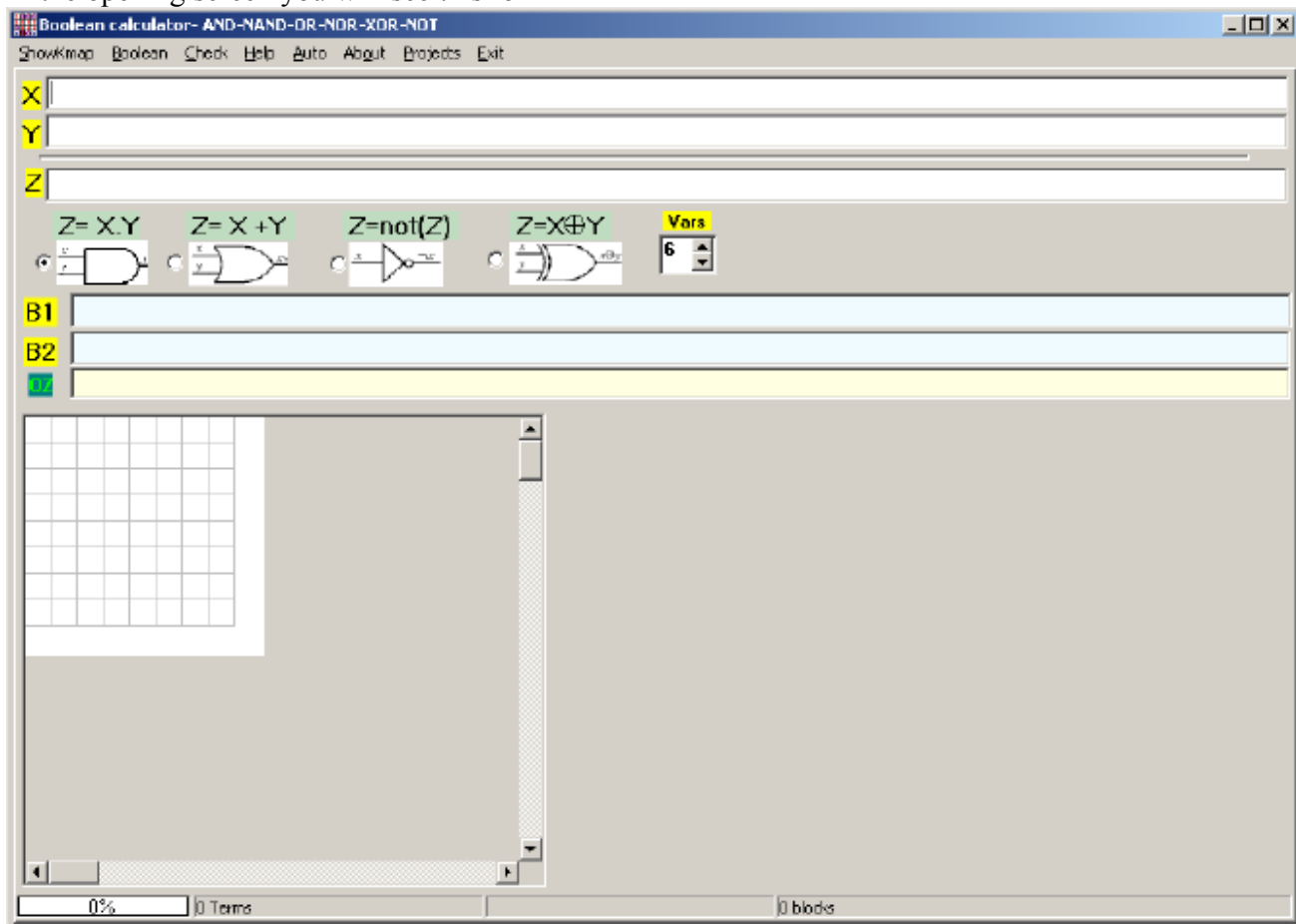


**KMAP3** is a very powerful Boolean calculator is a very powerful tool for evaluating, calculating of Boolean Algebra expressions. The **best** Boolean calculator on internet ! Even wolframalpha.com does not have such powerful tool. This software is a very useful tool for testing **DeMorgan** theorems and investigating **set theory** identities. In addition to handling of Boolean algebra calculation it can also handle more basic **Karnaugh Map(KMAP)** problems, in which better optimization are done. This Boolean calculator can handle terms with 4 up to 16 variables. No of vars can be controlled by the **up/dn** switch box. Currently up to 16 var terms are allowed. Instead of *Venn diagram* a KMAP diagram is used for better visualization of the terms and expressions. This app is also very useful for digital electronic engineers to optimize their digital circuitry. Some previous knowledge of Boolean Algebra and variable name format readers are assumed to have. In the opening screen you will see this form



### Basic Operation.

This calculator has 2 input editors X and Y and an output editor Z . The Z edit box is updated when corresponding Boolean operations AND, OR, NOT icons are clicked. B1(buffer 1) and B2(buffer 2) edit boxes are simply 2 temporary buffers to store the intermediate results. The OZ (Optimized Z) is just to notify you that the value in Z edit box can be further optimized and reduced to shorter expressions . Using drag and drop values of edit boxes X, Y, Z, B1 and B2 and OZ can be copied into another edit box without using copy and paste. Drag & Drop click on the yellow labels.

