Karnaugh envolvendo operações XOR.

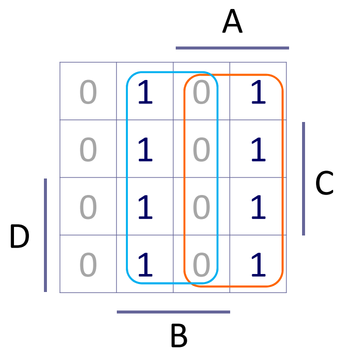
Mapa de Karnaugh: Padrões envolvendo XOR (2)



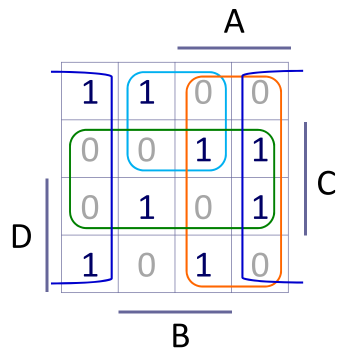
$F = A \oplus B \oplus C \oplus D$



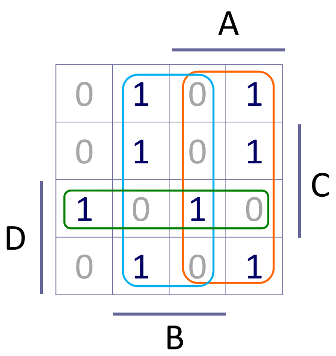
$F = (A \oplus B \oplus C \oplus D)'$



$F = A \oplus B$

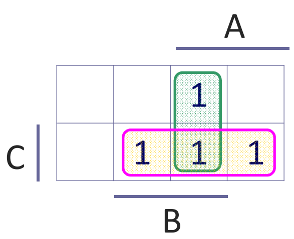


$F = A \oplus B' \oplus C \oplus BD'$

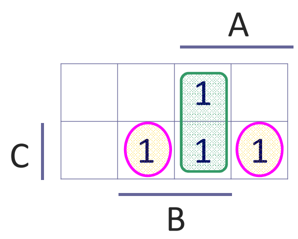


$F = A \oplus B \oplus CD$

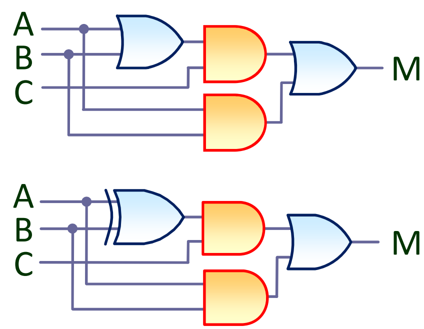
Exemplos de funções extraídas do mapa de Karnaugh envolvendo operações XOR.



$M = (A + B)\, C + AB$



$M = (A \oplus B)\, C + AB$



Os dois circuitos dos 2 quadros acima correspondem a duas ‘**assinaturas**’ distintas da mesma função M (Maioria), uma fazendo uso de um XOR, importantes no estudo do Carry-out do somador.

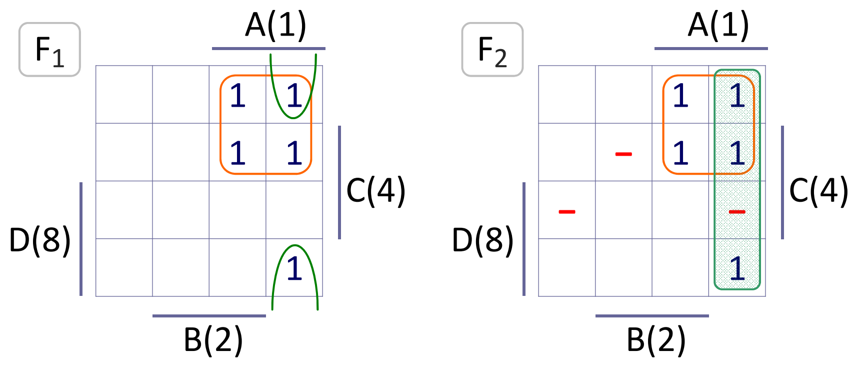
FUNÇÕES INCOMPLETAMENTE ESPECIFICADAS (1)

Quando não é definido o valor da função, por não se considerar possível a ocorrência de determinadas configurações à entrada, ou por se julgar irrelevante a resposta da função a essas combinações, usa-se:

- um símbolo de indefinição '**–**' ou '**X**' (**indiferença**, **irrelevância** ou **don't care**) na tabela de verdade ou mapa de Karnaugh da função, e
- a letra **d** na representação sob a forma de lista de mintermos (como em F_2), ou **D** para o caso de lista de maxtermos.

Na simplificação por mapa de Karnaugh, a estratégia é a de atribuir aos **don't care** valor lógico 0 ou 1 conforme mais conveniente para uma maior simplificação ('X' não existe, apenas existem '0' ou '1') .

A existência de indiferenças em F_2 permitiu reduzir o número de entradas da porta AB' e a complexidade do circuito relativamente à mesma função implementada em F1 sem indiferenças: atribuiu-se o valor 1 à indiferença correspondente ao mintermo 13 e o valor 0 às correspondentes a 6 e 12.



$F_1 = A\,D' + AB'C'$

$F_2 = A\,D' + AB'$

Mapas de Karnaugh e equações algébricas de F_1 e F_2 assinalando a redução do número de entradas em F_2 atribuível ao uso de indiferenças. (ausentes em F_1).

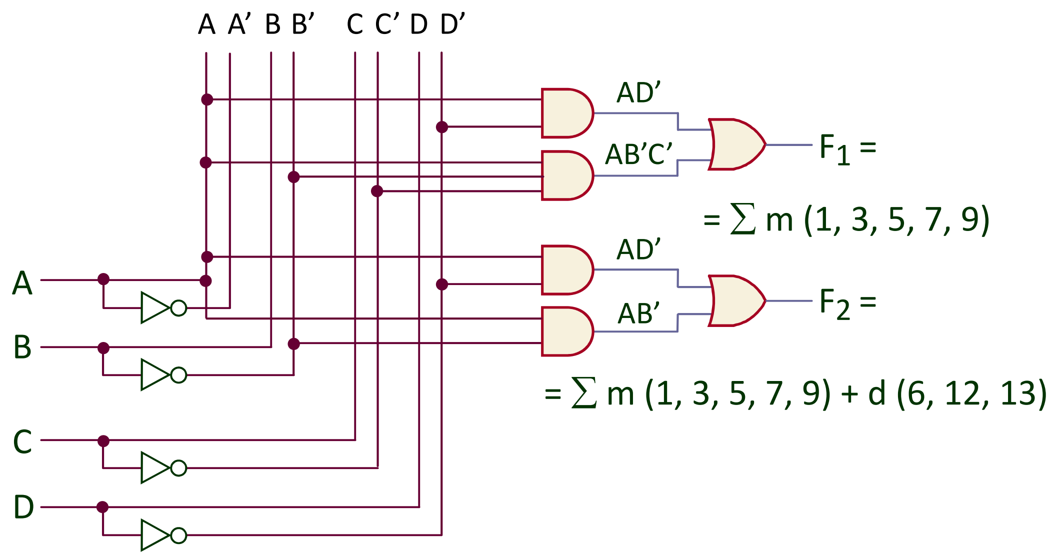


Diagrama lógico de duas funções F1 e F2 quase idênticas assinalando-se a redução do número de entradas em F2 atribuível ao uso de indiferenças (ausentes em F1).

Funções Incompletamente Especificadas (2)

$F_3 = \sum(1, 2) + d\,(3)$

F_3		A(1)	
	0	1	
	1	–	B(2)

$F_3 = A + B$

$F_4 = \sum(2, 4, 7) + d\,(1, 6)$

F_4			A(1)	A(1)
	0	1	0	–
C(4)	1	–	1	0
		B(2)	B(2)	

$F_4 = A'B + A'C + BC$

$F_5 = \sum(0, 2, 5) + d\,(3, 4)$

F_5			A	A
	1	1	–	0
C	–	0	0	1
		B	B	

$F_5 = A'C' + B'C$

$F_6 = \sum(1, 2, 5, 6, 10)$
$+ d\,(3, 4, 9, 11, 13, 14, 15)$

F_6			A(1)	A(1)	
	0	1	–	1	
	–	1	0	1	C(4)
D(8)	0	–	–	–	C(4)
D(8)	0	1	–	–	
		B(2)	B(2)		

$F_6 = A \oplus B$

$F_7 = \sum(0, 6, 7, 8, 9)$
$+ d\,(1, 4, 11, 14, 15)$

F_7			A	A	
	1	0	0	–	
	–	1	1	0	C
D	0	–	–	0	C
D	1	0	–	1	
		B	B		

$F_7 = BC + B'C'$

$F_8 = \sum(0, 1, 3, 5, 9, 10, 11\,)$
$+ d\,(2, 8, 13)$

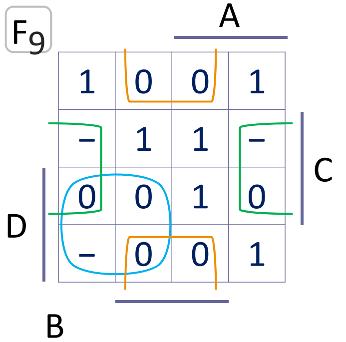
F_8			A	A	
	1	–	1	1	
	0	0	0	1	C
D	0	0	0	–	C
D	–	1	1	1	
		B	B		

$F_8 = C' + AB'$

Mapas de Karnaugh e expressões algébricas minimizadas de várias funções especificadas através da lista de mintermos, assinalando-se o uso criterioso das indiferenças.

