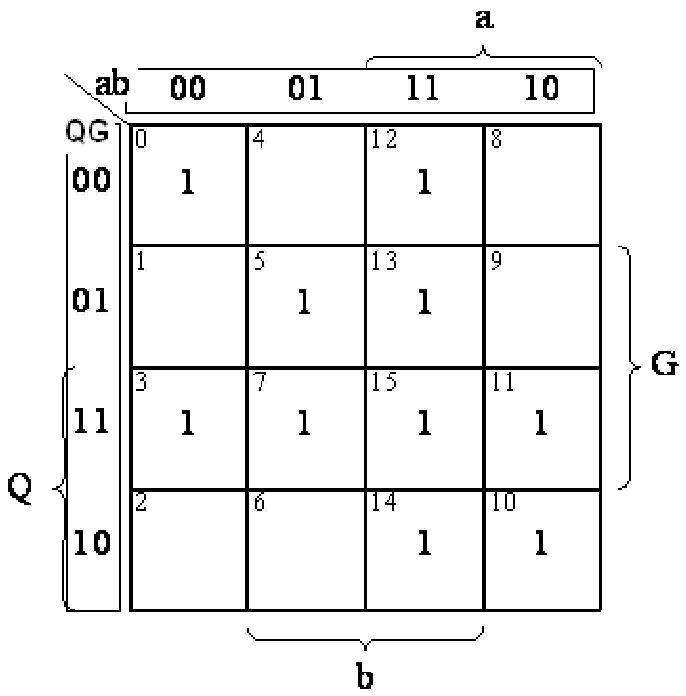
MINIMIZATION TECHNIQUES

- **Boolean Algebra:**
  - Used maximum upto 4-variables or 5-variable (beyond it, very becomes very cumbersome)
- **Karnaugh maps (K-maps):**
  - Used maximum upto 6-variables.
- **Variable Entered mapping (VEM technique):**
  - Used upto 7 or 8-variables.
- **Quine Mc-Cluskey method (QM method):**
  - Used for any number of variables.
- **Iterative Consensus method:**
- Method can be applied for function in standard or non-standard form.

K-MAP: 4-VAR AND 5-VAR:



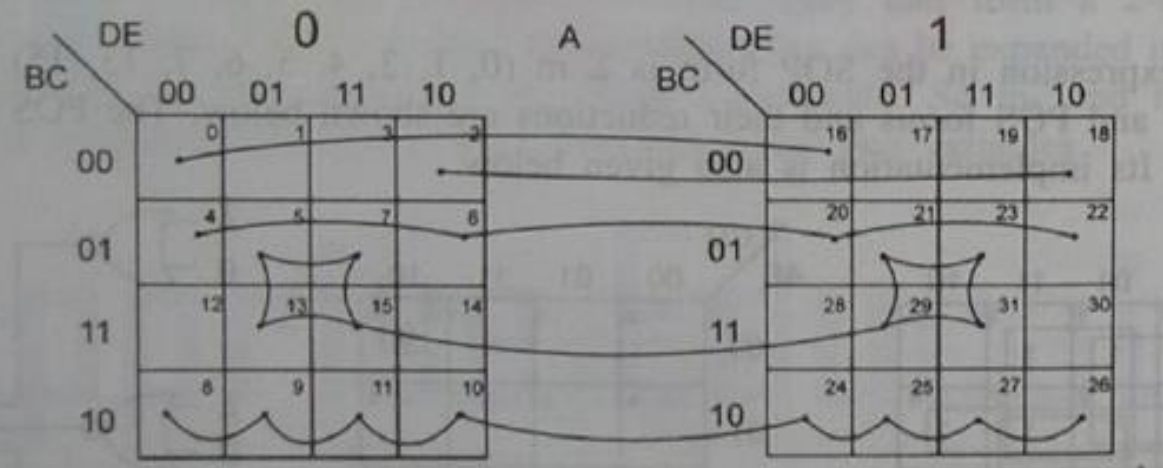


Fig. 6.10 Some possible groupings in a five-variable K-map.

K-MAP: 6-VAR:

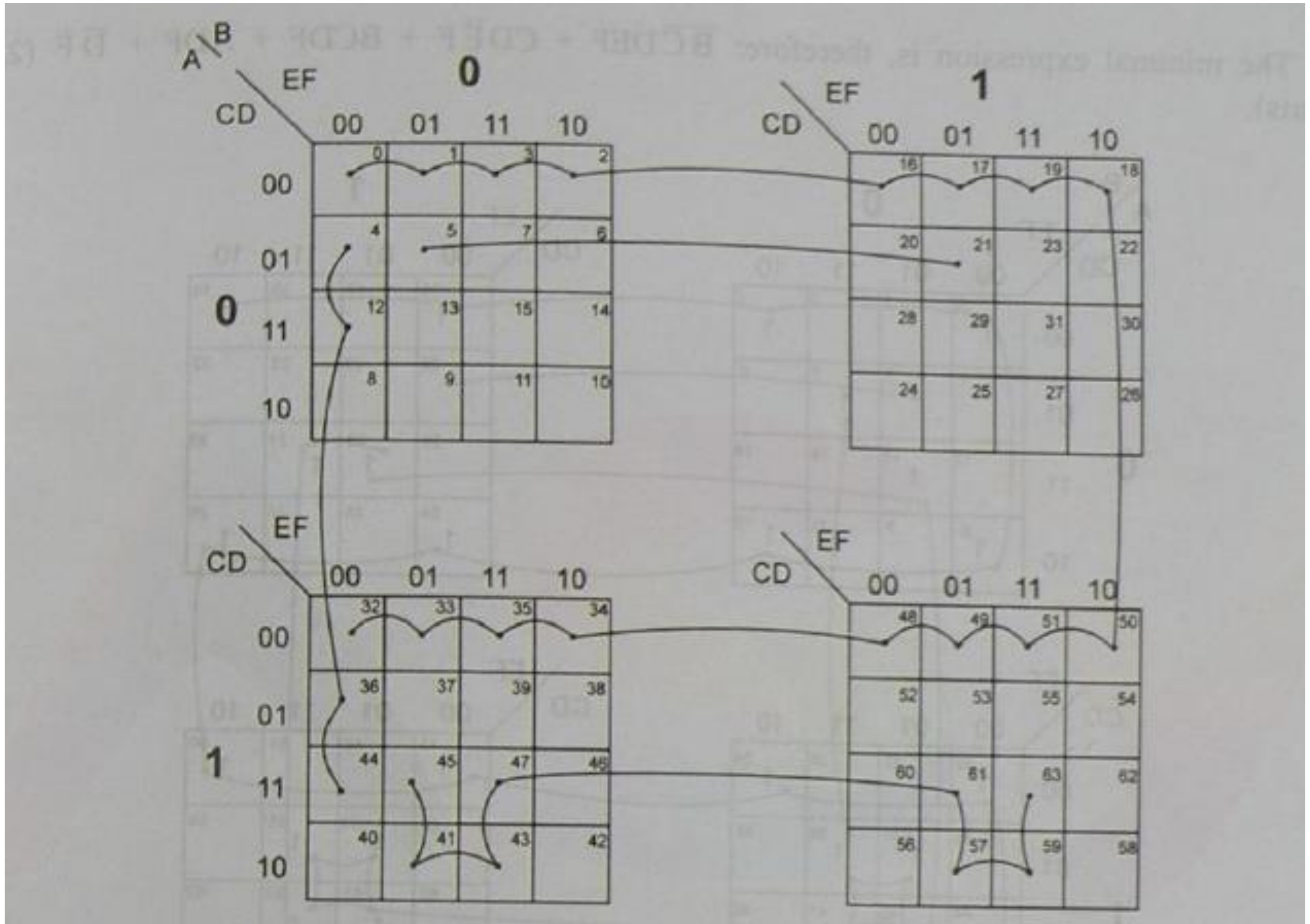


Fig. 6.11 Some possible groupings in a six-variable K-map.

AB = 00
ABCDEF:
00 “0000” TO 00”1111”

01”0000” TO 01”1111”
Xxxx
0 TO 15
16 TO 31

10 “0000” TO 10”1111”

WEIGHT:
MSB (LEFTMOST)
LSB (RIGHTMOST)
6 5 4 3 2 1 0
64 32 16 8 4 2 1

- 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. (<u>Maximize the number of elements in each grouping).</u>

- Make as few groupings as possible to cover all minterms. This will result in the fewest product terms. (<u>Minimize the number of groupings)</u>

K-MAPS WITH DON’T CARES:

- Minterms are assigned values as ‘1’ to get the Sum-of-Product (SOP) expression.
- Maxterms are assigned values as ‘0’ to get the Product-of-Product (POS) expression.
- Don’t cares are assigned values as ‘X’ or ‘d’ or ‘∅’ , alongwith minterms or maxterms to get SOP or POS expression respectively.

Rules:

1. Maximize the number of elements in each grouping.
2. Minimize the number of groupings.

Note: Don’t cares may or may not be a part of those groupings.

IMPORTANT TERMS:

Prime Implicant: set of minterms

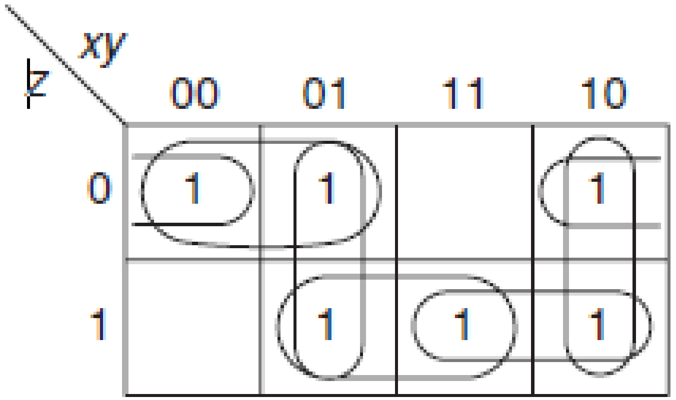
Essential Prime Implicant: set of minterms in which atleast one minterm is unique

Redundant Prime Implicant : set of minterms which are all already covered by some other PI.