In case of NOT(complement) if you don't want the NOT value overwrite the Z input, you can type the Boolean in the Y edit box , but you must enable *auto calculation* by clicking on its menu. Then as you type your Boolean expression in the Y input the Z input automatically updated with the complement value(NOT) of Y input. Notice when NOT icon is clicked the X edit box is automatically disabled and you can not type in it, unless you click on another Boolean icon.

**Important** in the bottom part of screen you see a grid which is automatically updated with the Karnaugh Map boolean expression in the active edit box(edit box that has the focus) as you type in to give you a visual feeling how the KMAP looks like. As it was mentioned KMAP grid is better boolean representation than **Venn Diagram**.

**Some examples**

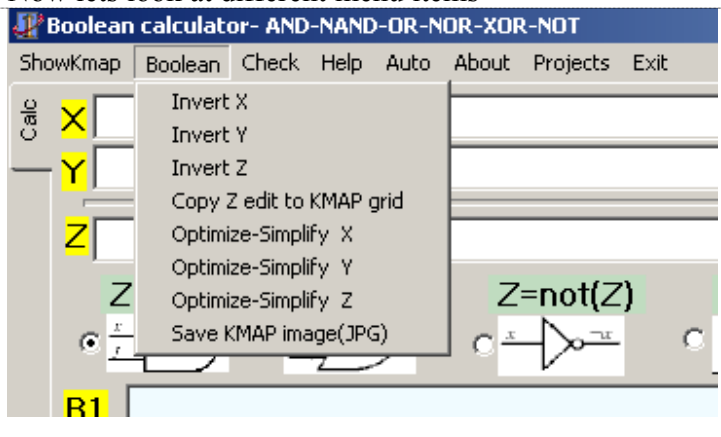
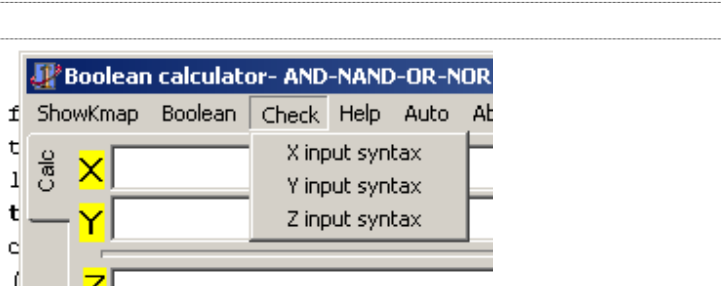
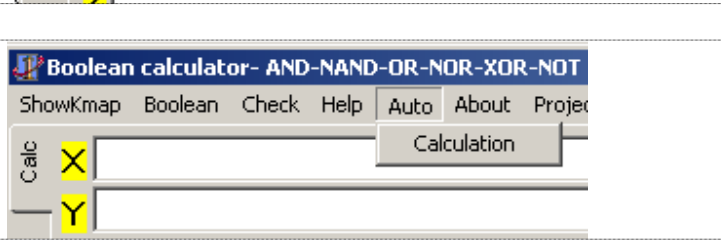
In these example lo state of logic is presented lower case characters for example instead of **A'** we use **a**  
 For all these for examples we assume  $X=Ab+Bc$  and  $Y=aB+bC$

**1-Boolean AND** operation. Click on the AND icon. You will see Z input is  $Z=AbC+aBc$  . If in the main menu you select Auto→ calculation then you don't need to click on the AND icon. As you type the Z output box is automatically updated with logical AND value of X AND Y . You can think of Z as the raw out put while OZ is optimized value of Z. If you want a NAND operation click on NOT and you get  $Z=AB+Ac+BC+aC+ab+bc$

**2-Boolean OR** operation. Click on **OR** icon . The new value of  $Z=Ab+Bc+aB+bC$ . If you want the **NOR** operation then you click on NOT and you get  $Z=ABC+abc$ . Notice in the case of **OR** the most optimized answer is  $Z=Ab+Bc+aC$ . Which is obtained by using more comprehensive **Karnaugh Map(KMAP)** method of simplification and optimization. To use this advanced optimization just Click on Boolean→Z input to kmap grid and then click on the **Simplify** in that form main menu.