FUNÇÕES INCOMPLETAMENTE ESPECIFICADAS (3)

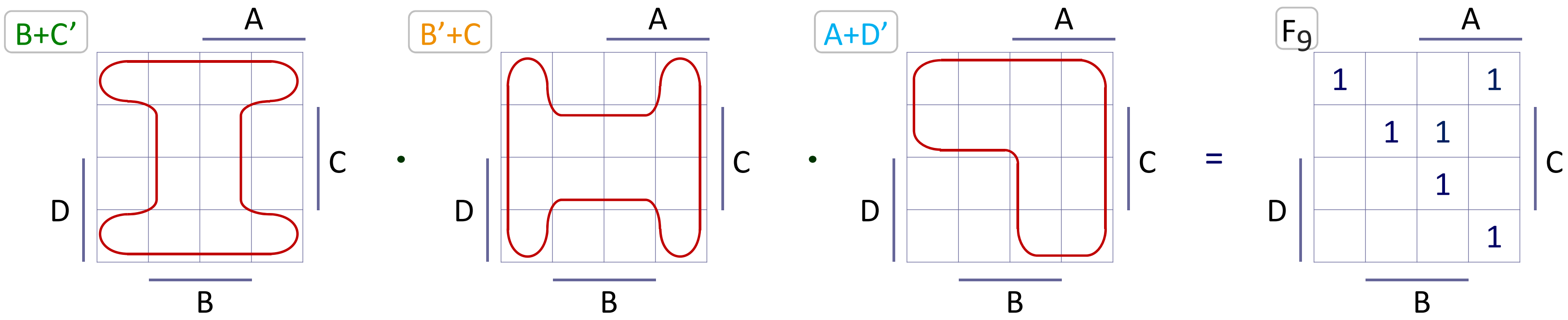
$$F_9 = \Pi M\,(2, 3, 10, 11, 12, 13, 14)\ \ \Pi D\,(4, 5, 8)$$



F_9 = (B'+C) (B+C') (A+D')

Mapa de Karnaugh e expressão algébrica minimizada na forma MPOS, assinalando-se o uso das indiferenças.

Verificação através da multiplicação lógica bit a bit dos mapas representando cada maxtermo para obtenção do mapa inicial (em cima):



Resumindo:

- No caso da representação sob a forma de lista de mintermos, ao somatório $\sum$m dos termos iguais a 1, junta-se o somatório dos termos opcionais (indiferenças), normalmente indicados por $\sum$d.
- No caso da representação sob a forma de lista de maxtermos, multiplicam-se os maxtermos (piatório ΠM) pelos termos opcionais (indiferenças), normalmente indicados por Π D.
- A leitura do mapa de Karnaugh feita directamente pelos zeros origina uma expressão minimizada na forma de produto de somas (MPOS).

Mapas de Karnaugh e expressões algébricas minimizadas de várias funções especificadas através da lista de maxtermos, assinalando-se o uso criterioso das indiferenças.

Implementação de uma Função – Circuitos Alternativos (Ex. 2-2-1)

Exemplo 2-2

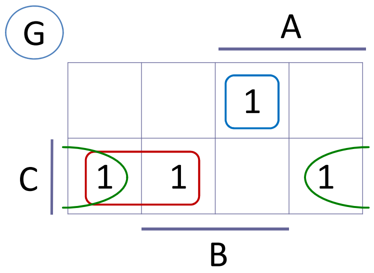
Objectivo

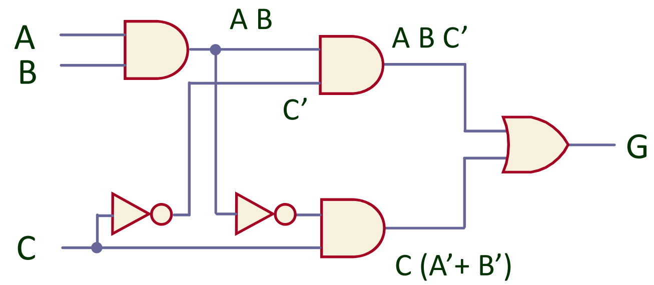
Desenhar o circuito da função G na tabela da Fig. nas várias formas correspondentes às representações em SOP, MSOP, soma de produtos após factorização e com utilização de uma porta XOR.

Mintermo	C	B	A	G
m_0	0	0	0	0
m_1	0	0	1	0
m_2	0	1	0	0
m_3	0	1	1	1
m_4	1	0	0	1
m_5	1	0	1	1
m_6	1	1	0	1
m_7	1	1	1	0

Tabela de Verdade funcional do circuito.

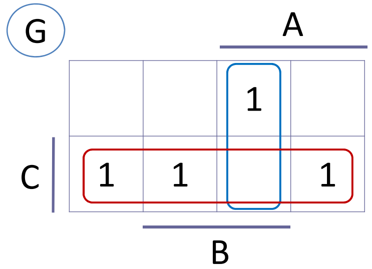
Uma manipulação feita a partir da expressão algébrica ou do mapa de Karnaugh da função permite obter expressões em várias formas, como a que usa o XOR e a factorização da variável C.

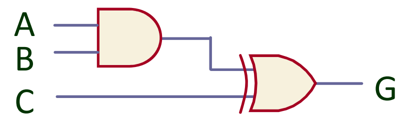




F = A B C’ + A’C + B’C

F = A B C’ + C (A’+ B’)

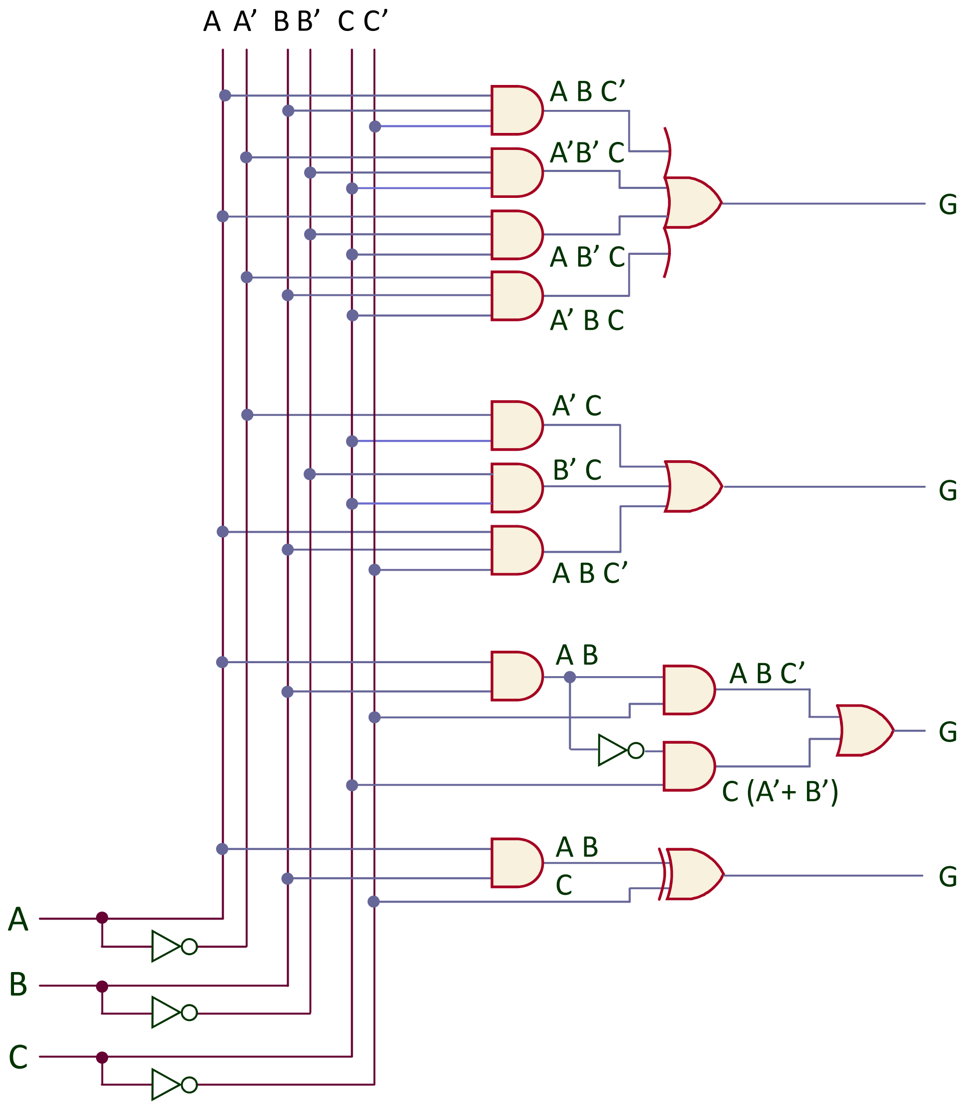




$F = A\,B \oplus C$

IMPLEMENTAÇÃO DE UMA FUNÇÃO – CIRCUITOS ALTERNATIVOS (EX. 2-2-2)

Exemplo 2-2



$G = \sum m\ (3, 4, 5, 6)$

$G = ABC' + A'B'C + AB'C + A'BC$

SOP - Forma Canónica AND-OR ou disjuntiva (soma de mintermos e 2 níveis).

MSOP - Soma de Produtos Minimizada (2 níveis).

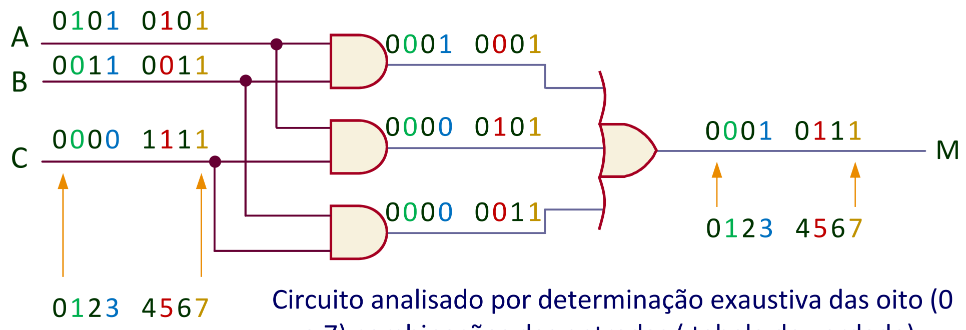
Soma de Produtos após simplificação e factorização (4 níveis).

Implementação utilizando um XOR (2 níveis).

