

Lecture 8

The Karnaugh Map or K-map

Minimization of Boolean expressions



- *The minimization will result in reduction of the number of gates (resulting from less number of terms) and the number of inputs per gate (resulting from less number of variables per term)*
- *The minimization will reduce cost, efficiency and power consumption.*
- **$y(x+x')=y.1=y$**
- **$y+xx'=y+0=y$**
- **$(x'y+xy')=x\oplus y$**
- **$(x'y'+xy)=(x\oplus y)'$**



Minimum SOP and POS

- The *minimum sum of products (MSOP)* of a function, f , is a SOP representation of f that contains the fewest number of product terms and fewest number of literals of any SOP representation of f .



Minimum SOP and POS

- $f = (xyz + x'yz + xy'z + \dots)$

Is called sum of products.

The + is sum operator which is an OR gate.

The product such as xy is an AND gate for the two inputs x and y .

Example

- **Minimize the following Boolean function using sum of products (SOP):**
- **$f(a,b,c,d) = \sum m(3,7,11,12,13,14,15)$**

	abcd	
3	0011	$a'b'cd$
7	0111	$a'bcd$
11	1011	$ab'cd$
12	1100	$abc'd'$
13	1101	$abc'd$
14	1110	$abcd'$
15	1111	$abcd$



Example

$$\begin{aligned}f(a,b,c,d) &= \sum m(3,7,11,12,13,14,15) \\&= a'b'cd + a'bcd + ab'cd + abc'd' + abc'd + abcd' + \\&\quad abcd \\&= cd(a'b' + a'b + ab') + ab(c'd' + c'd + cd' + cd) \\&= cd(a'[b' + b] + ab') + ab(c'[d' + d] + c[d' + d]) \\&= cd(a'[1] + ab') + ab(c'[1] + c[1]) \\&= ab + ab'cd + a'cd \\&= ab + cd(ab' + a') \\&= ab + cd(a + a')(a' + b') \\&= ab + a'cd + b'cd \\&= ab + cd(a' + b')\end{aligned}$$



Minimum product of sums (MPOS)

- The *minimum product of sums (MPOS)* of a function, f , is a POS representation of f that contains the fewest number of sum terms and the fewest number of literals of any POS representation of f .
- The zeros are considered exactly the same as ones in the case of sum of product (SOP)



Example

$$\begin{aligned} f(a,b,c,d) &= \prod M(0,1,2,4,5,6,8,9,10) \\ &= \sum m(3,7,11,12,13,14,15) \\ &= [(a+b+c+d)(a+b+c+d')(a+b'+c'+d') \\ &\quad (a'+b+c'+d')(a'+b'+c+d)(a'+b'+c+d') \\ &\quad (a'+b'+c'+d)(a'+b'+c'+d')] \end{aligned}$$



The Karnaugh Map

- Feel a little difficult using Boolean algebra laws, rules, and theorems to simplify logic?
- A K-map provides a systematic method for simplifying Boolean expressions and, if properly used, will produce the simplest SOP or POS expression possible, known as the minimum expression.



What is K-Map

- It's similar to truth table; instead of being organized (i/p and o/p) into columns and rows, the K-map is an array of cells in which each cell represents a binary value of the input variables.
- The cells are arranged in a way so that simplification of a given expression is simply a matter of properly grouping the cells.
- K-maps can be used for expressions with 2, 3, 4, and 5 variables.
 - 3 and 4 variables will be discussed to illustrate the principles.

The 3 Variable K-Map

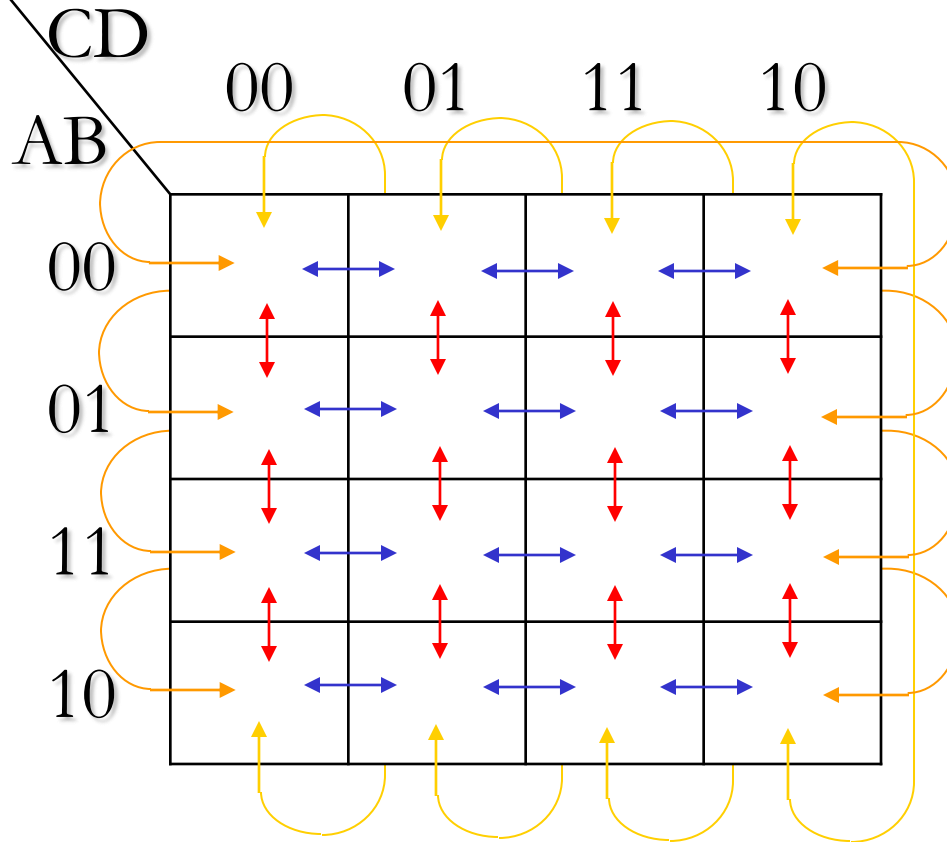
There are 8 cells as shown:

		C	
		0	1
AB	00	$\bar{A}\bar{B}\bar{C}$	$\bar{A}\bar{B}C$
	01	$\bar{A}B\bar{C}$	$\bar{A}BC$
11	11	$AB\bar{C}$	ABC
	10	$A\bar{B}\bar{C}$	$A\bar{B}C$

The 4-Variable K-Map

		CD			
		00	01	11	10
AB	00	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}CD$	$\bar{A}\bar{B}C\bar{D}$
	01	$\bar{A}B\bar{C}\bar{D}$	$\bar{A}B\bar{C}D$	$\bar{A}BCD$	$\bar{A}BC\bar{D}$
	11	$AB\bar{C}\bar{D}$	$AB\bar{C}D$	$ABCD$	$ABC\bar{D}$
	10	$A\bar{B}\bar{C}\bar{D}$	$A\bar{B}\bar{C}D$	$A\bar{B}CD$	$A\bar{B}C\bar{D}$

Cell Adjacency





K-Map SOP Minimization

- The K-Map is used for simplifying Boolean expressions to their minimal form.
- A minimized SOP expression contains the fewest possible terms with fewest possible variables per term.
- Generally, a minimum SOP expression can be implemented with fewer logic gates than a standard expression.

