Episode 6.04 – Four-Variable Karnaugh Map Example

Welcome to the Geek Author series on Computer Organization and Design Fundamentals. I'm David Tarnoff, and in this series, we are working our way through the topics of Computer Organization, Computer Architecture, Digital Design, and Embedded System Design. If you're interested in the inner workings of a computer, then you're in the right place. The only background you'll need for this series is an understanding of integer math, and if possible, a little experience with a programming language such as Java. And one more thing. Due to the computational nature of this episode, you might want to visit the transcript page found at intermation.com to download the episode worksheet.

Let's put this Karnaugh map experience into practice by designing the most simplified sum-of-products expression from a system description. Assume we are asked to design the logic circuit for a simple security system. It will have three alarm points: a glass break sensor we're going to call G, a motion detector we're going to call M, and a door open sensor we're going to call D. A fourth input, A, will act as a signal to indicate whether the alarm is armed or not. This might take the form of a key switch that controls a binary signal. A logic one from this key switch would indicate that the system is armed. A logic zero would mean that the system is disarmed allowing people to occupy the room and pass in and out of the door. Our output, X, is going to control an alarm siren such that when X is a logic one, the siren will sound, and when X is a logic zero, the siren will be silent.

Before we can create the truth table, we need to define the system's operation a little better. When do we want X to equal one? Well, when A equals one, the system is armed, and we want the siren to go off if any alarm point is active. This will be done by setting X to a logic one. The only time we don't want the alarm to sound when the system is armed, is when all three of the inputs equal zero indicating that they are not tripped.

If we order the input variable columns of our truth table A, G, M, and then D, the bottom eight rows of the sixteen truth table rows represent A equals one. The first of these eight rows is where A equals one, G equals zero, M equals zero, and D equals zero. This is the case when the system is armed, but no points have been tripped. This row will have a zero output for X. The remaining seven rows should all have one as the output since one or more alarm points have been tripped while the system is armed. Now what about the top eight rows of our truth table where A equals zero? If the system isn't armed, we wouldn't want the siren going off, right? Well, maybe. I don't know about you, but I would like the siren to sound if the glass is broken, even if the system is disarmed. More than likely, broken glass means something has gone horribly wrong. So, for the top eight rows of our truth table, we'd like to output a one if the glass is broken. If G is positioned as the second column of inputs on our truth table, then the top four rows represent A equals zero and G equals zero. This means the system is disarmed and the glass is intact, so regardless of what the motion detector, M, and the door sensor, D, are telling us, we don't want an alarm going off, and X should equal zero. The next four rows also have A equal to zero, but G equals one. That means glass has been broken and the alarm should be going off. This defines the values for the output X for this four variable truth table with inputs A, G, M, and D from top to bottom as 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1.

We could come up with the simplified sum-of-products solution by creating an implicant or product for each of the eleven truth table rows with ones, then going through the process of simplification to come up with the prime implicants. Or we could use the Karnaugh map.

Remember that the four-variable Karnaugh map is a four-by-four array of cells with the rows identified using the four possible binary patterns of the first two variables and ordered in the two-bit Gray code sequence: 0-0, 0-1, 1–1, and 1-0. We use the Gray code sequence so that adjacent cells from top to bottom only differ by a single variable. It also means that the cells in the bottom row of the Karnaugh map are considered adjacent to the top rows. The columns are identified using the four possible binary patterns of the last two variables. They are also ordered using Gray code. In the case of the four variables of our alarm system, A, G, M, and D, the cells of the Karnaugh map will be identified like this: the armed signal, A, equals zero for the top two rows and equals one for the bottom two rows; the glass break signal, G, equals zero for the top and bottom rows and equals one for the middle two rows; the motion detector signal, M, equals zero for the leftmost two columns and one for the rightmost two columns; and the door open signal, D, equals zero for the leftmost and rightmost columns and one for the two columns in the middle.

