Karnaugh Maps

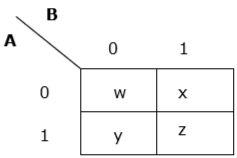
The Karnaugh map (K–map), introduced by Maurice Karnaughin in 1953, is a grid-like representation of a truth table which is used to simplify boolean algebra expressions. A Karnaugh map has zero and one entries at different positions. It provides grouping together Boolean expressions with common factors and eliminates unwanted variables from the expression. In a K-map, crossing a vertical or horizontal cell boundary is always a change of only one variable.

Example 1

An arbitrary truth table is taken below −

|  |  |  |
| --- | --- | --- |
| **A** | **B** | **A operation B** |
| 0 | 0 | w |
| 0 | 1 | x |
| 1 | 0 | y |
| 1 | 1 | z |

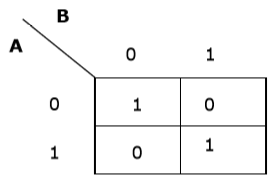
Now we will make a k-map for the above truth table −



Example 2

Now we will make a K-map for the expression – y= AB+ A’B’

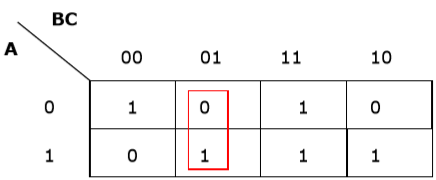
|  |  |  |
| --- | --- | --- |
| A | B | y |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |



Simplification Using K-map

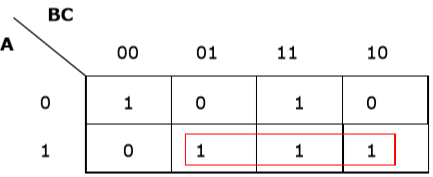
K-map uses some rules for the simplification of Boolean expressions by combining together adjacent cells into single term. The rules are described below −

**Rule 1** − Any cell containing a zero cannot be grouped.



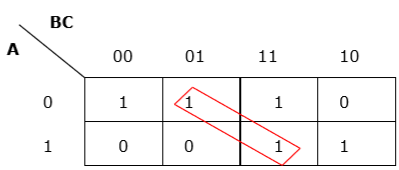
*Wrong grouping*

**Rule 2** − Groups must contain 2n cells (n starting from 1).

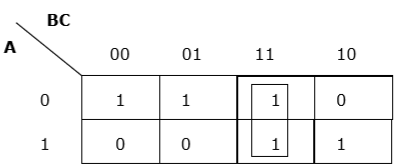


*Wrong grouping*

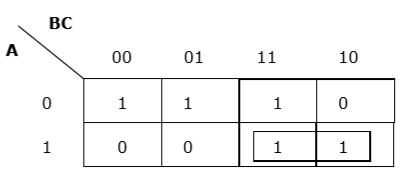
**Rule 3** − Grouping must be horizontal or vertical, but must not be diagonal.



