

REPORT
On
PROJECT DEVELOPMENT

Project Name: **K-MAP SOLVER**

Submitted By:

Muhammad Tasnim Mohiuddin

Student No. : 0805115

Mezbah Uddin

Student No. : 0805098

Submitted To:

Dr. Md. Mostofa Akbar

Professor, Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

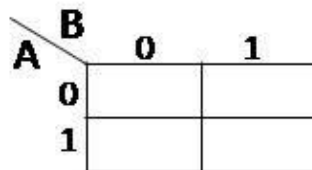
Introduction:

We have implemented “K-MAP SOLVER” by using java. It simplifies the Boolean function. In our program we have kept provision for simplifying upto eight variables Boolean function.

What is K-map:

The Karnaugh map (K-map for short) is a method to simplify Boolean algebra expressions. The Karnaugh map reduces the need for extensive calculations. In a Karnaugh map the Boolean variables are transferred (generally from a truth table) and ordered according to the principles of Gray code in which only one variable changes in between two adjacent numbers.

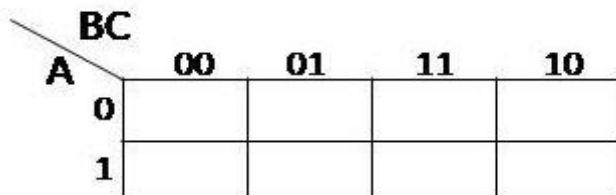
A Karnaugh map is a grid-like representation of a truth table. It is really just another way of presenting a truth table, but the mode of presentation gives more insight. A Karnaugh map has zero (0) ,one (1) and don't care(x) entries at different positions. Each position in a grid corresponds to a truth table entry.



A 2-variable Karnaugh map for variables A and B. The map is a 2x2 grid. The vertical axis is labeled A with values 0 and 1. The horizontal axis is labeled B with values 0 and 1. The grid cells are empty.

A \ B	0	1
0		
1		

Fig: 2 variable K-map



A 3-variable Karnaugh map for variables A, B, and C. The map is a 2x4 grid. The vertical axis is labeled A with values 0 and 1. The horizontal axis is labeled BC with values 00, 01, 11, and 10. The grid cells are empty.

A \ BC	00	01	11	10
0				
1				

Fig: 3 variable K-map

		CD			
		00	01	11	10
AB	00				
	01				
	11				
	10				

Fig: 4 variable K-map

Main features of our project:

➤ Number of variables:

It indicates the number of variables of our Boolean function. Base on it the next steps are designed. It indicates the number of variables is there in the Boolean function. There is provision for two to eight variables.

➤ Truth table:

After your selection of the number of variables, a truth table of your corresponding input will be shown. The first column shows the minterms in decimal form. The next some columns shows the minterms in binary form indicating the columns as A,B,C,D, The next column is named as “Value”. In this column you have to input the value of your Boolean function’s minterm. Here you can input only ‘0’, ‘1’ and ‘x’ (don’t care). In the combo box there is provision for selecting your desired value. You can also give input from your keyboard. For this, you have to double click on your desired row and press desired button from keyboard. The last column of the truth table is named as “Check”. At first it remains empty. After pressing the button “Simplify” it contains the value of minterms putting into

the resulting simplified expression. Its row shows 1 if that row's corresponding minterm is covered by the simplified expression. It shows 0 if that row's corresponding minterm is not covered by the simplified expression.

➤ **Simplify:**

By pressing the button simplify you will get the simplified expression of your Boolean function.

➤ **Simplified expression:**

In this text field you will see your desired simplified expression of the Boolean function. At first the simplified expression will be shown in Sum Of Product (SOP) form.

➤ **To POS:**

To see the simplified expression in Product Of Sum (POS) form you have to press "To POS" button. Thus you will see the answer in Product Of Sum (POS) form. The button then becomes "To SOP". If you want to get back the expression in Sum Of Product (SOP) form again then press the button "To SOP" and you will get the desired expression.

