ECE 2372 Modern Digital Design July 10

Reading assignment for today: Ch 2: section 2-4

On page 56 in your textbook is a table showing minterms for a system with 3 variables. The variables are XYZ in the table, and you will note the first minterm is ~X~Y~Z and is m_0 Likewise, the next combination ~X~YZ is minterm m_1 all the way thru to XYZ which is minterm m_7

Let's look at the same problem we looked at yesterday, a door that can be unlocked by A, or can be unlocked by B and C working together. The truth table is given below:

Binary code			Z= F(A,B,C)	
A	B	C		
0	0	0	0	m0
0	0	1	0	m1
0	1	0	0	m2
0	1	1	1	m3
1	0	0	1	m4
1	0	1	1	m5
1	1	0	1	m6
1	1	1	1	m7

We can write out this function in Sum of Products (SOP) form as shown below, where each minterm where the function is TRUE is included in the list of minterms:

Z = F(A,B,C) = ~ABC + A~B~C + A~BC + AB~C + ABC

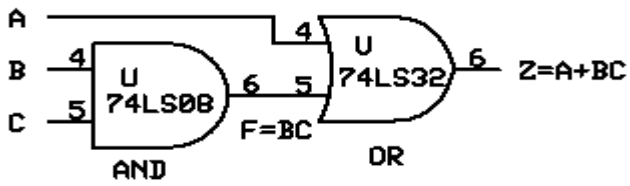
We could also write this as below, which provides the same information in more compact format:

Z = F(A,B,C) = Σm (3,4,5,6,7)

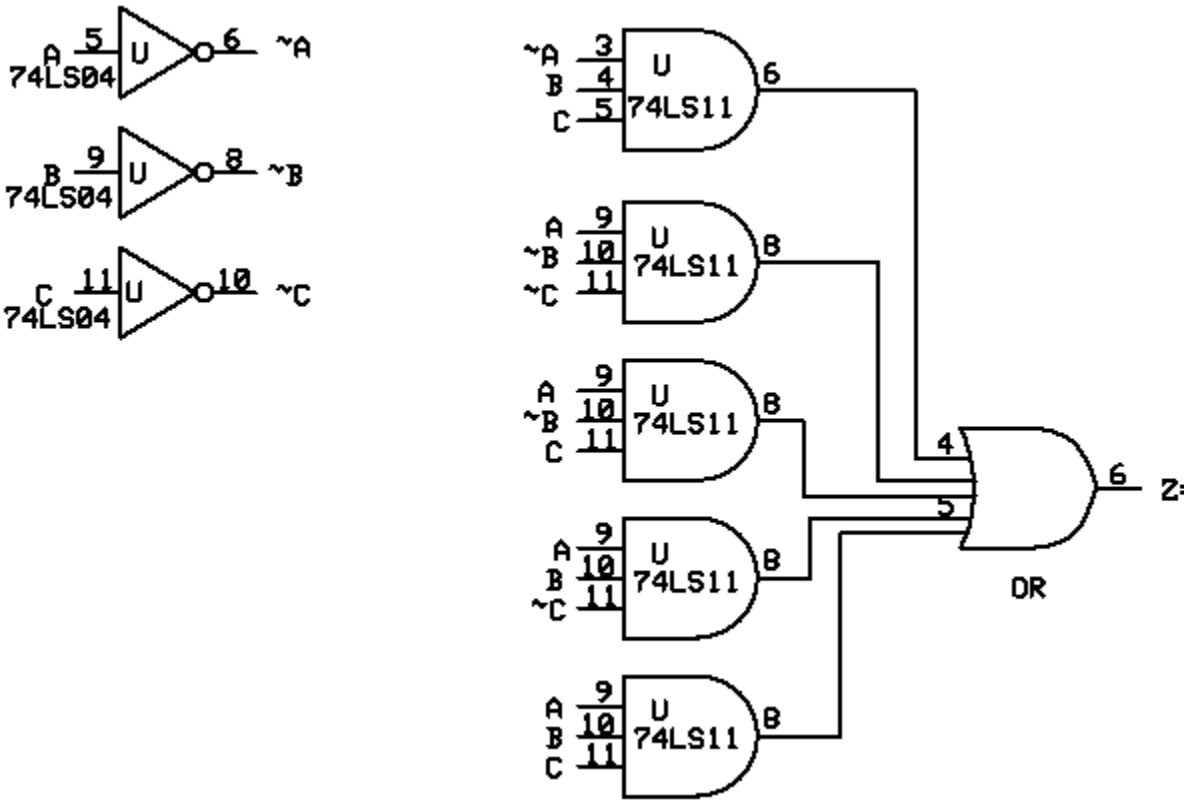
In yesterday's material, we applied some rules of Boolean Algebra and reduced the above equation to a simpler form. Our conclusion from yesterday is shown again below:

So, Z, a function of three variables A,B, and C is reduced to Z = F(A,B,C) = BC + A

We can implement this as a logic circuit shown here as we showed yesterday. This is in Sum of Products form:



If we had not reduced the problem using Boolean Algebra, we would have had this form for the solution:



In the diagram above, note that the function output is the sum of 5 minterms. Those are m3,m4,m5,m6,and m7

Z = F(A,B,C) = ~ABC + A~B~C + A~BC + AB~C + ABC

Or, the more compact format:

Z = F(A,B,C) = Σm (3,4,5,6,7)

However, recall that we simplified the design yesterday by applying some Boolean Algebra to reduce the overall equation to:
Z = F(A,B,C) = BC + A

That may have seemed somewhat tedious. With some practice from working several such problems, it becomes easier, but it is still somewhat tedious to reduce Boolean equations by that method. Someone, named Karnaugh, developed a graphical technique for simplifying Boolean Algebra equations and we refer to that method as Karnaugh Maps or often, just K-maps.

Below, from page 63 of your textbook is figure 2-12 which shows three different K-maps. Subfigures a and b show a 2-variable K-map, subfigures c and d show a 3-variable K-map and subfigure e and f show a 4-variable K-map. Let us consider the 2-variable K-map first.

Note that the 2-variable K-map is set up to have 4 "boxes" within it. Each of these 4 boxes represents one of the 4 minterms. Remember that with 2 variables (bits), we have 2^n combinations where n = the number of bits. So, with 2 variables, we have 4 possible combinations, or 4 minterms. Note we have input variables X and Y in this example. Looking at X first, X is related to the two rows in the table. The top row represents the case where the variable X is equal to ZERO. The bottom row is the case where the variable X is equal to ONE. Likewise, looking at variable Y, it designates the two columns. The leftmost column is where Y has the value ZERO, and the rightmost column is where Y has the value ONE. The 4 boxes, or cells within the table represent all 4 possible minterms for a function of two variables. Note, in subfigure a that the cells are labeled 0,1,2, and 3 representing the 4 minterms. The upper left corner cell is the case where X = 0 and Y = 0, and is minterm 0, representing the case ~X~Y. That is, both X and Y = ZERO is represented by the cell at X=0 and Y=0 intersection. Looking at subfigure b, note that same cell is labeled ~X~Y. The remaining 3 minterms are associated with their respective cells in the table or map. For example, minterm 3, representing the case of X = 1 and Y = 1 is the lower righthand corner cell, and in subfigure b, that cell is labeled XY.

Let's do a simple K-map for a 2-variable problem. Let's do a variation on the door problem and this time just two players. Player X can open the door with their key (X = 1) alone. Player Y can only open the door if player X is present with key as well. So the truth table for this problem is:

X	Y	Z=F(X,Y)	minterm
0	0	0	0
0	1	0	1
1	0	1	2
1	1	1	3

This can also be written as:

$Z = F(X,Y) = \Sigma m\ (2,3)$

Or

$Z = F(X,Y) = X\sim Y + XY$

Using Boolean Algebra, we could rewrite this as:

$Z = F(X,Y) = X\sim Y + XY$

$= X(\sim Y + Y)$

$= X(1)$

$= X$

~X~Y	~XY
0	0
X~Y	XY
1	1

Here we have mapped the function into the K-map. That is, we have put 1's in every minterm cell where the function is TRUE and 0's in the other minterm cells. We then simply group the 1's together in the largest groups of 2 or a power of 2 that we can. In this case, there is only one grouping possible, and that is the two cells in the lower row, where X = 1. Notice that it doesn't matter what the state of Y is in this case. Y = 1 is not sufficient for the function Z to equal 1, but regardless of the state of Y, if X = 1, then the function Z is true.