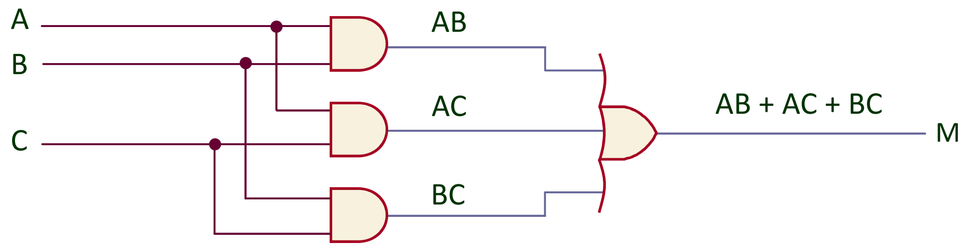
As 4 implementações são funcionalmente equivalentes: com os mesmos estímulos à entrada, as formas de onda, obtidas nas saídas dos 4 circuitos são quase idênticas.

As pequenas variações devem-se aos atrasos e aos GLITCHES que podem surgir devido às diferenças no número de níveis de portas e na estrutura interna das mesmas.

MÉTODOS DE ANÁLISE DE CIRCUITOS COMBINATÓRIOS



Circuito analisado por determinação exaustiva das oito (0 a 7) combinações das entradas (tabela de verdade).



Circuito analisado por determinação das expressões algébricas.

m	C	B	A	AB	AC	BC	M
0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0
2	0	1	0	0	0	0	0
3	0	1	1	1	0	0	1
4	1	0	0	0	0	0	0
5	1	0	1	0	1	0	1
6	1	1	0	0	0	1	1
7	1	1	1	1	1	1	1

Tabela de Verdade funcional obtida a partir dos métodos de análise do circuito acima.

A **análise** consiste na obtenção de uma descrição formal da função lógica do circuito. A descrição funcional mais descritiva corresponde à tabela de verdade. Um dos métodos de obter a tabela de verdade consiste em exaustivamente determinar as saídas parciais de todas as combinações de valores das variáveis de entrada ao longo do circuito, e propagar essa informação até à saída (desenho de cima).

Outro método consiste em ir obtendo as expressões algébricas para cada troço do circuito correspondente aos operadores lógicos envolvidos, e propagar estas expressões parciais até à saída. Este método é simples se o circuito tiver sido manipulado graficamente até à obtenção de ANDs, ORs e NOTs para simplificar a análise algébrica (como exemplificado no capítulo anterior – desenho do meio).

A partir daí é possível:

- Determinar o comportamento do circuito para diferentes combinações das entradas.
- Manipular a sua descrição algébrica e obter diferentes topologias de circuito para a mesma função lógica.
- Utilizar a descrição algébrica na análise de um sistema de complexidade superior que inclua o circuito num dos seus módulos.

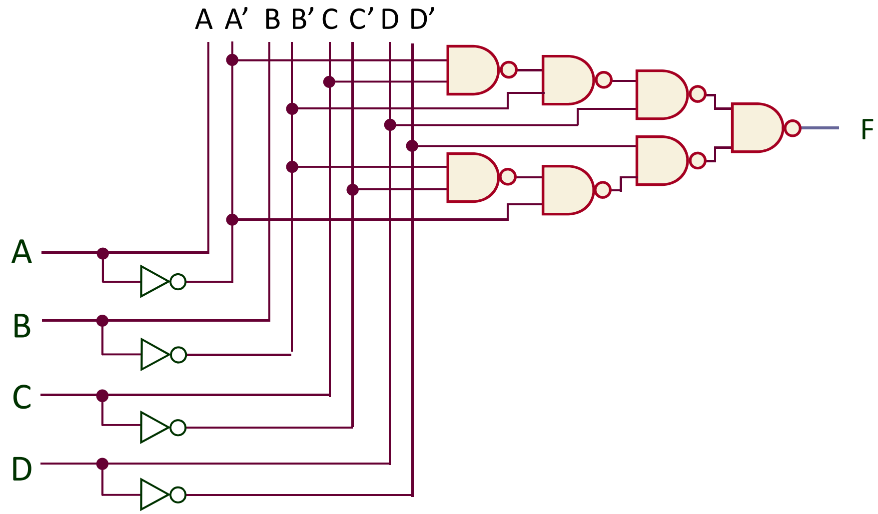
ANÁLISE DE UM CIRCUITO COMBINATÓRIO (EX. 2-3-1)

Exemplo 2-3

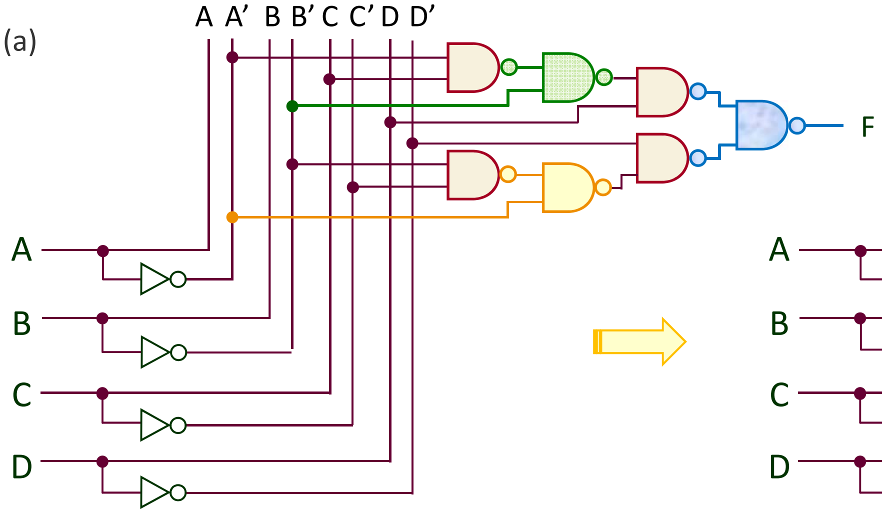
OBJECTIVO

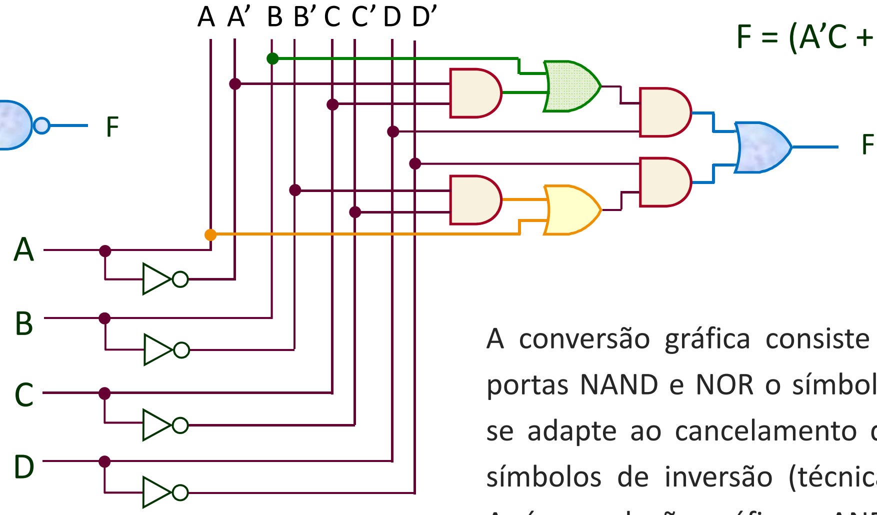
Para o circuito da Fig. realizado só com portas NAND de 2 entradas, obter:

a) Por manipulação gráfica, a expressão algébrica na forma AND-OR simplificada (MSOP).
b) A lista de mintermos.
c) A expressão algébrica na forma OR-AND simplificada (MPOS).
d) A lista de maxtermos.



(a)





F = (A'C + B) D + (B'C'+ A) D'

A conversão gráfica consiste em adoptar para as portas NAND e NOR o símbolo gráfico que melhor se adapte ao cancelamento de pares dos círculos símbolos de inversão (técnica **bubble-to-bubble**). Após a redução gráfica a AND, OR e NOT é muito mais fácil extrair a expressão booleana.

Análise de um Circuito Combinatório (Ex. 2-3-2)

Exemplo 2-3

(a)

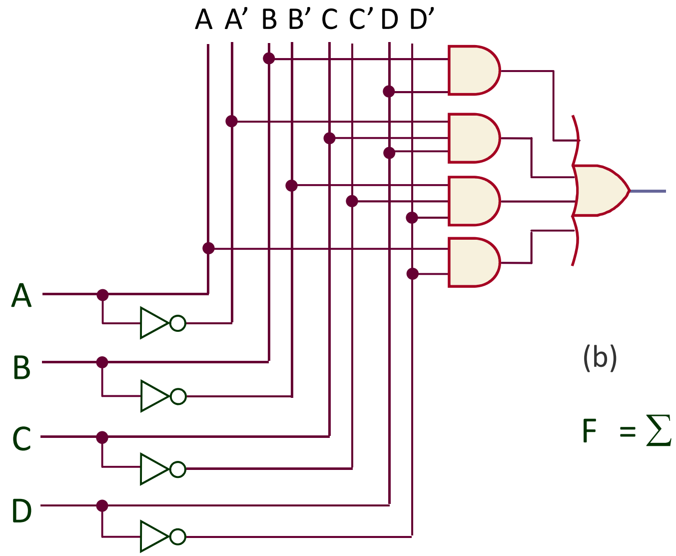
F

1		1	1
		1	1
1	1	1	
	1	1	

A, B, C, D

$F = (A'C + B)\,D + (B'C' + A)\,D' =$

$= BD + A'CD + B'C'D' + AD'$



F = BD + A'CD + B'C'D' + AD'

(b)

$F = \sum m\,(0, 1, 3, 5, 7, 10, 11, 12, 14, 15)$

(c)

