Lectured Five: Minimization Methods

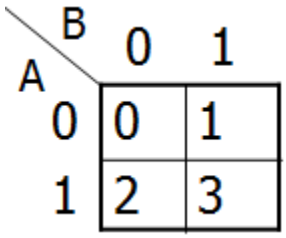
- Karnaugh map method.
- Quine McClusky method.

1. Karnaugh map:

A Karnaugh Map is a method of mapping truth tables onto a matrix that identifies places where two or more different combinations of the input variables yield the same result. In addition to identifying redundant terms, the K map also cancels them, leaving only the minimized Boolean algebra expressions that will yield the same truth table outputs as the unreduced terms. In the same time, Its primary limitation is that is not effective for more than four variables.

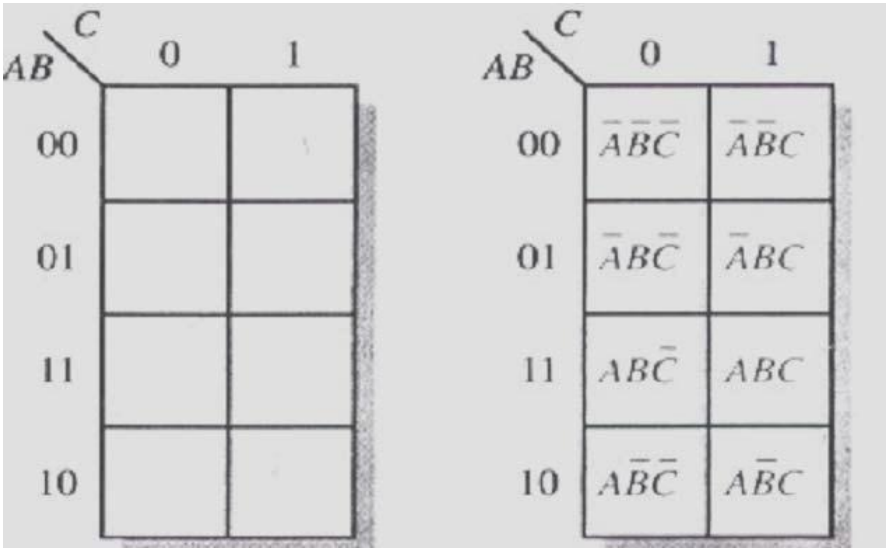
A K-map for a function of **n inputs variables** consists of 2^n cells, and, in every row and column, two adjacent cells should differ in the value of only one of the logic variables. Below we take an example about how can build the K-map for different inputs variables:

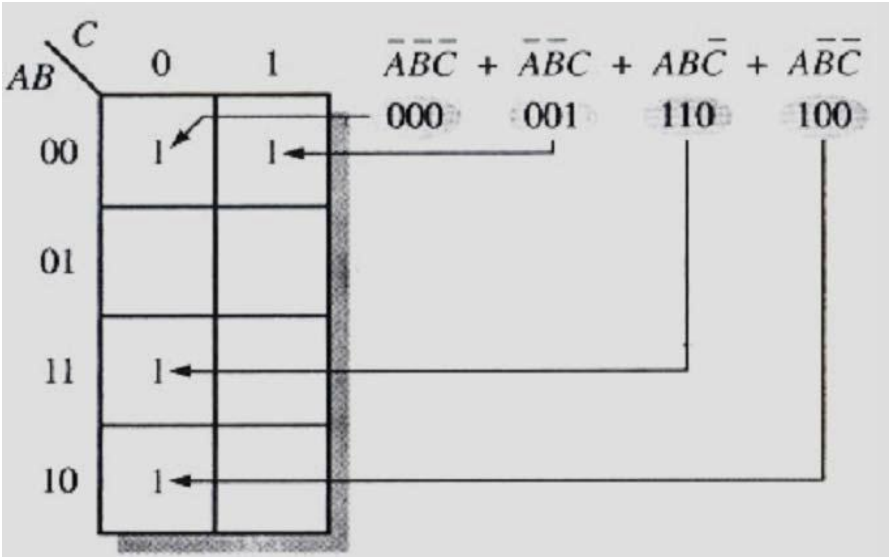
❖ 2-Variable K-Map:



❖ 3-Variable K-Map:

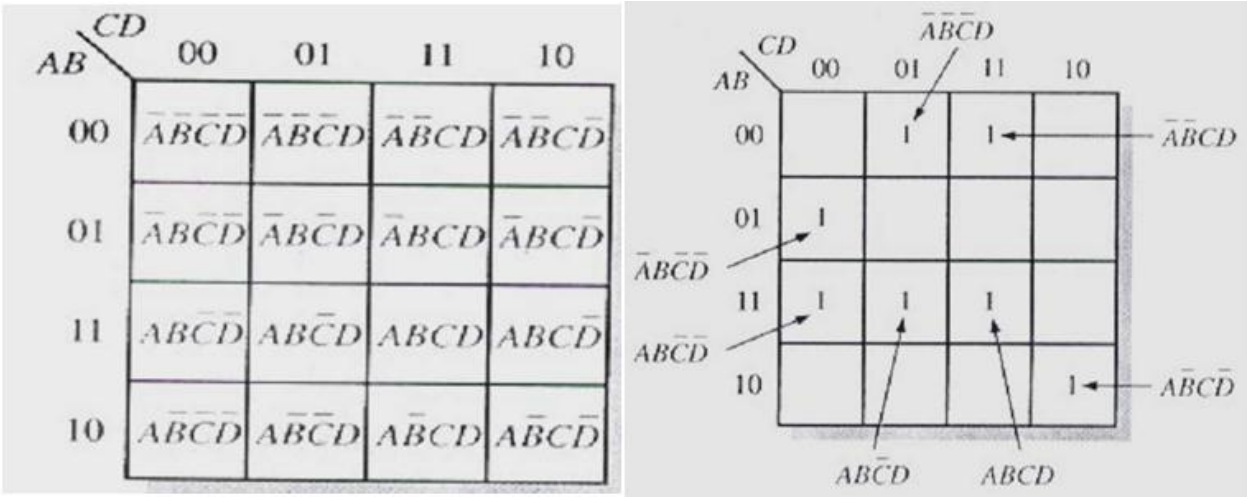
A \ BC	00	01	11	10
0	0	1	3	2
1	4	5	7	6



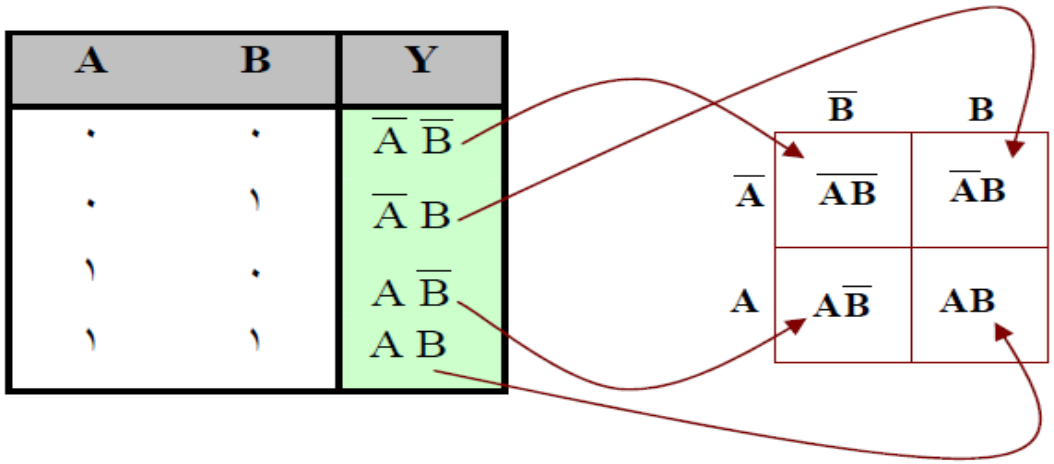


❖ 4-Variable K-map:

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



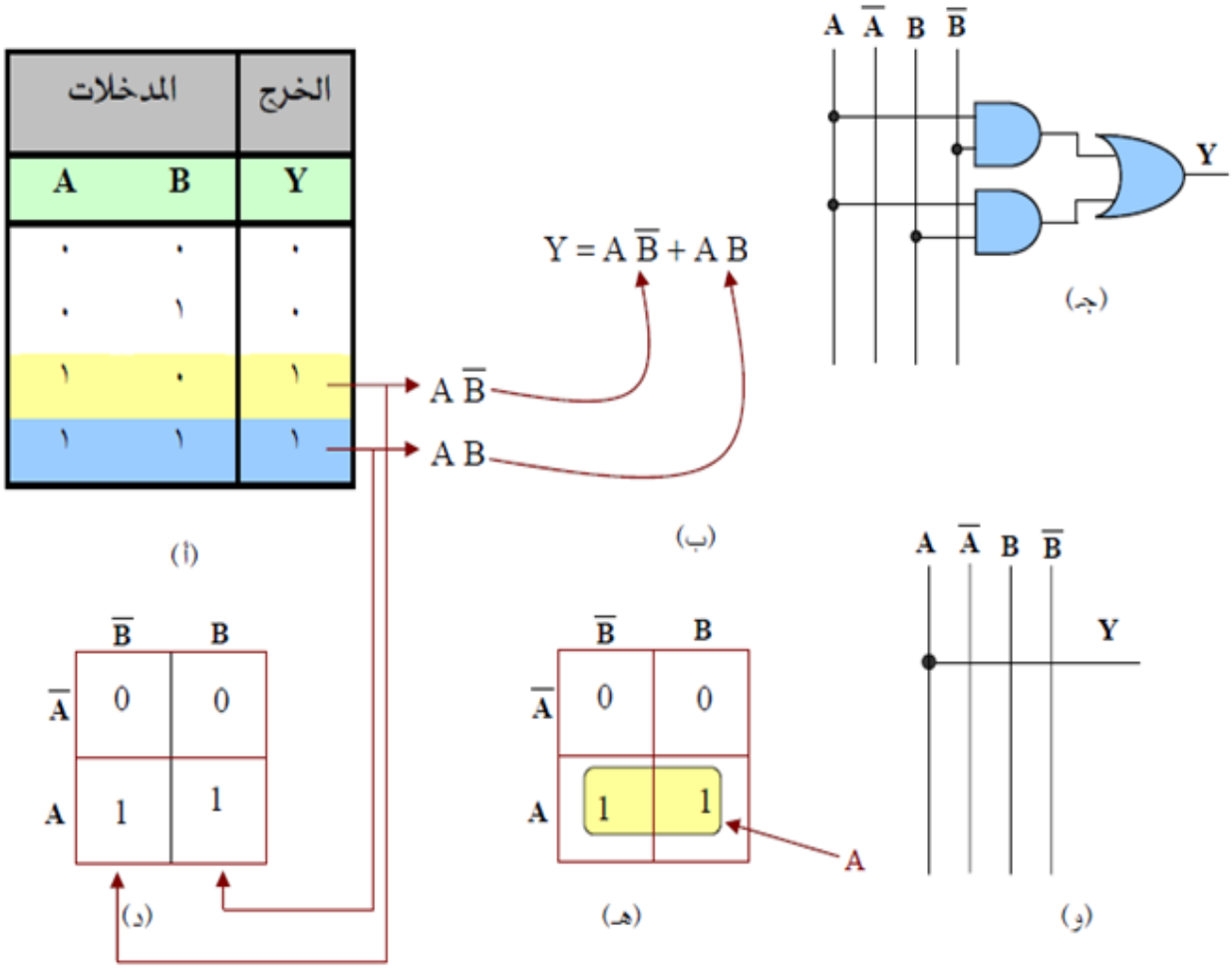
❖ Next the following figure explains how can representation the truth table on the K-map:



Ex1// Find the logic expression for the following truth table and then simplify it by using the K-map:

A	B	Y
0	0	0
0	1	0
1	0	1
1	1	1

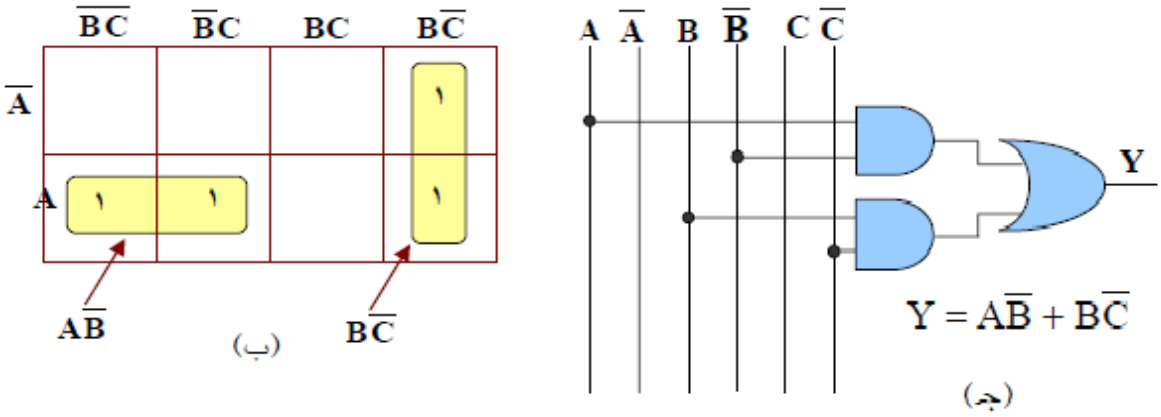
Sol//



Ex2// Design the logic circuit in the simplest form for the following truth table:

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Sol//

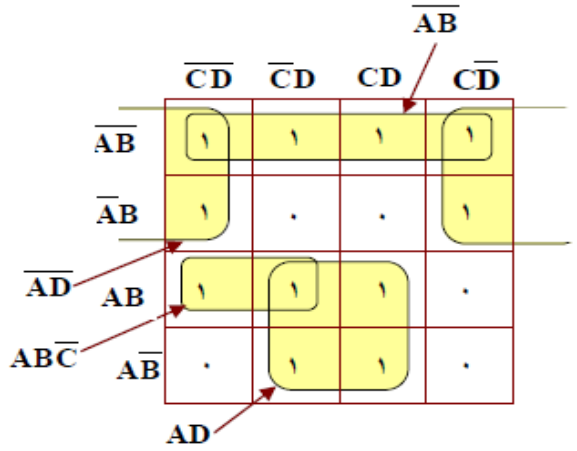


Ex3// Find the simpler expression for the following Boolean expressions:

1. $Y= \overline{A}\overline{B}CD + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}\overline{C}D + A\overline{B}C\overline{D} + AB\overline{C}\overline{D} + ABC\overline{D} + ABCD + A\overline{B}\overline{C}D + A\overline{B}CD$

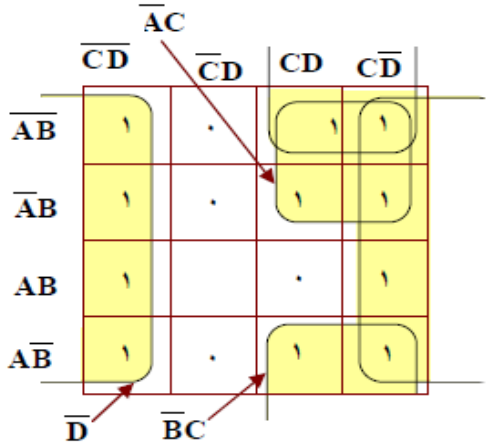
2. $Y= \overline{A}\overline{B}CD + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}\overline{C}D + A\overline{B}C\overline{D} + AB\overline{C}\overline{D} + ABC\overline{D} + ABCD + A\overline{B}\overline{C}D + A\overline{B}CD$

Sol//



$Y = AB\overline{C} + AD + \overline{A}B\overline{D} + \overline{A}\overline{B}$ (بعد التبسيط)

(1)



$Y = \overline{A}C + \overline{B}C + \overline{D}$ (بعد التبسيط)

(2)