➤ **K-map:**

After pressing the button simplify the simplified expression and the K-map corresponding to the truth table will be shown.

There is a text field which indicates whether our simplified expression is right or wrong. If the simplified expression is right it shows the message "All the minterms are checked. It is found the simplified expression is correct! ". Otherwise it shows "All the minterms are checked. It is found the simplified expression is wrong!"

Screen shot of our project:

K Map Solver

K-MAP SOLVER

Number of variables

4

Select a row and specify a truth value

Rectangular Snip

Minterm	A	B	C	D	Value	Check
0	0	0	0	0	0	0
1	0	0	0	1	x	1
2	0	0	1	0	1	1
3	0	0	1	1	1	1
4	0	1	0	0	0	0
5	0	1	0	1	0	0
6	0	1	1	0	0	0
7	0	1	1	1	1	1
8	1	0	0	0	0	0
9	1	0	0	1	1	1
10	1	0	1	0	x	1
11	1	0	1	1	1	1
12	1	1	0	0	0	0
13	1	1	0	1	0	0
14	1	1	1	0	1	1
15	1	1	1	1	x	1

Simplify>>

AB

00

01

11

10

00

01

11

10

x

0

0

1

1

0

x

x

1

0

1

x

Simplified Expression

To POS

B'C + CD + B'D + AC

All the minterms are checked.It is found the simplified expression is correct!

Our program:

We have implemented our project “K-MAP SOLVER” by using java. We have used “The Quine Mccluskey Algorithm” to develop our project. It is a method used for minimization of Boolean functions. It is functionally identical to Karnaugh mapping. The main features of this algorithm are:

1. Generate Prime Implicants
2. Construct Prime Implicant Table
3. Reduce Prime Implicant Table
4. Solve Prime Implicant Table

➤ **Generate Prime Implicants:**

The Prime Implicants are generated from the minterms and don't cares of the Boolean function. This is done by combining two minterms or don't cares having one bit difference recursively until no more is possible.

➤ **Construct Prime Implicant Table:**

After generating Prime Implicants we have to construct Prime Implicant Table. It is a table in which Prime Implicants are placed along row head and minterms are placed along column head. In the table if a minterm is covered by a Prime Implicant that cell is ticked or marked.

➤ **Reduce Prime Implicant Table:**

It has three steps:

❖ **Remove Essential Prime Implicants:**

If a minterm is covered by only one Prime Implicant then that Prime Implicant is considered as “Essential Prime Implicant”. It is removed from the table and put in the answer.

❖ **Remove dominated minterm:**

If a minterm dominates some other minterm then that dominated minterm is removed from the Prime Implicant table.

❖ **Remove dominating Prime Implicants:**

If a Prime Implicant is dominated by some other Prime Implicant then that dominating Prime Implicant is removed from the Prime Implicant table.

