Contents

Logic Simplification Using De Morgans Law and other Boolean algebra properties

When we simplify algebraic expressions using Boolean Algebra, we decrease the number of gates needed to implement a circuit.

Boolean algebra allows us to apply provable mathematical principles to design logic circuits.

The final result/output in the truth table of the original expression when compared to the truth table after reducing an expression should be ***identical***.

This demonstrated ***circuit equivalence***.

That is, both circuits produce exactly the same output for each input value combination.

The complexity of the expression representing a Boolean function has a direct effect on the complexity of the resulting digital circuit: more complex the expression, the more complex the circuit.

De Morgans Law and other ***Boolean algebra properties*** provide a formal mechanism for defining, managing and evaluating logic circuit design.

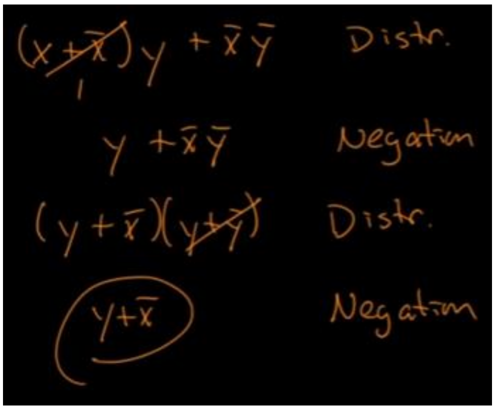
Circuits are not typically simplified using Boolean laws of algebra: they can be difficult and time consuming.

Instead, ***Karnaugh maps (K-maps)*** are used by designers as a more automated method to do this

<u>Example 1.</u>

Simplify the Boolean Expr. $\mathbf{xy + \bar{x}y + \bar{x}\bar{y}}$

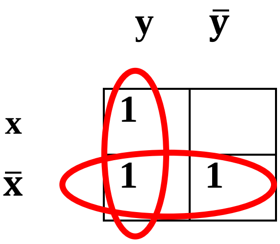
(1) Simplified using Boolean Rules

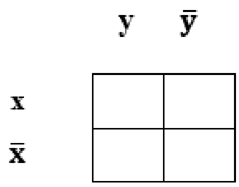


Therefore $\mathbf{xy + \bar{x}y + \bar{x}\bar{y} = y+\bar{x}}$

(2) Simplified using Karnaugh Maps

Karnaugh Maps allow you to write this expression as a picture and simplify from there:





As above, $\mathbf{xy + \bar{x}y + \bar{x}\bar{y} = y+\bar{x}}$

Logical Simplification Using Karnaugh Maps

So far we can see that applying Boolean algebra can be awkward in order to simplify expressions.

Apart from being laborious (and requiring the remembering all the laws) the method can lead to solutions which, though they appear minimal, are not.

The Karnaugh map provides a simple and straight-forward method of minimising Boolean expressions. With the Karnaugh map Boolean expressions having up to four and even six variables can be simplified.

So what is a Karnaugh map?

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

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*

Karnaugh mapping is a ***systematic step-by-step approach***.

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

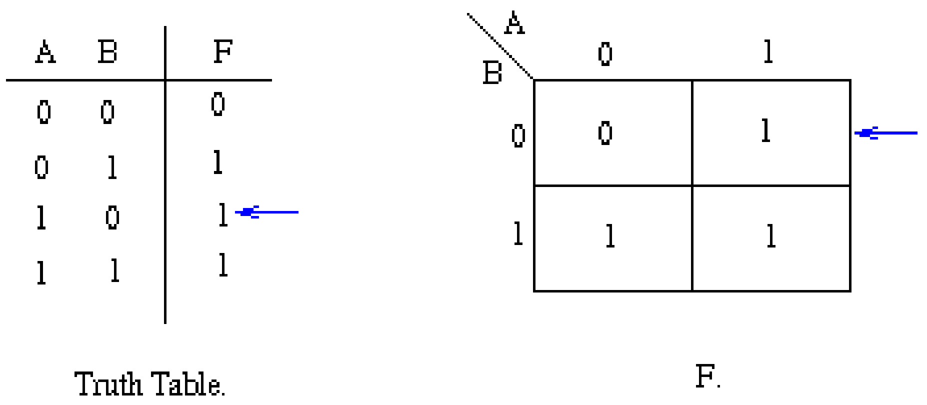


Figure 1 K-Map of truth table

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.

Karnaugh Mapping

Logical AND = . = ∩

Logical OR = * = U

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.