Mapping a Standard SOP Expression

For an SOP expression in standard form:

- A 1 is placed on the K-map for each product term in the expression.
- Each 1 is placed in a cell corresponding to the value of a product term $A\bar{B}C$
- Example: for the product term $A\bar{B}C$, a 1 goes in the 101 cell on a 3-variable map.

		C	
		0	1
AB	00	$\bar{A}\bar{B}\bar{C}$	$\bar{A}\bar{B}C$
	01	$\bar{A}B\bar{C}$	$\bar{A}BC$
11	$AB\bar{C}$	ABC	
10	$A\bar{B}\bar{C}$	$A\bar{B}C$	

Mapping a Standard SOP Expression (full example)

The expression:

$$\overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + A\overline{B}\overline{C} + A\overline{B}C$$



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

Practice:

$$\overline{A}BC + \overline{A}BC + A\overline{B}\overline{C} + ABC$$

$$\overline{A}BC + \overline{A}BC + A\overline{B}\overline{C}$$

$$\overline{A}\overline{B}CD + \overline{A}\overline{B}\overline{C}D + A\overline{B}\overline{C}D + ABCD + A\overline{B}C\overline{D} + \overline{A}\overline{B}C\overline{D} + A\overline{B}C\overline{D}$$



Mapping a Nonstandard SOP Expression

- A Boolean expression must be in standard form before you use a K-map.
 - If one is not in standard form, it must be converted.
- You may use the procedure mentioned earlier or use numerical expansion.

Mapping a Nonstandard SOP Expression

Numerical Expansion of a Nonstandard product term

- Assume that one of the product terms in a certain 3-variable SOP expression is $A\bar{B}\bar{C}$.
- It can be expanded numerically to standard form as follows:
 - **Step 1:** Write the binary value of the two variables and attach a 0 for the missing variable C : 100.
 - **Step 2:** Write the binary value of the two variables and attach a 1 for the missing variable \bar{C} : 101.
- The two resulting binary numbers are the values of the standard SOP terms \rightarrow m_4 and m_5 .
- If the assumption that one of the product term

Mapping a Nonstandard SOP Expression

Map the following SOP expressions on K-maps:

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

$$\bar{B}\bar{C} + A\bar{B} + AB\bar{C} + A\bar{B}C\bar{D} + \bar{A}\bar{B}\bar{C}D + A\bar{B}CD$$



K-Map Simplification of SOP Expressions

- After an SOP expression has been mapped, we can do the process of *minimization*:
 - Grouping the 1s
 - Determining the minimum SOP expression from the map



Grouping the 1s

- You can group 1s on the K-map according to the following rules by enclosing those adjacent cells containing 1s.
- **The goal** is to maximize the size of the groups and to minimize the number of groups.

Grouping the 1s (rules)

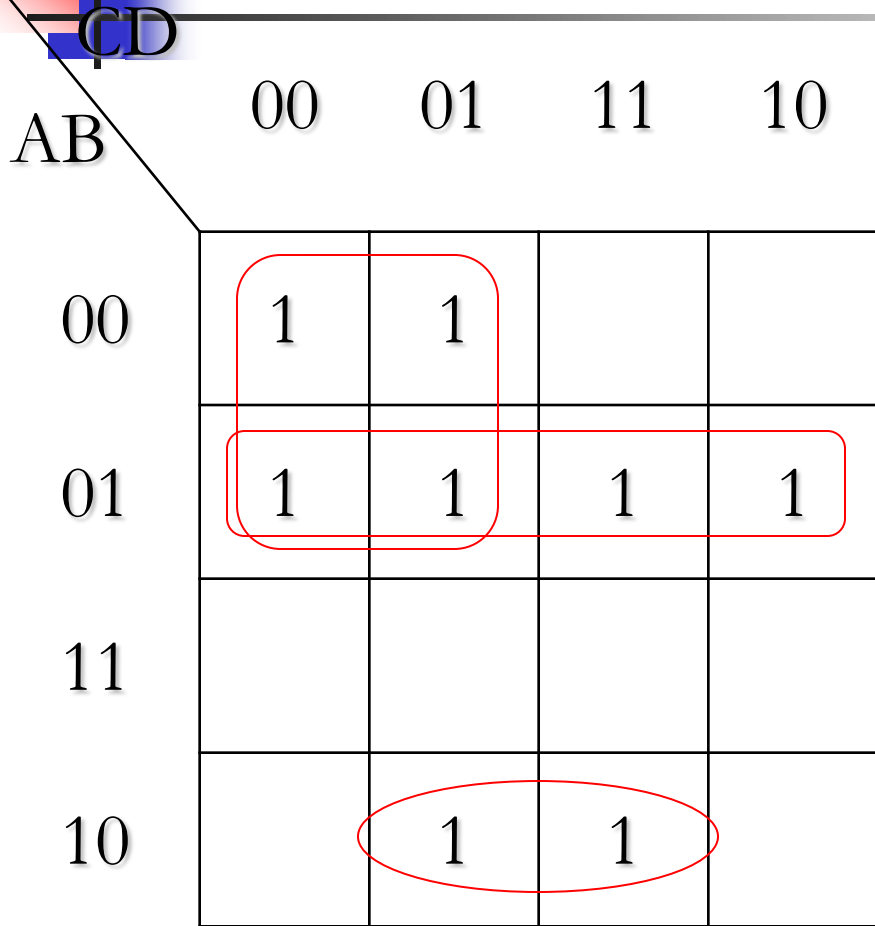
1. A group must contain either 1,2,4,8,or 16 cells (depending on number of variables in the expression)
2. Each cell in a group must be adjacent to one or more cells in that same group, but all cells in the group do not have to be adjacent to each other.
3. Always include the largest possible number of 1s in a group in accordance with rule 1.
4. Each 1 on the map must be included in at least one group. The 1s already in a group can be included in another group as long as the overlapping groups include noncommon 1s.

Grouping the 1s (example)