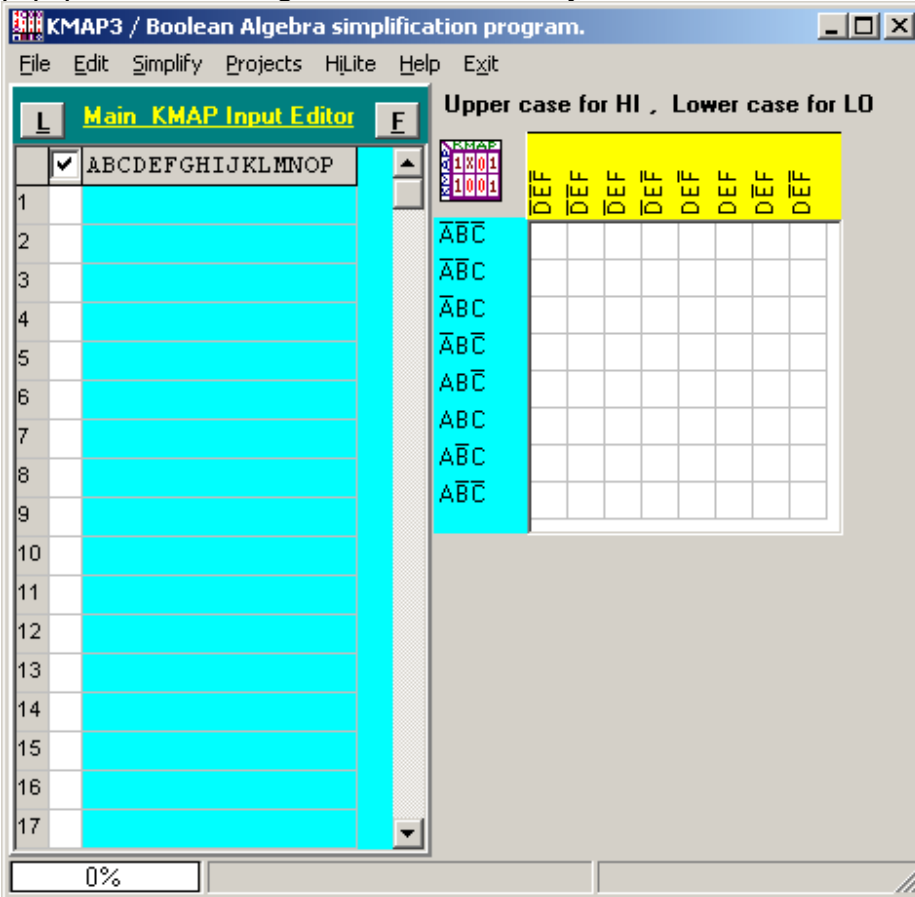
**3-Boolean XOR** operation . Click on **XOR** icon. The new value of  $Z=ABc+Abc+aBC+abC$  but in the optimized edit box(OZ) you will see value of  $Ac+aC$  which is the same as **A XOR C**. If you want **XNOR** instead just click on NOT icon again and you will see result  $AC+ac$  . But in this case the optimized answer was calculated in the first run.

Now lets look at different menu items

	<p>Commands available from <b>Boolean menu</b> . The last menu item will allow you to save the visual image of current edit box Karnaugh Map (KMAP) representation. Later we will talk about the 4<sup>th</sup> item in the menu <b>Copy Z edit to KMAP grid</b> . Which will invoke the KMAP editor, that will allow you to do a better optimization and enter the data by clicking on KMAP grid instead of typing it. Clicking on this menu the same effect as ShowKmap but the data will be copied.</p>
	<p>Commands available from <b>Check menu</b> to check for Boolean syntax error.</p>
	<p><b>Auto Calculation</b> will calculate the Z input as you type in the X or Y inputs without needing to click on any icon. If you want this handy feature make sure the right icon was clicked on first.</p>
<p>By clicking the <b>Projects</b> menu simply a new form will popup that have the boolean equations of a 4 bit adder. Here you will see one complex project of 8 input . This page is very experimental and still under construction and</p>	

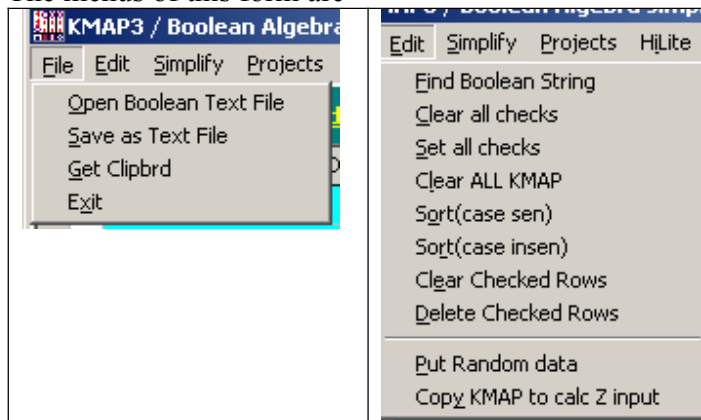
consideration. It is mainly for information only.

