**Karnaugh Maps/K-maps**

Before going through the Karnaugh maps, some terms need to be clarified. These are as follows—

**• Literals :**

A literal is a single logic variable or its complement. For example— X, Y, A’, Z, X’, etc.

**• Minterms:**

A minterm is the **product** of all the literals with or without complement involved in a logic system.  
  
For example—  
AB, A’B, AB’, A’B’ (for a problem containing only A and B),  
A’BC, ABC, AB’C… (for a problem containing A, B and C)

When the values of different variables are given, minterms can be easily formed as-

If X=0, Y=0 minterm would be X’Y’  
If X=1, Y=0, Z=1 minterm would be XY’Z

So, use the variable with value 1 as it is and variable with value 0 as complemented to find the minterm.

**• Maxterm:**

A maxterm is the **sum** of all the literals with or without complement involved in a logic system.

For example—

A+B, A+B’, A’+B, A’+B’ (for problem containing only A and B)  
A+B+C, A’+B+C’, … (for problem containing A, B and C)

When the values of different variables are given, maxterms can be easily formed as-

If X=0, Y=0 maxterm would be X+Y  
If X=1, Y=0, Z=1 maxterm would be X’+Y+Z’

So use the variable with value 1 as complemented and variable with value 0 as it is to find the maxterm.

**• Canonical expressions**:

A Boolean expression containing entirely of minterms or maxterms is known as canonical expression. These are of two types—

**1. Sum of Product(SOP form)**

It is the sum of all the **minterms** that result in a **true value** of the output variable.

For example—

XY’+X’Y+XY  
XYZ’+X’YZ+X’Y’Z+XYZ  
AB+AB’

**2. Product of Sums(POS form)**

It is the product of all the **maxterms** that result in a **false value** of the output variable.

For example—

(X+Y’)(X’+Y)(X+Y)  
(X+Y+Z’)(X’+Y+Z)(X’+Y’+Z)(X+Y+Z)  
(A+B)(A+B’)

**• Shorthand notation**

The minterms and the maxterms can be represented by shorthand notation which makes it very easy and fast to write and work with. Shorthand notations can be obtained as follows-

**For minterm representation**

To represent a minterm as shorthand notation following steps are to be followed-

1. Write 0 for a complemented term and 1 for non-complemented term. This will give us a binary number.
2. The shorthand notation will be an ‘m’ with the decimal equivalent of the binary number as subscript of ‘m’.  
     
   Eg. The minterm XYZ’ is represented as- XYZ’ will be represented as— XYZ’ --> 110 => So  we get m**6**

**For maxterm representation**

**T**o represent a maxterm as shorthand notation following steps are to be followed-

1. Write 1 for a complemented term and 0 for non-complemented term.

2. The shorthand notation will be a capital ‘M’ with the decimal term. This will give you a binary number equivalent of the binary number as subscript of ‘M’.  
  
Eg. The minterm X+Y’ is represented as- X+Y+Z’ will be represented as— X+Y’+Z --> 010 => So **M2**

**Minterm expansion**

Any expression can be represented using minterms. To find the minterm expansion of an expression following steps have to be followed-

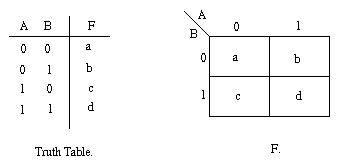
1. Write down all the terms in the expression  
2. Put X where ever a literal is missing to convert the terms to minterms  
3. Use all the combinations of Xs to find minterms  
4. Remove the duplicate/repeated terms and write the terms together.

**KarnaughMap:Definition**

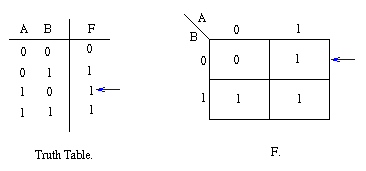
Maurice Karnaugh, a telecommunications engineer, developed the Karnaugh map at Bell Labs in 1953 while designing digital logic based telephone switching circuits.

The Karnaugh map provides a simple and straight-forward method of minimizing Boolean expressions. With the Karnaugh map Boolean expressions having up to four and even six variables can be simplified.A Karnaugh map provides a pictorial method of grouping together expressions with common factors and therefore eliminating unwanted variables. The Karnaugh map can also be described as a special arrangement of a [truth table](http://www.ee.surrey.ac.uk/Projects/Labview/common/glossary.html" \l "tt).