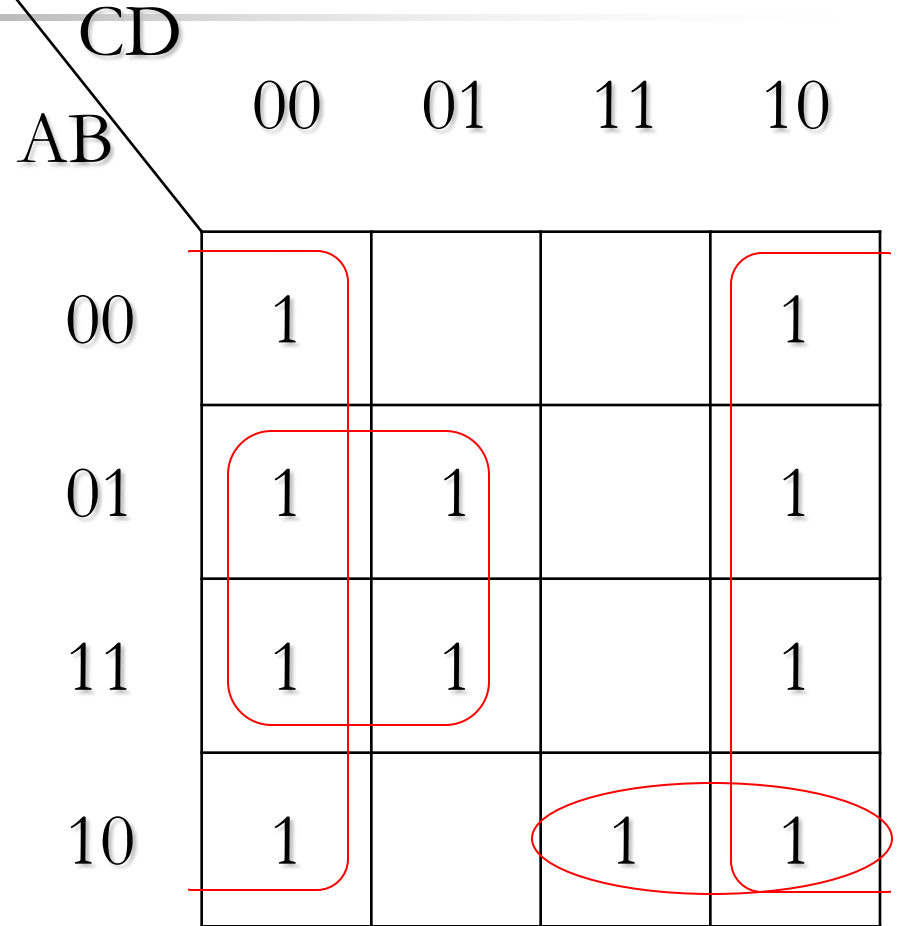
		C	
		0	1
AB	00	1	
	01		1
11	1	1	
10			

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

Grouping the 1s (example)



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



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

Determining the Minimum SOP Expression from the Map



- The following rules are applied to find the minimum product terms and the minimum SOP expression:
 1. Group the cells that have 1s. Each group of cell containing 1s creates one product term composed of all variables that occur in only one form (either complemented or complemented) within the group. Variables that occur both complemented and uncomplemented within the group are eliminated → called *contradictory variables*.

Determining the Minimum SOP Expression from the Map

2. Determine the minimum product term for each group.

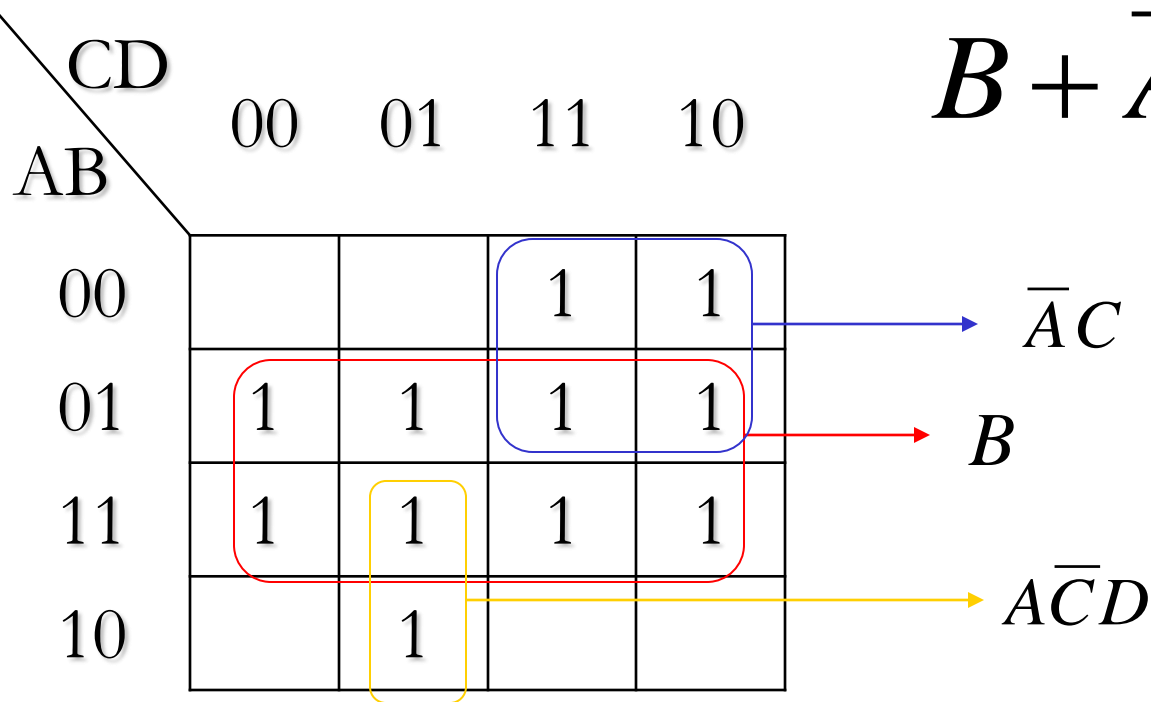
- For a 3-variable map:
 1. A 1-cell group yields a 3-variable product term
 2. A 2-cell group yields a 2-variable product term
 3. A 4-cell group yields a 1-variable product term
 4. An 8-cell group yields a value of 1 for the expression.
- For a 4-variable map:
 1. A 1-cell group yields a 4-variable product term
 2. A 2-cell group yields a 3-variable product term
 3. A 4-cell group yields a 2-variable product term
 4. An 8-cell group yields a a 1-variable product term
 5. A 16-cell group yields a value of 1 for the expression.



Determining the Minimum SOP Expression from the Map

3. When all the minimum product terms are derived from the K-map, they are summed to form the minimum SOP expression.

Determining the Minimum SOP Expression from the Map (example)



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

Determining the Minimum SOP Expression from the Map (exercises)

