Don’t Care Outputs

Some Input Combinations May Not Matter

Sometimes, we don’t care whether a particular input combination generates a 0 or a 1. For example,
- when an input combination is impossible to generate, or
- when outputs are ignored in the case of an input combination.

For Such Inputs, Use ‘x’ to Indicate “Don’t Care”

In such cases, we use ‘x’ (called a “don’t care”) in place of the desired output. Indicates that either 0 or 1 is acceptable. However: whatever we implement will generate a 0 or a 1, not a “don’t care.” So we need to be sure that we really do not care.

Why Are “Don’t’ Cares” Useful?

More choices often means a “better” answer (for any choice of metric). Say that you optimize a K-map for a function $F$. Then you consider several other functions $G$, $H$, and $J$.

If you have to pick one of the four functions ($F$, $G$, $H$, or $J$), the choice can’t get worse, since you can always pick $F$, but the best choice may be better than $F$. 
N “Don’t Cares” Allows $2^N$ Different Functions

Using x's for outputs means allowing more than one function to be chosen.
Each x can become a 0 or a 1.
So optimizing with N x's means choosing from among $2^N$ possible functions.

An Example with Two “Don’t Cares”

Let’s do an example.
The function $F$ appears to the right, partially specified.
Let’s say that we don’t care about the value of $F$ when $AB=01$.

<table>
<thead>
<tr>
<th>$AB$</th>
<th>$F$</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
</tr>
</tbody>
</table>

Let’s say that we don’t care about the value of $F$ when $AB=01$.*

*This notation means $A=0$ AND $B=1$. You can infer that $AB$ in this case does not mean $A$ AND $B$ because the product $AB$ has a single truth value (0 or 1).

Solution for $F$ with 0s: $AB + B'C$

One option is to fill the blanks with 0s.*
Then we can solve.

$F = AB + B'C$

But we could have chosen values other than 0, too.

*Without more information about $F$, filling with 0s is no better nor worse than any other choice.

Solution for $F$ with a 0 and a 1: $AB + C$

For example, we could put a 0 and a 1...
And then solve.

$F = AB + C$

This function is better than the first one (it has one fewer literal).
Solution for F with “Don’t Cares”: B + C

Rather than solving for all four possibilities, let’s write x’s into the K-map.

The x’s can be 0s or 1s, so to solve the K-map,

• we can grow loops to include x’s,
• but we do not need to cover x’s.

F = B + C  (the best possible answer)

Always Check that “Don’t Cares” Have No Ill Effects

When designing with x’s, it’s a good habit to verify that the 0s and 1s generated in place of x’s do not cause any adverse effects.

For our function, both x’s become 1s because they are inside a loop.
(We don’t have any more context for this example, so we are done.)