The diagram below illustrates the correspondence between the Karnaugh map and the truth table for the general case of a two variable problem.

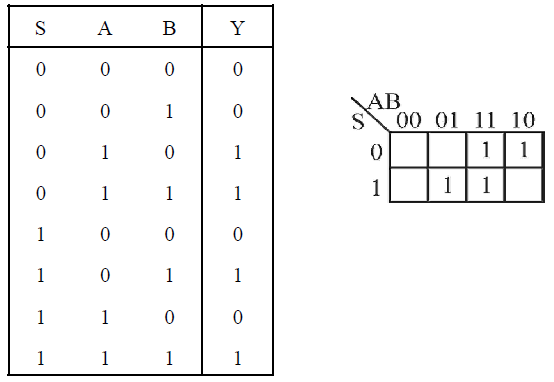


The values inside the squares are copied from the output column of the truth table, therefore there is one square in the map for every row in the truth table. Around the edge of the Karnaugh map are the values of the two input variable. A is along the top and B is down the left hand side. The diagram below explains this:

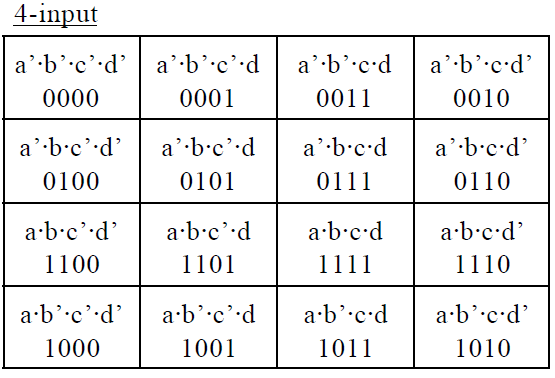


The values around the edge of the map can be thought of as coordinates. So as an example, the square on the top right hand corner of the map in the above diagram has coordinates A=1 and B=0. This square corresponds to the row in the truth table where A=1 and B=0 and F=1. Note that the value in the F column represents a particular function to which the Karnaugh map corresponds.

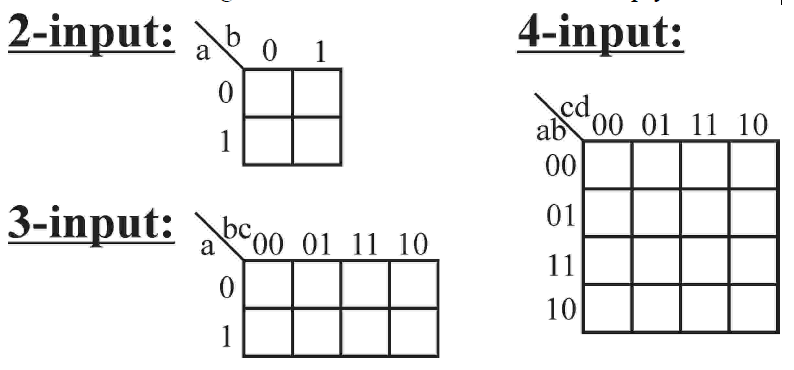
**For 3 input**

****

# 



The cells are arranged as above, but we write them empty, like this:



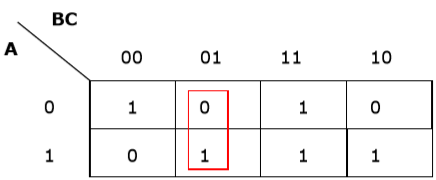
Note that the numbers are *not* in binary order, but are arranged so that

only a single bit changes between neighbours.

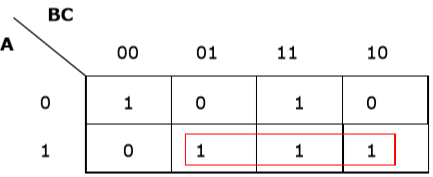
**Simplification Using K- map**

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

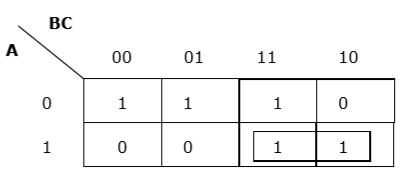
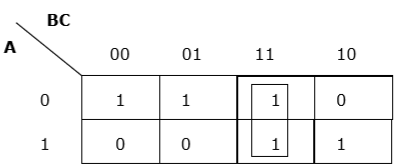
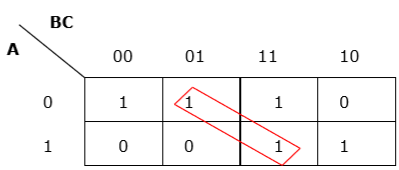
**Rule 1** − Any cell containing a zero cannot be grouped.