The first menu item **ShowKmap** is a very important part of this application. The **ShowKmap** menu is very similar to **Copy Z edit to KMAP grid** that was mentioned in the **boolean menu** . When you click on it this form will popup. This form using File menu will allow you to save the KMAP data as both text and binary file .

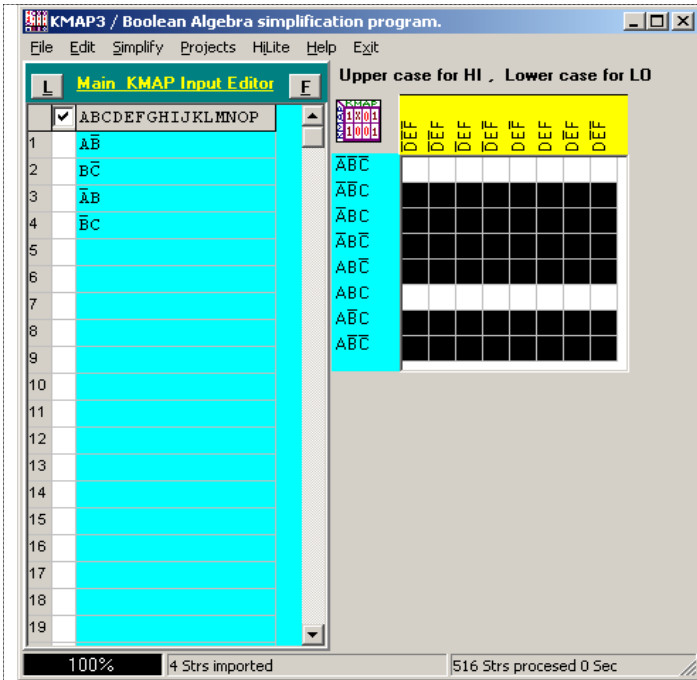


In the left you see an editor box in which each line can only contain one term only. You can not type + sign in it. And in the right you see the KMAP visual presentation of the terms in the left hand editor. Unlike the editor in the calculator form you will see true visual presentation of *low logic* by seeing an upper case character with a bar above it. To see how this form works lets go back to the 2<sup>nd</sup> example using **OR** operation. We have  $X=Ab+Bc$  ,  $Y=aB+bC$  the  $Z=Ab+Bc+aB+bC$  . But we are not sure if Z is really optimized. By clicking on the **Copy Z edit to KMAP grid** you will see this populated form