*Wrong diagonal grouping*

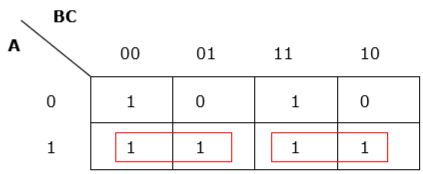


*Proper vertical grouping*



*Proper horizontal grouping*

**Rule 4** − Groups must be covered as largely as possible.

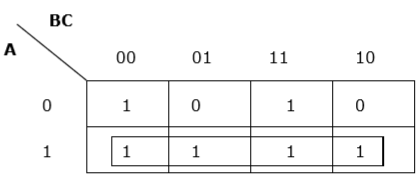


*Insufficient grouping*

8 1’s === octet =3 variables are elimated

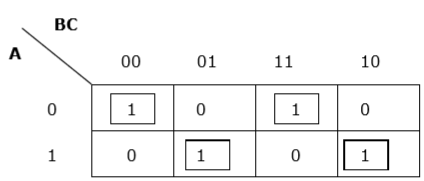
4 1’s =quad =2

2 1’s=pair = 1



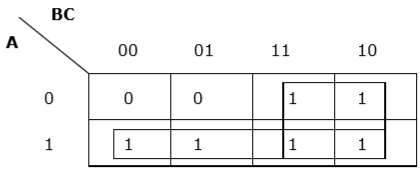
*Proper grouping*

**Rule 5** − If 1 of any cell cannot be grouped with any other cell, it will act as a group itself.



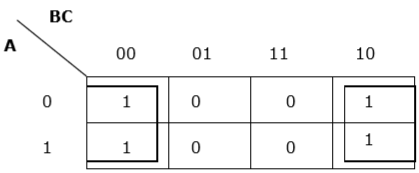
*Proper grouping*

**Rule 6** − Groups may overlap but there should be as few groups as possible.



*Proper grouping*

**Rule 7** − The leftmost cell/cells can be grouped with the rightmost cell/cells and the topmost cell/cells can be grouped with the bottommost cell/cells.



*Proper grouping*

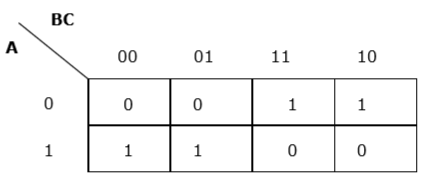
Problem

Minimize the following Boolean expression using K-map −

F(A,B,C)=A′BC+A′BC′+AB′C′+AB′C

Solution

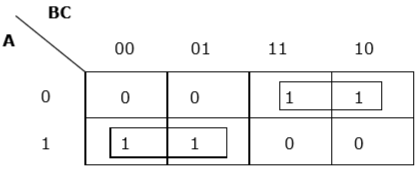
Each term is put into k-map and we get the following −



*K-map for F (A, B, C)*

Now we will group the cells of 1 according to the rules stated above –

B’C’ B’C BC BC’



~A

A

*K-map for F (A, B, C)*

We have got two groups which are termed as A′BA′B and AB′AB′. Hence, F(A,B,C)=A′B+AB′

=A⊕BF(A,B,C)=A′B+AB′=A⊕B. It is the minimized form.

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

## ****421****

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

**Z=** ∑m M,N,O(0,2,4,6)=?

1. Z= ∑A,B,C(1,3,6,7)=A’B’C+A’BC+ABC’+ABC

001 011 110 111

### de1

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

A’C

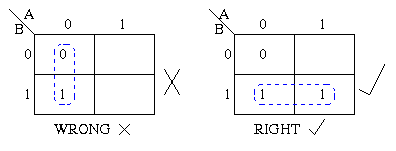
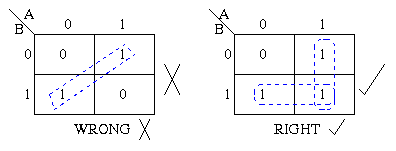
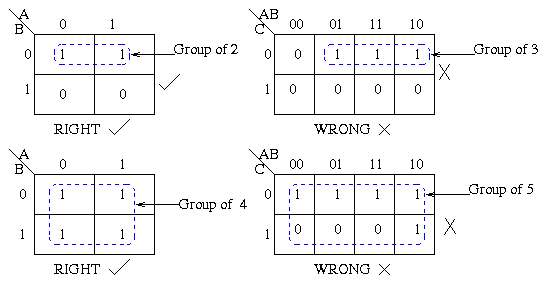
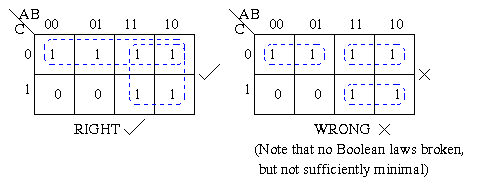
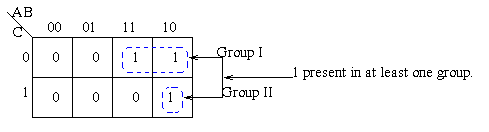
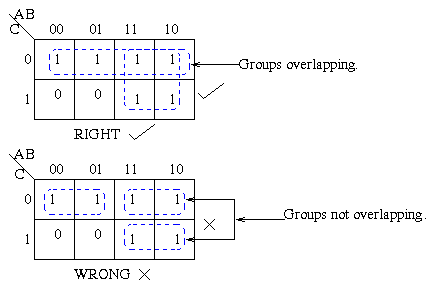
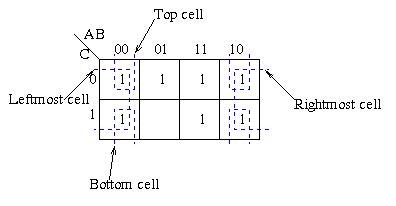
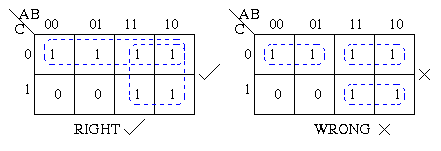
From**green** group we get product term—