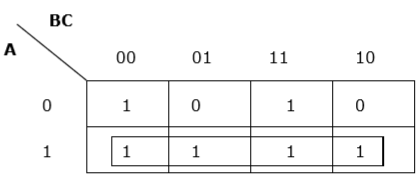
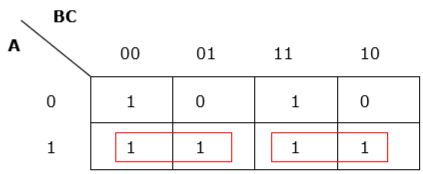
**Rule 2** − Groups must contain 2^n cells (n starting from 1).



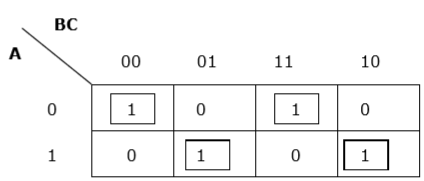
**Rule 3** − Grouping must be horizontal or vertical, but must not be diagonal.



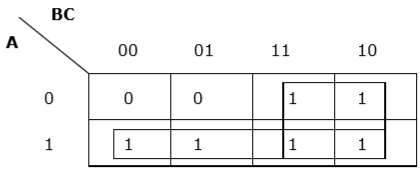
**Rule 4** − Groups must be covered as largely as possible.



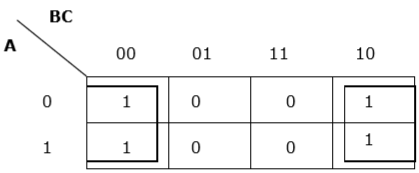
**Rule 5** − If 1 of any cell cannot be grouped with any other cell, it will act as a group itself.



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



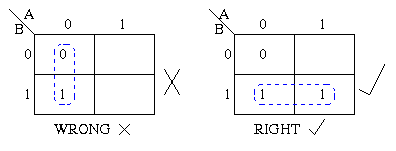
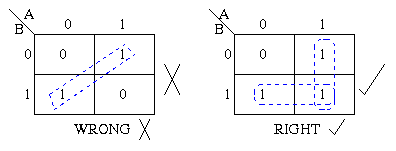
**Rule 7** − The leftmost cell/cells can be grouped with the rightmost cell/cells and the topmost cell/cells can be grouped with the bottommost cell/cells.

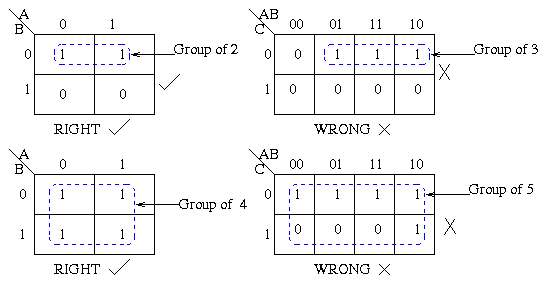


**Summmary:**

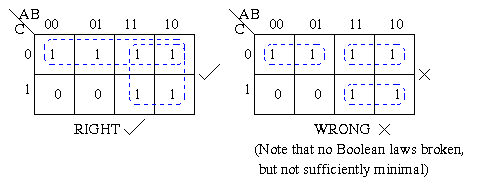
1. No zeros allowed.
2. No diagonals.
3. Only power of 2 number of cells in each group.
4. Groups should be as large as possible.
5. Everyone must be in at least one group.
6. Overlapping allowed.
7. Wrap around allowed.
8. Fewest number of groups possible.

