Computer System Architecture
COMP201TH
Lecture-4
Karnaugh Maps (K-Map)

- **K Map:**
  - Karnaugh map is a method of simplifying Boolean algebra expressions.
  - It is actually a truth table in another form.
  - It offers a graphical method of reducing a digital circuit to its minimum number of gates.
  - Karnaugh maps can be used on small circuits having 2 to 3 inputs as an alternative to Boolean algebra and on more complex circuits having up to 6 inputs; it can provide quicker and simpler minimisation than Boolean algebra.

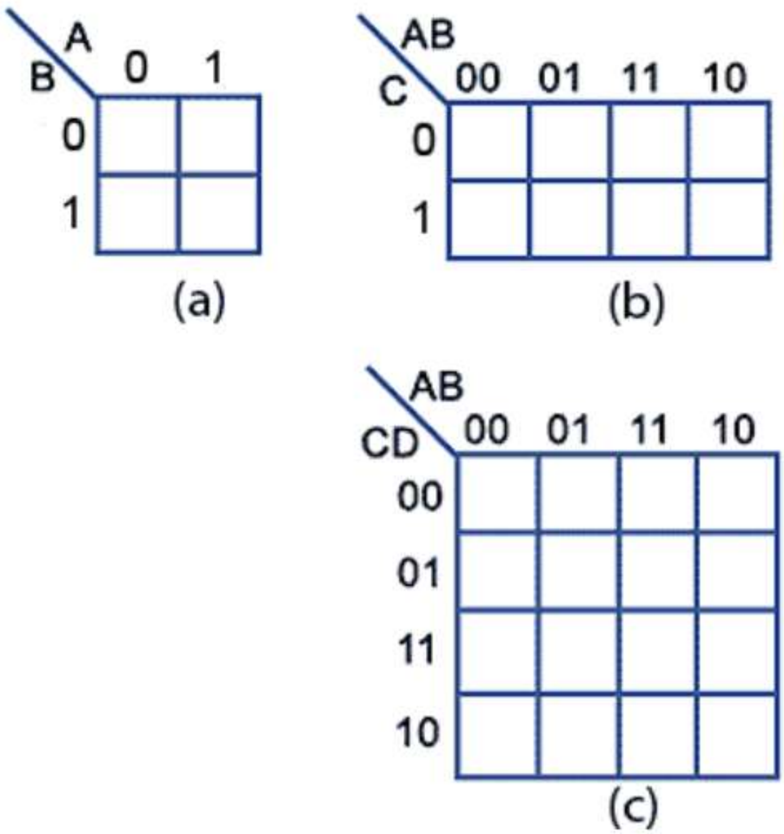


Fig. : Representation of K-map for 2,3 and 4 variables respectively

- Constructing K-maps:
  - The shape and size of the map is dependent on the number of binary inputs in the circuit to be analysed.
  - 2 input circuits with inputs A and B require maps with 2^2 = 4cells.
  - n input circuit will require map with 2^n cells.

- In K maps, the cells are ordered in Gray Code and each cell position represents on combination of input conditions while each cell value represents the corresponding output value.
  - Gray code is an ordering of the binary number system such that two successive values differ in only one bit. E.g. the representation of the $(1)_{10}$ in binary would normally be 01 and $(2)_{10}$ would be 10.
  - In Gray code, these values are represented as 01 and 11, respectively.

Binary	Gray	Binary	Gray	Binary	Gray
0	0	00	00	000	000
1	1	01	01	001	001
		10	11	010	011
		11	10	011	010
				100	110
				101	111
				110	101
				111	100

Fig.: Corresponding Gray codes of Binary numbers

- Example of K-Map:
  - Simplify A + A` B by use of Karnaugh Map.

f = A+A`B → minterm expression → A`=0 and A=1.

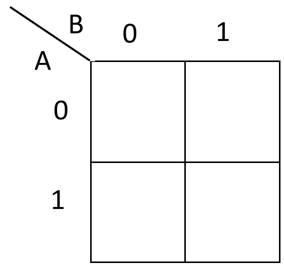
In the expression, first see first term is A, here we will take value of B as both 0 and 1.

Remember, while using K-map if one variable is not specified then we should consider its value both 0 and 1.

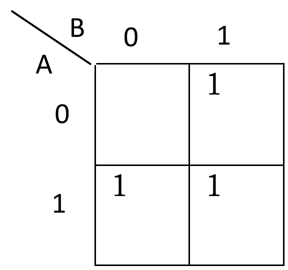
e.g. if f = AB +AB`C+A`BC`

here in the first term AB value of C is not specified, so when computing for K map, we will consider value of C as both 0 and 1.

K map for two variable is:



Now, K-map for f = A + A`B will be:



In K-map, we will put 1 in the cell where in value of A and B are specified in the Boolean expression i.e. f = A + A`B -> its in minterm expression,

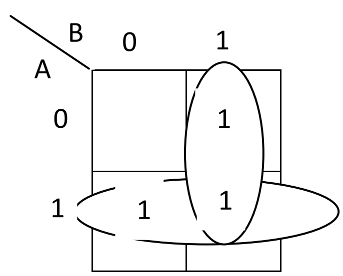
So f = 1 + 01.

Now, as said earlier: In the expression, first see first term is A, here we will take value of B as both 0 and 1.

So, for the given Boolean expression we will take first term as 10 and 11 as value of B is not given so we are considering it both 0 and 1.

Now see, we have put 1 in the cell where cell positions are 10, 11 and 01.

Next step is: we will group adjacent 1's and choose common between them from their cell positions.



Now , first group (vertically): 01

11

Common is 1 which is B

Now, second group (horizontally): 10

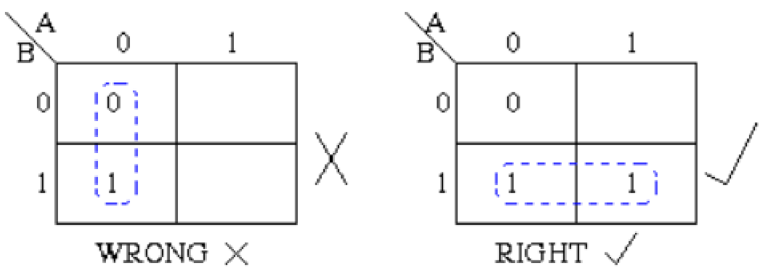
11

Common is 1 which is A.

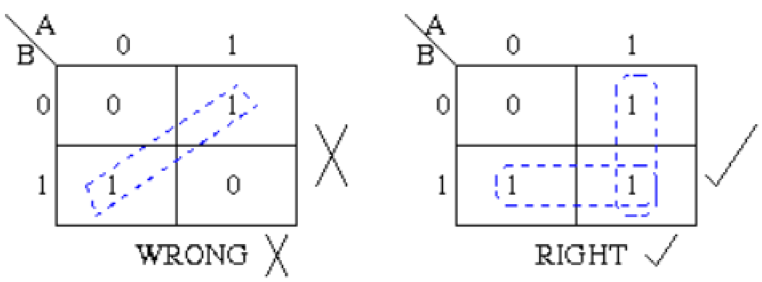
So, the solution is A+B.

- **Rules for simplifying Boolean expressions using K Maps [1, 2]:**

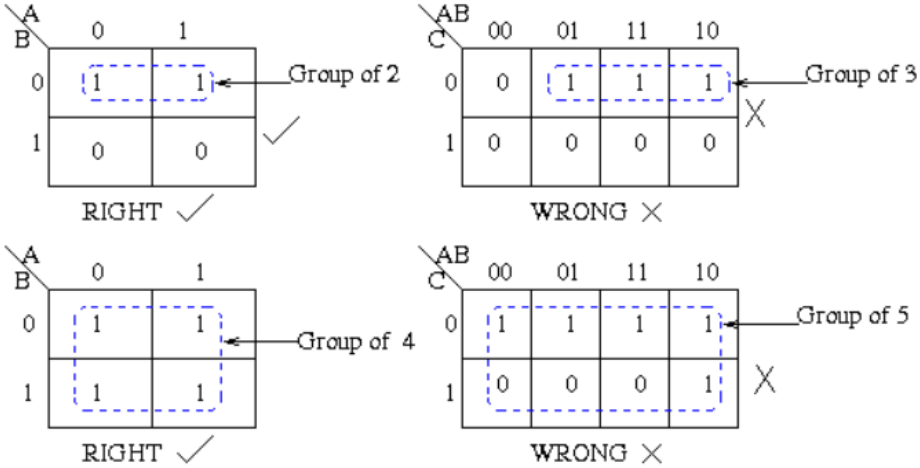
- **Groups may not include any cell containing a zero**



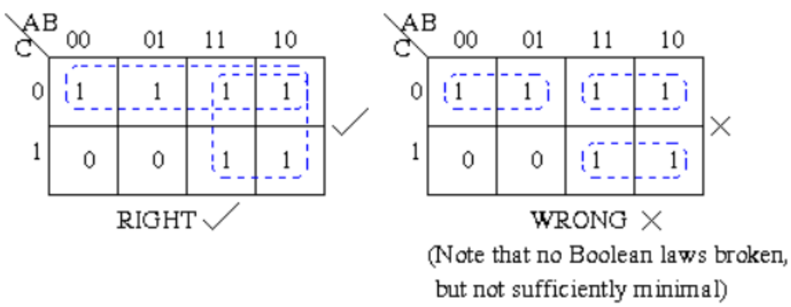
- **Groups may be horizontal or vertical, but not diagonal.**



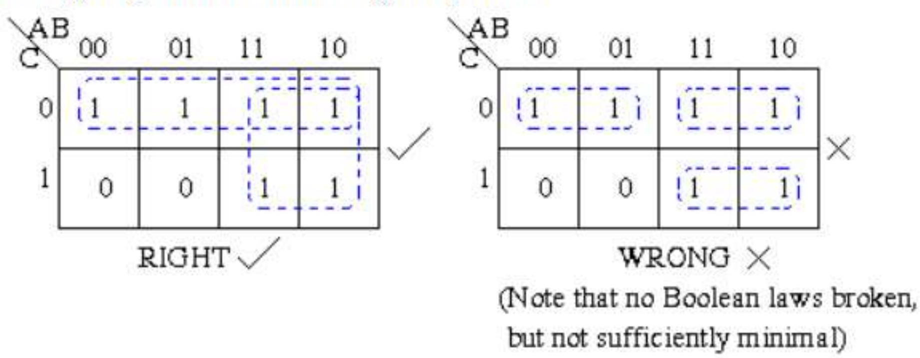
- **Groups must contain 1, 2, 4, 8, or in general 2^n cells.**
  That is if n = 1, a group will contain two 1's since $2^1 = 2$.
  If n = 2, a group will contain four 1's since $2^2 = 4$.



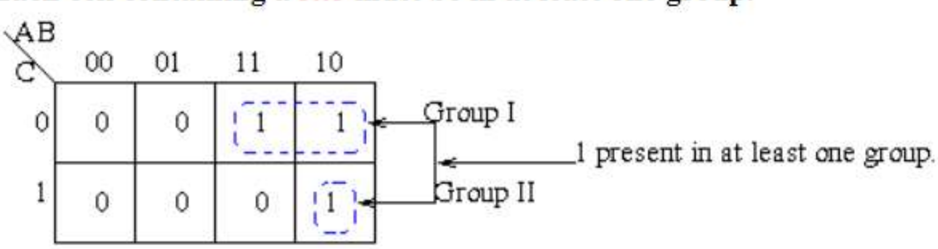
- **Each group should be as large as possible.**



- **Each group should be as large as possible.**

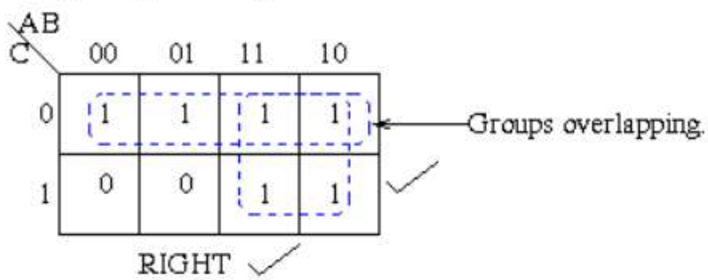


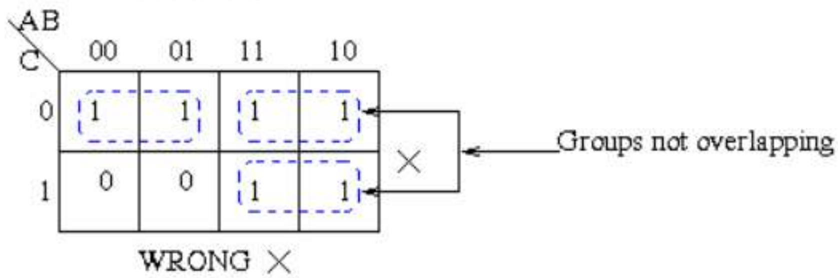
- **Each cell containing a *one* must be in at least one group.**



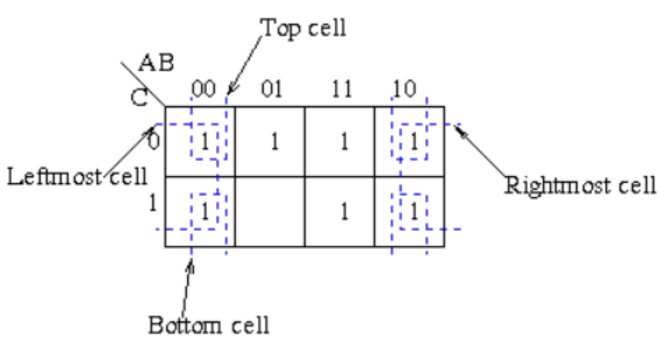
- **A square containing 1 should not be left alone to be included in the final expression if there is a possibility of its inclusion in a group of two squares containing 1s. Similarly, a group of two 1-squares (i.e. square containing 1) should not be made if these 1-square can be included in a group of four 1-squares and so on.**

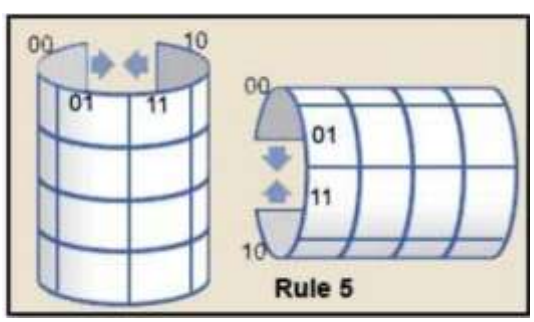
- **Groups may overlap.**



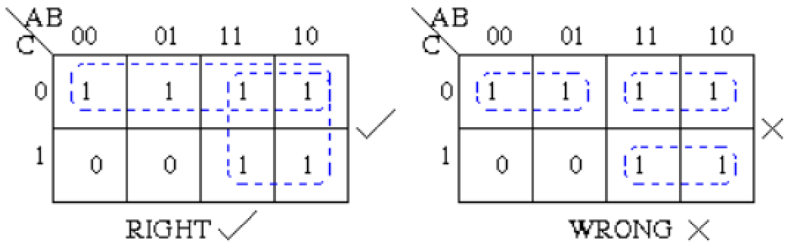


- **Groups may wrap around the table. The leftmost cell in a row may be grouped with the rightmost cell and the top cell in a column may be grouped with the bottom cell.**





- **There should be as few groups as possible, as long as this does not contradict any of the previous rules.**



Q: Simplify Boolean Function:

$$F = \bar{x}yz + \bar{x}y\bar{z} + x\bar{y}\bar{z} + x\bar{y}z$$

Sol: For 3-variables, K-map is:

$x \backslash yz$	00	01	11	10
0				
1				

Now, $f = \bar{x}yz + \bar{x}y\bar{z} + x\bar{y}\bar{z} + x\bar{y}z$

In the above expression, all the terms contain all the variables.

We can also write f as:

$$f = \underbrace{011}_{\bar{x}yz} + \underbrace{010}_{\bar{x}y\bar{z}} + \underbrace{100}_{x\bar{y}\bar{z}} + \underbrace{101}_{x\bar{y}z}$$

Cell Positions for which we will put cell value as 1.

∴ K-map will be:

$x \backslash yz$	00	01	11	10
0			1	1
1	1	1		

$x = 1, y = 0, z = 1$
i.e. 101

Now, next step is to group adjacent 1's.

$x \backslash yz$	00	01	11	10
0			1	1
1	1	1		

Group 1

Group 2

Now, we will choose common in the group i.e. common among the cell positions.

e.g. ∴ Group 1 →

x	y	z
0	1	1
0	1	0

common and which is $\bar{x}y$

∵ 0 in minterm is represented in complement form.

IIly Group 2 →

x	y	z
1	0	0
1	0	1

$= x\bar{y}$

∴ Solution is: $f = \bar{x}y + x\bar{y}$ Ans

Now, lets solve this using boolean identities:

$$f = \bar{x}yz + \bar{x}y\bar{z} + x\bar{y}\bar{z} + x\bar{y}z$$

$$= \bar{x}y(z+\bar{z}) + x\bar{y}(z+\bar{z})$$

$$= \bar{x}y + x\bar{y}$$

Q. Simplify: $f = \bar{x}yz + x\bar{y}\bar{z} + xyz + xy\bar{z}$

Sol:

$x \backslash yz$	00	01	11	10
0			1	
1	1		1	1

Group: 1

Group: 2

$f = 011 + 100 + 111 + 110$

$\therefore$ Group 1 : $\begin{array}{ccc} 0 & 1 & 1 \\ 1 & 1 & 1 \end{array}$

$= yz$

Group 2 : $\begin{array}{ccc} 1 & 0 & 0 \\ 1 & 1 & 0 \end{array}$

$= x\bar{z}$

$\therefore$ simplified expression is: $yz + x\bar{z}$

Q Simplify: $f = \bar{A}\bar{B}\bar{C} + \bar{B}C\bar{D} + \bar{A}BC\bar{D} + A\bar{B}\bar{C}$

Sol: 4- variable K map is

AB \ CD	00	01	11	10
00				
01				
11				
10				

Now, we know if in the terms of boolean expression value of any of variable is missing, then while making K-map, we consider the value of missing variable as both 0 & 1

$\therefore f = \underbrace{\bar{A}\bar{B}\bar{C}}_{\text{D is missing}} + \underbrace{\bar{B}C\bar{D}}_{\text{A is missing}} + \bar{A}BC\bar{D} + \underbrace{A\bar{B}\bar{C}}_{\text{D is missing}}$

$= 000 + 010 + 0110 + 100$

Now, consider value of missing variable as both 0 & 1.

$= \underbrace{(0000, 0001) + (0010, 1010) + 0110 + (1000, 1001)}_{\text{cell positions whose value will be 1.}}$

$\therefore$

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

Fig. Cell value = 1 for cell position 0000

Now, cell value = 1 for cell position 0001

⇒ K-map :

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

IIly. putting value = 1, for the cell positions:

0010, 1010, 0110, 1000, 1001

⇒ K-map will be :

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

Now, as per rules of K-map:

- Groups must contain 1, 2, 4, 8 or in general 2^n cells.
- Groups may wrap around the table. The leftmost cell in a row may be grouped with the rightmost cell and the top cell in a column may be grouped with the bottom cell.

∴

Group: 1

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

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

$\bar{B}\ \bar{D}$

Group: 2

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

0	0	0	0
0	0	0	1
1	0	0	0
1	0	0	1

$\bar{B}\ \bar{C}$

Group: 3

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

0	0	1	0
0	1	1	0

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

∴ $f = \bar{B}\bar{D} + \bar{B}\bar{C} + \bar{A}C\bar{D}$

References :

[1] Composed by David Belton
http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/karnaugh.html

[2] Eric Coates (Revision 14.01 18th July 2020)
https://learnabout-electronics.org/Digital/dig24.php