<u>Don't Cares</u> Sometimes, for one or more lines of a truth table, the output might not matter. This might be because the input corresponding to this output cannot occur, or it might be because the output simply does not matter for specific inputs. In a truth table we depict such a condition, called a "*don't care*" condition, with an X.

Don't care conditions are nice because we can make the output (denoted as X) whatever value we want (0 or 1) depending on what results in an easier solution.

Note that if a truth table has several *don't cares*, we can make each don't care a 0 or 1 independently of the other *don't cares*.

If a function has don't cares in its truth table, we can treat those K-map cells as holding a 1 or a 0, depending on which values (1 or 0) allow us to construct the simplest Boolean expression.

When considering the implicants and prime implicants of a K-map, it is most beneficial to treat all of the don't care values (usually denoted as X) as holding the value:

- Larger groupings

- The Boolean expression corresponding to a larger

When considering the *essential* prime implicants of a K-map, it is most beneficial to treat all of the don't care values (denoted as X) as holding the value: ___ . Why?

<u>Example</u> Determine the minimal Boolean expression for a truth table that has the SOP expression:

$$f(A,B,C,D) = \sum m(1,7,10,11,13) + \sum d(5,8,15)$$

Recall that the *m* in the expression above stands for _____ , so the 11 in the expression $\sum m(1,7,10,11,13)$ means that:

The d in the expression above stands for _____, so the 5 in the expression $\sum d(5,8,15)$ means that:

What is the K-map for this canonical SOP expression?

CD \ AB	00	01	11	10
00				
01				
11				
10				

Circle the prime implicants:

CD \ AB	00	01	11	10
00				
01				
11				
10				

Now...which of these prime implicants are essential? Remember, when looking for 1's not covered by any other prime implicants, you do not consider the cells holding X's. In other words, a cell holding an X will never force a prime implicants to be essential.

Draw arrows pointing to the 1's that are covered by only one prime implicant:

CD \ AB	00	01	11	10
00				
01				
11				
10				

So, the essential prime implicants are:

CD \ AB	00	01	11	10
00				
01				
11				
10				

How would you sweep up all the additional 1's?

CD \ AB	00	01	11	10
00				
01				
11				
10				

Thus, the minimal SOP expression is:

Now...think about the benefit of using don't cares. What would have been the minimal SOP expression for the original K-map if we decided to treat all don't cares as 0's when forming the prime implicants?

CD \ AB	00	01	11	10
00				
01				
11				
10				

So, by considering the don't cares as 1's when forming prime implicants we have reduced the SOP expression from four terms with 14 literals down to three terms with 8 literals.

What would have happened if we had treated don't cares as 1's when examining essential prime implicants?

<u>Example</u> (from Marcovitz text) Determine the minimal SOP expression for the K-map shown below.

yz \ wx	00	01	11	10
00	**X**	**1**	**1**	
01	**X**		**1**	**1**
11	**X**	**1**		**1**
10	**X**			

Let's first circle the prime implicants.

yz \ wx	00	01	11	10
00	**X**	**1**	**1**	
01	**X**		**1**	**1**
11	**X**	**1**		**1**
10	**X**			

Draw arrows pointing to the 1's that are covered by only one prime implicants:

yz \ wx	00	01	11	10
00	**X**	**1**	**1**	
01	**X**		**1**	**1**
11	**X**	**1**		**1**
10	**X**			

So, the essential prime implicants are:

yz \ wx	00	01	11	10
00	**X**	**1**	**1**	
01	**X**		**1**	**1**
11	**X**	**1**		**1**
10	**X**			

How would you sweep up the remaining 1's?

yz \ wx	00	01	11	10
00	**X**	**1**	**1**	
01	**X**		**1**	**1**
11	**X**	**1**		**1**
10	**X**			

Thus, the minimal SOP expression is:

Note that there are two other equally good ways that we could have swept up the remaining 1's (after circling the essential prime implicants), so there are three different correct answers to this question.

<u>Product of Sums</u>

Sometimes we might want the minimum POS expression for a digital logic circuit. Recall how we found a POS expression for a function *f* using its truth table :

- We formed the SOP expression for f' (the complement of *f*) using the lines of the truth table that had an output of 0.
- We then complemented f' to obtain *f*. If we use DeMorgan's Law, the expression we obtain for *f* will be in POS form.

How do you think we can obtain a minimal POS expression using a K-map?

-
-
-

Example Find a minimal POS expression for a digital logic circuit whose K-map is shown below.

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

CD \ AB	00	01	11	10
00				
01				
11				
10				

We can readily see two essential prime implicants:

but we have two equally good choices for covering the remaining 1:

Let's flip a coin and go with _____.

Then our minimal SOP expression for the complemented K-map is

and therefore the minimal POS expression for the original K-map is:

Example (from Marcovitz text) Find a minimal POS solution for

$$g(w,x,y,z) = \sum m(1,3,4,6,11) + \sum d(0,8,10,12,13)$$

yz \ wx	00	01	11	10
00				
01				
11				
10				

yz \ wx	00	01	11	10
00				
01				
11				
10				

yz \ wx	00	01	11	10
00				
01				
11				
10				

Now, are there any 1's that are covered only by one prime implicant? (These will become essential prime implicants.)

yz \ wx	00	01	11	10
00				
01				
11				
10				

Let's circle the essential prime implicants:

yz \ wx	00	01	11	10
00				
01				
11				
10				

Now there is only one remaining 1 still to be covered, and it can be covered in two equally good ways. Let's go with the third column.

Thus the final SOP expression for g' is

and therefore the minimal POS expression for g is