

If the Karnaugh map corresponds to a function with don't-care conditions, then the don't-care squares on the map can be left blank or assigned the value 1, whichever aids the minimization process.

Quine–McCluskey Procedure

Remember that the key to reducing the canonical sum-of-products form for a truth function lies in recognizing terms of the sum that differ in only one factor. In the Karnaugh map, we see where such terms occur. A second method of reduction, the *Quine–McCluskey procedure*, organizes information from the canonical sum-of-products form into a table to simplify the search for terms differing by only one factor.

The procedure is a two-step process paralleling the use of the Karnaugh map. First we find groupings of terms (just as we looped together marked squares in the Karnaugh map); then we eliminate redundant groupings and make choices for terms that can belong to several groups.

EXAMPLE 27 Let's illustrate the Quine–McCluskey procedure by using the truth function for Example 21. We did not write the actual truth function there, but the information is contained in the Karnaugh map. The truth function is shown in Table 7.13. The eight 4-tuples of 0s and 1s producing a function value of 1 are listed in Table 7.14, which is separated into four groupings according to the number of 1s. Note that terms of the canonical sum-of-products form differing by only one factor must be in adjacent groupings, which simplifies the search for such terms.

TABLE 7.13

x_1	x_2	x_3	x_4	$f(x_1, x_2, x_3, x_4)$
1	1	1	1	0
1	1	1	0	0
1	1	0	1	0
1	1	0	0	0
1	0	1	1	1
1	0	1	0	0
1	0	0	1	0
1	0	0	0	1
0	1	1	1	0
0	1	1	0	1
0	1	0	1	1
0	1	0	0	0
0	0	1	1	1
0	0	1	0	1
0	0	0	1	1
0	0	0	0	1

TABLE 7.14

Number of 1s	x_1	x_2	x_3	x_4
Three	1	0	1	1
Two	0	1	1	0
	0	1	0	1
	0	0	1	1
One	1	0	0	0
	0	0	1	0
	0	0	0	1
None	0	0	0	0

We compare the first term, 1011, with each of the three terms of the second group, 0110, 0101, and 0011, to locate terms differing by only one factor. Such a term is 0011. The combination 1011 and 0011 reduces to $\bar{0}11$ when the changing variable x_1 is eliminated. We shall write this reduced term with a dash in the x_1 position in the first row of a new table. The new

Table 7.15b, where we've just seen how the first row was obtained. We've rewritten the original Table 7.14 as Table 7.15a, but we have also marked the two terms 1011 and 0011 in this table with a superscript 1. This superscript 1 is a pointer that indicates the row number of the reduced term in Table 7.15b that is formed from these two terms (numbering terms corresponds to putting loops in the Karnaugh map).

TABLE 7.15

Number of 1s	x_1	x_2	x_3	x_4		Number of 1s	x_1	x_2	x_3	x_4
Three	1	0	1	1	¹	Two	—	0	1	1
Two	0	1	1	0	²	One	0	—	1	0
	0	1	0	1	³		0	—	0	1
	0	0	1	1	^{1,4,5}		0	0	1	—
One	1	0	0	0	⁶		0	0	—	1
	0	0	1	0	^{2,4,7}	None	—	0	0	0
	0	0	0	1	^{3,5,8}		0	0	—	0
None	0	0	0	0	^{6,7,8}		0	0	0	—

(a)

(b)

We continue this process with all the terms in Table 7.15a. A numbered term may still be used in other combinations, just as a marked square in a Karnaugh map can be in more than one loop. When we are done, the result is the completed Table 7.15b shown, where the terms in this table are again grouped by the number of 1s.

We now build still another table by processing the terms in Table 7.15b. Here not only the groupings but also the dashes help organize the search process, since terms differing by only one variable must have dashes in the same location. Tables 7.16a and 7.16b are the same as Tables 7.15a and 7.15b, and Table 7.16c is the new table. Again, numbers on terms in Table 7.16b that combine serve as pointers to the reduced terms in Table 7.16c. When we have processed all the terms in Table 7.16b, the reduction process cannot be continued. The unnumbered terms are irreducible, so they represent the possible maximum-sized loops on a Karnaugh map.

