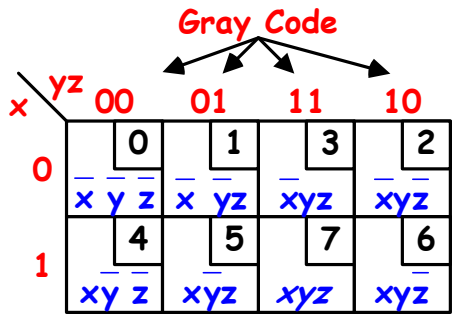
The 3-variable K-map (B)

The 3-variable **K-map** is created much the same way as the 2-variable **K-map** with one major difference. Note the logic values along the top of the map below representing **y** and **z**. **Note that they are in the following sequence: (0, 1, 3, 2).** This is a special sequence which you may remember from earlier when we discussed **Gray Code**.

x	y	z		
0	0	0	$\bar{x}\,\bar{y}\,\bar{z}$	m_0
0	0	1	$\bar{x}\,\bar{y}\,z$	m_1
0	1	0	$\bar{x}\,y\,\bar{z}$	m_2
0	1	1	$\bar{x}\,y\,z$	m_3
1	0	0	$x\,\bar{y}\,\bar{z}$	m_4
1	0	1	$x\,\bar{y}\,z$	m_5
1	1	0	$x\,y\,\bar{z}$	m_6
1	1	1	$x\,y\,z$	m_7



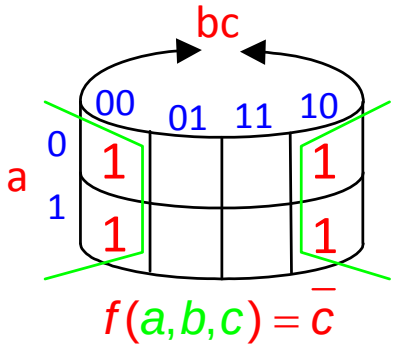
Note that **between any adjacent cell only one variable changes at a time**.

- Between **min-terms 3 and 2**, only the **z** changes.
- Between **min-terms 2 and 6** only the **x** changes.
- Between **min-terms 5 and 7 only the y changes**.

This is the feature which makes the **K-map** work. The same feature existed in the 2-variable map but it wasn't noticeable.

Before starting to map expressions into these new maps, we need to note something else about cell adjacency. In reality, the **K-map** is really a sphere (or can be visualized as a cylinder in some aspects) which has been flattened into a two-dimensional plane.

Note the figure to the right. This figure demonstrates that the **(10) column** (right edge) is in reality **adjacent to the (00) column** (left edge) even though they don't look like they are adjacent.



Note that **between the columns (00) and (10) only the 'b' variable changes** (which is the **K-map** requirement for adjacency.) Now, due to this, the four 1's in the map can be grouped into a single group of four where the variable **'a' falls out of the expression vertically** and the **variable 'b' falls out horizontally**. The only variable which is left is the **variable 'c' which is a '0'** in those two columns so the final expression is a $\bar{c}$.

It would also be possible to roll the map in the other direction (vertically) to demonstrate that the **top and bottom edges are adjacent** but for this map it isn't necessary since the inside cell edges are all that is needed. We will demonstrate the other adjacency when we look at the **4, 5** and **6** variable **k-maps**.

Now that we have discovered **the K-map** simplification of a Boolean expression, let's look at the same expression but this time we will solve it by **Boolean (switching) Algebra**.

$$
\begin{aligned}
f(a,b,c) &= \sum m(0,2,4,6) \\
&= \bar{a}\,\bar{b}\,\bar{c} + \bar{a}b\bar{c} + a\bar{b}\,\bar{c} + ab\bar{c} \\
&= \bar{a}\,\bar{c}\underbrace{\left(\bar{b}+b\right)}_{1} + a\bar{c}\underbrace{\left(\bar{b}+b\right)}_{1} \\
&= \bar{a}\,\bar{c} + a\bar{c} = \bar{c}\underbrace{\left(\bar{a}+a\right)}_{1} \\
&= \bar{c}
\end{aligned}
$$