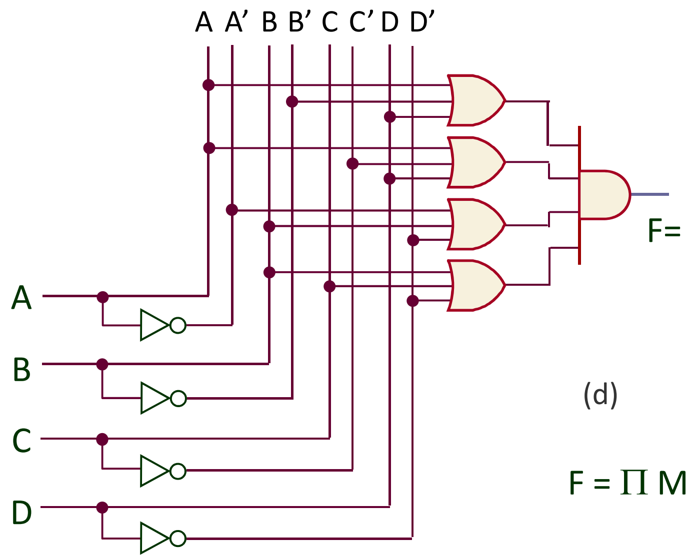
F'

	0		
0	0		
			0
0			0

A, B, C, D

$F' = A'BD' + A'CD' + AB'D + B'C'D$

$F = (A+B'+D)\,(A+C'+D)\,(A'+B+D')\,(B+C+D')$



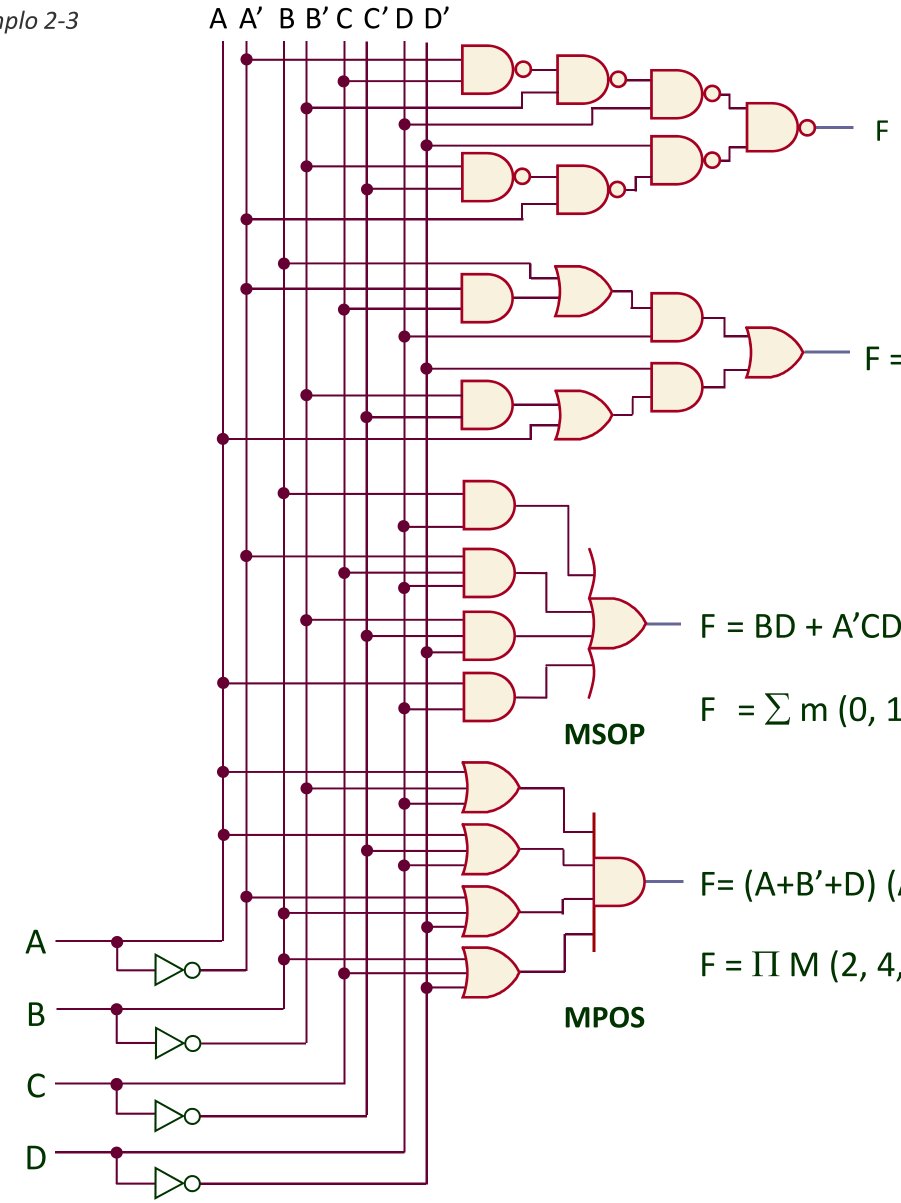
F= (A+B'+D) (A+C'+D) (A'+B+D') (B+C+D')

(d)

$F = \prod M\,(2, 4, 6, 8, 9, 13)$

ANÁLISE DE UM CIRCUITO COMBINATÓRIO (EX. 2-3-3)

Exemplo 2-3



F = (A’C + B) D + (B’C’+ A) D’

F = BD + A’CD + B’C’D’ + AD’

$F = \sum m\,(0, 1, 3, 5, 7, 10, 11, 12, 14, 15)$

F= (A+B’+D) (A+C’+D) (A’+B+D’) (B+C+D’)

$F = \prod M\,(2, 4, 6, 8, 9, 13)$

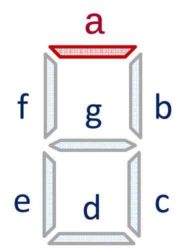
Comparando a forma MSOP com a forma MPOS, verifica-se que a forma MSOP se revela ligeiramente menos complexa.

Ambas apresentam um mesmo número de portas AND e OR, mas no total há menos dois pinos de entrada para as portas AND da forma MSOP do que para as portas OR da forma MPOS.

Exemplo 2-4

OBJECTIVO

Para o circuito conversor do código BCD-7 segmentos (do tipo ACTIVE HIGH) da Fig. abaixo obter, por manipulação gráfica, a expressão algébrica das saídas.



Disposição e identificação dos 7-segmentos de um display de caracteres decimais (segmento '**a**' iluminado).

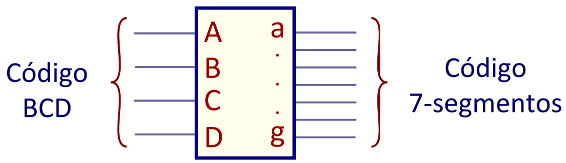
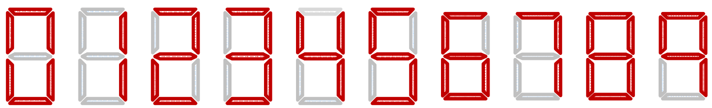


Diagrama de blocos do conversor.

Um segmento ilumina-se quando o circuito apresenta um 1 na saída correspondente, caso contrário o Conversor seria do tipo ACTIVE-LOW.

Pesos	8	4	2	1							
m	D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	0	0	1	1
10	1	0	1	0	-	-	-	-	-	-	-
11	1	0	1	1	-	-	-	-	-	-	-
12	1	1	0	0	-	-	-	-	-	-	-
13	1	1	0	1	-	-	-	-	-	-	-
14	1	1	1	0	-	-	-	-	-	-	-
15	1	1	1	1	-	-	-	-	-	-	-

Tabela de verdade de um conversor BCD-7-segmentos do tipo ACTIVE HIGH.



Caracteres decimais resultantes da iluminação dos segmentos num **display de** 7-segmentos.

A(1), B(2), C(4), D(8)

0	2	3	1
4	6	7	5
12	14	15	13
8	10	11	9

a

1	1	1	0
0	1	1	1
–	–	–	–
1	–	–	1

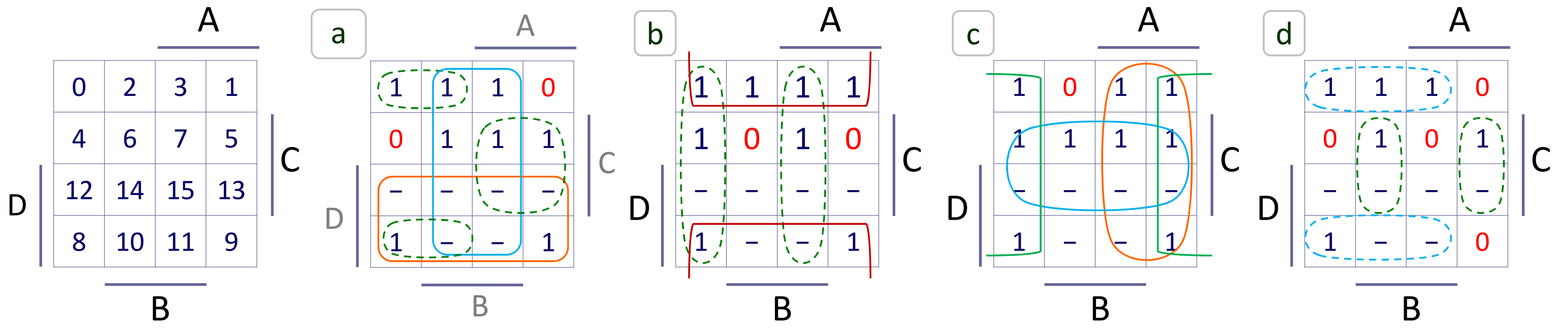
A, B, C, D

$a = B + D + A' \oplus C$

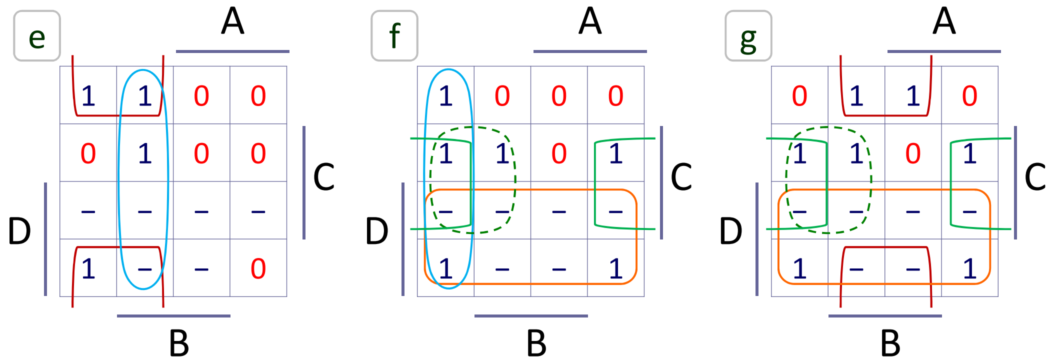
Mapas de Karnaugh mostrando a referência dos mintermos, o preenchimento dos 1s para o segmento **'a'** , e a equação algébrica do segmento '**a**' através da utilização de uma porta XOR.