AB

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

# Karnaugh Maps - Rules of Simplification

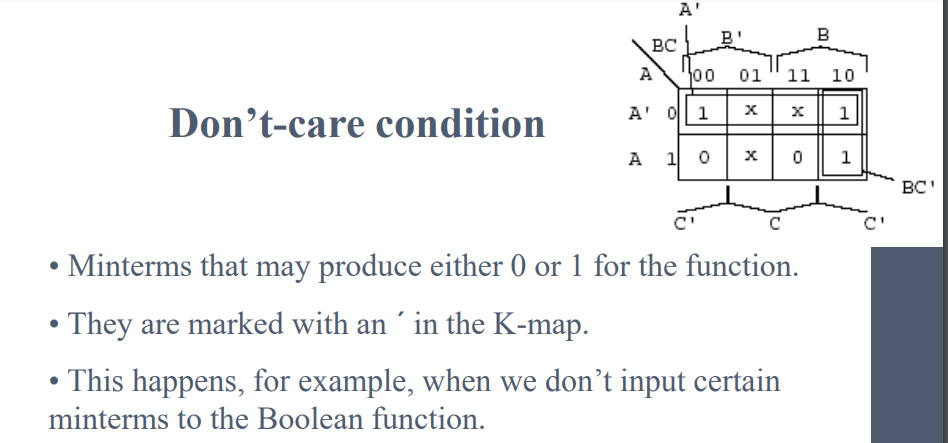
The Karnaugh map uses the following rules for the simplification of expressions by *grouping* together [adjacent](http://www.ee.surrey.ac.uk/Projects/Labview/common/glossary.html#Adj) cells containing *ones*

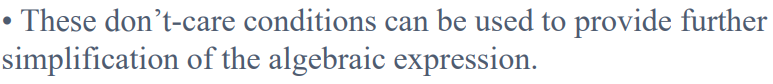
* **Groups may not include any cell containing a zero  
  **
* **Groups may be horizontal or vertical, but not diagonal.  
  **
* **Groups must contain 1, 2, 4, 8, or in general 2n cells.  
  That is if n = 1, a group will contain two 1's since 21 = 2.  
  If n = 2, a group will contain four 1's since 22 = 4.  
  **
* **Each group should be as large as possible.  
  **
* **Each cell containing a *one* must be in at least one group.  
  **
* **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.  
  **

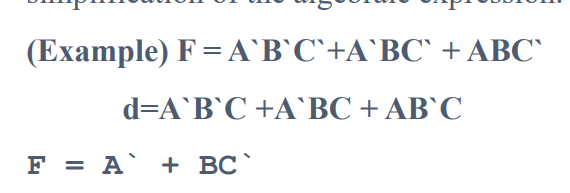
**Summmary:**

1. No zeros allowed.
2. No diagonals.
3. Only power of 2 number of cells in each group.
4. Groups should be as large as possible.
5. Every one must be in at least one group.
6. Overlapping allowed.
7. Wrap around allowed.
8. Fewest number of groups possible.