***It* sure was easier with a K-map, wasn't it?!!**

More "Grouping" rules

Now that we have a larger map, there are a couple more rules that have to be followed in order to get the minimal expression. But 1st, let's review the rules which have already discussed:

- **Fill in the K-map with either the 1's or the 0's depending on if you want an SOP expression or a POS expression.**
- **Group the adjacent '1' entries into group sizes of "powers of two." (Grouping Rule #1)**
- **Drop any variable which changes within a group out of the expression.**
- **For the SOP, 'OR' the resulting terms together.**

To these rules the following are added (as was promised earlier):

Grouping Rule #2

- **You must have the smallest number of group's possible**

Grouping Rule #3

- **The groups must be the largest group's possible.**

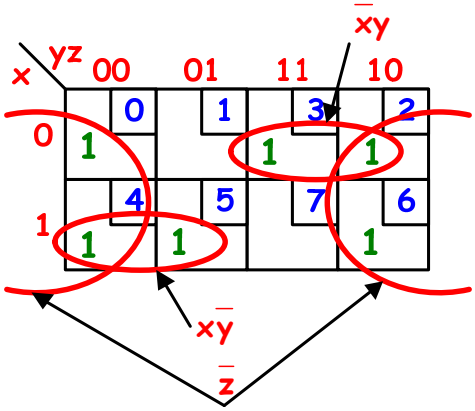
3-variable K-map examples

3-Variable Example 1 (SOP):

Problem Statement: **Minimize the following numerical canonical expression with a K-map:**

$$f(x,y,z) = \sum m(0,2,3,4,5,6)$$

The K-Map to the right demonstrates the groups that are obtained. Note carefully that you can share 1's. Also note the adjacency of the Right edge and the Left edge.



We can now get the following solution:

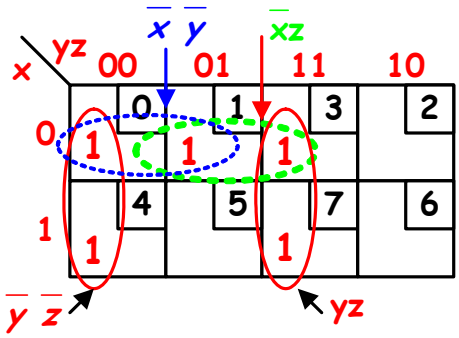
$$f(x,y,z) = \sum m(0,2,3,4,5,6) = \bar{x}y + x\bar{y} + \bar{z}$$

3-Variable K-map Example 2 (SOP):

Problem Statement: Minimize the following numerical canonical expression with a K-map:

$$f(x,y,z) = \sum m(0,1,3,4,7)$$

Note that in this example there are two possible minimum answers. The '1' in cell 1 can be grouped correctly with cell 0 or with cell 3.



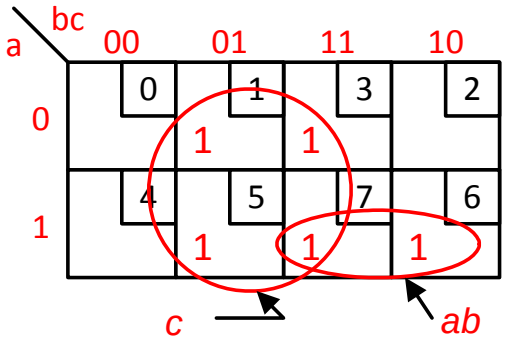
We can now get the following solution:

$$f(x,y,z) = \sum m(0,1,3,4,7) = \bar{y}\,\bar{z} + yz + \begin{cases} \bar{x}\,\bar{y} \\ \text{or} \\ \bar{x}z \end{cases}$$

Note the way the two solutions are included in the resulting expression. You will be required to show at least 2 solutions (if the additional solution exists) for any assigned homework problem. Showing 1 solution is fine for Quizzes, Tests, and Exams.