CONVERSOR BCD-7 SEGMENTOS (EX. 2-4-2)

Exemplo 2-4



$a = B + D + A' \oplus C$ $\quad b = C' + A' \oplus B$ $\quad c = A + B' + C$ $\quad d = C'(A' + B) + C\,(A \oplus B)$



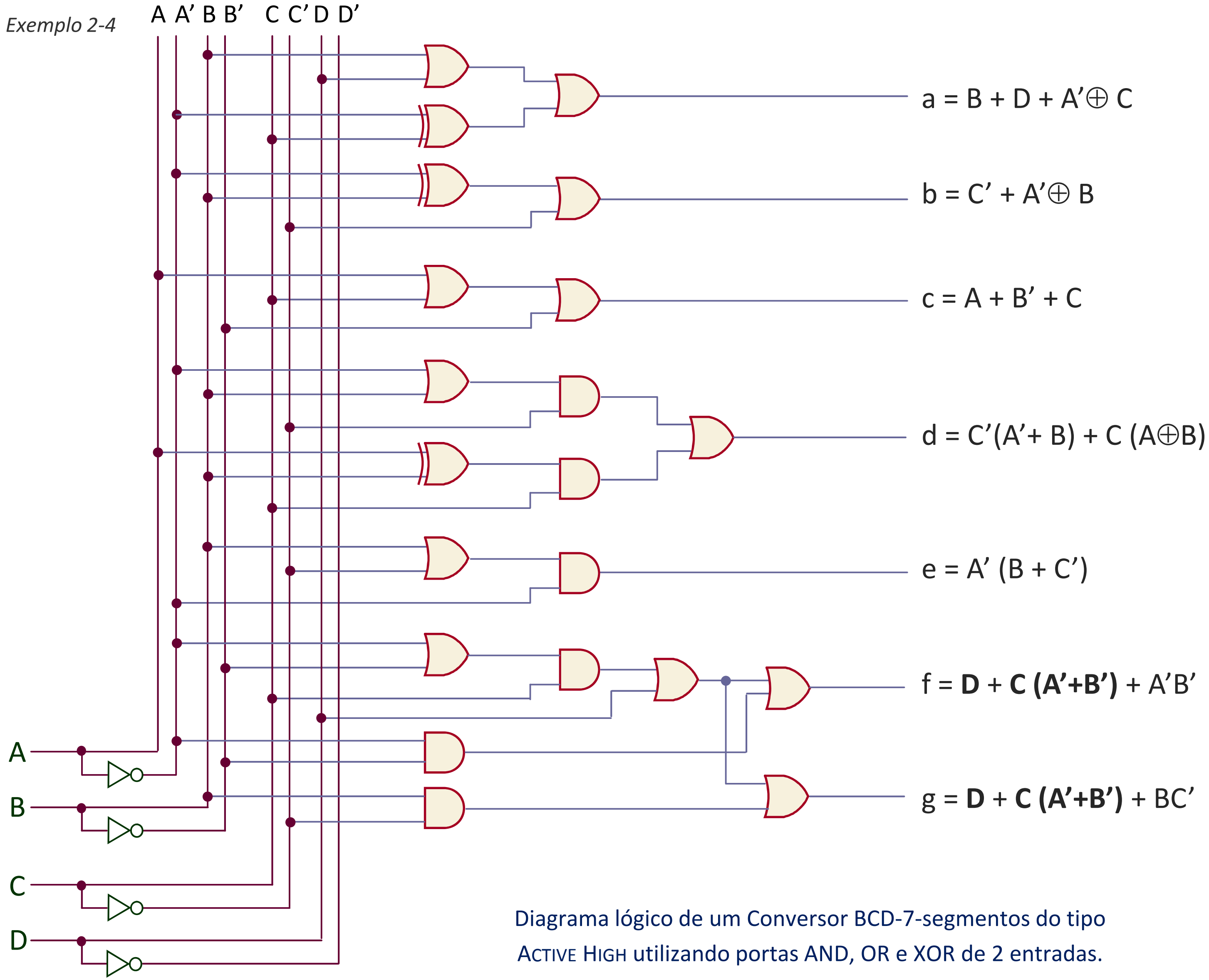
$e = A'\,(B + C')$ $\quad f = D + A'B' + C(A' + B')$ $\quad g = D + BC' + C(A' + B')$

Mapas de Karnaugh dos diversos segmentos e equações algébricas simplificadas fazendo uso intensivo das indiferenças. As expressões obtidas são do tipo AND-OR e XOR e tomam em conta a factorização.

Quando há várias funções dependentes das mesmas variáveis de entrada tem interesse sintetizar uma função a partir de outra já existente, para minimizar o número de portas. Há que identificar um padrão comum às várias funções e reutilizá-lo. Esta minimização utiliza o método multinível, ou de **BRIDGING**.

Nas expressões de **f** e **g** os termos D + C (A' + B') são recorrentes, sendo este bloco gerado uma só vez em f, e reutilizado em g.

CONVERSOR BCD-7 SEGMENTOS (EX. 2-4-3)



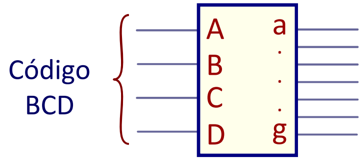


Diagrama de blocos.

Diagrama lógico de um Conversor BCD-7-segmentos do tipo
ACTIVE HIGH utilizando portas AND, OR e XOR de 2 entradas.

COMPARADOR DE NÚMEROS DE 2 BITS (EX. 2-5-1)

Exemplo 2-5

OBJECTIVO

Para o circuito comparador de 2 números (de 2 bits cada) da Fig. ao lado, obter a expressão algébrica das saídas.

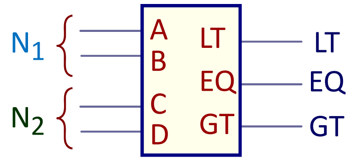


Diagrama de blocos do circuito comparador com 4 entradas e 3 saídas.

LT Less Than
EQ Equal
GT Greater Than

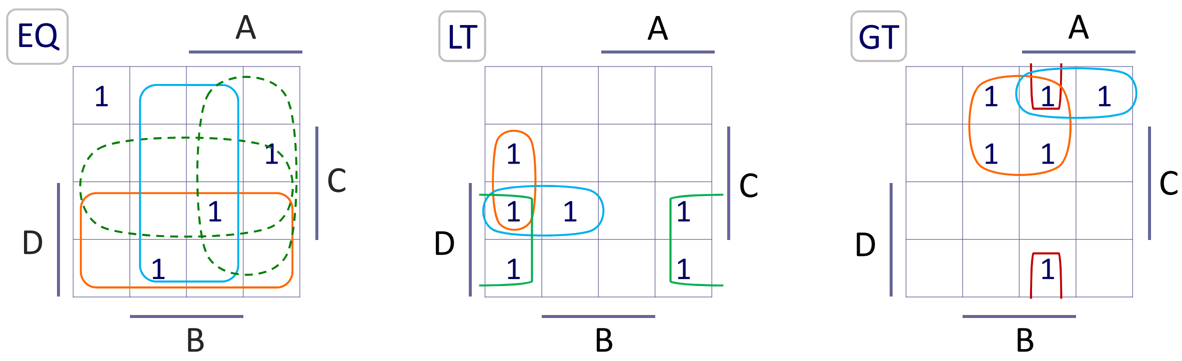
N_1 (bits B e A) e N_2 (bits D e C) são os 2 números a comparar.

LT =1 se $N_1 < N_2$ (BA < DC)

m	DC (N_2)	BA (N_1)	LT	EQ	GT
0	00	00	0	1	0
1		01	0	0	1
2		10	0	0	1
3		11	0	0	1
4	01	00	1	0	0
5		01	0	1	0
6		10	0	0	1
7		11	0	0	1
8	10	00	1	0	0
9		01	1	0	0
10		10	0	1	0
11		11	0	0	1
12	11	00	1	0	0
13		01	1	0	0
14		10	1	0	0
15		11	0	1	0

Tabela Funcional.

$$EQ = A'B'C'D' + AB'CD' + ABCD + A'BC'D =$$
$$= (AC + A'C')(BD + B'D') =$$
$$= (A \oplus C)' (B \oplus D)'$$



