Karnaugh Maps

For K-Maps I honestly recommend all of you to check
https://www.youtube.com/watch?v=RO5alU6PpSU

The video is well explained and very clear.
The idea of using K-Maps is to simplify our work in order to speed up the simplification process. To do this we are going to use the best rule ever! We ignore zeros! But now step by step!

An n-variable K-map has 2^n cells with each cell corresponding to an n-variable truth table value.

K-map cells are labeled with the corresponding truth-table row

K-map cells are arranged such that adjacent cells correspond to truth rows that differ in only one bit position (logical adjacency).

Now before we build a K-map I need to explain how you get the little numbers in each cell and what is Gray Code!

Consider the following
A product term in which all the variables appear exactly once, either complemented or uncomplemented, is called a **minterm**
A **minterm** represents exactly one combination of the binary variables in a truth table. It has the value of 1 for that combination and 0 for the others

In essence

X	Y	Z	Product Term	Symbol
0	0	0	$\overline{X}\,\overline{Y}\,\overline{Z}$	m_0
0	0	1	$\overline{X}\,\overline{Y}Z$	m_1
0	1	0	$\overline{X}Y\overline{Z}$	m_2
0	1	1	$\overline{X}YZ$	m_3
1	0	0	$X\overline{Y}\,\overline{Z}$	m_4
1	0	1	$X\overline{Y}Z$	m_5
1	1	0	$XY\overline{Z}$	m_6
1	1	1	XYZ	m_7

A Boolean function can be represented algebraically from a given truth table by forming the logical sum of all the **minterms** that produce a 1 in the function. This expression is called a **sum of minterms**

And with that we get

X	Y	Z	F	$\overline{F}$
0	0	0	1	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	0	1
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0

$$F = X'Y'Z' + X'YZ' + XY'Z + XYZ$$

$$= m_0 + m_2 + m_5 + m_7$$

$$F(X,Y,Z) = \Sigma m(0,2,5,7)$$

All of this is important because as you can see the expression is only concerned with where the 1's exist!

Now Gray Code!
Gray code is a set of rules to use when we move beyond a basic 2 input Logic system!

Gray code means that nearest neighbours in the K-map differ from one-another by only one bit. This means that the images of products of predicates (A and B and ...) on the K-map are connected subsets of the K-map as long as you define the K-map on the surface of a torus - so you can spot these factors easily.

So a basic two input looks like this

x	y	minterm
0	0	x'y'
0	1	x'y
1	0	xy'
1	1	xy

X \ Y	0	1
0	x'y'	x'y
1	xy'	xy

But when we look into a three input we have the following

X \ YZ	00	01	11	10
0	x'y'z'	x'y'z	x'yz	x'yz'
1	xy'z'	xy'z	xyz	xyz'

X \ YZ	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

Remember that the neighbours can only change by 1 bit! Hence why on the top you have

00 01 11 10

Now let's try and simplify from a truth table using **minterms**
Consider the following example

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$$f(x,y,z) = x'y'z + xy'z + xyz' + xyz$$
$$= m_1 + m_5 + m_6 + m_7$$

Remember we are only worried with 1's!!!!

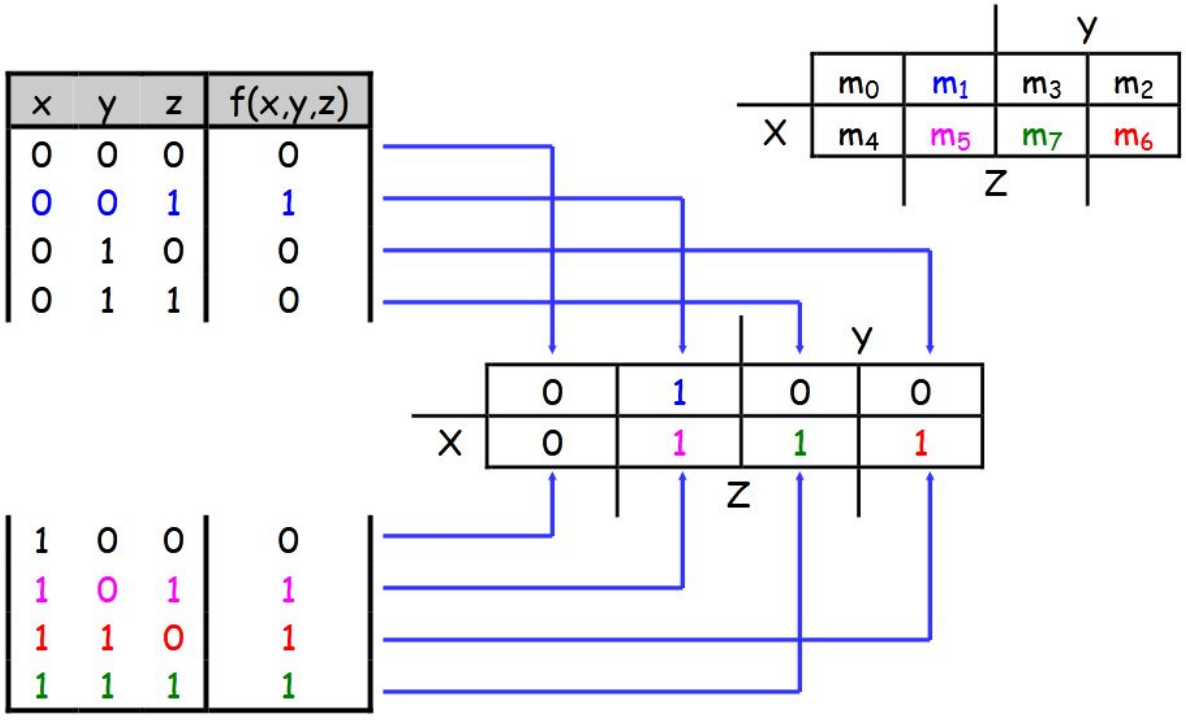
But we need to simplify

$$xyz' + xyz + x'y'z + xy'z$$
$$(xyz' + xyz) + (x'y'z + xy'z) + (xy'z + xyz)$$
$$(xy \bullet (z' + z)) + (y'z \bullet (x' + x)) + (xz \bullet (y' + y))$$
$$(xy \bullet 1) + (y'z \bullet 1) + (xz \bullet 1)$$
$$xy + y'z + xz$$

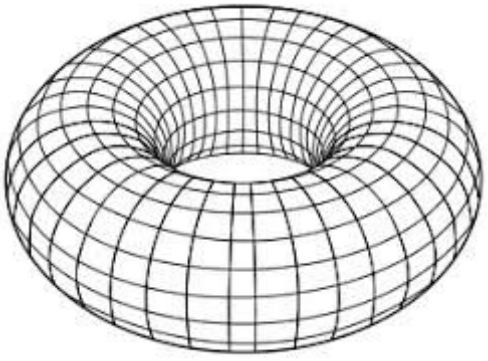
In order to simplify I had to add in constructs above! This is fair as it can allow for better simplification!

Let's see if I am right or can I even simplify more using a K-Map?

Consider the following

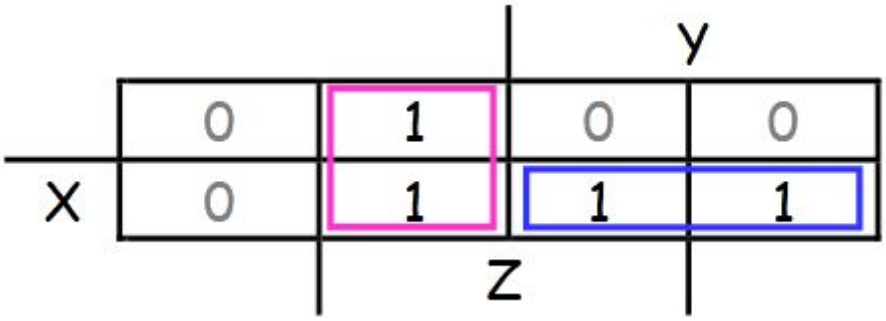


Now the rules for K-map manipulation!!!
Pairings have to be done in Powers of 2
Pairings **<u>can</u>** overlap!
Pairings can be done with the edges… remember a K-map functions as a Torus