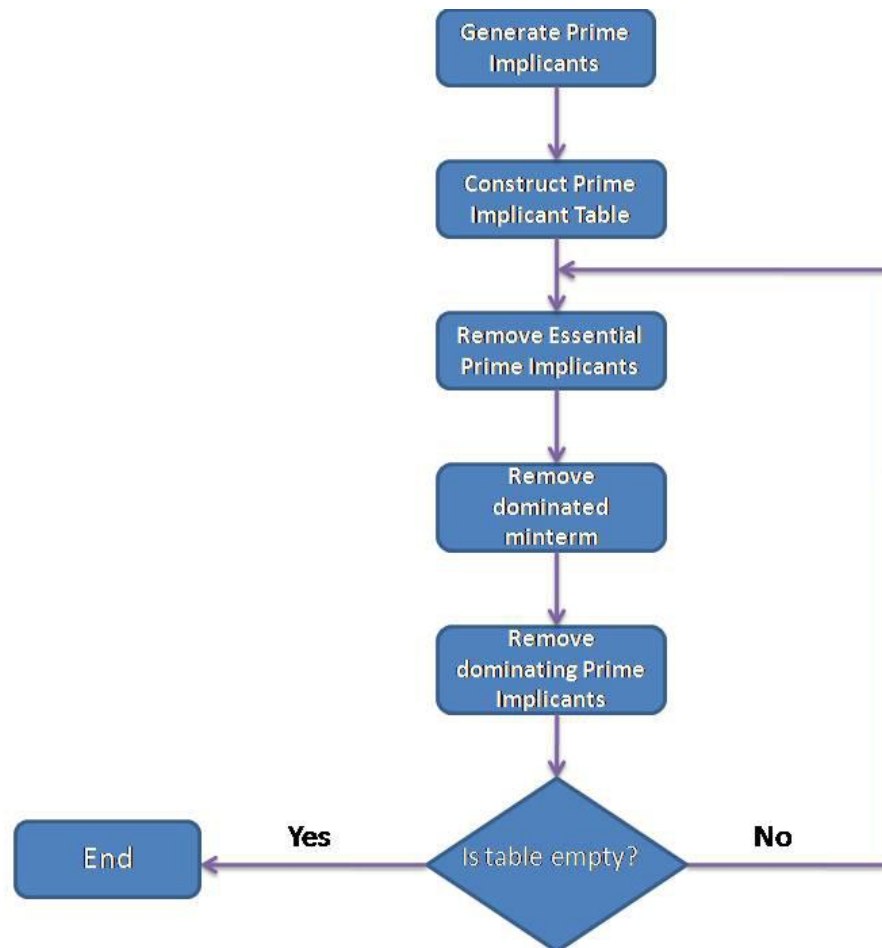


Fig: Flow-chart of “Quine Mccluskey Algorithm”

➤ **Solve Prime Implicant Table:**

Step 3 is done recursively until the Prime Implicants table becomes empty. Thus simplified expression is found.

In our program we have used ArrayList and array to implement this algorithm.

The main features of our program are given below:

We have created five java classes to make this K-map solver.

1. Term.java
2. TermFunction.java
3. Kmap.java
4. PITable.java
5. MainFrame.java

All these are put in “mykmapsolver” package. A brief description of our java classes are given below:

➤ **Term.java:**

In this class we have created three constructors and some methods. It contains all about of a term e.g. value, minterm expression, maxterm expression, Boolean expression of the term etc.

➤ **TermFunction.java:**

It contains four functions. They are append(), exists(), minimized(), removeDuplicate() function. The append function appends a term to an array of term. The exists function returns true if a term exists in a term array. The minimized function discards the checked terms from a term array. The removeDuplicate function removes the duplicate from a term array.

➤ **Kmap.java:**

It is for drawing kmap. There are two constructor and some related methods in this class.

➤ **PTable.java:**

It is the class where we have implemented “The Queen Mccluskey Algorithm”. There are three constructors and some methods for the implementation of the algorithm.

➤ **MainFrame.java:**

It is the main class. It creates the GUI and handles the total program.

In our program we have kept provision for checking whether the simplified expression is right or wrong. For this we have created “Check” column in the truth table. If the “Check’s” row’s corresponding minterm is covered by the simplified expression then its value becomes 1 otherwise it is 0. Then we check our input values and checked values. If they are identical (except don’t care) then we print the message “All the minterms are checked. It is found the simplified expression is correct! ”. Otherwise we print “All the minterms are checked. It is found the simplified expression is wrong! ”.

How to use:

After running this kmapsolver you have to follow the following steps:

- ☑ First you have to select the number of variables. It indicates how many variables are there in your Boolean function.
- ☑ After your selection of number of variables, a truth table of your corresponding input will be shown. Here you have to input the value of your Boolean function's minterm in the "Value" column. Here you will get three options – '0', '1' and 'x' (don't care). According to your function input your values for corresponding minterms in truth table.
- ☑ After putting value in truth table you have to click on "Simplify>>" button to get the result. A K-map corresponding to the truth table (where you have put value of the Boolean function) and simplified expression of your Boolean function will be shown. The simplified expression will be shown in Sum Of Product (SOP) form. If you want to get the simplified expression in Product Of Sum (POS) form then you have to click on "To POS" button.