Consider the following map. The function plotted is:

$$Z = f(A,B) = A\overline{B} + AB$$

$\overline{A}\overline{B}$	$A\overline{B}$
00	**01**
$\overline{A}B$	AB
10	**11**

Figure 3 Layout of general 2-input K-Map

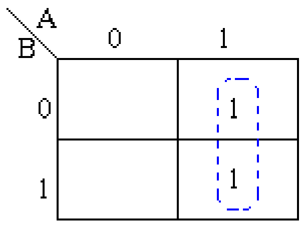
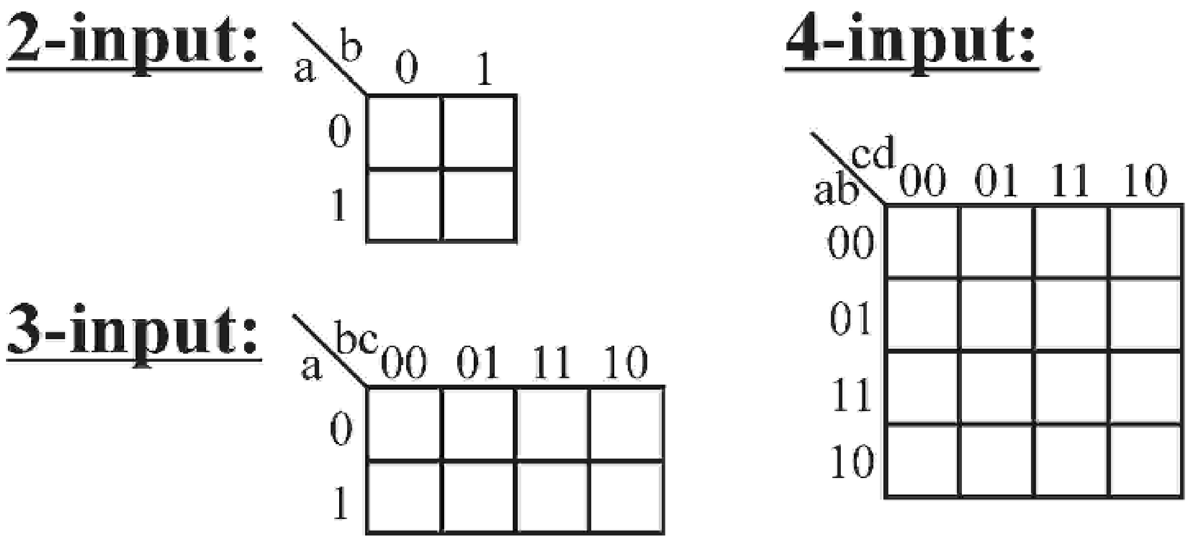


Figure 2 Boolean expr as K-Map

- 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.



Simplification using algebraic simplification and also its equivalent K-map:

$\bar{A}\bar{B}$	$A\bar{B}$
00	**01**
$\bar{A}B$	AB
10	**11**

$Z = A\bar{B} + AB$

$Z = A(\bar{B} + B)$

$Z = A$

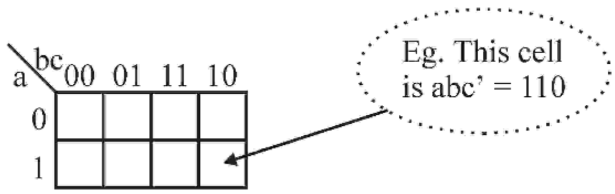
0	**0**
1	**1**

Variable B becomes redundant (due to which Boolean rule?)

Referring to the map above, the two adjacent 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 minimised answer therefore is Z = A.

Note that the numbers are not in binary order, but are arranged so that only ***a single bit changes between adjacent cells***

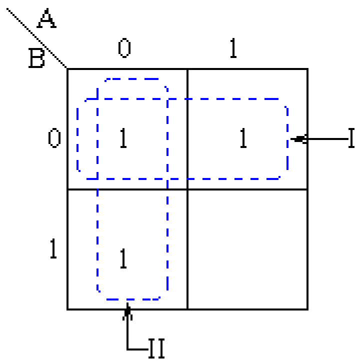
So cells in the same row on the left and right edges of the array also only differ by one bit.



Note: The value of a particular cell is found by combining the numbers at the edges of the row and column.

Example 2.

Consider the expression Z = f(A,B) = $\bar{A}\bar{B} + A\bar{B} + \bar{A}B$ 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.

1. 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 $\bar{B}$

2. For group labelled II corresponds to the area of the map where A = 0. The group can therefore be defined as $\bar{A}$.
   This implies that when A = 0 the output is 1. The output is therefore 1 whenever B = 0 and A = 0

Hence the simplified answer is Z = $\bar{A} + \bar{B}$

More examples of groupings

Example 3. $AB+\bar{C}D$

AB\CD	00	01	11	10
00	0	1	0	0
01	0	1	0	0
11	1	1	1	1
10	0	1	0	0

Example 4. $B\overline{D}+ABC$

AB\CD	00	01	11	10
00	0	0	0	0
01	1	0	0	1
11	1	0	1	1
10	0	0	0	0

Compared to the algebraic method, the K-map process is a ***more orderly process*** requiring fewer steps and ***always*** producing a minimum expression.