<u>**Reduced expression = set of EPI**</u>

CYCLIC PRIME IMPLICANT:

Find the reduced map for the function:



$f(x, y, z) = (0, 2, 3, 4, 5, 7)$

Answer: x’z’ + yz + xy’ or y’z’ + x’y + xz

Cyclic prime implicant map: No PI is essential prime implicants have the same size, and every cell is covered by exactly two prime implicants.

DON'T CARES PROBLEM:

Design a code converter that converts BCD messages into Excess-3 code.

Input lines: Four - w, x, y and z, and

Output lines: Four - $f1$, $f2$, $f3$, and $f4$.

Input combinations: Decimal values 0 – 9: Value of '1'

Remaining six combinations: 10 -15: Don't-care combinations (Never occur)

Code converter is designed by considering each output function separately.

TRUTH TABLE:

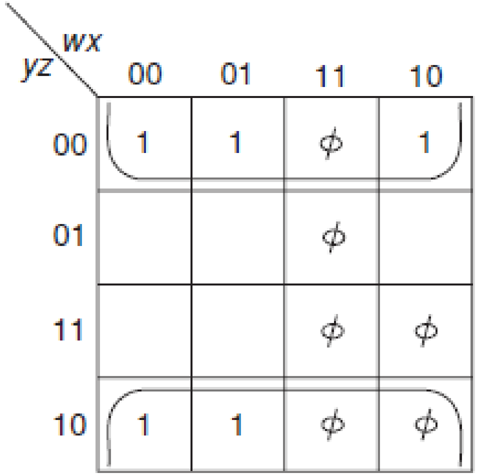
Decimal number	BCD inputs				Excess-3 outputs			
	w	x	y	z	f_4	f_3	f_2	f_1
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

(*a*) Truth table for BCD and Excess-3 codes

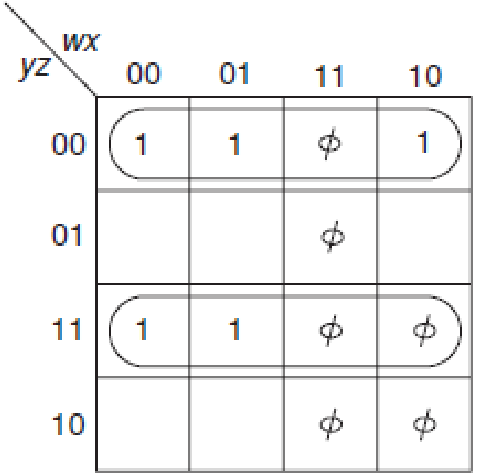
$$f_1 = \sum(0, 2, 4, 6, 8) + \sum_{\phi}(10, 11, 12, 13, 14, 15)$$
$$f_2 = \sum(0, 3, 4, 7, 8) + \sum_{\phi}(10, 11, 12, 13, 14, 15)$$
$$f_3 = \sum(1, 2, 3, 4, 9) + \sum_{\phi}(10, 11, 12, 13, 14, 15)$$
$$f_4 = \sum(5, 6, 7, 8, 9) + \sum_{\phi}(10, 11, 12, 13, 14, 15)$$

(*b*) Output functions

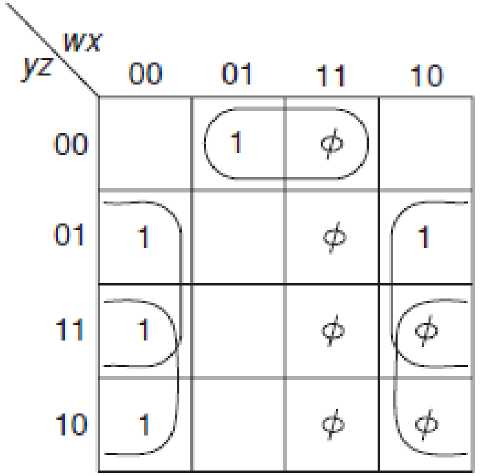
K-MAPS:



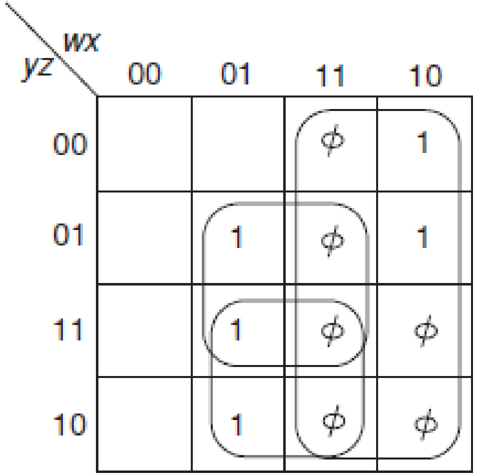
f_1 map



f_2 map



f_3 map



f_4 map

CIRCUIT DIAGRAM: