**8th June, 2020 JESUS AND MARY SCHOOL & COLLEGE MODULE 3**

**CLASS 12**

**COMPUTER SCIENCE**

# KARNAUGH MAP

# A Karnaugh map provides a pictorial method of grouping together expressions with common factors and therefore eliminating unwanted variables.

## Karnaugh Map Definition

Karnaugh maps were designed by Maurice Karnaugh in 1953 when he was working on the digital logic for telephone switching circuits at Bell Labs. These maps helped make the logic a lot simpler. This simplicity transferred into the design of the digital logic circuit itself by reducing the number of needed inputs and gates or electronic components. By definition, a **Karnaugh map** simplifies Boolean logic expressions. A Karnaugh map (or K-map for short) looks like this.

# ****Explain K-MAP and its significance. Explain 2-variables, 3-variables and 4-variables K-MAP.****

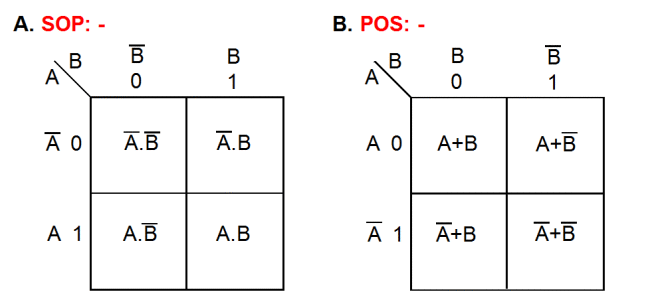
|  |
| --- |
|  |

The Karnaugh map or k-map is a graphical technique for simplifying Boolean function. The k-map in a two dimensional representation of a truth table it provides a simpler method for minimizing logic expressions. The map method is ideally suited for four or less variables. But it becomes cumbersome for five or more variable. A karnaugh map is a diagram consisting of squares. Each square of the map represent either min term or max term. Any logic expression can be written as either sum of product term or product of sum term which is also called sum of min term and product of max term respectively. Therefore a logic expression can be easily represented on a karnaugh map.

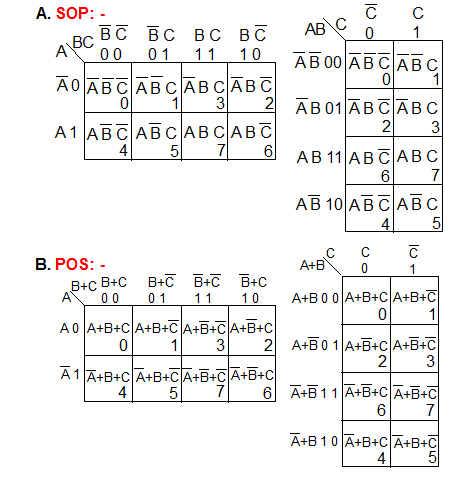
A KARNAUGGH map for n variables made up of 2n squares. Each square designate a product term in Boolean expression. For product term which are present in the expression. In case of sum of product, 1s can be written in Present Square and in case of product of sum term, 0s can be written in Present Square. Blank Square or opposite square indicate. That then contains nothing.

We can construct KARNAUGH map in two different forms for sum of product term and for **product of sum** term. Because in sum of product contains zero as complement value and one as un-complement value similarly in the product of sum term we use 1 as complement and 0 as un-complemented value function of both sum of product and product of sum about same except that the complement and un-complement value.

**Two variable K-map: –** In the 2 variable k-map, four squares are constructed. Each square contains one term of expression with two variables. **A 2 variable k-map for SOP and POS form are as follows: –**

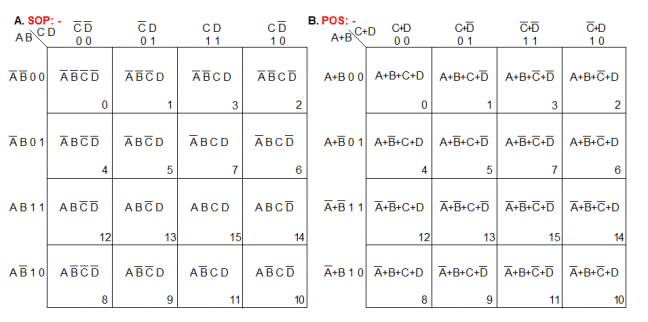


**Three variables K-map: –** In the three variable k-map, 8 square is required. The 3-variables k-map can content either horizontally or vertically. **An example of 3 variables k-map for SOP and POS form are as follows-**



**Four variable K-map: –** In the 4 variable k-map, 16 square are required. **An example of 4 variable**

**k-map for SOP and POS form are as follows: –**



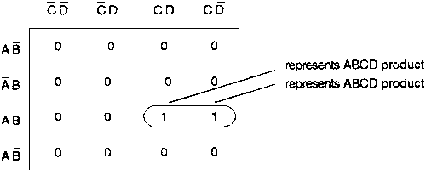
In many digital circuits and practical problems we need to find expression with minimum variables. We can minimize Boolean expressions of 3, 4 variables very easily using K-map without using any Boolean algebra theorems. K-map can take two forms Sum of Product (SOP) and Product of Sum (POS) according to the need of problem. K-map is table like representation but it gives more information than TRUTH TABLE. We fill grid of K-map with 0’s and 1’s then solve it by making groups.

**Steps to solve expression using K-map-**

1. Select K-map according to the number of variables.
2. Identify minterms or maxterms as given in problem.
3. For SOP put 1’s in blocks of K-map respective to the minterms (0’s elsewhere).
4. For POS put 0’s in blocks of K-map respective to the maxterms(1’s elsewhere).
5. Make rectangular groups containing total terms in power of two like 2,4,8 ..(Except 1) and try to cover as many elements as you can in one group.
6. From the groups made in step 5 find the product terms and sum them up for SOP form.

**Framing Groups :**  The groups can be formed in the following ways **Pairs, Quads, and Octets .**

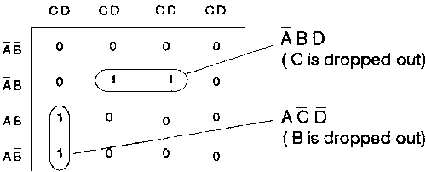
**Pairs**



As you see in above mentioned Fig. only one variable goes from uncomplement to complement. Whenever this happens, you can eliminate the variable that changes form.

|  |  |
| --- | --- |
| Proof: | http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img048.gif |
|  | http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img049.gif |
|  | **X = A B C** |

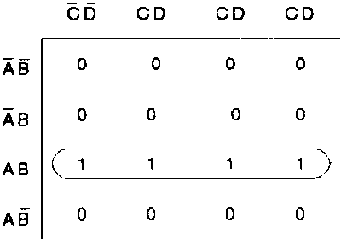
Ex:



Whenever you see a pair first encircles it and then simplifies to get the simplified Boolean expression:

http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img050.gif

**Quad**

  
 *Quad*

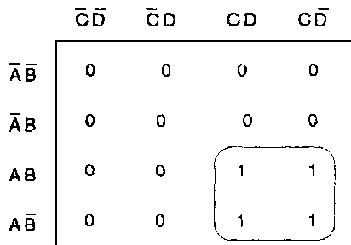
Quad: A group of 4 one's that are horizontally or vertically adjacent. End to end or in form of a square.

A quad eliminates two variables and their complements.

|  |  |  |
| --- | --- | --- |
| Proof: | http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img051.gif | *(two pairs)* |
|  | X = A B (C + C) |  |
|  | X = A B |  |

Encircle the quad and step through the different one's in the quad and determine which two variables go from complement to uncomplement (or vs), these are the variables that drop out.

Ex:

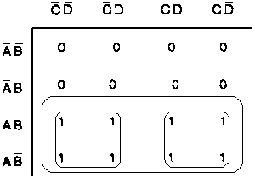


*Quad*

The variables B and D can be eliminated. So we get the following equation:

X = A C

**Octet**



*Octet*

An octet eliminates three variables and their complements.