Example 5. Simplification Example

a. Unsimplified Boolean Expression

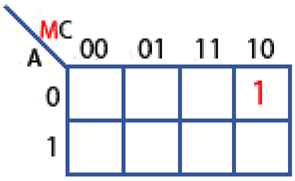
X = M + M•C + A•C + A•M + A•C•M

b. Truth Table

A	M	C	X	Boolean
0	0	0	0	
0	0	1	0	
0	1	0	1	M
0	1	1	1	M•C
1	0	0	0	
1	0	1	1	A•C
1	1	0	1	A•M
1	1	1	1	A•M•C

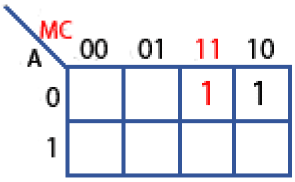
c. Karnaugh Map

i. From the truth table row 3, inputs AMC have values of 010, producing a logic 1 at the output (X) and giving the Boolean expression M in the Boolean column. Therefore 1 is placed in the map cell corresponding to A=0 and MC=10



ii. row 4, inputs AMC have values of 011, producing a logic 1 at the output (X)

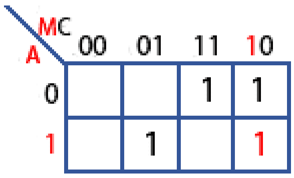
1 is placed in the map cell corresponding to A=0 and MC=11



iii. row 5, output (X), is 0 so this row is ignored.

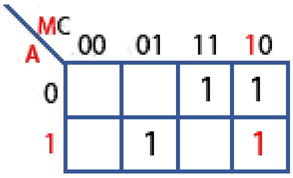
iv. row 6, inputs AMC have values 101, producing a logic 1 at the output (X)

1 is placed in the map cell corresponding to A=1 and MC=01



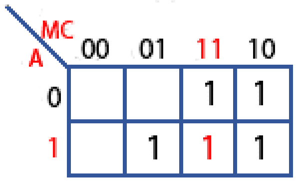
v. row 7, the inputs AMC have values of 110, producing a logic 1 at the output (X)

1 is placed in the map cell corresponding to A=1 and MC=10



vi. row 8 the inputs AMC have values of 111 producing a logic 1 at the output (X)

1 is placed in the map cell corresponding to A=1 and MC=11



All the truth table rows that produced a logic 1 have now been entered into the map and those lines that produced a logic 0 can be ignored, so the remaining three cells are left blank

The map is ready for **simplification**

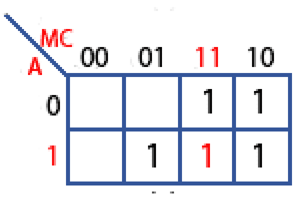
Circuit simplification in any Karnaugh map is achieved by ***combining the cells containing 1 to make groups of cells***.

In grouping the cells it is necessary to follow **six rules**:

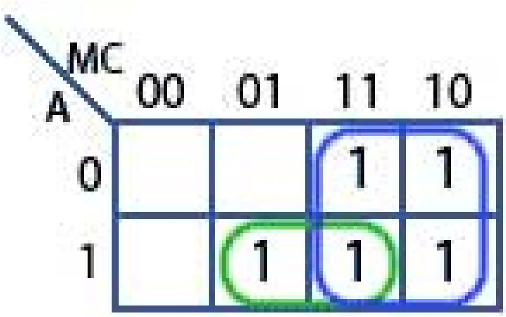
Rules for K-Maps

(1) Each cell with a 1 must be included in at least one group.
(2) Try to form the largest possible groups.
(3) Try to end up with as few groups as possible.
(4) Groups may be in sizes that are powers of 2: $2^0 = 1$, $2^1 = 2$, $2^2 = 4$, $2^3 = 8$, $2^4 = 16$, ...
(5) Groups may be square or rectangular only (including wrap-around at the grid edges). No diagonals or zig-zags can be used to form a group.
(6) The larger a group is, the more redundant inputs there are:
   a. A group of 1 has no redundant inputs.
   b. A group of 2 has 1 redundant input.
   c. A group of 4 has 2 redundant inputs.
   d. A group of 8 has 3 redundant inputs.
   e. A group of 16 has 4 redundant inputs

We are ready to simplify the K-map for the expression **X = M + M•C + A•C + A•M + A•M•C**



Converting the two groups in the Karnaugh map to Boolean expressions is done by discovering which input or inputs (A, M or C) ***does NOT change*** within each group.



vii. Notice that the blue group spans two rows vertically, and so contains rows A=0 and A=1, therefore A ***changes*** within the group so cannot appear in the expression.

The blue group also spans two columns and so contains MC=11 and MC=10. Here, C = both 1 and 0, but M=1 in both columns.

Therefore the **only input that does not change** in the blue group is **M**, so the Boolean expression for the blue group is simply **M**.

viii. Looking at the (green) group of 2, **A** does not change but MC changes from 01 to 11. This indicates that although M changes, C does not. Therefore there are two **non-changing inputs** in this group A and C.

Putting the results of the simplification together by 'ANDing' any non-changing inputs within a single group, and 'ORing' the different groups, produces the simplified Boolean equation for the whole circuit:

X = M + A•C

The main advantage of using a Karnaugh map for circuit simplification is that the Karnaugh method uses fewer rules, and these rules can be applied systematically rather than intuitively as with Boolean algebra.

These advantages become more apparent when minimising more complex circuits.

Example 6. Simplify this following larger Boolean expression and associated truth table.