Simplest SOP Expressions (1/3)

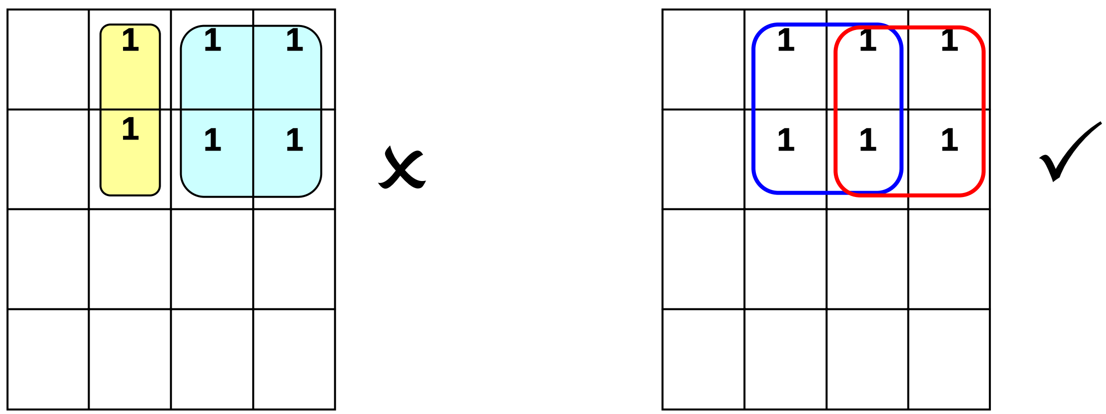
- To find the simplest possible *sum of products* (SOP) expression from a K-map, you need to obtain:
  - ❖ minimum number of literals per product term; and
  - ❖ minimum number of product terms
- This is achieved in K-map using
  - ❖ *bigger groupings* of minterms (prime implicants) where possible; and
  - ❖ *no redundant groupings* (look for essential prime implicants)

Implicant: a product term that could be used to cover minterms of the function

- a product term of a Boolean function if the function has an output 1 for all minterms of the product term

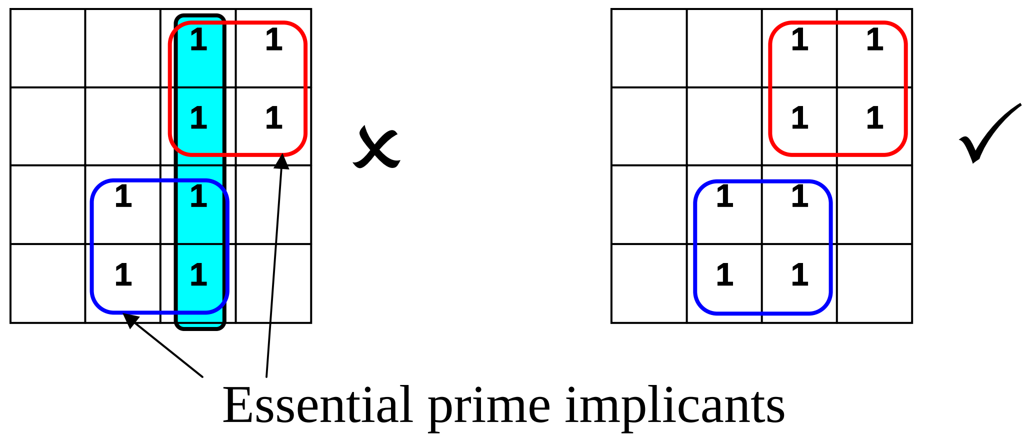
Simplest SOP Expressions (2/3)

- A *prime implicant* is a product term obtained by combining the maximum possible number of minterms from adjacent squares in the map.
- A *Prime Implicant (PI)* is group that is expanded as big as possible (group size must be a power of 2)
- Use bigger groupings (prime implicants) where possible.



Simplest SOP Expressions (3/3)

- No redundant groups:



- An *essential prime implicant* is a prime implicant that includes at least one minterm that is not covered by any other prime implicant.

Simplest SOP Expressions (1/6)

- Algorithm 1 (non optimal):

  1. Count the number of adjacencies for each minterm on the K-map.
  2. Select an uncovered minterm with the fewest number of adjacencies. Make an arbitrary choice if more than one choice is possible.
  3. Generate a prime implicant for this minterm and put it in the cover. If this minterm is covered by more than one prime implicant, select the one that covers the most uncovered minterms.
  4. Repeat steps 2 and 3 until all the minterms have been covered.

Simplest SOP Expressions (2/6)

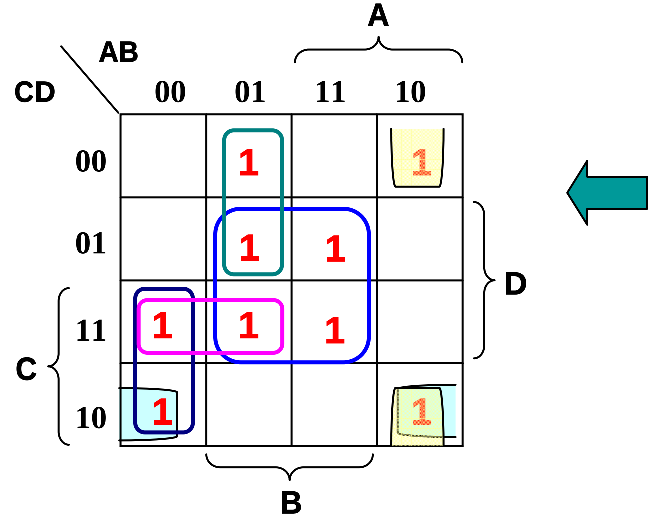
- Algorithm 2 (non optimal):

  1. Circle all prime implicants on the K-map.
  2. Identify and select all essential prime implicants for the cover.
  3. Select a minimum subset of the remaining prime implicants to complete the cover, that is, to cover those minterms not covered by the essential prime implicants.

Simplest SOP Expressions (3/6)

- Example:

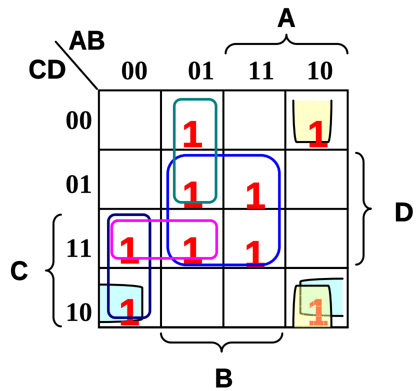
$$f(A,B,C,D) = \sum m(2,3,4,5,7,8,10,13,15)$$

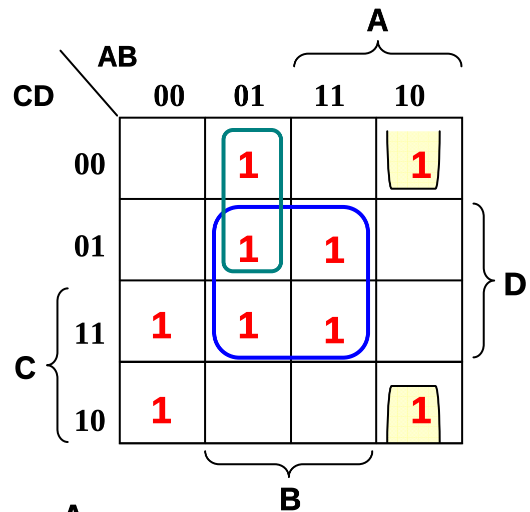


All prime implicants

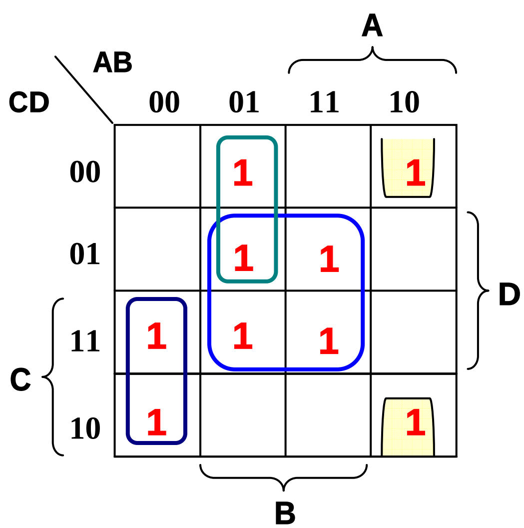
These are all the prime implicants; but do we need them all?

Simplest SOP Expressions (4/6)



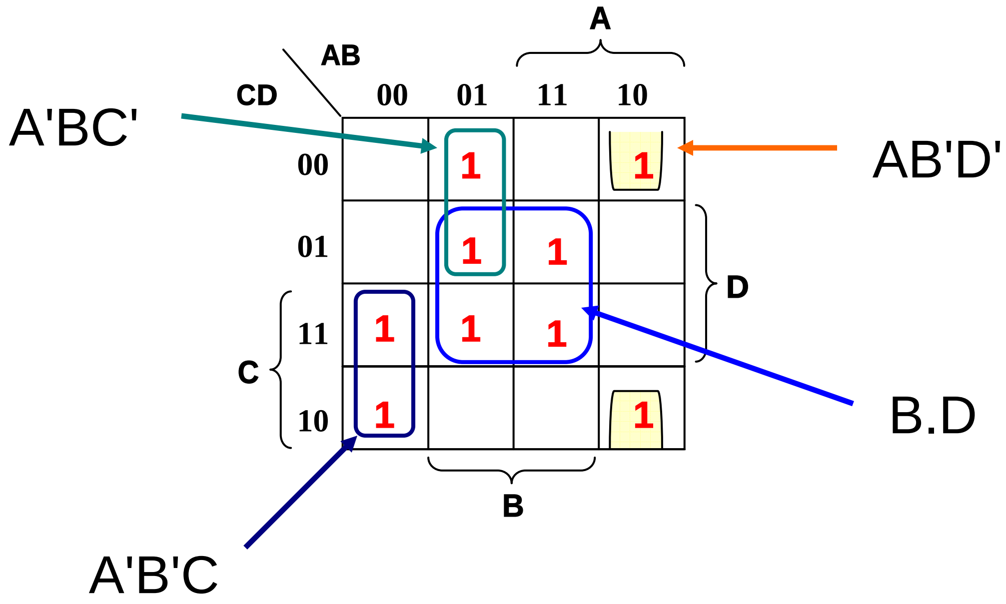


Essential prime implicants



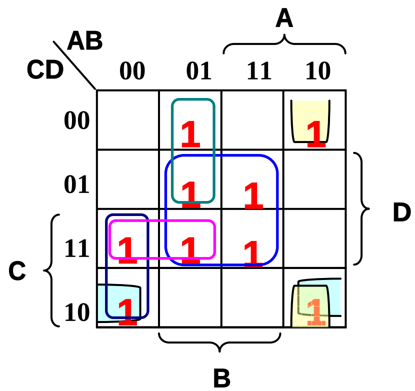
Minimum cover

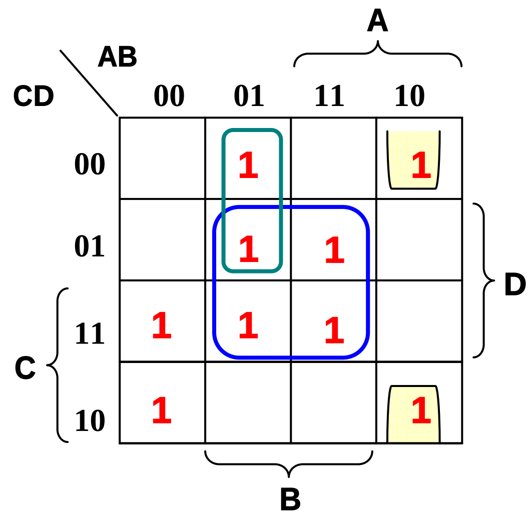
Simplest SOP Expressions (5/6)



f(A,B,C,D) = B.D + A'.B'.C + A.B'.D' + A'.B.C'