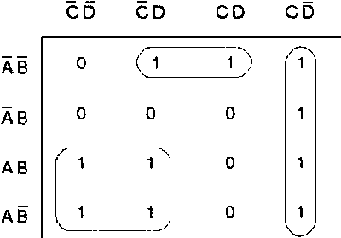
|  |  |  |
| --- | --- | --- |
| Proof: | http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img052.gif | *(two quads)* |
|  | X = A (C + C) |  |
|  | X = A |  |

**Karnaugh Simplifications**

Process:

1. Draw the Karnaugh map  
2. Look for octets and encircle them.  
3. Look for quads and encircle them.  
4. Look for pairs and encircle them.  
5. Simplify and write down the equation.

Exp:

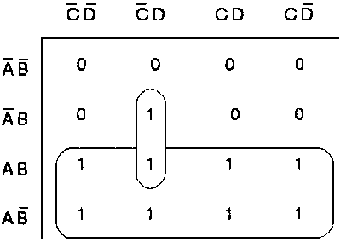
  
 *Karnaugh map*

http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img053.gif

**Overlapping and Rolling**

Overlapping groups

Ex:

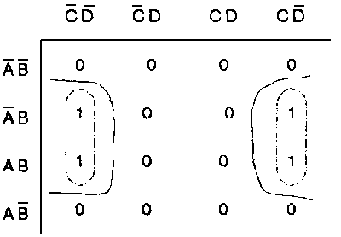
  
 *Karnaugh map*

Groups can overlap to get a simpler equation:

http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img054.gif

Rolling the map

Ex:

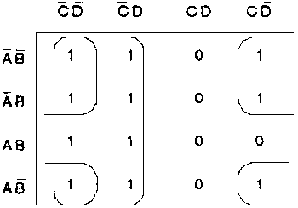
  
 *Karnaugh map*

Instead of encircling two pairs:

http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img055.gif

We can roll the map and encircle a quad:

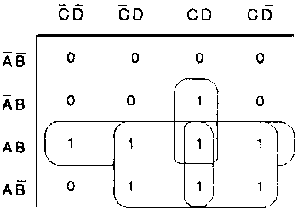
http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img056.gif



HO: Simplify the following map.

Solution:

http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img057.gif

  
HO: Simplify the following map.

Solution:

http://www.nzdl.org/gsdl/collect/cdl/archives/HASHfb0a.dir/img058.gif

## ****SOP FORM****

1. **K-map of 3 variables-**

Z= ∑A,B,C(1,3,6,7)

### de1

From **red** group we get product term—

A’C

From**green** group we get product term—

AB

Summing these product terms we get- **Final expression (A’C+AB)**

1. **K-map for 4 variables**

F(P,Q,R,S)=∑(0,2,5,7,8,10,13,15)

### [de2](https://media.geeksforgeeks.org/wp-content/uploads/K-Map-Karnaugh-Map-2-1.png)

From **red** group we get product term—

QS

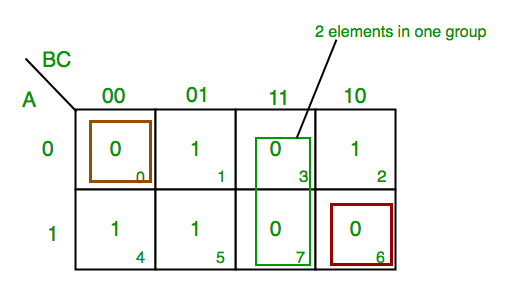
From **green** group we get product term—

Q’S’

Summing these product terms we get- **Final expression (QS+Q’S’)**

## ****POS FORM****

1. **K-map of 3 variables-**

F(A,B,C)=π(0,3,6,7)[](https://media.geeksforgeeks.org/wp-content/uploads/K-Map-Karnaugh-Map-4.png)

From **red**group we find  terms

A    B      C’

Taking complement of these two

A’     B’     C

Now **sum** up them

(A’ + B’ + C)

From**green** group we find  terms

B         C

Taking complement of these two terms

B’         C’

Now sum up them

(B’+C’)

From **brown**group we find terms

A’ B’ C’