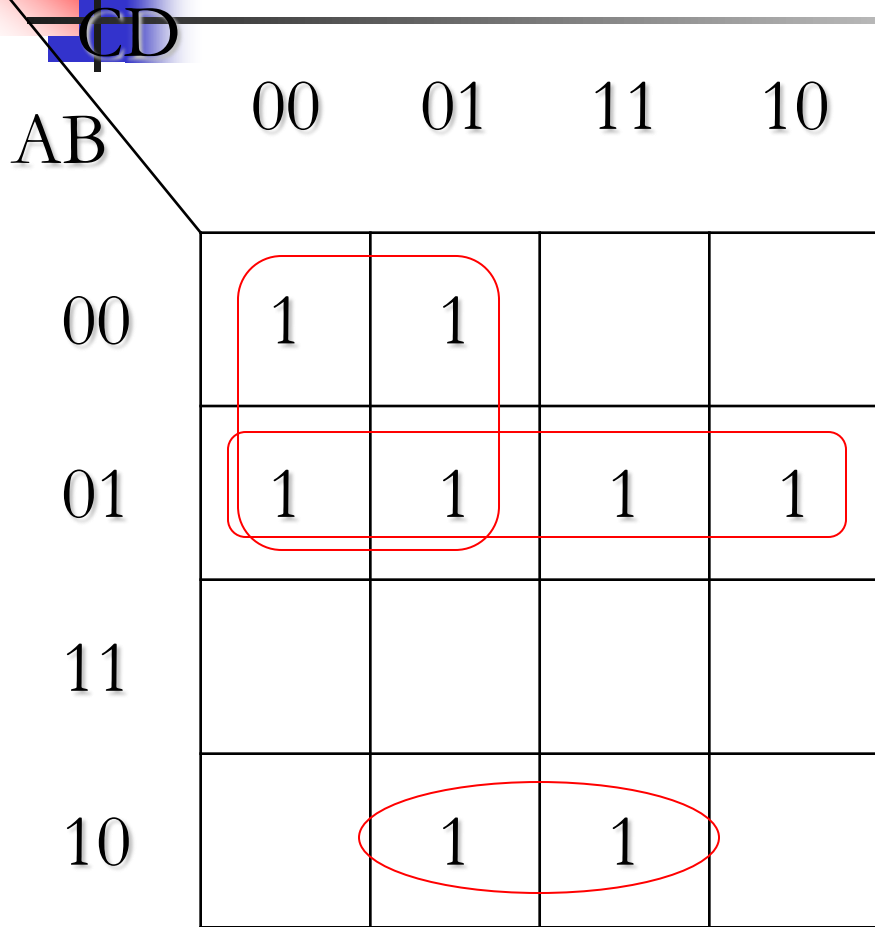
		C	
		0	1
AB	00	1	
	01		1
11	1	1	
10			

$$AB + BC + \overline{A}\overline{B}\overline{C}$$

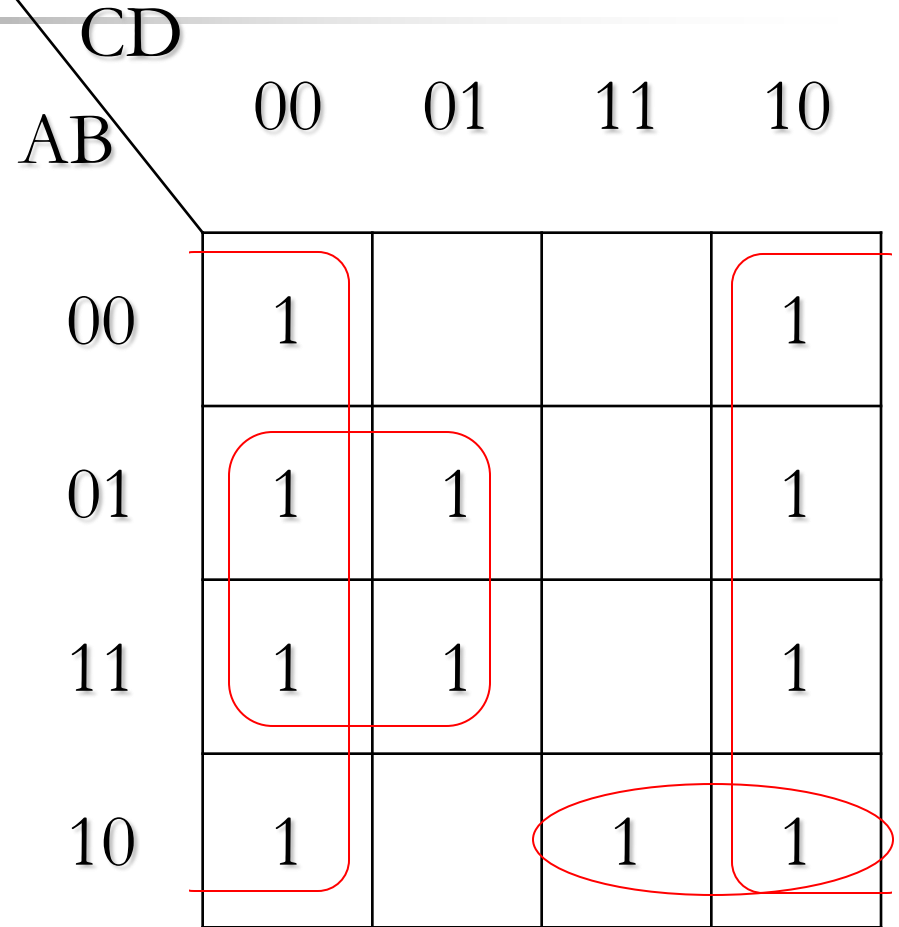
		C	
		0	1
AB	00	1	1
	01	1	
11		1	
10	1	1	

$$\overline{B} + \overline{A}\overline{C} + AC$$

Expression from the Map (exercises)



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



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

Practicing K-Map (SOP)

$$\overline{A}\overline{B}C + \overline{A}B\overline{C} + \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} + A\overline{B}\overline{C}$$

$$\overline{B} + \overline{A}C$$

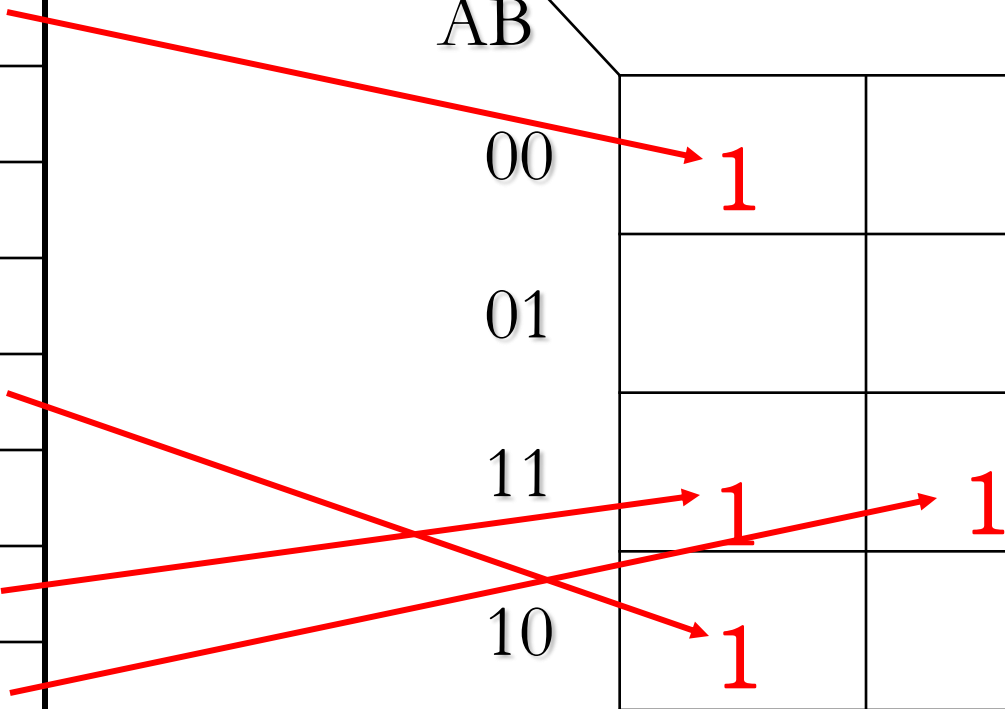
$$\overline{B}\overline{C}\overline{D} + \overline{A}B\overline{C}\overline{D} + A\overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}C\overline{D} + A\overline{B}C\overline{D} +$$
$$\overline{A}\overline{B}C\overline{D} + \overline{A}B\overline{C}\overline{D} + A\overline{B}\overline{C}\overline{D} + A\overline{B}C\overline{D}$$

$$\overline{D} + \overline{B}C$$

Mapping Directly from a Truth Table

I/P			O/P
A	B	C	X
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

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





“Don’t Care” Conditions

- Sometimes a situation arises in which some input variable combinations are not allowed, i.e. BCD code:
 - There are six invalid combinations: 1010, 1011, 1100, 1101, 1110, and 1111.
- Since these unallowed states will never occur in an application involving the BCD code → they can be treated as “don’t care” terms with respect to their effect on the output.
- The “don’t care” terms can be used to advantage on the K-map (how? see the next