8 حالات متغير واحد
4 حالات متغيرين
حالتين 3 متغيرات

Ex4// Find the truth table for the following expression and simplify it by using K-map:

$$F(ABCD) = \sum(2, 3, 8, 12, 13, 14)$$

Sol //

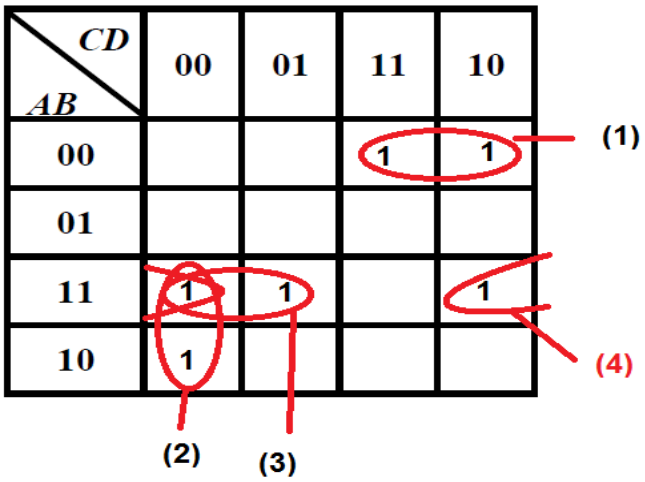
- **لحل مثل هذه الأسئلة فان هذا التعبير يمثل المستوى الذي يكون فيه الإخراج يساوي واحد وكذلك فانه يشير الى (SOP) وأحيانا يعطي تعبير أخر مثلا:**

$$F(ABCD) = \prod(2, 3, 8, 12, 13, 14)$$

حيث يمثل التعبير الثاني المستوى الذي يكون فيه الإخراج صفر ويشير الى (POS). وكلاهما يسميان بال (canonical form).

A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Sol //



$$F = \overline{A}\overline{B}C + A\overline{C}\overline{D} + AB\overline{C} + AB\overline{D}$$

H.W:

1. Find the simpler expression for the following Boolean expressions:

(1) $$Y = \overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}C\overline{D} + \overline{A}\overline{B}CD + \overline{A}B\overline{C}\overline{D} + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}\overline{C}\overline{D} + A\overline{B}\overline{C}D + A\overline{B}C\overline{D} + A\overline{B}CD + AB\overline{C}\overline{D} + AB\overline{C}D + ABC\overline{D} + ABCD$$

(2) $$Y = \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}C\overline{D} + \overline{A}\overline{B}CD + \overline{A}B\overline{C}\overline{D} + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}\overline{C}\overline{D} + A\overline{B}\overline{C}D + A\overline{B}C\overline{D} + A\overline{B}CD$$

2. Reduce the following equation using both Boolean algebra and K-Maps to a circuit with the fewest number of gates.

$$Y = A.B.C + A.\overline{B}.C + \overline{B}.\overline{C} + \overline{A}.\overline{B}$$

3. Find the truth table for the following expression and simplify it by using K-map:

$$\mathbf{F} = \prod(\mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{10}, \mathbf{11}, \mathbf{12}, \mathbf{13}, \mathbf{14}, \mathbf{15})$$

2. Don't Care Conditions:

(Are sets of inputs for which the designer doesn't care what the output is). For example, when dealing with BCD (Binary Coded Decimal) numbers encoded as four bits, we may not care about any codes above the BCD range of (0, 1, 2...9). The 4-bit binary codes for the hexadecimal numbers (A, B, C, E, F) are not valid BCD codes. Thus, we do not have to fill in those codes at the end of a truth table, or K-map, if we do not care to. We would not normally care to fill in those codes because those

codes (1010, 1011, 1100, 1101, 1110, 1111) will never exist as long as we are dealing only with BCD encoded numbers. These six invalid codes are don't cares as far as we are concerned. That is, we do not care what output our logic circuit produces for these don't cares

– In such cases we fill in the K-map with and **X** meaning **don't care**.
– When minimizing an **X** is like a **"joker". X** can be **0** or **1**whatever helps best with the minimization.

Ex1// Design logic circuit to convert the BCD to Excess^{-3} code?