Thru graphical means, we have mapped the function Z onto a 2-variable K-map. By grouping the largest powers of 2 groups of 1's together, we find the minimal number of terms to satisfy the equation. We find a reduced form. In this case, it reduces to simply Z = F(X,Y) = X

Two variable problems are rather simple, so let's look at a 3 variable problem

X	YZ			
	00	01	11	10
0				
1				

Note we use a Gray Code here. Look at the columns. Notice the leftmost column is the case of YZ = 00, next column to the right we change only one bit to get to YZ = 01 (Y = 0, Z = 1), then the next column to the right is the case where XY = 11 (again, note only 1 bit position change from the previous column). Then the last column on the rightm, YZ = 10, again, note there is only one bit position change from the previous column. In fact, note that if you wrap the right most column back around to form a cylinder that the rightmost and leftmost columns are now adjacent.

It is also true that moving from one row to the next results in only one bit position change. Moving from any one cell to an adjacent cell only involves one bit position change.

Now let's map a function into this K-map

Binary code			Z= F(A,B,C)	
A	B	C		
0	0	0	0	m0
0	0	1	0	m1
0	1	0	1	m2
0	1	1	1	m3
1	0	0	1	m4
1	0	1	1	m5
1	1	0	1	m6
1	1	1	0	m7

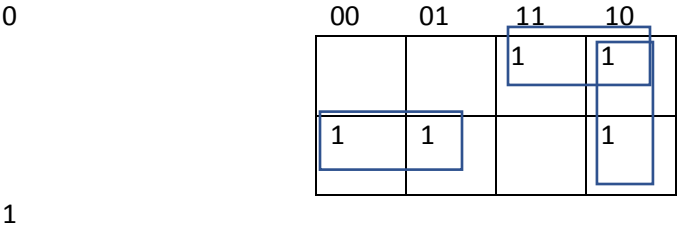
I have just made up an arbitrary function shown in the truth table above. We can also write it out in other forms.

Z = F(A,B,C) = ~AB~C + ~ABC + A~B~C + A~BC + AB~C

Or, the more compact format:

Z = F(A,B,C) = Σm (2,3,4,5,6)

A BC



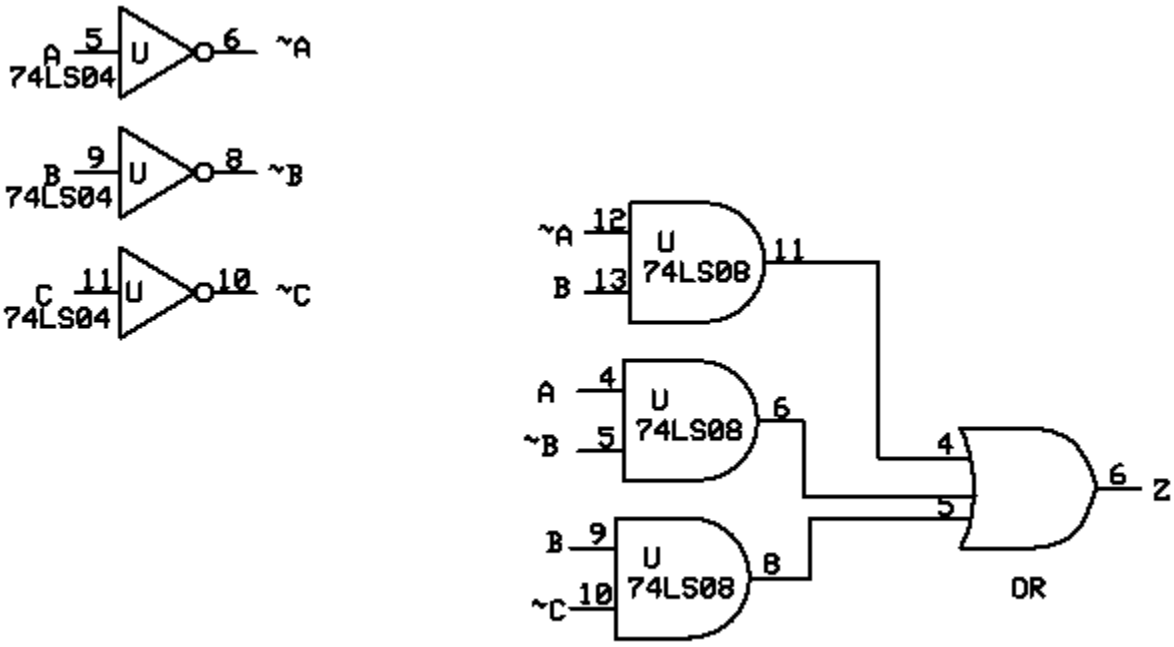
Note we get three groupings this time. We have some overlap. That is OK. It is acceptable to "cover" a given cell position in more than one way, but all cells with 1's in them have to be included in the final solution, even if it is just a single cell. If two cells are covered in a 3-variable map, then only two variables or terms are necessary to define that cell grouping. For example, let's look at the upper right hand corner grouping of the minterm cells 011 (~ABC) and 010 (~AB~C). This grouping is in the row where A = 0, so A has to be included. Notice that this same grouping overlaps the case of C = 0 and C = 1, so the variable C is not relevant here since the grouping covers both cases for C. However, note this grouping covers the case where B = 1 but not where B = 0. So, this grouping covers the term X = 0 and B = 1. We have two other groupings. Bottom left corner is similar, but covers the case where A = 1 and B

= 0, but for either case of C. Then the last grouping, the vertical rectangle that groups together the case where C = 0, B = 1, and A = either case.

So we wind up with 3 groupings of two each:

~AB + A~B + B~C

This is now in Sum of Products (SOP) form, and the SOP implementation with logic gates is given below:



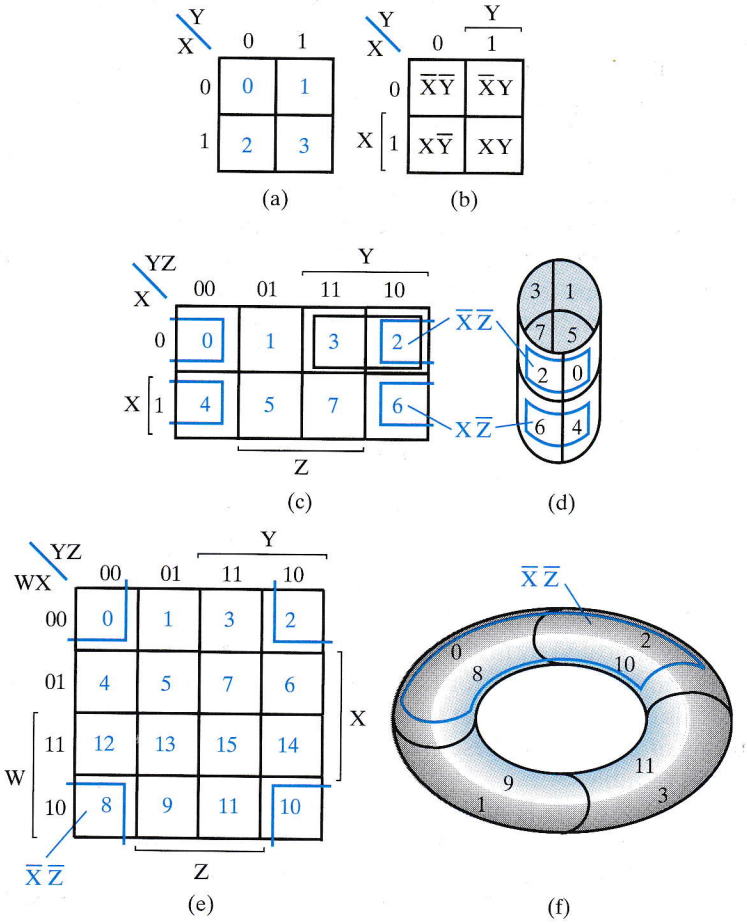
We will stop here for today. Next time we will look at more K-map examples, and also cover the 4-variable K-map. I will also post some additional Boolean Algebra examples, hopefully later today.

than sum-of-products and product-of-sums, the gate-input count must be determined directly from the implementation.

Regardless of the cost criteria used, we see later that the simplest expression is not necessarily unique. It is sometimes possible to find two or more expressions that satisfy the cost criterion applied. In that case, either solution is satisfactory from the cost standpoint.

Map Structures

We will consider maps for two, three, and four variables as shown in Figure 2-12. The number of squares in each map is equal to the number of minterms in the corresponding function. In our discussion of minterms, we defined a minterm m_i to go with the row of the truth table with i in binary as the variable values. This use of i to



□ **FIGURE 2-12**
Map Structures