3-Variable K-map Example 3 (SOP)

Problem Statement: Find the minimal solution for $f(a,b,c) = \sum m(1,3,5,6,7)$



$$f(a,b,c) = ab + c$$

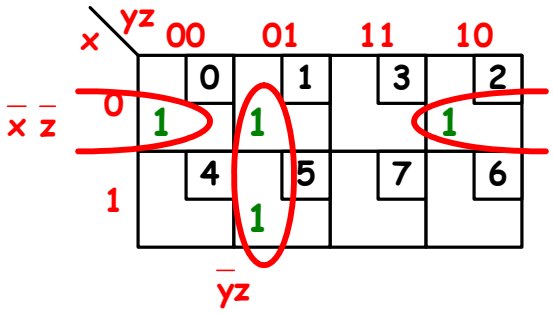
Absolute Minimum Results

Before going further, it is necessary to point out that the result of properly **K-mapping** and grouping an expression will be the **absolute minimum of that expression** (possibly one of several different minimums, but still the minimum). **It is not possible to cause this resulting minimum to become even more of a minimum with algebra!**

3-Variable K-map Example 4 (SOP):

Problem Statement: **Minimize** $f(x,y,z) = \sum m(0,1,2,5)$

This example demonstrates that **cell 2** and **cell 0** are indeed adjacent.



$$f(x,y,z) = \bar{x}\ \bar{z} + \bar{y}z$$

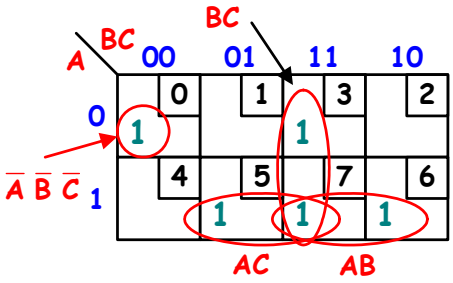
3-Variable K-map Example 5 (SOP):

Problem Statement: **Minimize the following:**

$$f(A,B,C) = ABC + AB\bar{C} + A\bar{B}C + \bar{A}BC + \bar{A}\ \bar{B}\ \bar{C}$$

First step is to note that the equation is already in an algebraic canonical form (fully expanded). So, all that is necessary is to quickly convert each term to its min-term equivalent, map the min-terms, and solve.

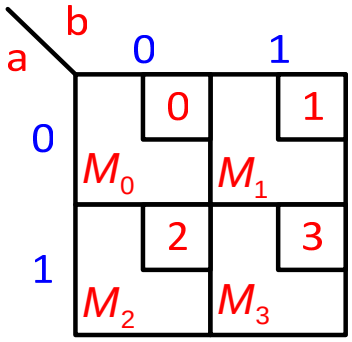
$$f(A,B,C) = \underbrace{ABC}_{7} + \underbrace{AB\bar{C}}_{6} + \underbrace{A\bar{B}C}_{5} + \underbrace{\bar{A}BC}_{3} + \underbrace{\bar{A}\ \bar{B}\ \bar{C}}_{0}$$



$$f(A,B,C) = AC + AB + BC + \bar{A}\ \bar{B}\ \bar{C}$$

The K-map and the POS solution

Remember the **max-term**? Well, we can **K-map max-terms** as well as we can **min-terms**.

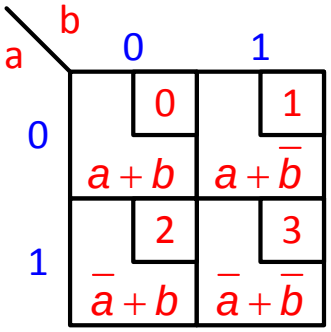


All that is required is that the **0's** from the table get mapped instead **1's**.

Of course, you have to remember that when the answer is read off of the map, a "**0**" is the **inverse** of a "**1**" so we have to invert each term. For instance, since the '**0**' is being mapped in, they are representing the '**0**' rows of the table vice the '**1**' rows. These '**0**' rows are the **MAX-TERM's** of the expression.

Remember that the **max-term** list results in a **POS** expression which is equivalent to the **min-term SOP** expression resulting from the same table. The **max-term** is represented by a capital **M** with a subscript representing the cell location or table row number.

