Andrew Liles
EEL4705L
Prelab 4

Expression Reduction Techniques

Objective

Learn how to reduce Boolean expressions to their simplest form using algebraic manipulation and Karnaugh maps.

Algebraic Reduction

- Allows us to reduce expressions to simpler form by factoring out values of 1 (i.e., any variable OR'ed with its complement)
  - For Example:
    $F = ABC + \overline{A}BC + A\overline{B}C + A\overline{B}\overline{C}$
  - $F = (A + \overline{A})BC + A\overline{B}(C + \overline{C})$
    $F = BC + A\overline{B}$
- Basically, if there are any two terms that are otherwise the same apart from one variable that is present uncomplemented in one, but complemented in the other, they can be reduced into a simpler form as demonstrated above.
- Using the algebraic method may not always yield results, or will be tedious and difficult; the Karnaugh Map method produces results with less effort.

Karnaugh Maps

- Can be used to reduce expressions even if there are no apparent algebraic reductions
-

Two-Input K-Map

A \ B	**0**	**1**
0		
1		

Three-Input K-Map

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

Four-Input K-Map

AB \ CD	00	01	11	10
00				
01				
11				
10				

- These tables are formulated so that for each adjacent cell only one variable value changes.
- The column and row headings represent the values of the input variables. For instance, in the third column of the Four-Input K-Map, which is labeled with '11', indicates that variables C and D both have the value 1.
- To represent the function values at the various input values, we put 1's and 0's in their appropriate places on the map. If the input combination A=1, B=0, C=0, D=1 results in a function value of 1, we would place a 1 in the cell of the table in column 01 and row 10 (second column, fourth row). Repeat this process until all combinations of the input variables are entered into the table.

<u>Reduction using K-Maps</u>

- Fill the values into the K-Map as described above.

Truth Table

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

K-Map

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

- Then, circle values of 1 that are in adjacent squares. You must circle a number of squares that is equivalent to a power of 2: 1, 2, 4, 8, 16. The more squares you can circle indicates how much the expression can be simplified.

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

- The final SOP expression will consist of a product term for each of the circled groups. So, for this instance, we will have two product terms.
- We write product terms by observing which variables stay constant and which change within a circled group. In this example, the circled pair in the A=1 row have a change in the value of the C variable. This is evident by observing the column headings. When the column heading changes from 00 to 01, this means that the value of C has changed from 0 to 1, while B has stayed constant at 0.
- The variables that stay constant are the ones that are written in our product terms. So, for this example, we would end up with $F = A\overline{B} + BC$.
- Circling values of 1 can wrap around any side of the table, as long as an appropriate number of squares is circled. For example:

AB \ CD	00	01	11	10
00	0	1	0	0
01	1	0	1	1
11	1	0	0	1
10	0	1	0	0

- Thus, the reduced expression for this example is $F = \overline{B}\overline{C}D + B\overline{D} + \overline{A}BC$.
- In some cases, we will have “don’t care” conditions–either combinations of inputs that will not occur or that we are not concerned about the outcome of. We place these on our K-Map as an ‘x’, and proceed with the process as before, only when circling values of 1, we can consider an ‘x’ to be a 1 if it will result in a larger groups for us to circle. Consider the following example that produces a 1 when the value of the BCD input is larger than 5:

Truth Table:

A	**B**	**C**	**D**	**F**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

K-Map:

AB \ **CD**	**00**	**01**	**11**	**10**
00	0	0	0	0
01	0	0	1	1
11	X	X	X	X
10	1	1	X	X

- If we didn't consider the don't care values as being 1's, we would end up being able to circle two groups of two squares, resulting in a function $F = A\overline{B}\overline{C} + \overline{A}BC$. However, if we consider the don't care values to be 1's, we can circle a large group of 8, and one smaller group of 4:

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	X	X	X	X
10	1	1	X	X

- This would result in a function $F = A + BC$, which is decidedly simpler, and logically the same as the previous function.