Figurative demonstration:

Suppose we are going to simplify the following Boolean function:

$$F(A,B,C,D) = \sum (1,3,7,11,15)$$

$$d(A,B,C,D) = \sum (0,2,5)$$

Step 1:

As it is 4 variable Boolean function we have to select 4 from the variable selection portion of the K-map.



Step 2:

Then we have to input the function's value in the truth table's "Value" column.

Input your Boolean
function's Value here

Truth table

K-MAP SOLVER

Number of variables: 4

Select a row and specify a truth value

Minterm	A	B	C	D	Value	Check
0	0	0	0	0	0	
1	0	0	0	1	0	
2	0	0	1	0	0	
3	0	0	1	1	0	
4	0	1	0	0	0	
5	0	1	0	1	0	
6	0	1	1	0	0	
7	0	1	1	1	0	
8	1	0	0	0	0	
9	1	0	0	1	0	
10	1	0	1	0	0	
11	1	0	1	1	0	
12	1	1	0	0	0	
13	1	1	0	1	0	
14	1	1	1	0	0	
15	1	1	1	1	0	

Simplified Expression

To POS

Simplify>>

Step 3:

Now we have to press “Simplify>>” button to get the simplified expression.

Click on simplify button
to get simplified expression

The image shows the K-Map Solver application window. At the top, the title bar says 'K Map Solver'. Below it, the text 'K-MAP SOLVER' is displayed in large green letters. Underneath, there is a dropdown menu for 'Number of variables' set to '4'. Below that, the instruction 'Select a row and specify a truth value' is shown. A table with 16 rows (minterms 0 to 15) and 6 columns (A, B, C, D, Value, Check) is displayed. The table contains binary values for A, B, C, and D, and a 'Value' column with 'x' or '0'. The 'Check' column is empty. Below the table is a large empty box for the K-map. To the right of the table is another large empty box. At the bottom right, there is a 'Simplified Expression' label, a 'To POS' button, and two empty boxes for the simplified expression. At the bottom center, there is a 'Simplify>>' button. An arrow points from the text 'Click on simplify button to get simplified expression' to the 'Simplify>>' button.

Minterm	A	B	C	D	Value	Check
0	0	0	0	0	x	
1	0	0	0	1	1	
2	0	0	1	0	x	
3	0	0	1	1	1	
4	0	1	0	0	0	
5	0	1	0	1	x	
6	0	1	1	0	0	
7	0	1	1	1	1	
8	1	0	0	0	0	
9	1	0	0	1	0	
10	1	0	1	0	0	
11	1	0	1	1	1	
12	1	1	0	0	0	
13	1	1	0	1	0	
14	1	1	1	0	0	
15	1	1	1	1	1	

Thus we will get our simplified expression in Sum Of Product form. We will also see the K-map. To get the simplified expression in Product Of Sum form we have to press on the “To POS” button.

K-MAP SOLVER

Number of variables: 4

Select a row and specify a truth value

Minterm	A	B	C	D	Value	Check
0	0	0	0	0	x	0
1	0	0	0	1	1	1
2	0	0	1	0	x	0
3	0	0	1	1	1	1
4	0	1	0	0	0	0
5	0	1	0	1	x	1
6	0	1	1	0	0	0
7	0	1	1	1	1	1
8	1	0	0	0	0	0
9	1	0	0	1	0	0
10	1	0	1	0	0	0
11	1	0	1	1	1	1
12	1	1	0	0	0	0
13	1	1	0	1	0	0
14	1	1	1	0	0	0
15	1	1	1	1	1	1

K-map

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

To get result in Product Of Sum form press this button

To POS

Simplified Expression

$A'D + CD$

Simplified expression in Sum Of Product form

Message of correctness of Simplified expression

All the minterms are checked.It is found the simplified expression is correct

Simplify>>