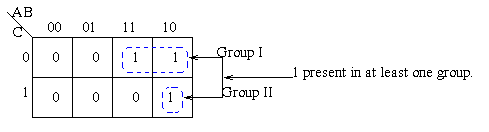
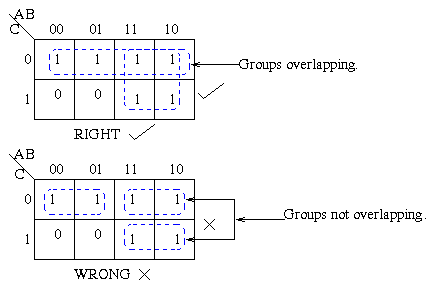
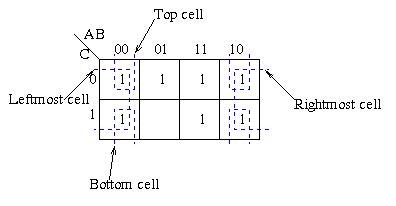
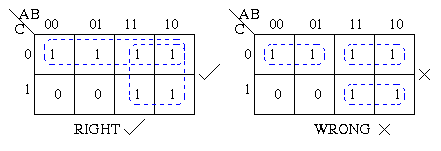
**Some Examples**

* 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 previou.
* s rules.   
  ****

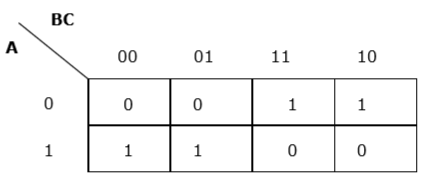
**Problem: Boolean expression**

Minimize the following Boolean expression using K-map –

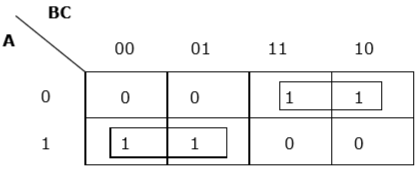
F(A,B,C)=A′BC+A′BC′+AB′C′+AB′C

**Solution**

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



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

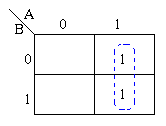


We have got two groups which are termed as A′B and AB′A′B and AB′. Hence, F(A,B,C)=A′B+AB′=A⊕B. It is the minimized form.

**2.1 Adjacent cells**

**Example 1**

Consider the following map. The function plotted is: Z = f(A,B) = Ahttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gif + AB



* Note that values of the input variables form the rows and columns. That is the logic values of the variables A and B (with one denoting true form and zero denoting false form) form the head of the rows and columns respectively.
* Bear in mind that the above map is a one dimensional type which can be used to simplify an expression in two variables.
* There is a two-dimensional map that can be used for up to four variables, and a three-dimensional map for up to six variables.

Using algebraic simplification,

Z = Ahttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gif + AB

Z = A(http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gif + B)

Z = A

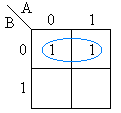
Variable B becomes redundant due to [Boolean Theorem T9a](http://www.ee.surrey.ac.uk/Projects/Labview/boolalgebra/index.html#booleantheorems).

Referring to the map above, the two [adjacent](http://www.ee.surrey.ac.uk/Projects/Labview/common/glossary.html#Adj) 1's are grouped together. Through inspection it can be seen that variable B has its true and false form within the group. This eliminates variable B leaving only variable A which only has its true form. The minimized answer therefore is Z = A.

### 

### Example 2

Consider the following map. The function plotted is http://www.ee.surrey.ac.uk/Projects/Labview/common/graphics/adjac1.gif

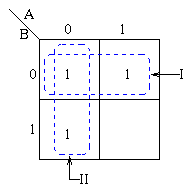
 

Using algebraic simplification, http://www.ee.surrey.ac.uk/Projects/Labview/common/graphics/adjac3.gif by using T9a of the Boolean Laws (A + http://www.ee.surrey.ac.uk/Projects/Labview/common/graphics/adjac4.gif = 1). Referring to the map we can encircle the adjacent cells and infer that A and http://www.ee.surrey.ac.uk/Projects/Labview/common/graphics/adjac4.gif are not required.

If two occupied cells of a Karnaugh are adjacent, horizontally or vertically (but not diagonally) then one variable is redundant. This has resulted by labelling the map as shown, i.e. adjacent cells satisfy the condition A + http://www.ee.surrey.ac.uk/Projects/Labview/common/graphics/adjac4.gif = 1.