"Don't Care" Conditions

INPUTS				O/P
A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

	CD			
	00	01	11	10
AB				
00				
01			1	
11	x	x	x	x
10	1	1	x	x

Without "don't care"
 $Y = \overline{A}BC + \overline{A}BCD$

With "don't care"
 $Y = A + BCD$



K-Map POS Minimization

- The approaches are much the same (as SOP) except that with POS expression, 0s representing the standard sum terms are placed on the K-map instead of 1s.

Mapping a Standard POS Expression (full example)

The expression:

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

000

010

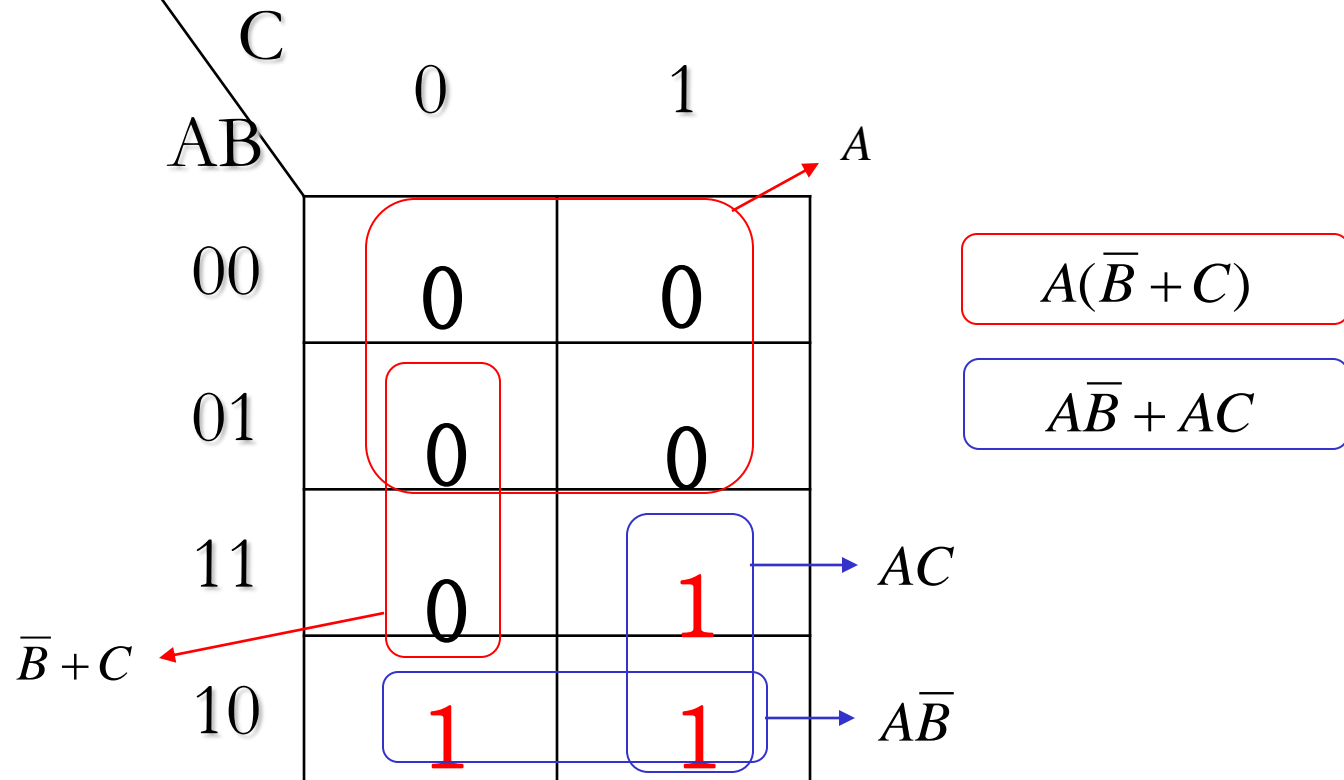
110

101

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

K-map Simplification of POS Expression

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





Karnaugh Maps (K-maps)

- Karnaugh maps -- A tool for representing Boolean functions of up to six variables.
- K-maps are tables of rows and columns with entries represent 1`s or 0`s of SOP and POS representations.



Karnaugh Maps (K-maps)

- An n -variable K-map has 2^n cells with each cell corresponding to an n -variable truth table value.
- K-map cells are labeled with the corresponding truth-table row.
- K-map cells are arranged such that adjacent cells correspond to truth rows that differ in only one bit position (*logical adjacency*).



Karnaugh Maps (K-maps)

- If m_i is a minterm of f , then place a 1 in cell i of the K-map.
- If M_i is a maxterm of f , then place a 0 in cell i .
- If d_i is a don't care of f , then place a d or x in cell i .

Examples

- *Two variable K-map*

$$f(A,B) = \sum m(0,1,3) = A'B' + A'B + AB$$

	A	0	1
B	0	1	0
1		1	1

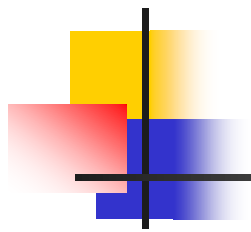
Three variable map

- $f(A,B,C) =$
 $\Sigma m(0,3,5) =$
 $A'B'C' + A'BC + AB'C$

A`B`	A`B	A B	A B`
0 0	0 1	1 1	1 0

C` 0	1 A`B`C`		
C 1		1 A`BC	1 AB`C

Maxterm example



		$(A+B)$	$(A+B')$	$(A'+B')$	$(A'+B)$
		$A'B'$	$A'B$	AB	AB'
C	C'		0	0	0
C'	C	0		0	

$$f(A,B,C) = \prod M(1,2,4,6,7)$$

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

Note that the complements are (0,3,5) which are the minterms of the previous example

Four variable example

(a) Minterm form. (b) Maxterm form.

$$f(a,b,Q,G) = \sum m(0,3,5,7,10,11,12,13,14,15) = \prod M(1,2,4,6,8,9)$$

		a			
		ab	00	01	11
Q	QG	0	4	12	8
	00	1		1	
	01	1	5	13	9
	11	3	7	15	11
	10	2	6	14	10

b

(a)

		a			
		ab	00	01	11
Q	QG	0	4	12	8
	00		0		0
	01	1	5	13	9
	11	3	7	15	11
	10	2	6	14	10

b

(b)



Simplification of Boolean Functions Using K-maps

- K-map cells that are physically adjacent are also logically adjacent. Also, cells on an edge of a K-map are logically adjacent to cells on the opposite edge of the map.
- If two logically adjacent cells both contain logical 1s, the two cells can be combined to eliminate the variable that has value 1 in one cell's label and value 0 in the other.