Sol// Firstly, we must find the truth table to convert the BCD to Excess^{-3} code

A	B	C	D	X	Y	W	Z
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

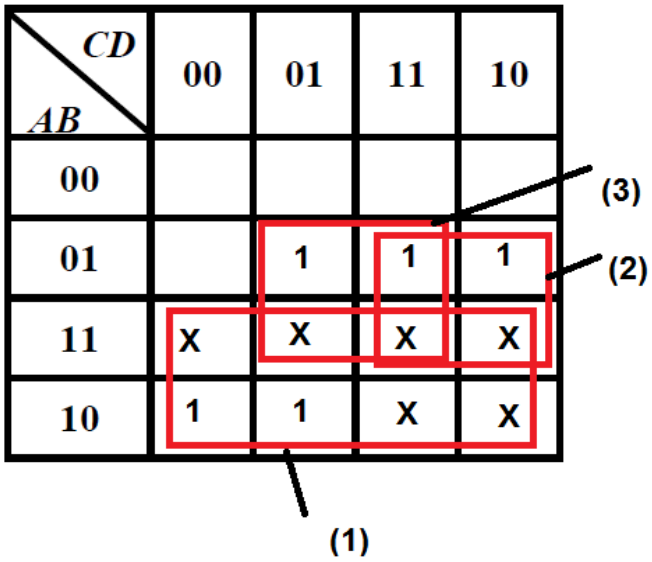
In the second step, find the Boolean expression for (**XYWZ**) as follow:

$$\mathbf{X} = \sum (5,6,7,8,9) + \sum X\,(10,11,12,13,14,15)$$

$$\mathbf{Y} = \sum (1,2,3,4,9) + \sum Y(10,11,12,13,14,15)$$

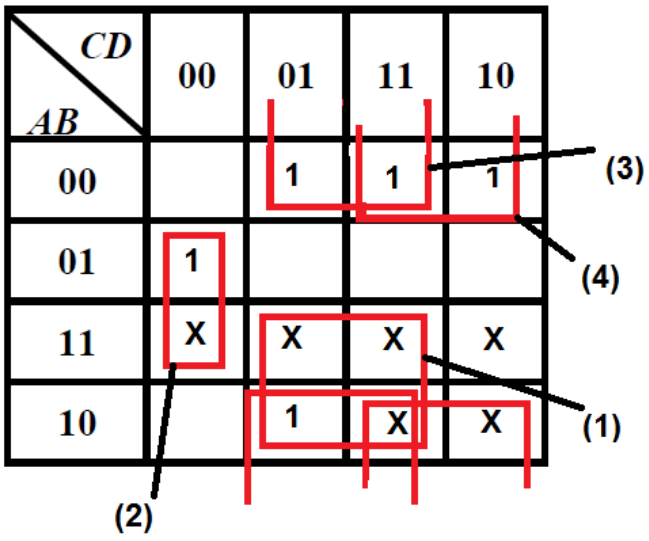
$$\mathbf{W} = \sum (0,3,4,7,8) + \sum W(10,11,12,13,14,15)$$

$$\mathbf{Z} = \sum (0,2,4,6,8) + \sum Z(10,11,12,13,14,15)$$



❖ **X** Boolean expression

$$\mathbf{X = A + BC + BD}$$



❖ **Y** Boolean expression

$$Y = AD + B\overline{CD} + \bar{B}D + \bar{B}C$$

H.W: Find the Boolean expression for W and Z?

Ex2// Simplify the Boolean function:

$$F(W, X, Y, Z) = \Sigma(1,3,7,11,15),\ dc(W, X, Y, Z) = \Sigma(0,2,5)$$

WX \ YZ	00	01	11	10
00	X	1	1	X
01	0	X	1	0
11	0	0	1	0
10	0	0	1	0

(2) (3) (1)

$$F = YZ + \overline{WX} + \bar{W}Z$$

3. Quine-McClusky Minimization Procedure:

This is basically a tabular method of minimization and as much it is suitable for computer applications. The procedure for optimization as follows:

Step1: Describe individual minterms of the given expression by their equivalent binary numbers.

Step2: Form a table by grouping numbers with equivalent number of 1's in them, i.e. first numbers with no 1's, then numbers with one 1, and then numbers with two 1's, … etc.