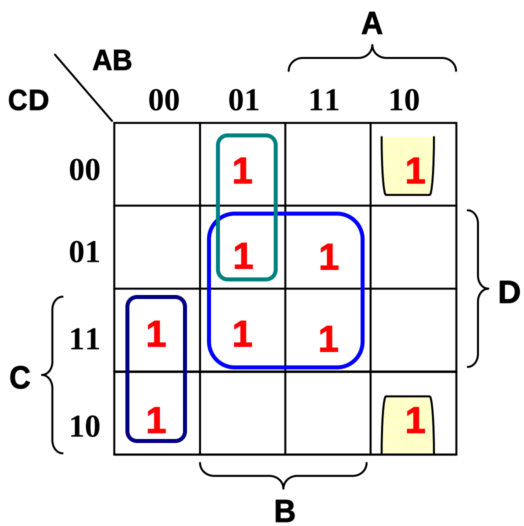
Simplest SOP Expressions (6/6)





Essential prime implicants

SUMMARY



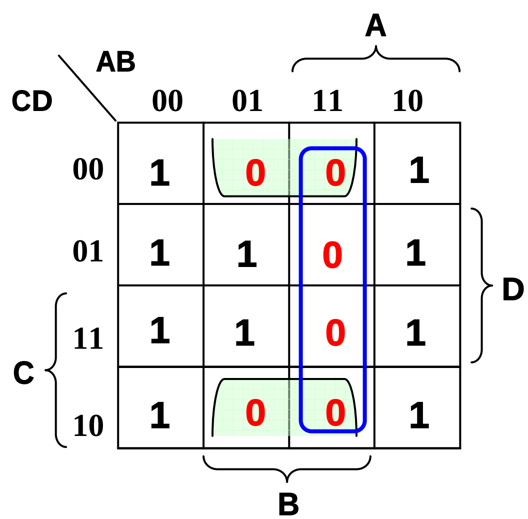
Minimum cover

f(A,B,C,D) = B.D + A'.B'.C + A.B'.D' + A'.B.C'

Getting POS Expressions (1/2)

- Simplified POS expression can be obtained by grouping the maxterms (i.e. 0s) of given function.
- Example:

  Given $F = \sum m(0,1,2,3,5,7,8,9,10,11)$, we first draw the K-map, then group the maxterms together:



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

Getting POS Expressions (2/2)

K-map of **F**

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

K-map of **F'**

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

- This gives the SOP of F' to be:

  F' = B.D' + A.B

- To get POS of F, we have:

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

  = (B.D')'.(A.B)' DeMorgan

  = (B'+D).(A'+B') DeMorgan

Don’t-care Conditions (1/3)

- In certain problems, some outputs are not specified.
- These outputs can be either ‘1’ or ‘0’.
- They are called don’t-care conditions, denoted by X (or sometimes, d).
- Example: An odd parity generator for BCD code which has 6 unused combinations.

No.	A	B	C	D	**P**
0	0	0	0	0	**1**
1	0	0	0	1	**0**
2	0	0	1	0	**0**
3	0	0	1	1	**1**
4	0	1	0	0	**0**
5	0	1	0	1	**1**
6	0	1	1	0	**1**
7	0	1	1	1	**0**
8	1	0	0	0	**0**
9	1	0	0	1	**1**
10	1	0	1	0	**X**
11	1	0	1	1	**X**
12	1	1	0	0	**X**
13	1	1	0	1	**X**
14	1	1	1	0	**X**
15	1	1	1	1	**X**

Don’t-care Conditions (2/3)

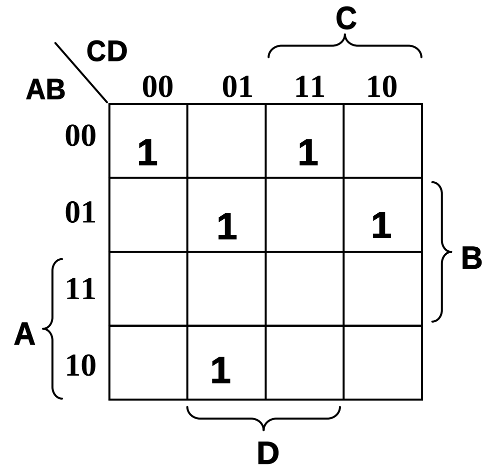
- Don’t-care conditions can be used to help simplify Boolean expression further in K-maps.
- They could be chosen to be either ‘1’ or ‘0’, depending on which gives the simpler expression.
- We usually use the notation Σd to denote the set of don’t-care minterms. For example, the function P in the odd-parity generator for BCD can be written as:

  $P = \Sigma m(0, 3, 5, 6, 9) + \Sigma d(10, 11, 12, 13, 14, 15)$

Don’t-care Conditions (3/3)

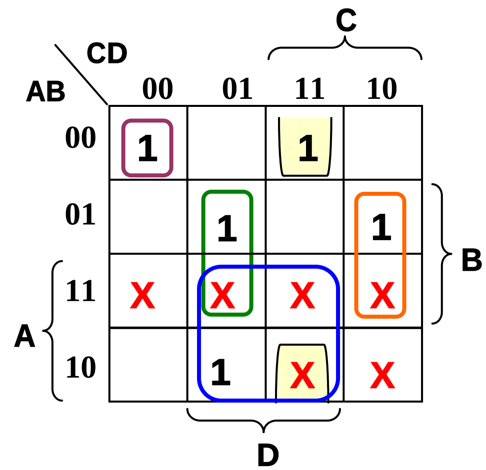
- For comparison:
  - ❖WITHOUT don’t-cares:

    P = A'.B'.C'.D’ + A'.B'.C.D + A'.B.C'.D + A'.B.C.D' + A.B'.C'.D



  - ❖WITH don’t-cares:

    P = A'.B'.C'.D' + B'.C.D + B.C'.D + B.C.D' + A.D



Examples (1/6)

- Example #1:

  $f(A,B,C,D) = A.B.C + B'.C.D' + A.D + B'.C'.D'$

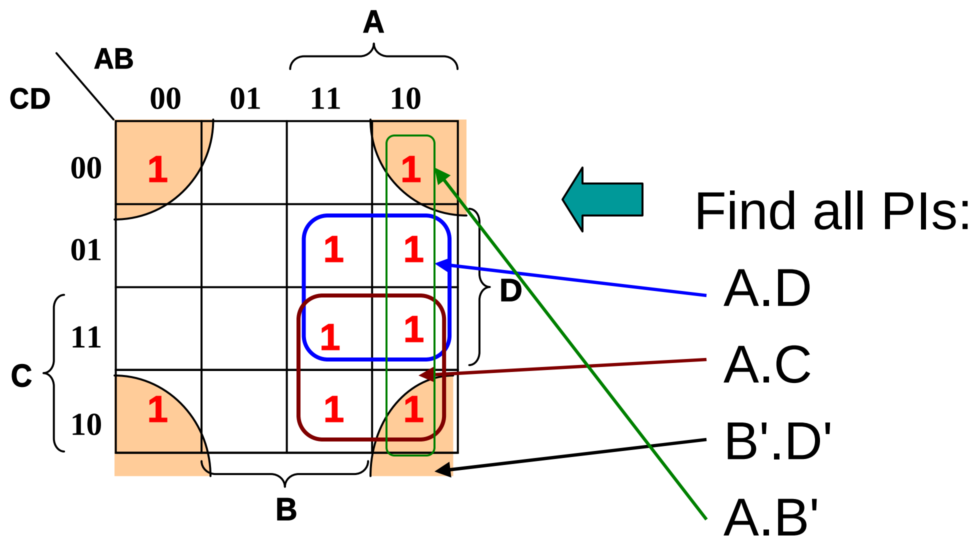
CD \ AB	00	01	11	10
00	1			1
01			1	1
11			1	1
10	1		1	1

Fill in the 1's.

Examples (2/6)

- Example #1:

  f(A,B,C,D) = A.B.C + B'.C.D' + A.D + B'.C'.D'



A.D, A.C and B'.D' are EPIs, and they cover all the minterms.