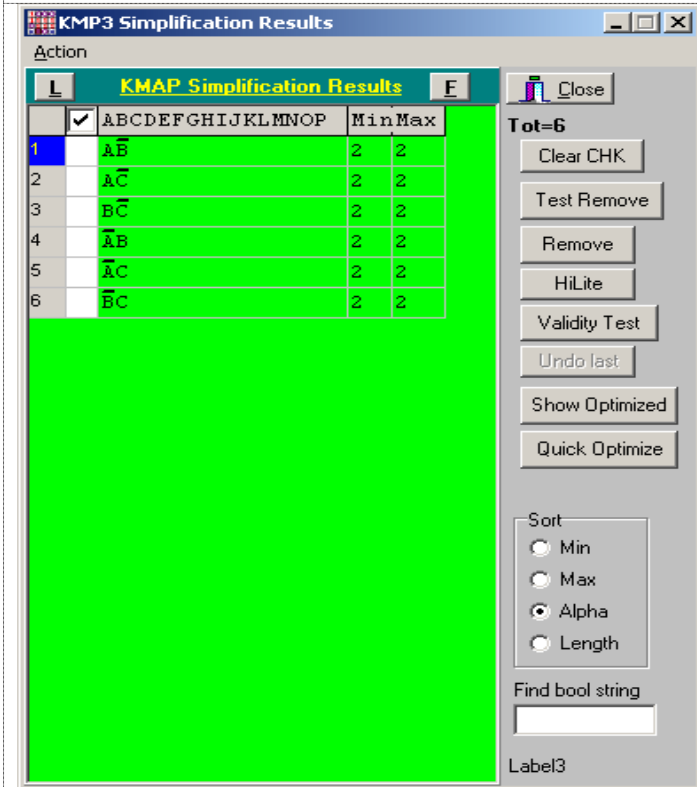
The menus of this form are



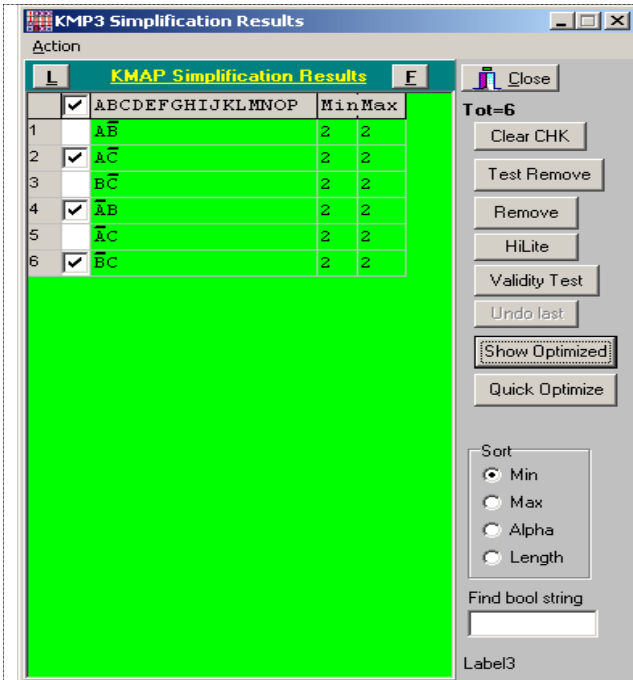
All the features and commands are available from the main menu, which are self explanatory. The 4<sup>th</sup> item in the main menu is **HiLite** . By clicking on this menu the terms that are checked in the left editor are temporary flashed yellow for better visual understanding and distinguishing and overlapping that boolean terms can have.



The terms  $Ab$ ,  $Bc$ ,  $aB$ ,  $bC$  are displayed in the left editor with correct boolean visual presentation. And in the right side the KMAP visual representation. In this example 6 variables from A ..F are used but we could have simply just use 4 variables from A to D using the *vars* switch. In-order to simplify and optimize this KMAP to the max now you click on the Simplify menu. You will see the new screen bellow

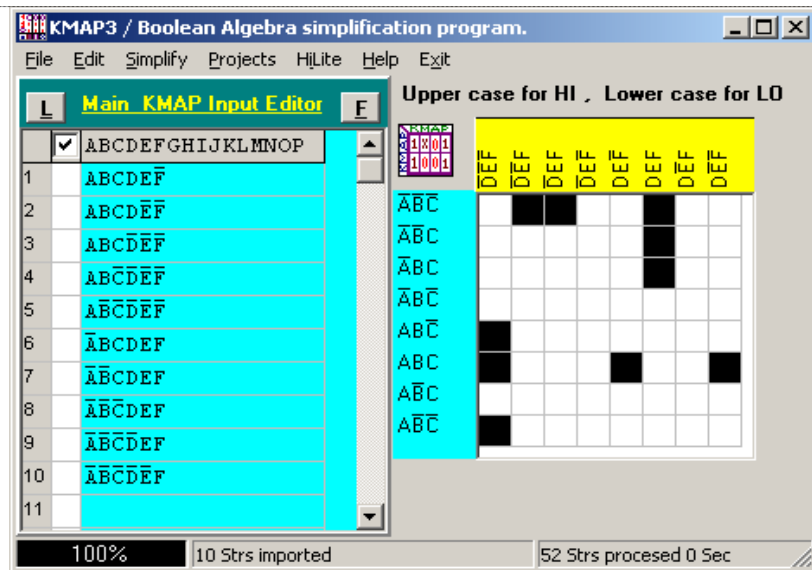


This is the screen where unlike calculator which uses string comparison and reduction method used a binary brute force calculations are done to find the minimized number of terms. We notice 2 more new terms  $Ac$  and  $aC$  are also added because they are valid terms that will cover the cells of KMAP. Here you can do some manual tweaking to see which terms can be removed or click of **Show Optimized** to see which one of these 6 terms can be removed. If you click on **Show Optimized** you will see the same screen with suggested terms that can be removed.



The software will suggest to you that 2<sup>nd</sup>(Ac) , 4<sup>th</sup>(aB) and 6<sup>th</sup>(bC) terms can be removed. Next you click on remove . This KMAP is now optimized to  $Ab+Bc+aC$  . By clicking on the Action menu you see selection of choices with where and how save and copy these 3 optimized terms.