Simplification of Boolean Functions Using K-maps

- This is equivalent to the algebraic operation, $aP + a'P = P$ where P is a product term not containing a or a' .
- A group of cells can be combined only if all cells in the group have the same value for some set of variables.



Simplification Guidelines for K-maps

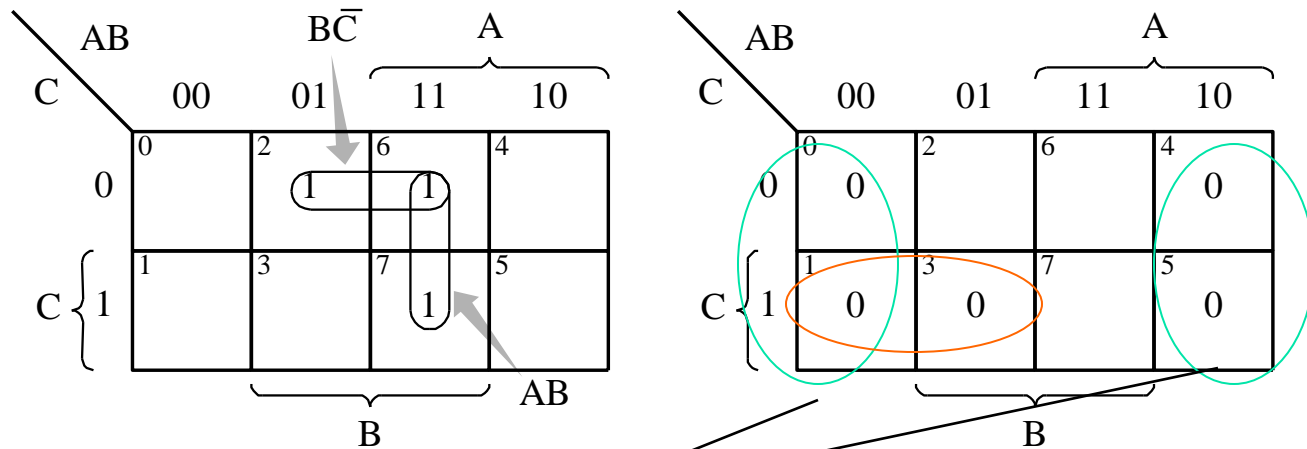
- Always combine as many cells in a group as possible. This will result in the fewest number of literals in the term that represents the group.
- Make as few groupings as possible to cover all minterms. This will result in the fewest product terms.
- Always begin with the largest group, which means if you can find eight members group is better than two four groups and one four group is better than pair of two-group.

Example

Simplify $f = A'BC' + ABC' + ABC$ using;

(a) Sum of minterms. (b) Maxterms.

- Each cell of an n -variable K-map has n logically adjacent cells.



$$F' = B' + A'C$$

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

a- $f(A,B,C) = AB + BC'$

b- $f(A,B,C) = B(A + C')$

Example simplify

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

CD \ AB		A			
		00	01	11	10
C	00	0	4 1	12	8 1
	01	1	5 1	13 1	9
	11	3 1	7 1	15 1	11
	10	2 1	6	14	10 1
		B		D	

(a)

CD \ AB		A			
		00	01	11	10
C	00	0	4 1	12	8 1
	01	1	5 1	13 1	9
	11	3 1	7 1	15 1	11
	10	2 1	6	14	10 1
		B		D	

(b)

CD \ AB		A			
		00	01	11	10
C	00	0	4 1	12	8 1
	01	1	5 1	13 1	9
	11	3 1	7 1	15 1	11
	10	2 1	6	14	10 1
		B		D	

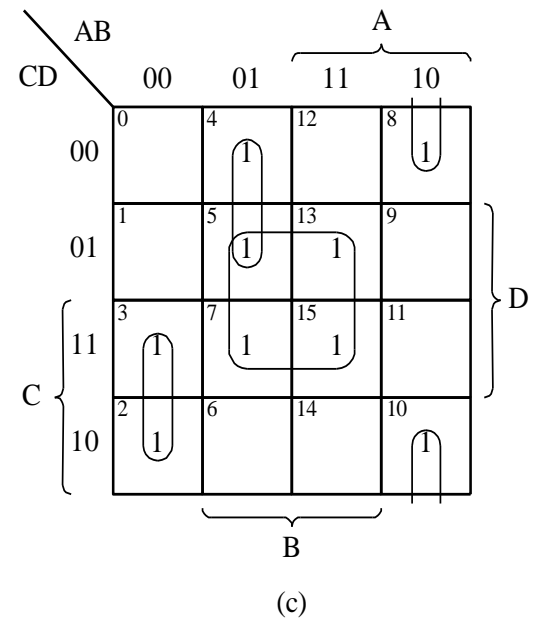
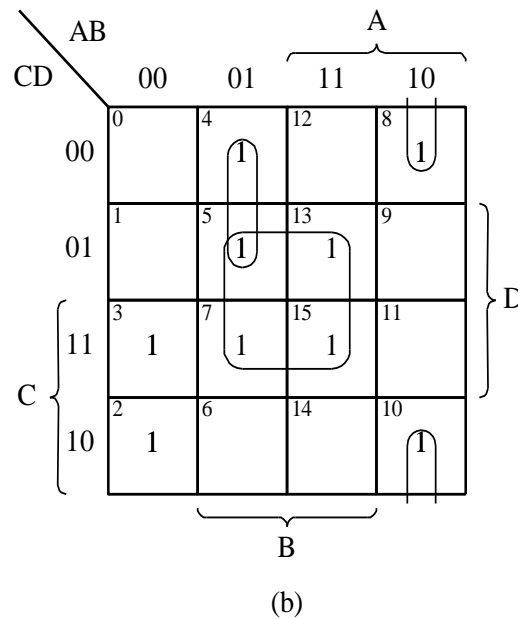
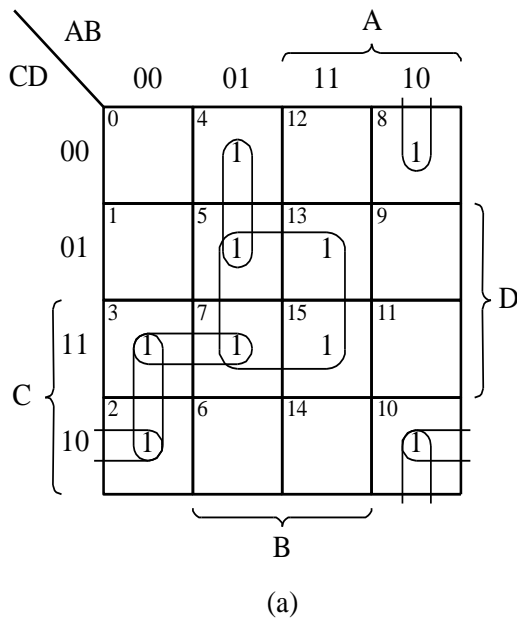
(c)

CD \ AB		A			
		00	01	11	10
C	00	0	4 1	12	8 1
	01	1	5 1	13 1	9
	11	3 1	7 1	15 1	11
	10	2 1	6	14	10 1
		B		D	

(d)