**Example 3:**

Consider the expression Z = f(A,B) = http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gifhttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gif + A http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gif + http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gifB plotted on the Karnaugh map:

Pairs of 1's are *grouped* as shown above, and the simplified answer is obtained by using the following steps:

Note that two groups can be formed for the example given above, bearing in mind that the largest rectangular clusters that can be made consist of two 1s. Notice that a 1 can belong to more than one group.

For group I

The first group labelled I, consists of two 1s which correspond to A = 0, B = 0 and A = 1, B = 0.

Put in another way:

All squares in this example that correspond to the area of the map where

B = 0 contains 1s, independent of the value of A.

So when B = 0 the output is 1.

The expression of the output will contain the term http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gif

For group II

For group labelled II corresponds to the area of the map where A = 0.

The group can therefore be defined as http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gif.

This implies that when A = 0 the output is 1.

Therefore, the output is 1 whenever B = 0 and A = 0

Hence the simplified answer is Z = http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gif + http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gif

**Problem 4**

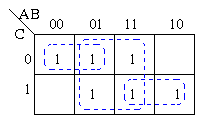
Minimize the following problems using the Karnaugh maps method.

Z = f(A,B,C) = http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gifhttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gif http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/c.gif + http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gifB + ABhttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/c.gif + AC

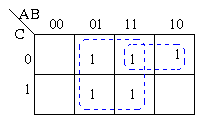
Z = f(A,B,C) = http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gifB + Bhttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/c.gif + BC + Ahttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gifhttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/c.gif

**Answer**

1. Z = f(A,B,C) = http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gifhttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gif http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/c.gif + http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gifB + ABhttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/c.gif + AC

   
By using the [rules](http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/karrules.html) of simplification and ringing of [adjacent](http://www.ee.surrey.ac.uk/Projects/Labview/common/glossary.html#Adj) cells in order to make as many variables redundant, the minimized result obtained is B + AC+ http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gifhttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/c.gif

1. Z = f(A,B,C) = http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/a.gifB + Bhttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/c.gif + BC + Ahttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/b.gifhttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/c.gif

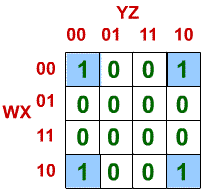
   
By using the [rules](http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/karrules.html) of simplification and ringing of [adjacent](http://www.ee.surrey.ac.uk/Projects/Labview/common/glossary.html#Adj) cells in order to make as many variables redundant, the minimized result obtained is B + Ahttp://www.ee.surrey.ac.uk/Projects/Labview/minimisation/graphics/c.gif

**Four Variables**

**Example 1**

Larger groups in Karnaugh Maps of any size can lead to greater simplification.  Let's consider the group shown shaded below.

There are four terms covered by the shaded area.



 In the upper left:-

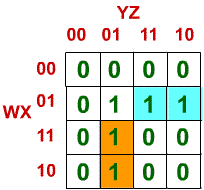
* + https://www.facstaff.bucknell.edu/mastascu/eLessonsHTML/Logic/Logic3A07A.gif
* In the upper right;
  + https://www.facstaff.bucknell.edu/mastascu/eLessonsHTML/Logic/Logic3A07B.gif
* In the lower left;
  + https://www.facstaff.bucknell.edu/mastascu/eLessonsHTML/Logic/Logic3A07C.gif
* In the lower right;
  + https://www.facstaff.bucknell.edu/mastascu/eLessonsHTML/Logic/Logic3A07D.gif

These terms can be combined (assuming they are all ones in the Karnaugh Map!).  The result is

By combining the first two terms above (the two terms at the top of the Karnaugh Map):-

* + https://www.facstaff.bucknell.edu/mastascu/eLessonsHTML/Logic/Logic3A07E.gif
* By combining the last two terms above (the two terms at the bottom of the Karnaugh Map):-
  + https://www.facstaff.bucknell.edu/mastascu/eLessonsHTML/Logic/Logic3A07F.gif
* Then, these two germs can be combined to give:
  + https://www.facstaff.bucknell.edu/mastascu/eLessonsHTML/Logic/Logic3A07G.gif

**Example 2**



* + In this example, the two terms shown are:
    - https://www.facstaff.bucknell.edu/mastascu/eLessonsHTML/Logic/Logic3A09A.gif
    - https://www.facstaff.bucknell.edu/mastascu/eLessonsHTML/Logic/Logic3A09B.gif