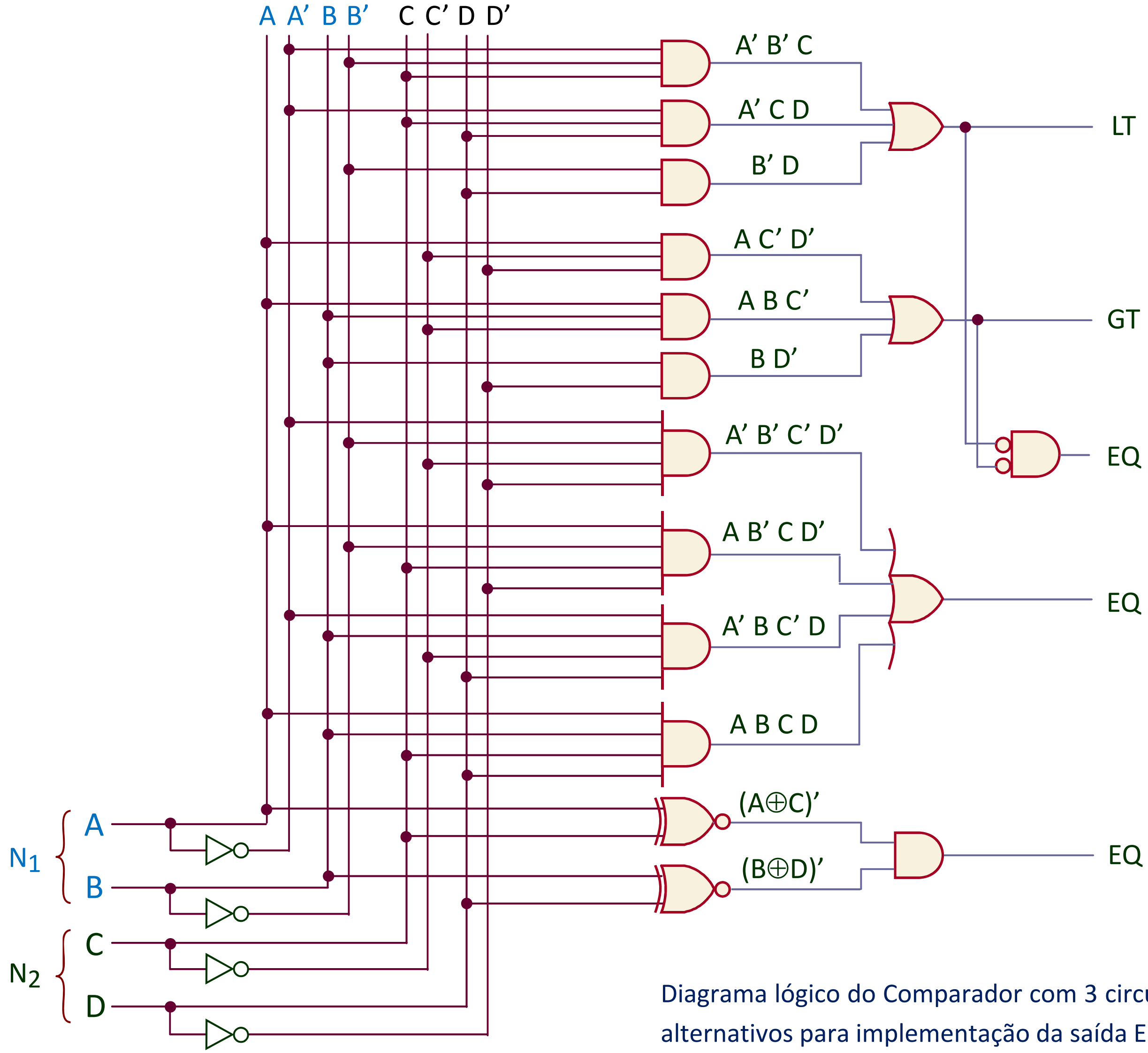
LT = A'B'C + A'CD + B'D

GT = BD' + AC'D' + ABC'

Tabela Funcional, Mapas de Karnaugh e Equações Algébricas das Saídas do Comparador.

COMPARADOR DE NÚMEROS DE 2 BITS (EX. 2-5-2)

Exemplo 2-5



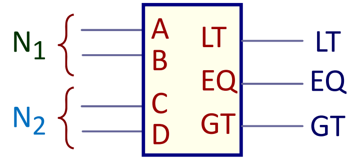


Diagrama de blocos de um comparador de números de 2 bits.

$$LT = A'B'C + A'CD + B'D$$

$$GT = BD' + AC'D' + ABC'$$

$$EQ = A'B'C'D' + AB'CD' + ABCD + A'BC'D = (A \oplus C)' (B \oplus D)'$$

Diagrama lógico do Comparador com 3 circuitos alternativos para implementação da saída EQ.

MULTIPLICADOR DE NÚMEROS DE 2 BITS (EX. 2-6-1)

Exemplo 2-6

OBJECTIVO

Para o circuito multiplicador de 2 números de 2 bits cada da Fig. ao lado, obter a expressão algébrica das saídas.

N_1 e N_2: números a multiplicar

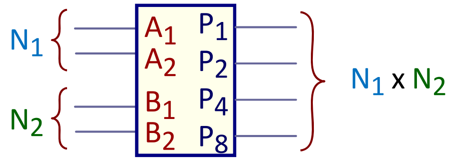
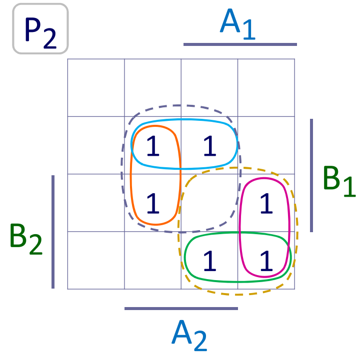


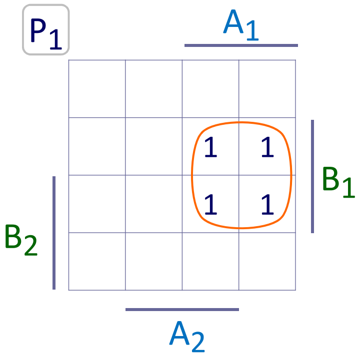
Diagrama de blocos.

	N_2		N_1					
m	B_2	B_1	A_2	A_1	P_8	P_4	P_2	P_1
0	0	0	0	0	0	0	0	0
1			0	1	0	0	0	0
2			1	0	0	0	0	0
3			1	1	0	0	0	0
4	0	1	0	0	0	0	0	0
5			0	1	0	0	0	1
6			1	0	0	0	1	0
7			1	1	0	0	1	1
8	1	0	0	0	0	0	0	0
9			0	1	0	0	1	0
10			1	0	0	1	0	0
11			1	1	0	1	1	0
12	1	1	0	0	0	0	0	0
13			0	1	0	0	1	1
14			1	0	0	1	1	0
15			1	1	1	0	0	1

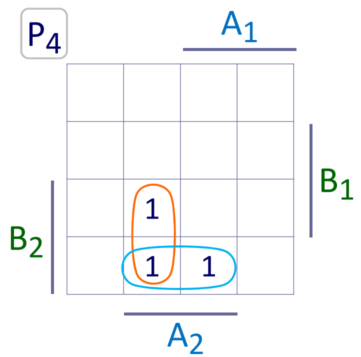
Tabela funcional.



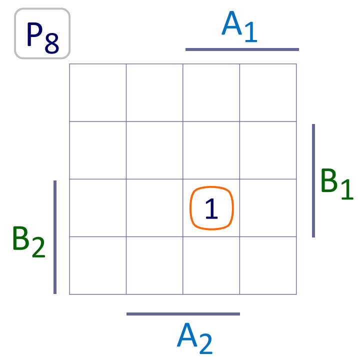
$$P_2 = A_2B_1B_2' + A_1'A_2B_1 + A_1A_2'B_2 + A_1B_1'B_2 = $$
$$= A_2B_1 \oplus A_1B_2$$



$$P_1 = A_1B_1$$



$$P_4 = A_1'A_2B_2 + A_2B_1'B_2 = $$
$$= A_2B_2 \oplus A_1A_2B_1B_2$$



$$P_8 = A_1A_2B_1B_2$$

Mapas de Karnaugh e equações simplificadas das saídas.

Instituto Superior de Engenharia de Lisboa

MULTIPLICADOR DE NÚMEROS DE 2 BITS (EX. 2-6-2)

Exemplo 2-6

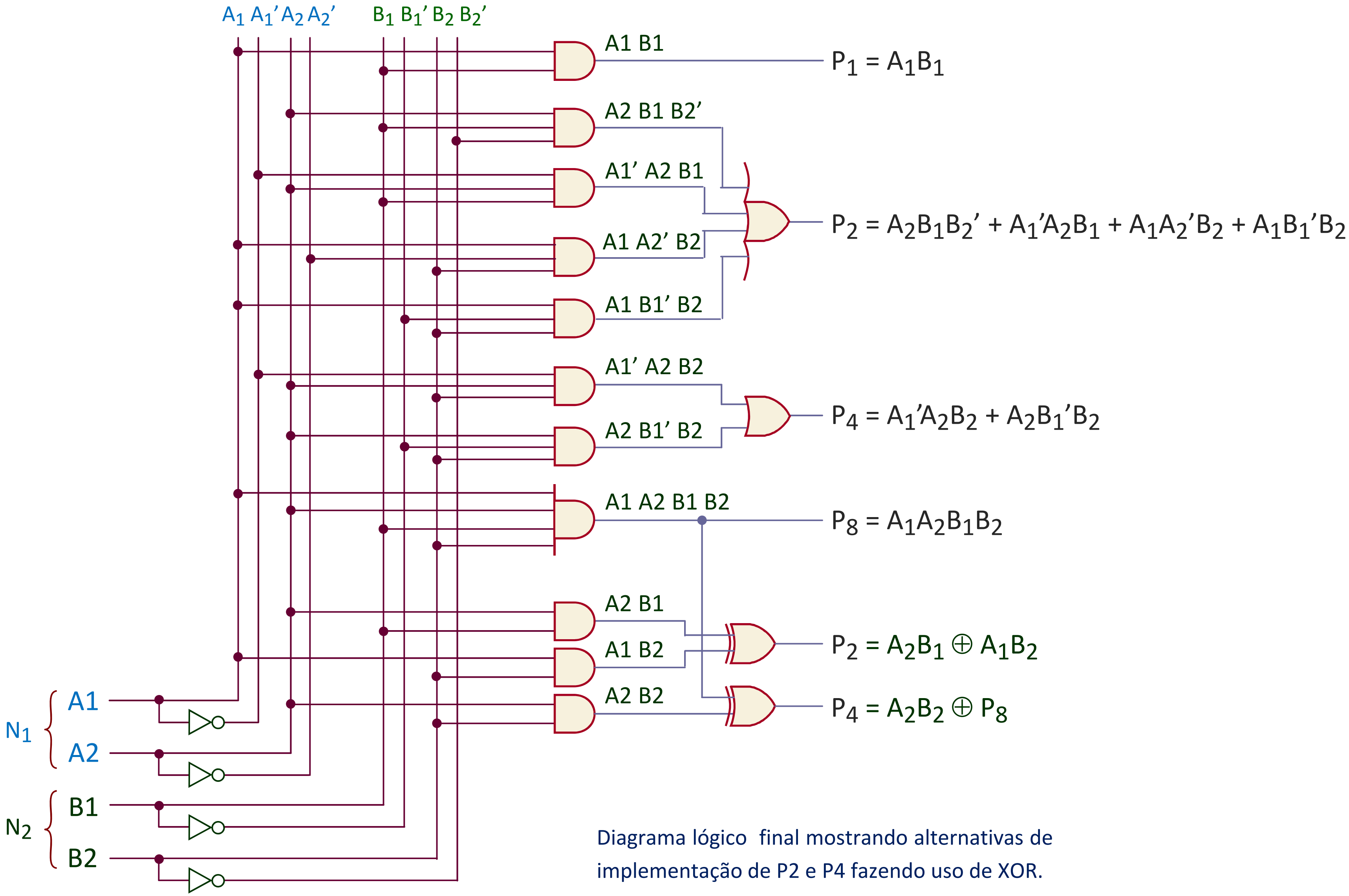


Diagrama lógico final mostrando alternativas de implementação de P2 e P4 fazendo uso de XOR.