Example Multiple selections

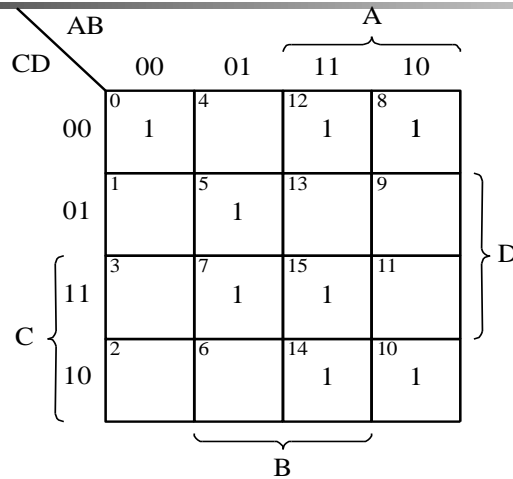
$$f(A,B,C,D) = \sum m(2,3,4,5,7,8,10,13,15)$$



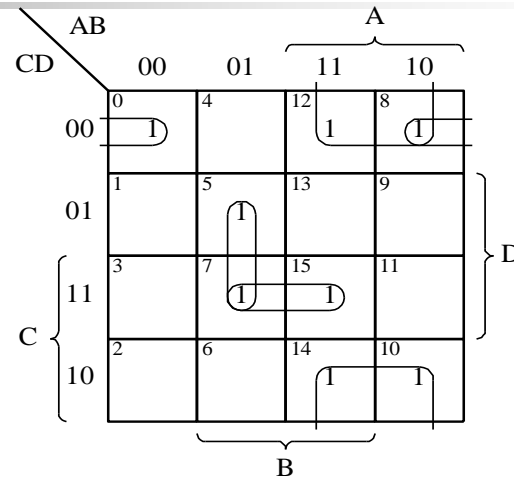
c produces less terms than a

Example Redundant selections

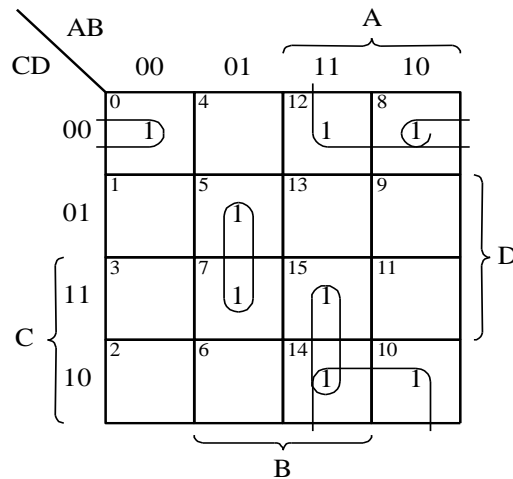
$$f(A,B,C,D) = \sum m(0,5,7,8,10,12,14,15)$$



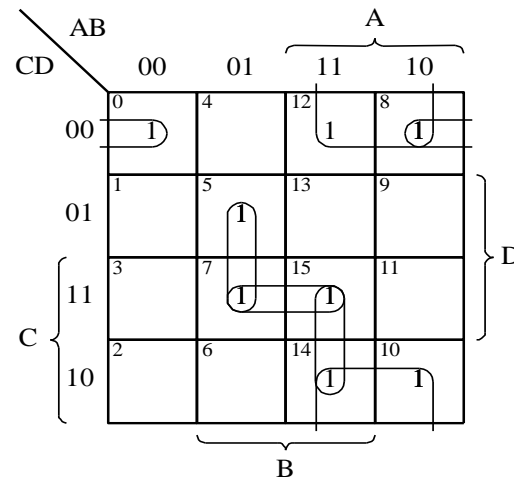
(a)



(b)

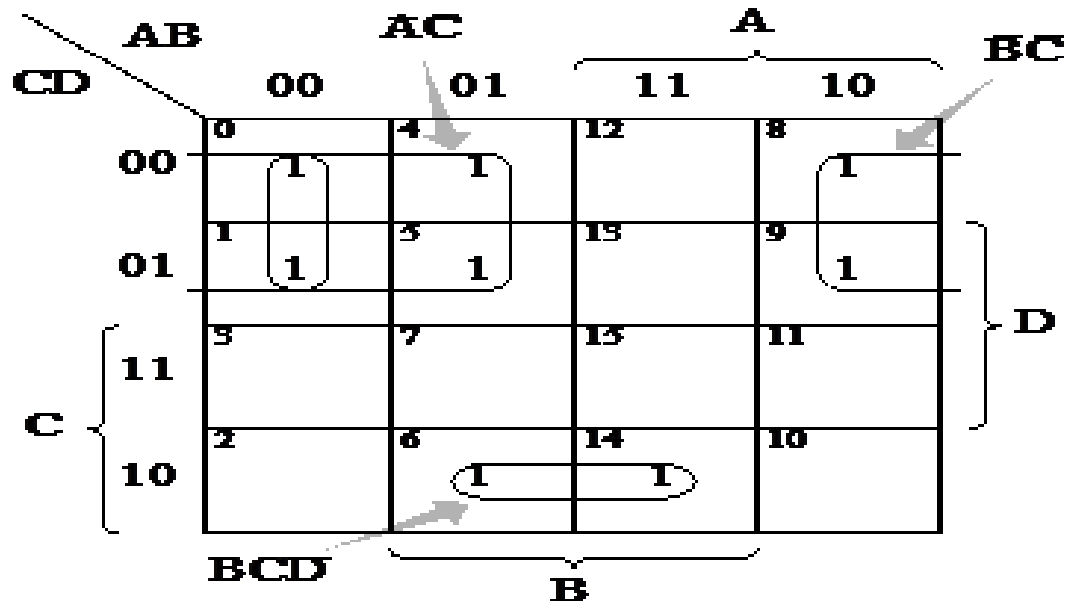


(c)

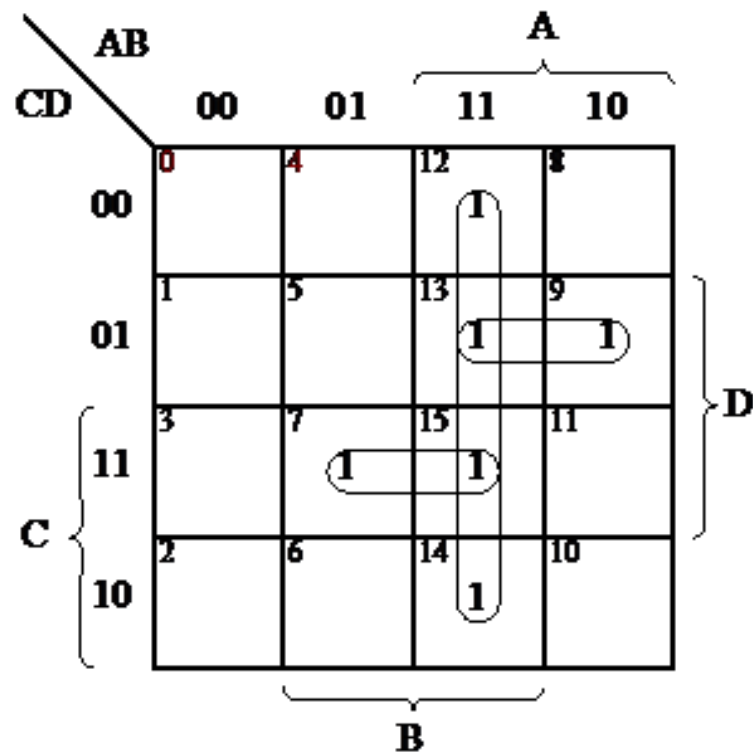


(d)