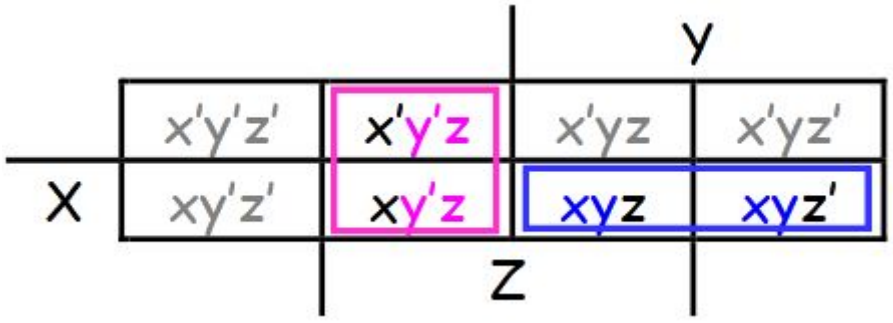


They won't be this complex!!!

So let’s try and play with our K-map!



Pairings are done



Now is the easiest bit.

After the pairings you need to look at what does not change? So for the pink section so the X part changes so we don't need that! And for the blue the Z changes so we can ignore that as well!

So we get

$$xy + y'z + xz = y'z + xy$$

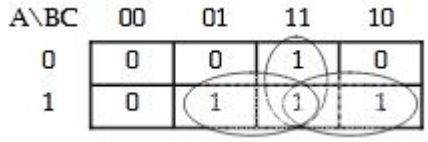
Another example

$$X = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

From here we obtain

A\BC	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Let’s maximize all possible groupings, respecting the power of 2 rule!!!



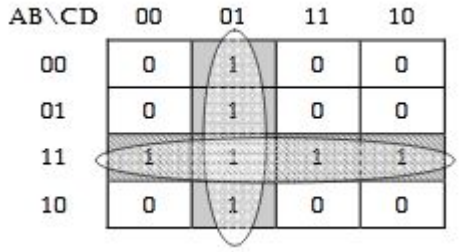
Look for what does not change in each group (the inputs)

$$X = AC + BC + AB$$

And that is plain and simple K-Maps!!!!

A few more examples of Groupings

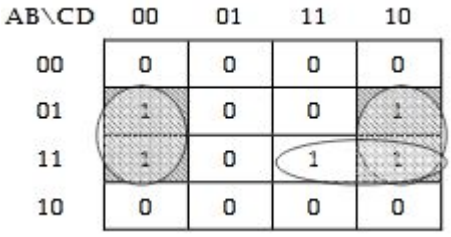
1.



Results in

$$A\,B + \overline{C}\,D$$

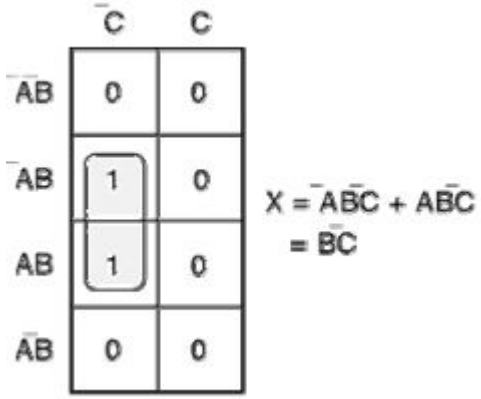
2.