When the **max-terms** are listed algebraically, note that they are read off as the inverse of what they would be if listed as **min-terms**. For instance, let's look at **cell 0**. If there were a '**1**' mapped in that cell, it would be represented by the algebraic expression $\bar{a}\ \bar{b}$, while if there were a '**0**' mapped in the same cell, it would be represented as $a + b$. It is no coincidence that the two expressions are related to each other by **DeMorgan's Theorem**.



$$f(a,b) = \bar{a}\ \bar{b}$$

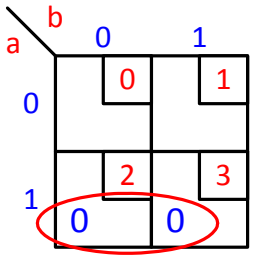
$$\overline{f(a,b)} = \overline{\bar{a}\ \bar{b}} = \bar{\bar{a}} + \bar{\bar{b}} = a + b$$

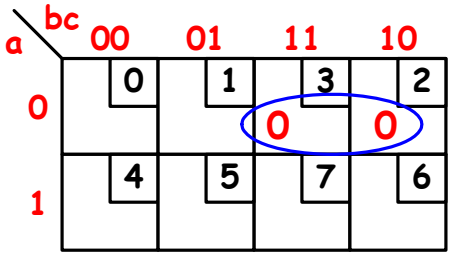
Other **max-term** algebraic expressions are shown in the **K-map** above.

Let us suppose that we were to place 0's in cells 2 and 3 of a two-variable map and group as shown to the right. The variable 'b' drops out of the expression because it changes within the **group of two** so the expression would become:

$$f(a,b) = \overline{a}\ .$$

(If it had been two 1's grouped together in the map the expression would have been $f(a,b) = a$).





Let's fill the same two cells in a **3-variable map**. If they were 1's, we would decode the term as $\overline{A}B$. But it isn't a group of 1's, it's a group of 0's. So the term is the inverse of what it would have been:

$$\overline{f(a,b,c)} = \overline{\overline{A}B} = \overline{\overline{A}} + \overline{B} = A + \overline{B}$$

3-Variable K-map Example 6 (POS):

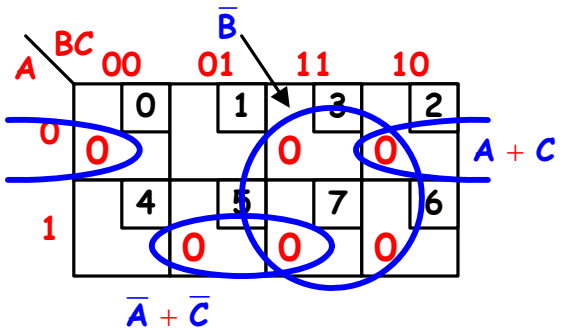
Problem Statement: **Find the minimal POS expression for the following function:**

$$f(A,B,C) = \Sigma m(1,4)$$

Since the minimum POS expression is desired, it is necessary to convert the given min-term list to a max-term list. In this case, if it is not a 1, then it must be a 0.

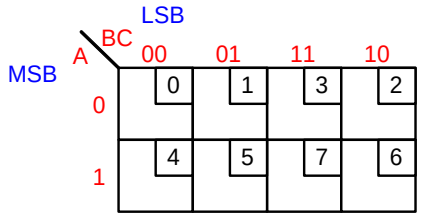
$$f(A,B,C) = \Sigma m(1,4) = \prod M(0,2,3,5,6,7)$$

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

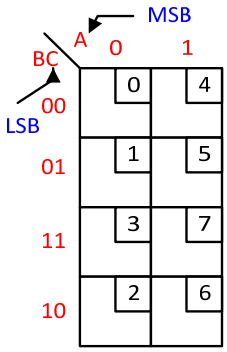


Another way to display the 3-variable K-map

These lecture notes would be incomplete if I did not point out the difference between my representation of the **K-map** and a lot of older textbooks (or newer ones with older authors).