In the 2<sup>nd</sup> example given  $X=Ab+Bc+Cd+De+Ef$  and  $(1)Y=aB+bC+cD+dE+eF$  find  $Z=X \text{ XOR } Y$  . The Z output of this XOR is  $Z=ABCDEF+ABCDEF+ABCdef+ABcdef+Abcdef+aBCDEF+abCDEF+abcDEF+abcdEF+abcdeF$  and  $OZ=ABCdf+ABCef+ABdef+Acdef+aCDEF+abDEF+abcEF+abcdF$  . Z has 10 and OZ has 8 terms . We Click on the menu Boolean-→Copy Z edit to kmap grid . We will see the KMAP for with populated date



Next we will click on the Simplify menu and we will see a form that has a new optimized solution with 8 terms

**KMP3 Simplification Results**

Action

L	AB CDEFGHIJKLMNOP	Min	Max
1	ABCD $\bar{F}$	1	2
2	$\bar{A}$ CDEF	1	2
3	$\bar{A}$ CDEF	1	2
4	$\bar{A}$ BCD $\bar{F}$	1	2
5	ABCE $\bar{F}$	2	2
6	ABDE $\bar{F}$	2	2
7	$\bar{A}$ BDE $\bar{F}$	2	2
8	$\bar{A}$ BCE $\bar{F}$	2	2

Tot=8

Buttons: Clear CHK, Test Remove, Remove, HiLite, Validity Test, Undo last, Show Optimized, Quick Optimize

Sort:  Min,  Max,  Alpha,  Length

Find bool string:

0 selected

Min and Max column indicates how many times that block of variable is shared with other variables. If Min is 1 then that term can not be removed. But if Min greater than 1 it is possible to remove it if KMAP blocks of those terms cover all of its blocks. We now click on Show Optimized button and we will see. 6<sup>th</sup> and 8<sup>th</sup> terms can be removed. Click on **Remove** button

**KMAP Simplification Results**

L	AB CDEFGHIJKLMNOP	Min	Max
1	ABCD $\bar{F}$	1	2
2	$\bar{A}$ CDEF	1	2
3	$\bar{A}$ CDEF	1	2
4	$\bar{A}$ BCD $\bar{F}$	1	2
5	ABCE $\bar{F}$	2	2
6	<input checked="" type="checkbox"/> ABDE $\bar{F}$	2	2
7	$\bar{A}$ BDE $\bar{F}$	2	2
8	<input checked="" type="checkbox"/> $\bar{A}$ BCE $\bar{F}$	2	2

from Action menu you can save this data or copy it back to calculator for further operation.

$Z = ABCDf + ABCef + Acdef + aCDEF + abDEF + abcdF$  has only 6 terms. To double check this calculation we can use the Boolean identity  $X = (X \text{ xor } Y) \text{ xor } Y$  where  $X = ABCDf + ABCef + Acdef + aCDEF + abDEF + abcdF$  and  $Y = Ab + Bc + Cd + De + Ef$ .  $X \text{ xor } Y$  should give us  $Z = aB + bC + cD + dE + eF$ . But we will notice  $Z = AbC + AbD + AbE + AbF + BcD + BcE + BcF + CdE + CdF + DeF + aBc + aCDEf + aCd + aDe + aEf + abDEF + abcdF + bCd + bDe + bEf + cDe + cEf + dEf$  and  $OZ = aB + aC + aD + aE + bC + bD + bE + bF + cD + cE + cF + dE + dF + eF$ , although both look very different than  $aB + bC + cD + dE + eF$  but indeed they are the same. With the same method we copy the OZ data into the KMAP form and we will see it populated

**KMAP3 / Boolean Algebra simplification program.**

File Edit Simplify Projects HiLite Help Exit

Main KMAP Input Editor

L	AB CDEFGHIJKLMNOP
1	$\bar{A}\bar{B}C$
2	$\bar{A}\bar{B}D$
3	$\bar{A}\bar{B}E$
4	$\bar{A}\bar{B}F$
5	$\bar{B}\bar{C}D$
6	$\bar{B}\bar{C}E$
7	$\bar{B}\bar{C}F$
8	$\bar{C}\bar{D}E$
9	$\bar{C}\bar{D}F$
10	$\bar{D}\bar{E}F$
11	$\bar{A}\bar{B}C$

Upper case for HI, Lower case for LO

DEF DEF DEF DEF DEF DEF DEF DEF

100% 23 Strs imported 670 Strs processed 0 Sec

**KMAP Simplification Results**

L	AB CDEFGHIJKLMNOP	Min	Max
1	$\bar{A}\bar{B}$	1	7
2	$\bar{B}\bar{C}$	1	8
3	$\bar{C}\bar{D}$	1	9
4	$\bar{D}\bar{E}$	1	8
5	$\bar{E}\bar{F}$	1	7
6	$\bar{A}\bar{C}$	2	8
7	$\bar{B}\bar{D}$	2	9
8	$\bar{C}\bar{E}$	2	9
9	$\bar{D}\bar{F}$	2	8
10	$\bar{A}\bar{D}$	3	9
11	$\bar{B}\bar{E}$	3	9
12	$\bar{C}\bar{F}$	3	9
13	$\bar{A}\bar{E}$	4	9
14	$\bar{B}\bar{F}$	4	9
15	$\bar{A}\bar{F}$	5	9