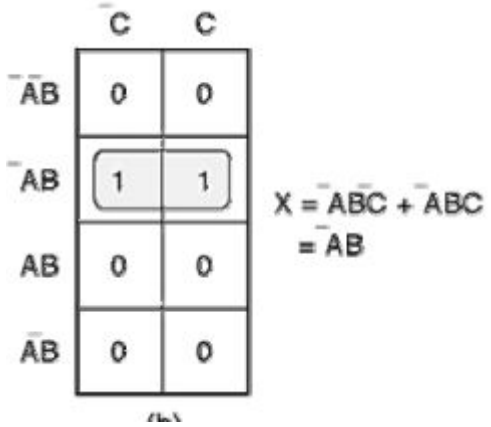
Results in

$$B\overline{D} + ABC$$

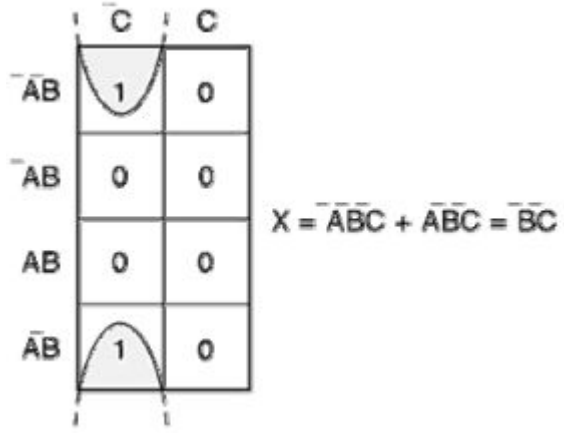
3.


C̄ C
ĀB̄ 0 0
ĀB 1 0
AB 1 0
AB̄ 0 0
X = ĀBC̄ + ABC̄
= BC̄

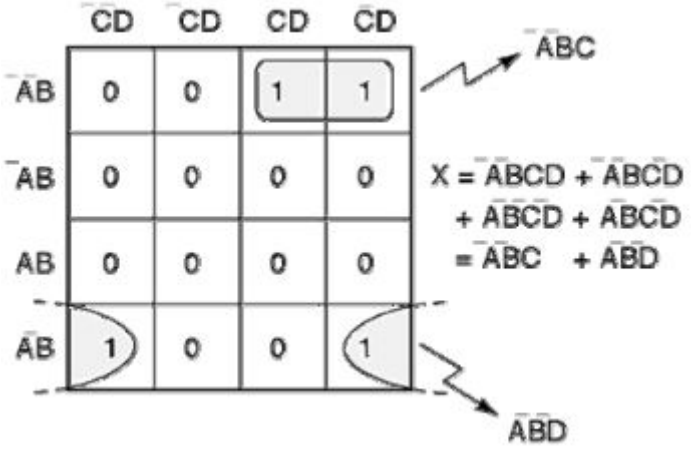
4.


C̄ C
ĀB̄ 0 0
ĀB 1 1
AB 0 0
AB̄ 0 0
X = ĀBC̄ + ĀBC
= ĀB

5.


C̄ C
ĀB̄ 1 0
ĀB 0 0
AB 0 0
AB̄ 1 0
X = ĀBC̄ + ĀBC = BC̄

6.


$\bar{C}\bar{D}$ $\bar{C}D$ CD $C\bar{D}$
$\bar{A}\bar{B}$ 0 0 1 1
$\bar{A}B$ 0 0 0 0
AB 0 0 0 0
$A\bar{B}$ 1 0 0 1
$\bar{A}BC$
$X = \bar{A}BCD + \bar{A}B\bar{C}D$
$+ \bar{A}\bar{B}CD + \bar{A}\bar{B}\bar{C}D$
$= \bar{A}BC + \bar{A}\bar{B}D$
$\bar{A}\bar{B}D$