My way



The books way

The difference is more than just that one is horizontal and the other is vertical. The difference lies in **where the MSB's and LSB's are**. **In my method, when you read a variable off of the graph, you read it left to right, MSB to LSB**, just the way that the English language is usually read. The other method requires the user to start at the right, read the **MSB** and then jump to the left, and read the last two variables from left to right. It just doesn't flow easily! More and more textbooks today are beginning to show this my way; however, our textbook still is using the old method.

No matter what you have been taught in past courses, **you are REQUIRED to show these my way**. There are 2 main reasons for this:

- My way reduces opportunity for error which is what all engineers should be searching for.
- I just don't want to shift back and forth between systems when grading. It is hard enough to meet grading deadlines as it is!

Using "Don't Care's" in K-maps

In many instances, the designer will come across situations where a certain **min-term** will never be allowed to occur. An example of this situation is **B**inary **C**oded **D**ecimal. Remember that in this code, the numbers **0 thru 9** are represented but **10 thru 15** are not. In this case, it's possible to allow the **min-terms** occupying rows **10 thru 15** in the table to take on **"either a 1 or a 0"** value **without adversely affecting the answer**. This not only doesn't affect the correctness of the answer but it most likely will **make the solution simpler**.

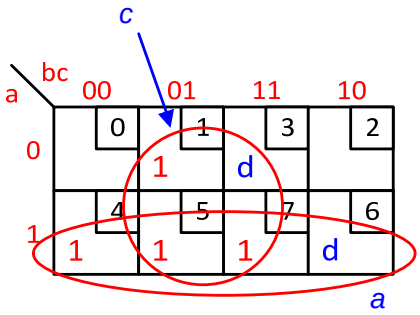
In K-maps, this **'Don't Care'** condition (also known as an Incompletely Specified Problem) is represented by either an **'x'** or a **'d'**. This **'Don't Care'** can be used to allow small groups to become larger groups, thus result in simpler answers. However, unlike the **min-terms** or **max-terms** that they will be helping to group, the "**Don't Care**" only is used when it is needed. **It doesn't get used unless it can make things simpler**.

The following example will demonstrate how the **'Don't Care'** can help in a **3-variable K-map**. It will be revisited later with larger **K-maps**.

3-Variable K-map Example 7 (SOP, Don't Care):

Problem Statement: Minimize: $f(a,b,c) = \sum m(1,4,5,7) + \sum d(3,6)$

Cells **3** and the **6** are **Don't Cares**. We will have a choice of using them as **0**'s or **1**'s depending on what gives us the minimum result. (We can use **d**'s or **X**'s to represent **Don't Cares** in a **K-Map**. The problem with **d**'s is that they might be confused for **1**'s.)



without : $f(a,b,c) = a\bar{b} + ac + \bar{b}c$

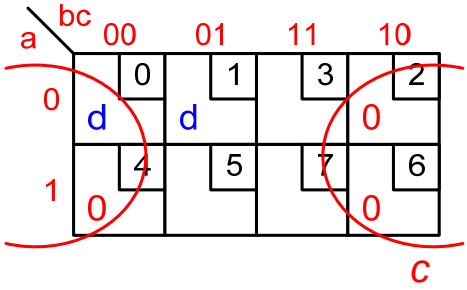
with : $f(a,b,c) = a + c$

Note that the groups will be **simpler and fewer** with the "**Don't Cares**" than without them.

3-Variable K-map Example 8 (POS, Don't Care):

Problem Statement: **Determine the minimal POS expression for:**

$$f(a,b,c) = \prod M(2,4,6) \bullet \prod d(0,1)$$



Note that the '**Don't Care**' in cell **1** didn't get used since it wouldn't help obtain a simpler answer.

without : $f(a,b,c) = (\bar{b} + c)(\bar{a} + c)$

with : $f(a,b,c) = c$

Plotting reduced algebraic expressions into K-maps.

Up until now, the popular way that these lesson notes have chosen to enter a **K-map** is to first have a **min-term** or a **max-term** list and then simplify the expression using the **K-map**. Unfortunately, **min and max term lists don't just occur naturally**. They usually start out with an algebraic form and then are turned into a **numerical canonical** form.