TABLE 7.16

Number of 1s	x_1	x_2	x_3	x_4		Number of 1s	x_1	x_2	x_3	x_4		Number of 1s	x_1	x_2	x_3	x_4
Three	1	0	1	1	¹	Two	—	0	1	1		None	0	0	—	—
Two	0	1	1	0	²	One	0	—	1	0						
	0	1	0	1	³		0	—	0	1						
	0	0	1	1	^{1,4,5}		0	0	1	—	¹					
One	1	0	0	0	⁶		0	0	—	1	¹					
	0	0	1	0	^{2,4,7}	None	—	0	0	0						
	0	0	0	1	^{3,5,8}		0	0	—	0	¹					
None	0	0	0	0	^{6,7,8}		0	0	0	—	¹					

(a)

(b)

(c)

For the second step of the process, we compare the original terms with the irreducible terms. We form a table with the original terms as column headers and the irreducible terms (the unnumbered terms in the reduction tables just constructed) as row labels. A check in the comparison table (Table 7.17) indicates that the original term in that column eventually led to the irreducible term in that row, which can be determined by following the pointers.

TABLE 7.17

	1011	0110	0101	0011	1000	0010	0001	0000
-011	✓			✓				
0-10		✓						
0-01			✓			✓		
-000					✓		✓	
00--				✓		✓	✓	✓

If a column in the comparison table has a check in only one row, the irreducible term for that row is the only one covering the original term, so it is an essential term and must appear in the final sum-of-products form. Thus, we see from Table 7.17 that the terms -011, 0-10, 0-01, and -000 are essential and must be in the final expression. We also note that all columns with a check in row 5 also have checks in another row and so are covered by an essential reduced term already in the expression. Thus, 00-- is redundant. As in Example 21, the minimal sum-of-products form is

$$\underline{x_2'x_3x_4} + x_1'x_3x_4' + x_1'x_3'x_4 + x_2'x_3'x_4'$$

In situations where there is more than one minimal sum-of-products form, the comparison table will have nonessential, nonredundant reduced terms. A selection must be made from these reduced terms to cover all columns not covered by essential terms.

EXAMPLE 28 We shall use the Quine-McCluskey procedure on the problem presented in Example 24. The reduction tables are given in Table 7.18, and the comparison table appears in Table 7.19.

TABLE 7.18

Number of 1s	x_1	x_2	x_3	x_4	
Three	0	1	1	1	1
Two	1	0	1	0	2,3
	0	1	1	0	1,4
One	0	0	1	0	2,4
	1	0	0	0	3

(a)

Number of 1s	x_1	x_2	x_3	x_4
Two	0	1	1	-
One	-	0	1	0
	1	0	-	0
	0	-	1	0

(b)

TABLE 7.19

	0111	1010	0110	0010	1000
011-	✓		✓		
-010		✓		✓	
10-0		✓			
0-10			✓	✓	✓

We see from the comparison table that 011- and 10-0 are essential reduced terms and that there are no redundant terms. The only original term not covered by essential terms is 0010, column 4, and the choice of the reduced term for row 2 or for row 4 will cover it. Thus, the minimal sum-of-products form, as before, is

$$x_1'x_2x_3 + x_1x_2'x_4 + x_2'x_3x_4$$

or

$$x_1'x_2x_3 + x_1x_2'x_4 + x_1'x_3x_4$$

PRACTICE 19 Use the Quine–McCluskey procedure to find a minimal sum-of-products form for the truth function in Table 7.20.

TABLE 7.20

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
1	1	1	1
1	1	0	1
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	0
0	0	1	1
0	0	0	1

The Quine–McCluskey procedure applies to truth functions with any number of input variables, but for a large number of variables, the procedure is extremely tedious to do by hand. However, it is exactly the kind of systematic, mechanical process that lends itself to a computerized solution. In contrast, Karnaugh maps make use of the human ability to quickly recognize visual patterns.

If the truth function f has few 0-values and a large number of 1-values, it may be simpler to implement the Quine–McCluskey procedure for the complement of the function, f' , which will have 1-values where f has 0-values, and vice versa. Once a minimal sum-of-products expression is obtained for f' , it can be complemented to obtain an expression for f , although the new expression will not be in sum-of-products form. (In fact, by De Morgan's laws, it will be equivalent to a product-of-sums form.) We can obtain the network for f from the sum-of-products network for f' by tacking an inverter on the end.

The whole object of minimizing a network is to simplify the internal configuration while preserving the external behavior. In Chapter 8 we shall attempt the same sort of minimization on finite-state machine structures.