Taking complement of these two

A B C

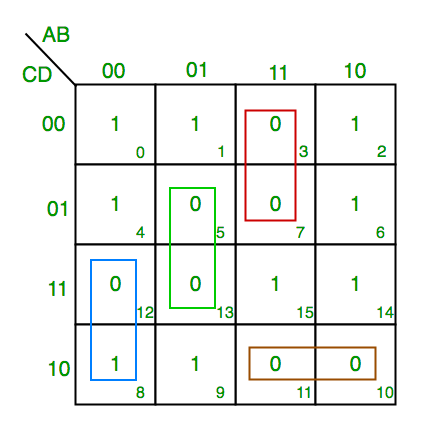
Now **sum** up them

(A + B + C)

We will take product of these three terms :**Final expression (A’ + B’ + C) (B’ + C’) (A + B + C)**

**2. K-map of  4 variables-**

F(A,B,C,D)=π(3,5,7,8,10,11,12,13)

[](https://media.geeksforgeeks.org/wp-content/uploads/K-Map-Karnaugh-Map-3.png%22)

From **green** group we find  terms

C’     D     B

Taking their complement and summing them

(C+D’+B’)

From **red** group we find terms

C     D    A’

Taking their complement and summing them

(C’+D’+A)

From **blue**  group we find  terms

A     C’     D’

Taking their complement and summing them

(A’+C+D)

From **brown** group we find  terms

A    B’    C

Taking their complement and summing them

(A’+B+C’)

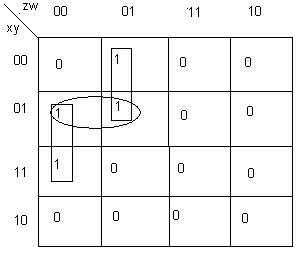
Finally we express these as product –**(C+D’+B’).(C’+D’+A).(A’+C+D).(A’+B+C’)**

**PITFALL–**  \*Always remember **POS ≠ (SOP)’**

\*The correct form is (**POS of F)=(SOP of F’)’**

**Redundant groups**

A groups of 1s or 0s whose all members are overlapped by other groups is called redundant group. We don’t consider this group while writing the simplified equations from the K-map.



In the above K-map the group which is represented by the oval is a redundant group and hence while writing the equations we ignore it or we don’t make this kind of group and the K-map representation becomes as given next:

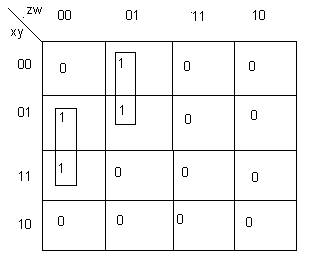
The equation we get is

F= yz’w’ + x’z’w (ignoring the redundant group)

If we consider this group then equation would be F=  yz’w’ + x’z’w + x’yz’

And this is a not the simplified expression and hence **WRONG**.

K-map without the redundant group is:



**Some facts of K-map:**

* The 1’s in the map represent min terms and 0’s represent max terms
* If we combine 1’s and make groups, we get simplified function.
* If we combine 0’s and then make groups, we get the simplified compliment function.
* To derive the simplified function in POS form from K map, we combine 0s and then get expressions for compliment function and then take compliment. We’ll get the simplified function in POS form.

Top of Form

**WORKSHEET 1**

**Assignment Work**

1. What do you understand by Karnaugh‘s Map?
2. What is framing groups? And explain its types.
3. Given the following function **F(A,B,C) = Σ (0,1,2,5,6)** obtain most simplified response by using K-map.
4. What is Redundant groups explain with example.
5. What is SOP and POS explain with example and notation.
6. Obtain max term expression for the Boolean function given below-

**F(A,B,C) = Π (7, 0, 3, 5)**

1. Given the Boolean function F(A,B,C,D) = ∑ (0,1,2,6,8,9,10) Use K-Map to reduce this function F, using the SOP form. Draw logic gate diagram for the reduced SOP form.
2. Given the Boolean Function X(A,B,C,D) =∏ (3,4,5,7,11,12,13,14.15) Use K-map to reduce this function X, using the POS form. Draw the logic gate diagram for the reduce POS form.
3. Given : F(x, y, z) = ∑ (1,3,7) Verify : F(x, y, z) = π (0,2,4,5,6)
4. Minimize the following function by using K-Map:

F(A,B,C) = A’BC’ + A’BC + ABC’ + ABC