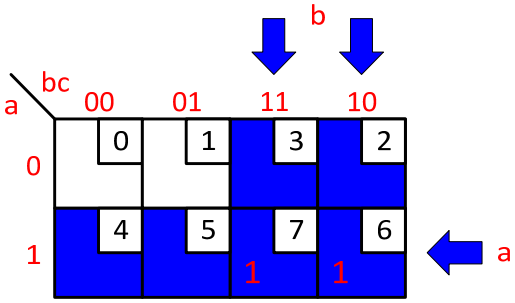
It should be noted that this isn't really necessary to do if we have six or fewer variable in the switching list. (Above six variables it becomes impossible for use mortals to use a **K-map** and **different methods which MUST be used which start with a min or max-term list**). All that is necessary to get an algebraic expression ready to be mapped into a **K-map** is to perform enough algebra on the expression to get it into a **Sum of Products form (if you want to map the 1's)** or a **Product of Sums form (if you intend to map the 0's).**

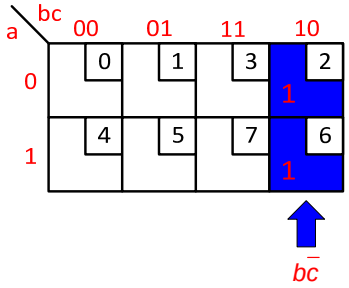
3-Variable K-map Mapping Example 1:

Problem Solution: **Map the following expression into a K-map (i.e., use a K-map to expand the following expression into a Numerical Canonical form)**

$$f(a,b,c) = ab + b\bar{c} + \bar{a}c + b$$

First, let's examine how the first term, ab, is mapped. The **'a'** part of the expression is represented by the bottom row while the **'b'** part is represented by the last two columns. **The intersection of these rows and columns are where the 1st two 1's are to be plotted.**



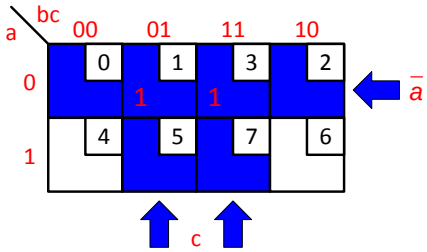


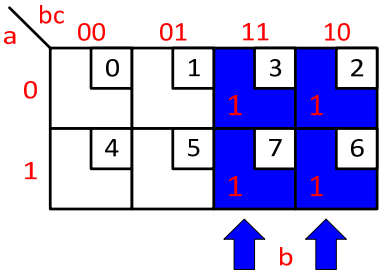
Next, we examine how the second term, $b\bar{c}$, is plotted:

This one isn't hard. It is represented by the **entire last column**.

Next, we examine how the third term, $\bar{a}c$, is plotted:

Again, you can see the intersection of the single top row and the two middle columns is where the two **1**'s are plotted.



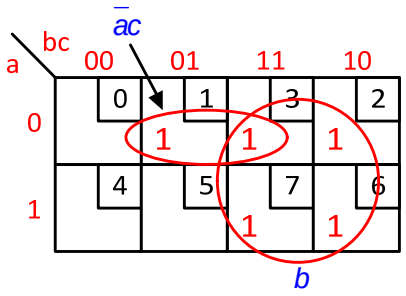


And, finally, let's map in the **'b'**:

The '**b**' is represented by the last two columns giving us four locations for the **1**'s.

Example continues on the next page)

We can now combine the results of all the variables together and we get the following **K-map**:



$$f(a,b,c) = \sum m(1,2,3,6,7)$$
$$= b + \bar{a}c$$

3-Variable K-map Conclusion

This chapter has introduced the **K-map** as a method to **simplify** and/or **expand** algebraic expressions of **two or three variables**. Following chapters will demonstrate how to do the same with larger numbers of variables. As usual, just as with computer programs, "**garbage in = garbage out!**" It is imperative that the terms get mapped into the **k-map** correctly or the result **will be wrong!**