Tot=15

Buttons: Clear CHK, Test Remove, Remove, HiLite, Validity Test, Undo last, Show Optimized, Quick Optimize

Sort:  Min,  Max,  Alpha

Next we click on Show Optimized

Term	Min	Max
$\overline{A}B$	1	7
$\overline{B}C$	1	8
$\overline{C}D$	1	9
$\overline{D}E$	1	8
$\overline{E}F$	1	7
$\overline{A}C$	2	8
$\overline{B}D$	2	9
$\overline{C}E$	2	9
$\overline{D}F$	2	8
$\overline{A}D$	3	9
$\overline{B}E$	3	9
$\overline{C}F$	3	9
$\overline{A}E$	4	9
$\overline{B}F$	4	9
$\overline{A}F$	5	9

Buttons: Close, Tot=15, Clear CHK, Test Remove, Remove, HiLite, Validity Test, Undo last, Show Optimized, Quick Optimize. Sort:  Min,  Max,  Alpha

As we the final optimization result is  $aB+bC+cD+dE+eF$  the same as (1) $Y=aB+bC+cD+dE+eF$

If we copy the Z data in the KMAP form, we will see the a different populated form with different terms, but same kmap grid.

Upper case for HI , Lower case for LO

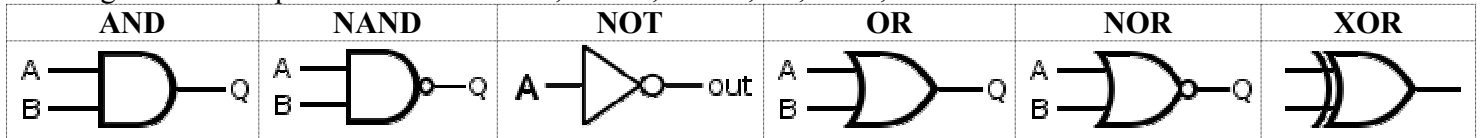
Term	DEF	DE	DF	CE	CD	CE	DE	DE	DE
$\overline{A}BC$	1	1	1	0	0	0	0	0	0
$\overline{A}BD$	1	1	0	0	0	0	0	0	0
$\overline{A}BE$	1	1	0	0	0	0	0	0	0
$\overline{A}BF$	1	1	0	0	0	0	0	0	0
$\overline{B}CD$	0	0	0	1	1	1	0	0	0
$\overline{B}CE$	0	0	0	1	1	0	0	0	0
$\overline{B}CF$	0	0	0	1	1	0	0	0	0
$\overline{C}DE$	0	0	0	0	0	1	1	1	0
$\overline{C}DF$	0	0	0	0	0	1	1	0	0
$\overline{D}EF$	0	0	0	0	0	0	0	1	1
$\overline{A}B\overline{C}$	0	0	0	0	0	0	0	0	0
$\overline{A}CDE$	0	0	0	0	0	0	0	0	0
$\overline{A}C\overline{D}$	0	0	0	0	0	0	0	0	0
$\overline{A}D\overline{E}$	0	0	0	0	0	0	0	0	0
$\overline{A}E\overline{F}$	0	0	0	0	0	0	0	0	0
$\overline{A}BDE$	0	0	0	0	0	0	0	0	0
$\overline{A}BCDF$	0	0	0	0	0	0	0	0	0
$\overline{B}C\overline{D}$	0	0	0	0	0	0	0	0	0
$\overline{B}D\overline{E}$	0	0	0	0	0	0	0	0	0

100% 23 Strs imported 670 Strs procesed 0.016 s

By repeating the same above procedure we will get the same final simplification and optimization.

$aB+bC+cD+dE+eF$  the same as (1) $Y=aB+bC+cD+dE+eF$

The digital circuit representation for AND, NAND, NOT, OR, NOR, XOR are



Using the Boolean calculator it is easy to establish these identities

using **set theory notation**  $\cup$  = OR  $\cap$  = AND  $\oplus$  = XOR

in the examples sometimes we use XOR abbreviation, sometimes  $\oplus$  symbol

$X = a \cup b \cup c \cup d$  is equivalent to boolean notation  $X = a + b + c + d$  equi  $X = a \text{ OR } b \text{ OR } c \text{ OR } d$

$Y = A \cup B \cup C \cup D$  is equivalent to boolean notation  $X = A + B + C + D$  equi  $X = A \text{ OR } B \text{ OR } C \text{ OR } D$

.....