Example



Example



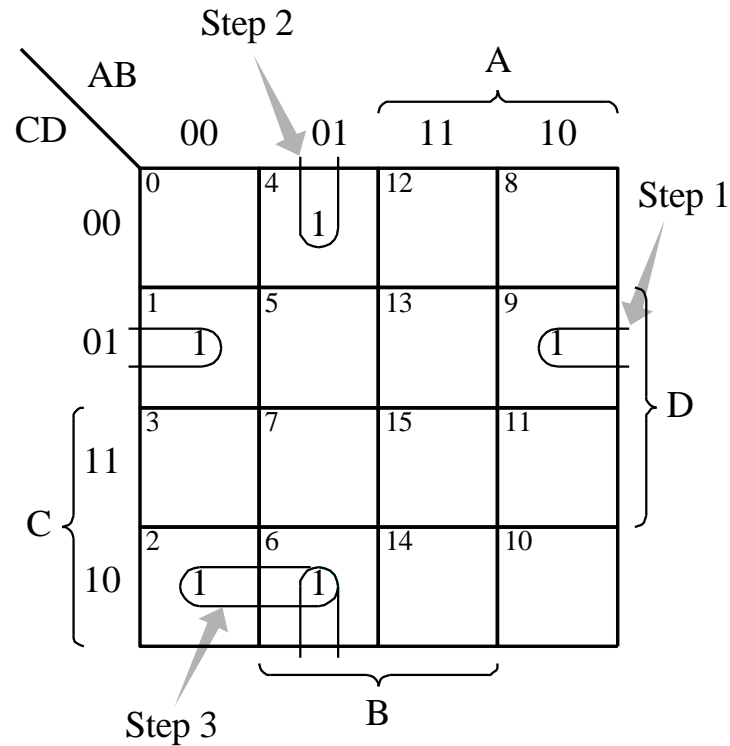
(a)

Example

CD \ AB	A			
	00	01	11	10
00	0 1	4 1	12 0	8 1
01	1 1	5 1	13 0	9 0
11	3 1	7 0	15 0	11 1
10	2 1	6 1	14 0	10 1

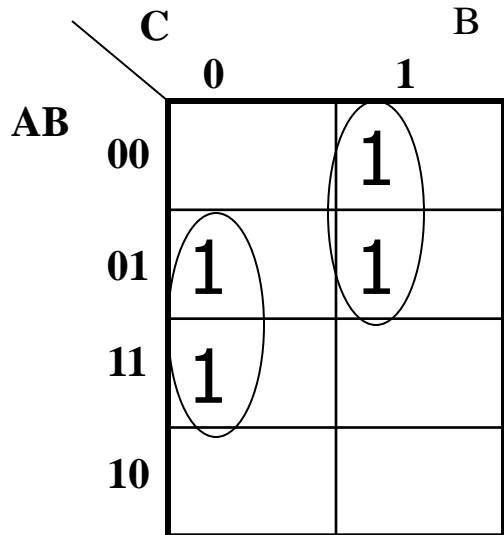
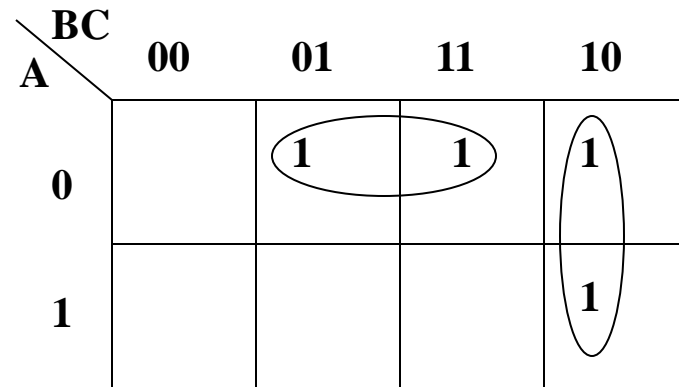
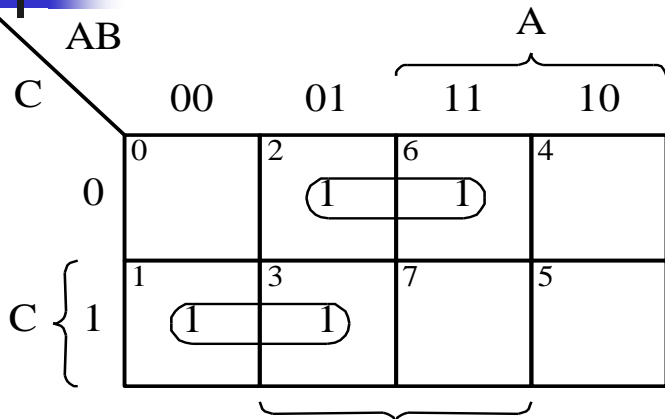
(b)

$$f(A, B, C, D) = \sum m(1, 2, 4, 6, 9)$$



Different styles of drawing maps

$$f(A,B,C) = \sum m(1,2,3,6) = A'C + BC'$$



Don't-care condition

		B'		B	
		00	01	11	10
A'	0	1	x	x	1
	1	0	x	0	1
		C'		C	

- Minterms that may produce either 0 or 1 for the function.
- They are marked with an 'x' in the K-map.
- This happens, for example, when we don't input certain minterms to the Boolean function.
- These don't-care conditions can be used to provide further simplification of the algebraic expression.

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

$$d = A'B'C + A'BC + AB'C$$

$$F = A' + BC'$$

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

Five variable K-maps

Use Two Four-variable K-Maps

$$f(a,b,c,d,e) = \sum m(0,5,7,13,15,16,21,23,29,31)$$

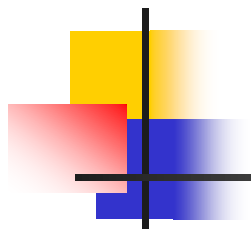
a	f
16	1
17	
18	
19	
20	
21	1
22	
23	1
24	
25	
26	
27	
28	
29	1
30	
31	1

a` = 0 map

bc \ de	00	01	11	10
00	1			
01		1	1	
11		1	1	
10				

a = 1 map

bc \ de	00	01	11	10
00	1			
01		1	1	
11		1	1	
10				



$a' = 0$ map

bc \ de	00	01	11	10
00	1			
01		1	1	
11		1	1	
10				

$a = 1$ map

bc \ de	00	01	11	10
00	1			
01		1	1	
11		1	1	
10				

$$F1 = a'b'c'd'e' + a'ce,$$
$$f(a,b,c,d,e) = f1 + f2$$

$$F2 = ace + ab'c'd'e'$$

$$F = (a + a')ce + (a + a')b'c'd'e'$$
$$= ce + b'c'd'e'$$