Ex : f(A,B,C) = ∑m(0,3,5)= A`B`C`+A`BC+AB`C







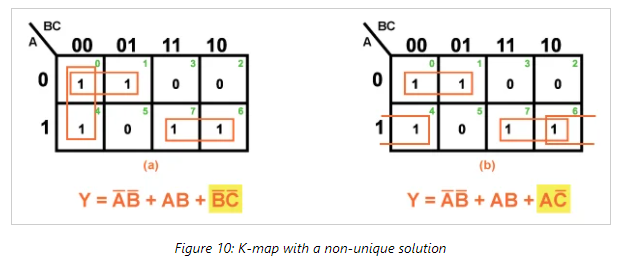
### ****Advantages of K-Maps****

1. The K-map simplification technique is simpler and less error-prone compared to the method of solving the logical expressions using Boolean laws.
2. It prevents the need to remember each and every Boolean algebraic theorem.
3. It involves fewer steps than the algebraic minimization technique to arrive at a simplified expression.
4. K-map simplification technique always results in minimum expression if carried out properly.

### 

### ****Disadvantages of K-Maps****

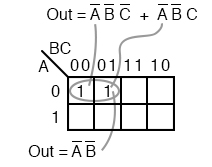
1. As the number of variables in the logical expression increases, the K-map simplification process becomes complicated.
2. The minimum logical expression arrived at by using the K-map simplification procedure may or may not be unique depending on the choices made while forming the groups. For example, for the output variable Y shown by the K-map in Figure 10, we can obtain two different, but accurate logical expressions. The variation in the solution obtained is observed in the third term, which may be either B̅C̅ or AC̅ (highlighted in Figure 10). This difference depends on whether one chooses to group the cells (0,4) or (4,6) to form a two-celled group in order to cover the one found in the K-map cell numbered 4.



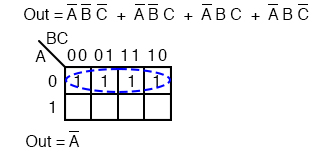
### Examples of Simplification with Karnaugh Maps

Let us move on to some examples of simplification with 3-variable Karnaugh maps. We show how to map the product terms of the unsimplified logic to the K-map.

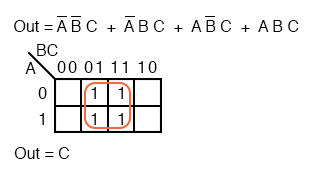
We illustrate how to identify groups of adjacent cells which leads to a Sum-of-Products simplification of the digital logic.



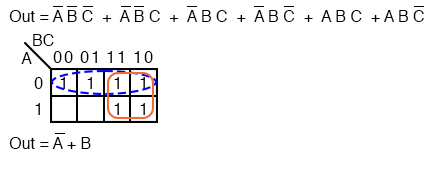
Above we, place the 1’s in the K-map for each of the product terms, identify a group of two, then write a p-term (product term) for the sole group as our simplified result.



Mapping the four product terms above yields a group of four covered by Boolean **A’**

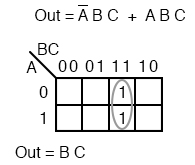


Mapping the four p-terms yields a group of four, which is covered by one variable **C**.

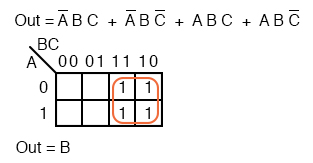


After mapping the six p-terms above, identify the upper group of four, pick up the lower two cells as a group of four by sharing the two with two more from the other group. Covering these two with a group of four gives a simpler result.

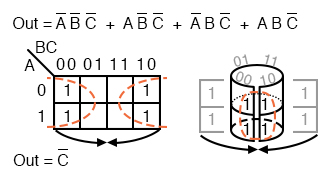
Since there are two groups, there will be two p-terms in the Sum-of-Products result **A’+B**



The two product terms above form one group of two and simplifies to **BC**

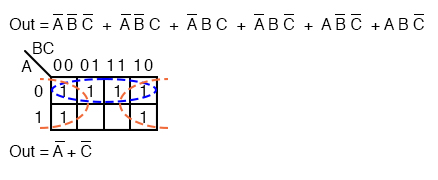


Mapping the four p-terms yields a single group of four, which is **B**



Mapping the four p-terms above yields a group of four. Visualize the group of four by rolling up the ends of the map to form a cylinder, then the cells are adjacent. We normally mark the group of four as above left.