$X = a \cap b \cap c \cap d$  is equivalent to  $X = abcd$  equivalent  $X = a \text{ AND } b \text{ AND } c \text{ AND } d$

$Y = A \cap B \cap C \cap D$  is equivalent to  $X = ABCD$  equivalent  $X = A \text{ AND } B \text{ AND } C \text{ AND } D$

### Example 1

Assume  $X = a + b + c + d$  and  $Y = A + B + C + D$

$X \text{ AND } Y = Ab + Ac + Ad + Bc + Bd + Cd + aB + aC + aD + bC + bD + Cd$  using basic optimization

$X \text{ AND } Y = Ab + Bc + Cd + aD + Ac$  A better optimization using KMAP form

$X \text{ NAND } Y = ABCD + abcd$

$X \text{ OR } Y = 1$

$X \text{ XOR } Y = ABCD + abcd$

$\text{NOT}(X \text{ XOR } Y) = Ab + Bc + Cd + aD + Ac$

$(Ab + Bc + Cd + aD + Ac) \text{ XOR } (a + b + c + d) = ABCD$

$(Ab + Bc + Cd + aD + Ac) \text{ XOR } (A + B + C + D) = abcd$

**Notice** in this example  $X \text{ XOR } Y = X \text{ NAND } Y$

### Example 2

Assume  $X = a + bc + d$  and  $Y = A + BC + D$

$X \text{ AND } Y = A bc + Ad + BCd + aBC + aD + bcD$  using basic optimization

$X \text{ AND } Y = A bc + Ad + BCd + aD$  A better optimization using KMAP form

$X \text{ NAND } Y = ABD + ACD + abd + acd$

$X \text{ OR } Y = 1$

$X \text{ XOR } Y = ABD + ACD + abd + acd$

**Notice** also in this example  $X \text{ XOR } Y = X \text{ NAND } Y$

### Example 3

Assume  $X = abc + bcd + acd + abd$  and  $Y = ABC + BCD + ACD + ABD$

$X \text{ AND } Y = 0$

$X \text{ OR } Y = ABC + ABD + ACD + BCD + abc + abd + acd + bcd$

$X \text{ NOR } Y = ABcd + AbCd + AbcD + aBCd + aBcD + abCD$

$X \text{ XOR } Y = ABC + ABD + ACD + BCD + abc + abd + acd + bcd$

**Notice** in this example  $X \text{ OR } Y = X \text{ XOR } Y \rightarrow X + Y = X \oplus Y$

### Example 4

Assume  $X = ab + ac + ad + bc + bd + cd$  and  $Y = AB + AC + AD + BC + BD + CD$

$X \text{ AND } Y = ABcd + AbCd + AbcD + aBCd + aBcD + abCD$

$X \text{ NAND } Y = ABC + ABD + ACD + BCD + abc + abd + acd + bcd$



X OR Y = 1    X NOR Y = 0  
 X XOR Y = ABC + ABD + ACD + BCD + abc + abd + acd + bcd

**Notice** in this example X AND Y = X XOR Y → XY = X ⊕ Y

**ExclusiveOR** have many interesting properties see [http://en.wikipedia.org/wiki/Exclusive\\_or](http://en.wikipedia.org/wiki/Exclusive_or)  
 XOR = ⊕ with Boolean Calculator you can easily investigate some like

- (A ⊕ B) ⊕ (C ⊕ D) = ABCd + ABcD + AbCD + Abcd + aBCD + aBcd + abCd + abcD
- (A ⊕ B) ⊕ A = B
- (A ⊕ B) ⊕ B = A
- A ⊕ b = NOT(A ⊕ B)
- (A ⊕ B) = (a ⊕ b)
- (A ⊕ B) ⊕ CD = ABCD + Abc + Abd + aBc + aBd + abCD
- (A ⊕ B) ⊕ (C + D) = ABC + ABD + Abcd + aBcd + abC + abD
- (A ⊕ B) ⊕ (A + B) = AB
- (A ⊕ B) ⊕ AB = A + B
- (ABCD + abcd) ⊕ (A + B + C + D) = a + b + c + d
- (ABCD + abcd) ⊕ (a + b + c + d) = A + B + C + D
- ABCD ⊕ abcd = ABCD + abcd

.....  
 In general if you define n Boolean single variables  $A_i$

$$P = \left( \prod_{i=1}^n A_i \right) Y \left( \prod_{i=1}^n \bar{A}_i \right)$$

Then

$$\left( \sum_{i=1}^n \bar{A}_i \right) I \left( \sum_{i=1}^n A_i \right) = \bar{P}$$

example

$$P = \bar{A}.\bar{B}.\bar{C}.\bar{D} + A.B.C.D \text{ then } (\bar{A} + \bar{B} + \bar{C} + \bar{D}).(A + B + C + D) = \bar{P}$$

.....

If you have any questions or comments and report a bug please contact [motahed1@yahoo.com](mailto:motahed1@yahoo.com)

**Kmap3 is a Freeware program - source code not available**

[Download program](#)

[Documentation\(DOC\)](#)

[Documentation\(PDF\)](#)