INCREMENTADOR DE UM NÚMERO BCD (EX. 2-7-1)

Exemplo 2-7

OBJECTIVO

Para o circuito incrementador de código BCD da Fig. ao lado, obter, por manipulação gráfica, a expressão algébrica das saídas na forma AND-OR simplificada (MSOP).

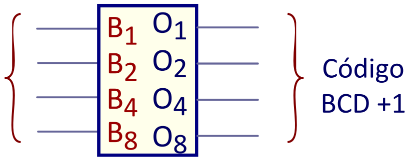
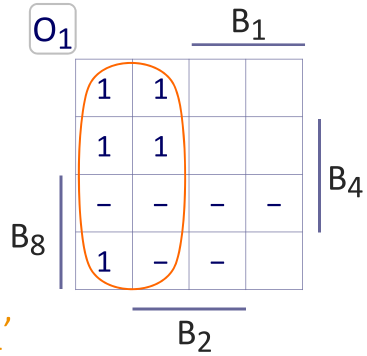


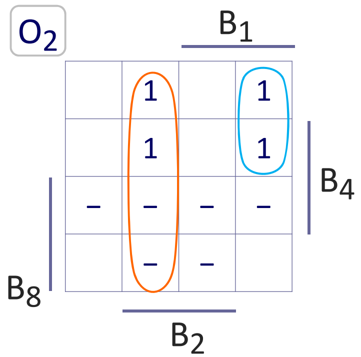
Diagrama de blocos do circuito Incrementador com 4 entradas em código BCD e 4 saídas apresentando o código BCD de entrada incrementado de 1 unidade.

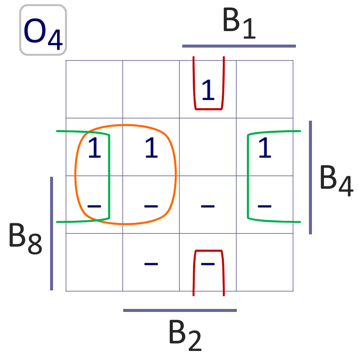
m	B_8	B_4	B_2	B_1	O_8	O_4	O_2	O_1
0	0	0	0	0	0	0	0	1
1	0	0	0	1	0	0	1	0
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	1	0	0
4	0	1	0	0	0	1	0	1
5	0	1	0	1	0	1	1	0
6	0	1	1	0	0	1	1	1
7	0	1	1	1	1	0	0	0
8	1	0	0	0	1	0	0	1
9	1	0	0	1	0	0	0	0
	1	0	1	0	–	–	–	–
	1	0	1	1	–	–	–	–
	1	1	0	0	–	–	–	–
	1	1	0	1	–	–	–	–
	1	1	1	0	–	–	–	–
	1	1	1	1	–	–	–	–

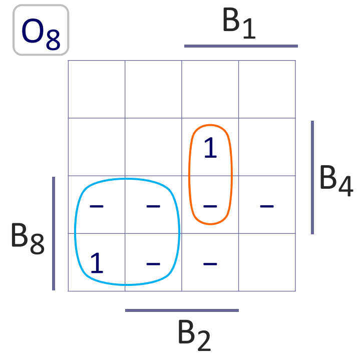
Tabela funcional.



$O_1 = B_1'$







$O_2 = B_1'B_2 + B_1B_2'B_8'$

$O_4 = B_1'B_4 + B_2'B_4 + B_1B_2B_4'$

$O_8 = B_1'B_8 + B_1B_2B_4$

Mapas de Karnaugh e equações simplificadas das saídas.

Nas configurações de entrada que não são BCD, não é importante considerar o valor da função, uma vez que essas configurações nunca ocorrem.

INCREMENTADOR DE UM NÚMERO BCD (EX. 2-7-2)

Exemplo 2-7

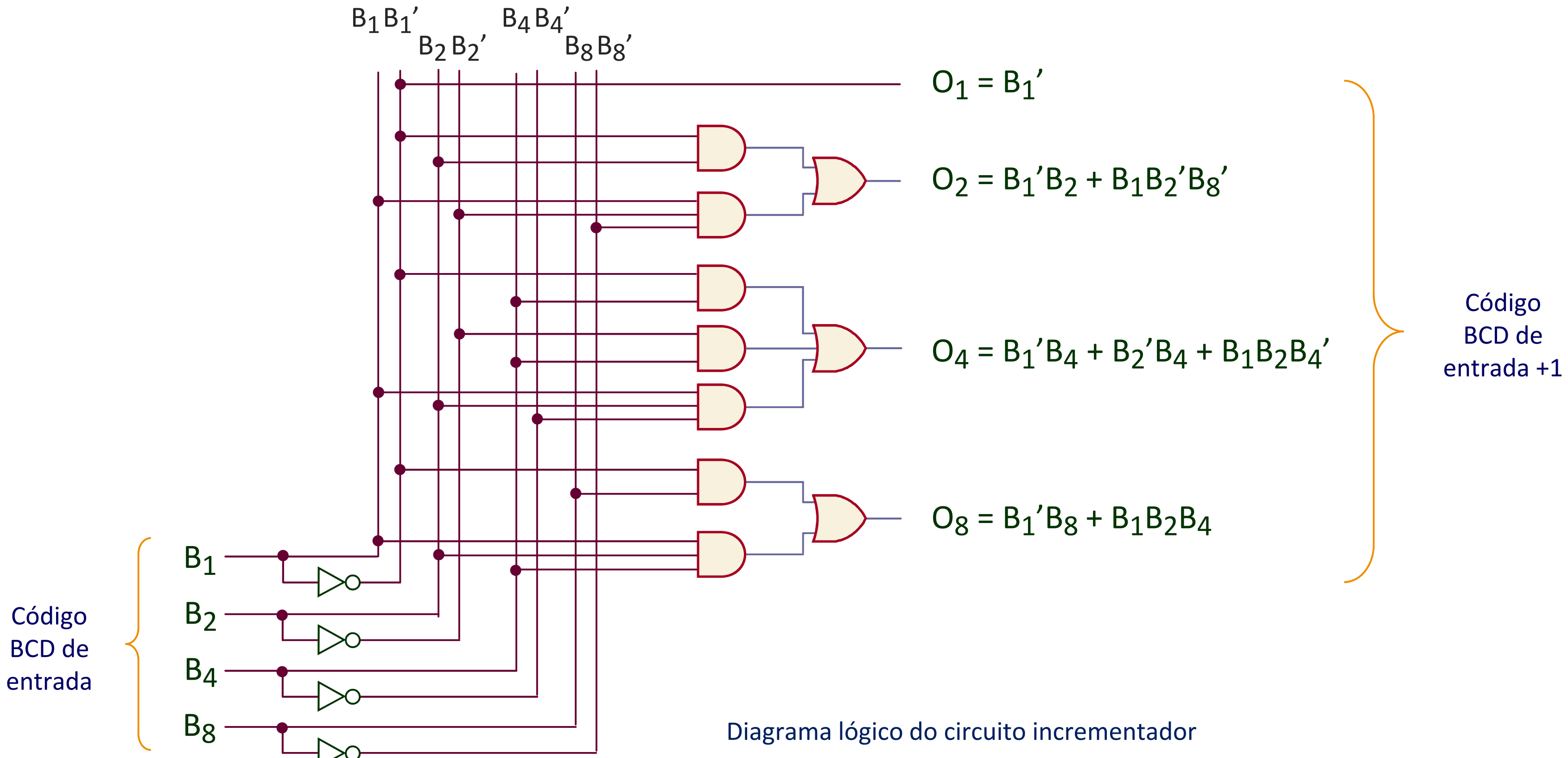


Diagrama lógico do circuito incrementador de um número BCD na forma AND-OR.

INDICADOR COM 4 SENSORES E 4 LÂMPADAS (EX. 2-8-1)

Exemplo 2-8

OBJECTIVO

Para o circuito da Fig. ao lado, obter a expressão algébrica das saídas e desenhar o diagrama lógico só com portas NAND de 2 e 3 entradas. O comportamento do circuito é o seguinte: uma lâmpada de saída Li ilumina-se quando:

1. estiverem activos pelo menos 2 sensores;
2. não estiver ativo o sensor Si correspondente;
3. a lâmpada se encontrar fora do intervalo dos sensores ativos.