Step3: Compare each number in the top group with each minterm in the next lower group. If the two numbers are the same in every position but one, place a check sign (√) to the right of both numbers to show that they have been paired and covered. Then enter the newly formed number in the next column (a new table). The new number is the old numbers but where the literal differ, an "x" is placed in the position of that literal.

Step4: Using (3) above, form a second table and repeat the process again until no further pairing is possible. (On second repeat, compare numbers to numbers in the next group that have the same "x" position).

Step5: Terms which were not covered are the prime implicants and are ORed and ANDed together to form final function.

Ex1// Minimize the function given below by Quine-McClusky method:

F= $\bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}B\bar{C}D + \bar{A}BC\bar{D} + A\bar{B}\bar{C}D + A\bar{B}C\bar{D} + AB\bar{C}D + ABC\bar{D} + ABCD + \bar{A}BCD$

Sol//

f(A,B,C,D)=	$\bar{A}\bar{B}\bar{C}\bar{D}$+	$\bar{A}\bar{B}C\bar{D}$+	$\bar{A}BC\bar{D}$+	$A\bar{B}C\bar{D}$+	$A\bar{B}C\bar{D}$+	$AB\bar{C}D$+	$ABC\bar{D}$+	ABCD+	$\bar{A}BCD$
Binary	0000	0101	0110	1001	1010	1101	1110	1111	0111
minterm	0	5	6	9	10	13	14	15	7
No of 1's	0	2	2	2	2	3	3	4	3
group	1	2	2	2	2	3	3	4	3

❖ **This can be written as a sum of minterms as follows:**

$$f(A,B,C,D) = \sum m(0,5,6,7,9,10,13,14,15)$$

Step1: Form a table of functions of minterms according to the number of **1's** in each minterm as shown in table **E1.a.**

minterm	A	B	C	D		
0	0	0	0	0		All numbers with no 1's in each minterm (a)
5	0	1	0	1	✓	All numbers with two 1's in each minterm
6	0	1	1	0	✓	
9	1	0	0	1	✓	
10	1	0	1	0	✓	
7	0	1	1	1	✓	All numbers with three 1's in each minterm
13	1	1	0	1	✓	
14	1	1	1	0	✓	
15	1	1	1	1	✓	All numbers with four 1's in each minterm

Step2: Start pairing off each element of first group with the next, however since m_0 has no 1's, it and the next group of numbers with one 1's are missing, therefore they cannot be paired off. Start by pairing elements of m_5 with m_7, m_{13}, m_{14}, and m_6 with m_7, m_{13}, m_{14}, and so on… If they pair off, write them in a separate table and √ the minterm that pair, i.e. m_5 and m_7 pair off 0101 and 0111 to produce 01x1, so in the next table **E1.b** under "minterm paired" we enter **"5, 7"** and under "ABCD" we enter **"01x1"** and place a √ sign in front of **5** and **7** in table **E1.a.**

minterms paired	A	B	C	D	
5, 7	0	1	X	1	✓
5, 13	X	1	0	1	✓
6, 7	0	1	1	X	✓
6, 14	X	1	1	0	✓
9, 13	1	X	0	1	..... (b)
10, 14	1	X	1	0	..... (c)
7, 15	X	1	1	1	✓
13, 15	1	1	X	1	✓
14, 15	1	1	1	X	✓

Table E1.b

Step 3: Now repeat the same procedure by pairing each element of a group with the elements of the next group for elements that have "x" in the same position. For example, **"5,7"** matches "**13,15"** to produce x1x1. These elements are placed in table **E1.c** as shown, and the above elements in table **E1.b** are √ checked. (The elements that produce the same ABCD pattern are eliminated.)

Paired minterms from E1.b	A	B	C	D	
5,7 – 13,15	x	1	x	1	... (d)
6,7 – 14,15	x	1	1	x	... (e)

Table E1.c

Step4: Since 9,13 and 10,14 in table **E1.b** do not pair off, they are prime implicants and with m0, from **E1.a**, and (d) and (e) from **E1.c** are unpaired individuals. Therefore, it is possible to write the minimized SOP as **a + b + c + d + e**

$$f(A,B,C,D) = \overline{A}\overline{B}\overline{C}\overline{D} + A\overline{C}D + AC\overline{D} + BD + BC$$

Ex2// Give a minimum logic expression of the 4-variables W,X,Y and Z to implement a function made up minterms:

$F(W, X, Y, Z) = \Sigma (0, 2, 5, 6, 7, 10, 13, 14, 15)$

Sol//

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

Binary	0000	0010	0101	0110	0111	1010	1101	1110	1111
Minterm	0	2	5	6	7	10	13	14	15
No of 1's	0	1	2	2	3	2	3	3	4
Group	1	2	3	3	4	3	4	4	5

Minterm Values	Binary Value	First Reduction		Second Reduction
	ABCD	ABCD		ABCD
0	0000	00X0	(0,2)... a	XX10 ((2,6),(10,14)) ….b
2	0010	0X10	(2,6)	
		X010	(2,10)	X1X1 ((5,13),(7,15))….c
5	0101			X11X ((6,7),(14,15))….d
6	0110	01X0	(5,7)	
10	1010	X101	(5,13)	
		011X	(6,7)	
7	0111	X110	(6,14)	
13	1101	1X10	(10,14)	
14	1110			
		X111	(7,15)	
15	1111	11X1	(13,15)	
		111X	(14,15)	

$$F(A,B,C,D) = a + b + c + d + e$$

$$\text{Or} \quad F(A,B,C,D) = \overline{ABD} + C\overline{D} + BD + BC$$

H.W: Use the Quine-McCluskey method to simplify the following expression

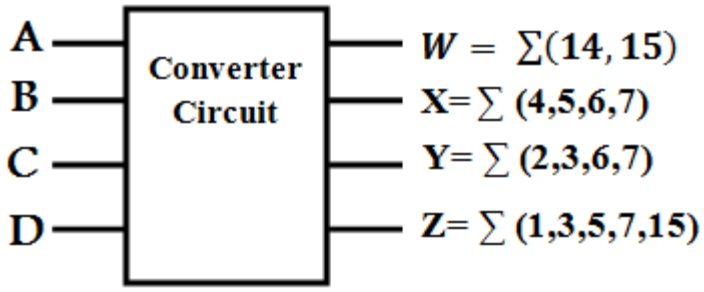
$$F(W, X, Y, Z) = \Sigma\,(0,2,3,6,7,8,9,10,13)$$

4. Code Converter Circuit:

Ex1// Design code converter logic circuit to convert 2421 to BCD code?

Sol//

Dec	A 2	B 4	C 2	D 1	W 8	X 4	Y 2	Z 1
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	0
3	0	0	1	1	0	0	1	1
4	0	1	0	0	0	1	0	0
5	0	1	0	1	0	1	0	1
6	0	1	1	0	0	1	1	0
7	0	1	1	1	0	1	1	1
14	1	1	1	0	1	0	0	0
15	1	1	1	1	1	0	0	1



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

W = A

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

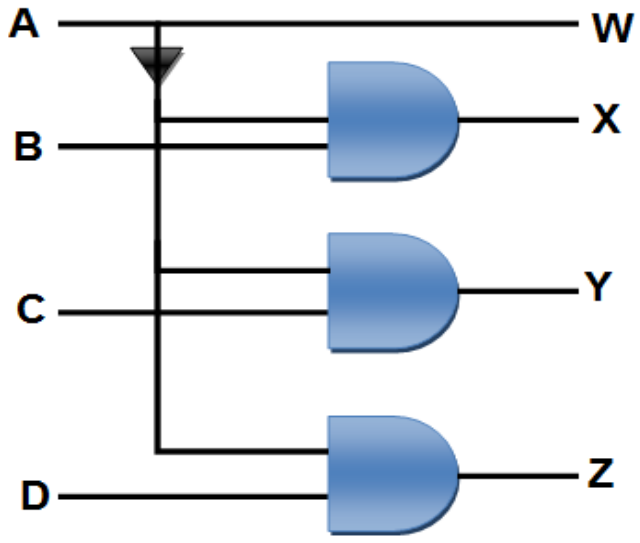
X = $\overline{A}B$

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

Y = $\overline{A}C$

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

Z = $\overline{A}D$



H.W: Design logic circuit to convert BCD code to Excess^{-3} Code ?