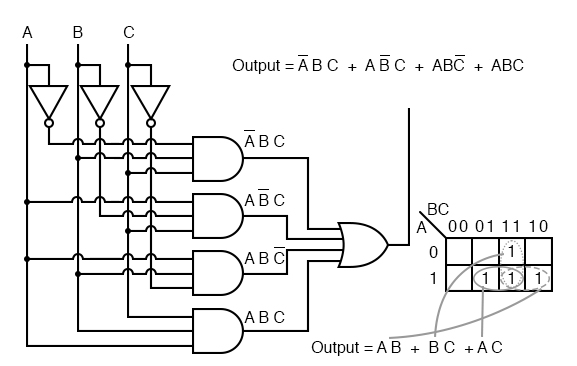
Out of the variables A, B, C, there is a common variable: C’. C’ is a 0 overall four cells. The final result is **C’**

. 

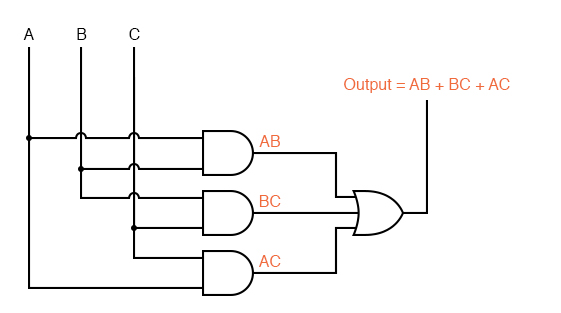
The six cells above from the unsimplified equation can be organized into two groups of four. These two groups should give us two p-terms in our simplified result of **A’ + C’**.

### Simplifying Boolean Equations with Karnaugh Maps

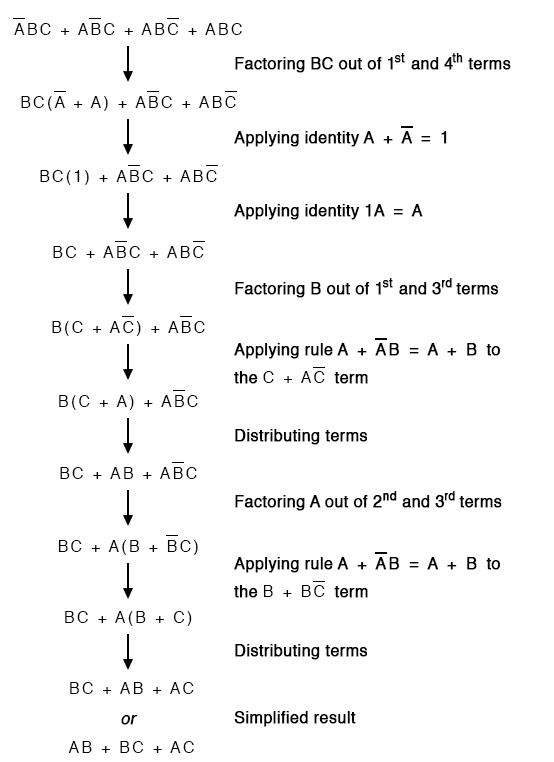
We will simplify the logic using a Karnaugh map.



The Boolean equation for the output has four product terms. Map four 1’s corresponding to the p-terms. Forming groups of cells, we have three groups of two. There will be three p-terms in the simplified result, one for each group. See [Converting Truth Tables into Boolean Expressions](https://www.allaboutcircuits.com/textbook/digital/chpt-7/converting-truth-tables-boolean-expressions/) from chapter 7 for a gate diagram of the result, which is reproduced below.



Below we repeat the Boolean algebra simplification of the toxic waste incinerator for comparison.



q..Design a digital circuit with 3 inputs and 1 output.

abc

Y= ∑m(0,2,4,5)+d(3,6)

Y= c’+(aꚚb)

q.3 inputs and 1 output,

y=1 for first 4 input conditions.

Y=0 for other input conditions

Y=~a

q. Design a digital circuit that will allow input signal A to pass through only when the control inputs b and c are same,otherwise the output is high.

Input control input output

A B C Y

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

Y= a+(CꚚb)