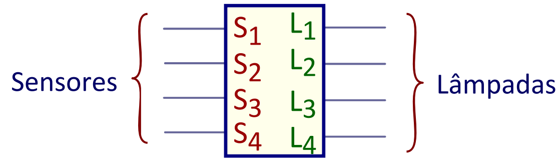
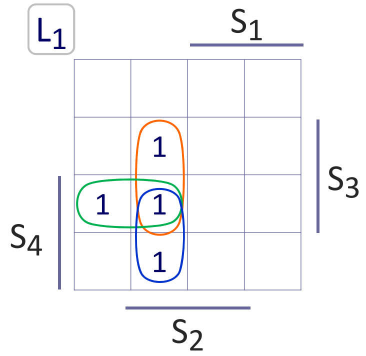


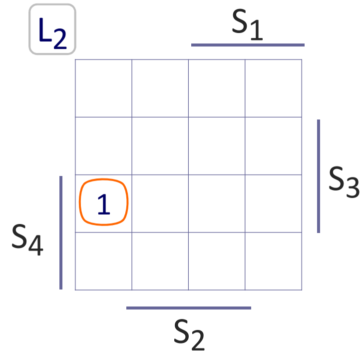
Diagrama de blocos do circuito indicador com 4 lâmpadas (saídas) que se iluminam consoante a configuração de 4 sensores (entradas).

m	S_4	S_3	S_2	S_1	L_4	L_3	L_2	L_1
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0
2	0	0	1	0	0	0	0	0
3	0	0	1	1	1	1	0	0
4	0	1	0	0	0	0	0	0
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	0	0
8	1	0	0	0	0	0	0	0
9	1	0	0	1	0	0	0	0
10	1	0	1	0	0	0	0	1
11	1	0	1	1	0	0	0	0
12	1	1	0	0	0	0	1	1
13	1	1	0	1	0	0	0	0
14	1	1	1	0	0	0	0	1
15	1	1	1	1	0	0	0	0

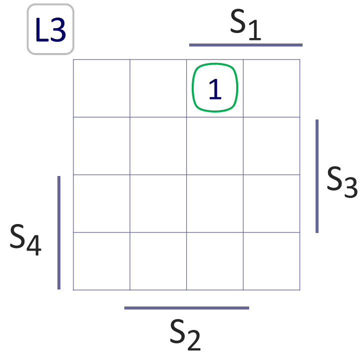
Tabela funcional.



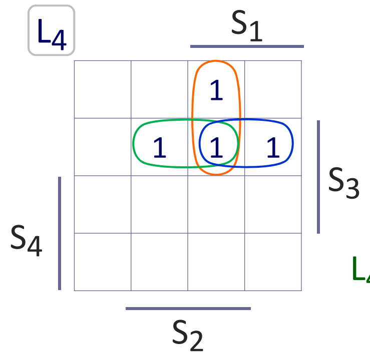
$L_1 = S_1'S_2S_3 + S_1'S_2S_4 + S_1'S_3S_4$



$L_2 = S_1'S_2'S_3S_4$



$L_3 = S_1S_2S_3'S_4'$



$L_4 = S_1S_2S_4' + S_2S_3S_4' + S_1S_3S_4'$

Instituto Superior de Engenharia de Lisboa

INDICADOR COM 4 SENSORES E 4 LÂMPADAS (EX. 2-8-2)

Exemplo 2-8

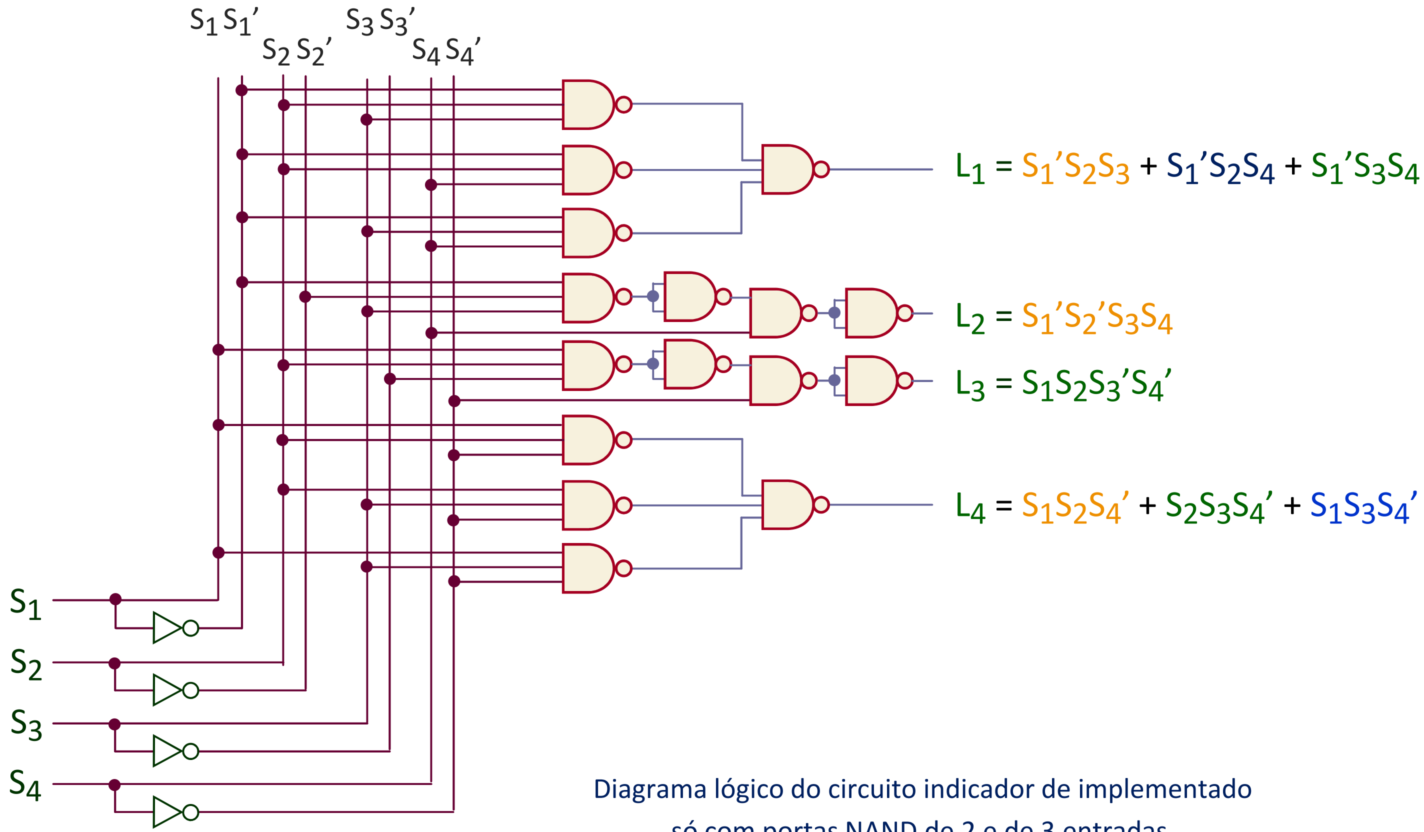
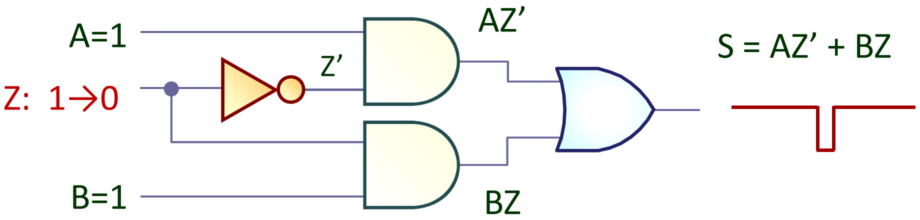
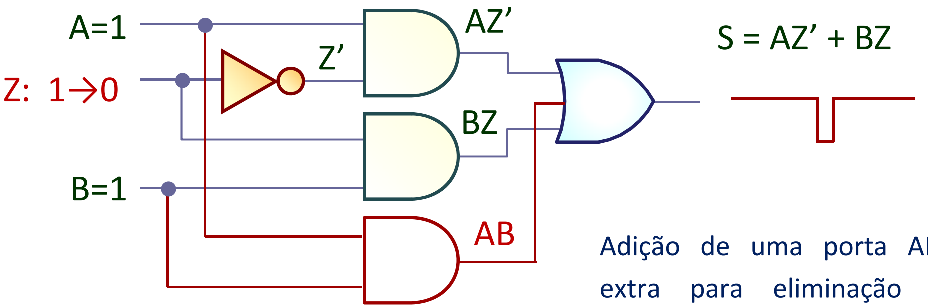


Diagrama lógico do circuito indicador de implementado
só com portas NAND de 2 e de 3 entradas.

Eliminação de Hazards e Glitches com Mapa de Karnaugh



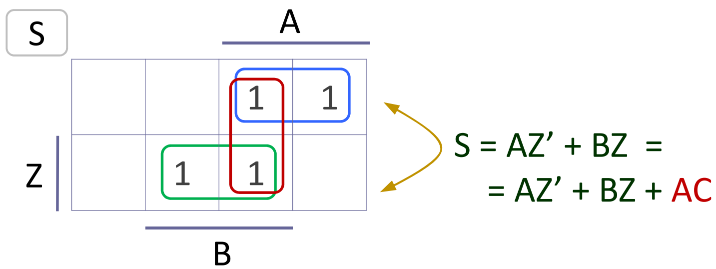
Circuito típico de **Hazard** estático evidenciando um **Glitch** a 0 de duração igual a uma unidade do tempo de atraso.



Adição de uma porta AND extra para eliminação do **Hazard** e do **Glitch**.

Num circuito que implemente uma soma de produtos a 2-níveis só podem ocorrer **Hazards Estáticos-1** (como se viu no capítulo anterior com o circuito da Fig.). **Hazards Estáticos-0** podem ocorrer num circuito OR-AND dual do representado.

O **Hazard** só aparece quando A=B=1, e o sinal Z viaja para a direita através das portas AND. Na mudança de Z de 1 para 0 há um período transitório diminuto em que as entradas da porta OR vão simultaneamente a 0 podendo produzir um **Glitch** a 0 (se o AND que transita para 0 o fizer antes do AND que transita para 1).



Os mapas de Karnaugh podem ser usados para detectar **Hazards** estáticos em circuitos com estrutura AND-OR ou OR-AND. O pressuposto de que apenas uma entrada varia em cada instante equivale a efectuar um deslocamento através de células vizinhas num mapa de Karnaugh (como indicado pela seta no mapa). No mapa da Fig. não há um termo produto que cubra simultaneamente as combinações ABZ=111 e ABZ=110. Para eliminar o **Hazard** e o **Glitch** tem de incluir-se no circuito uma porta AND redundante (correspondente a esse termo produto e à área a vermelho no mapa) que se mantém a 1 durante a transição da variável Z, forçando sempre a entrada do OR a 1.

LSD – 2 Índice 1

LSD – 2 ÍNDICE 2