So the answer is: f(A,B,C,D) = A.D + A.C + B'.D'

Examples (3/6)

- Example #2 (with don't cares):

$$f(A,B,C,D) = \sum m(2,8,10,15) + \sum d(0,1,3,7)$$

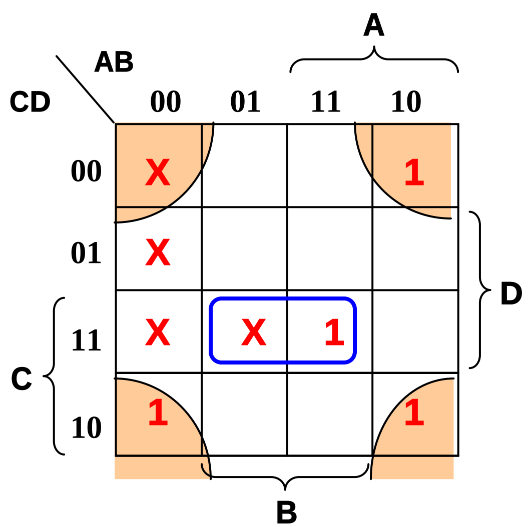
CD \ AB	00	01	11	10
00	X			1
01	X			
11	X	X	1	
10	1			1

Fill in the 1's and X's.

Examples (4/6)

- Example #2 (with don't cares):

  $f(A,B,C,D) = \sum m(2,8,10,15) + \sum d(0,1,3,7)$



Do we need to have an additional term A'.B' to cover the 2 remaining x's?

No, because all the 1's (minterms) have been covered.

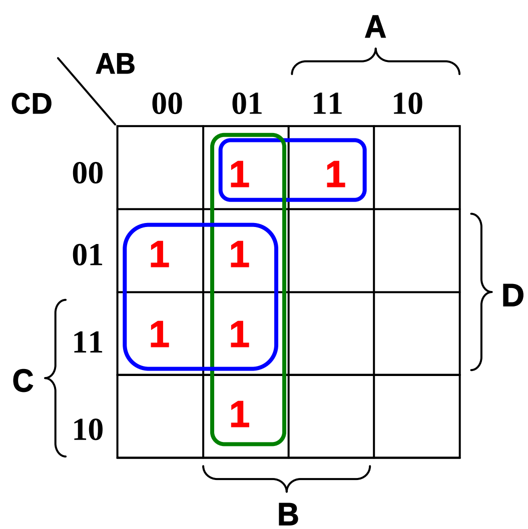
f(A,B,C,D) = B'.D' + B.C.D

Examples (5/6)

- To find simplest POS expression for example #1:

  f(A,B,C,D) = A.B.C + B'.C.D' + A.D + B'.C'.D'

- Draw the K-map of the complement of f, f '.



From K-map,

f ' = A'.B + A'.D + B.C'.D'

Using DeMorgan's theorem,

f = (A'.B + A'.D + B.C'.D')'

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

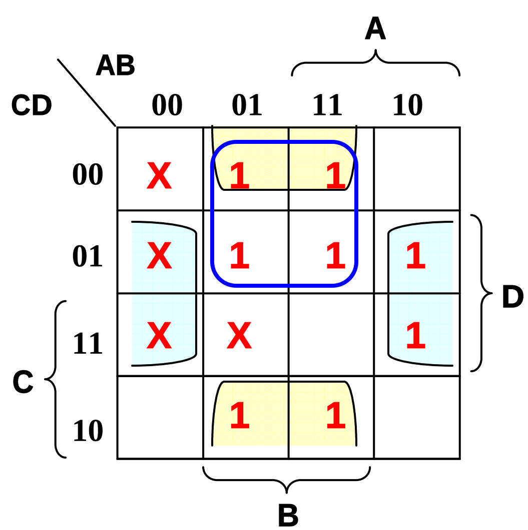
Examples (6/6)

- To find simplest POS expression for example #2:

  f(A,B,C,D) = $\sum m(2,8,10,15) + \sum d(0,1,3,7)$

- Draw the K-map of the complement of f, f '.

  f '(A,B,C,D) = $\sum m(4,5,6,9,11,12,13,14) + \sum d(0,1,3,7)$



From K-map,

f ' = B.C' + B.D' + B'.D

Using DeMorgan's theorem,

f = (B.C' + B.D' + B'.D)'

= (B'+C).(B'+D).(B+D')