Now to populate the Karnaugh map with the output values from our truth table. The top row of the Karnaugh map gets the outputs from the top four rows of the truth table where both A and G equal zero. The Gray code ordering makes it so that they don't map in order, but since all four of these rows have zeros, the top row of the Karnaugh map is also populated with all zeros.
Now for the second row where A equals zero and G equals one. These values come from the fifth to the eighth rows of the truth table, 0-1-0-0, 0-1-0-1, 0-1-1-0, and 0-1-1-1. Once again, the Gray code makes it so that they are transferred to the Karnaugh map going from left to right from truth table rows 0-1-0-0, 0-1-0-1, 0-1-1-1, and then 0-1-1-0. Once again, though, all four rows have the same value, in this case one. Therefore, just copy the four ones directly to the second row of the Karnaugh map.
The third row of the Karnaugh map represents the rows of the truth table where both A and G equal one. These values come from the bottom four rows of the truth table. All four of these rows have ones, so these four ones get copied to the third row of the Karnaugh map from left to right from truth table rows 1-1-0-0, 1-1-0-1, 1-1-1-1, and then 1-1-1-0.

The bottom row of the Karnaugh map represents the rows of the truth table where A equals one and G equals zero. These come from the ninth, tenth, eleventh, and twelfth rows of the truth table, 1-0-0-0, 1-0-0-1, 1-0-1-0, and 1-0-1-1. The leftmost cell of the bottom row of the Karnaugh map represents A equals one, G equals zero, M equals zero, and D equals zero. This is the case where the system is armed, but none of the alarm points are tripped. A zero goes in this cell. The next cell, the one in the bottom row, second column of the Karnaugh map, represents A equals one, G equals zero, M equals zero, and D equals one. Here we have the system armed and the door has been opened. This should turn on a siren, so a one goes in this cell. The next cell, the one in the bottom row, third column of the Karnaugh map, represents A equals one, G equals zero, M equals one, and D equals one. Here we have the system armed and both motion has been detected and the door has been opened. This also should turn on the siren, so a one goes in this cell. Lastly, the cell in the bottom right corner of the Karnaugh map, where A equals one, G equals zero, M equals one, and D equals zero, represents when the system is armed and only motion is detected. We also place a one here.

Now that rows of the truth table have been transferred to the cells of the Karnaugh map, the next step is to group the cells containing ones into the rectangles that will represent the prime implicants. Remember our rules for making rectangles. First, each rectangle or square must be placed either horizontally or vertically. Second, all cells in a rectangle must contain ones. Third, the number of cells in the rectangle must equal a power of two, in other words, only groups of 1, 2, 4, 8, or 16 are allowed. Fourth, the outside edges of Karnaugh maps are considered adjacent, so rectangles may wrap from one

side of the map to the other, either left to right or top to bottom. Fifth, cells may overlap, but every rectangle must have at least one cell unique to it. Sixth, every rectangle must be as large as possible. Lastly, every one in the Karnaugh map must be covered by at least one rectangle, even if it requires adding a rectangle of size one to cover a lone cell.

Looking at the Karnaugh map for our security system, it appears that our ones have grouped themselves nicely. We begin by looking for the largest possible rectangle we can create in order to cover the most ones. We have a nice one of size eight right in the middle of the map. The second and third rows both contain nothing but ones, so they can be grouped together into a single rectangle. This leaves only the three ones in the fourth row to be covered by rectangles.

Since three is not a valid size for a rectangle, we can't simply place a rectangle across these three cells and call it an implicant. Two is a valid rectangle size, but note that if we create a rectangle that just covers two of the uncovered ones, it violates a different rule, the rule that tells us to make the rectangles as large as possible. So, let's make a rectangle that covers the middle two columns of the bottom two rows. This two-by-two rectangle is the correct size, and even though it overlaps the large rectangle, it's perfectly valid since it has two cells that are unique to it.

We still have one cell in our Karnaugh map that is left uncovered: the cell in the bottom right corner. Once again, a rectangle of size one is legal, but it could be made larger. If we double it by including the one in the cell to the left, we've made a larger rectangle, and hence, a simpler implicant, but it's still not as large as we could make it. Let's double its size again by including the two cells above it. Yes, this two-by-two rectangle overlaps the other two rectangles, but all three rectangles have at least one cell unique to each. These three rectangles cover all the ones in the Karnaugh map, which means that our final sum-of-products expression will have exactly three implicants or products.

Now let's make the products. In Episode 6.03 – Makin' Rectangles, we showed that you can derive the product represented by a rectangle by listing the values of each of the variables for all cells in the rectangle. The variables that change in value between one and zero for the different cells should be eliminated from the product. In the case of the large eight-cell rectangle going across all four columns of the Karnaugh map, the values for A, G, M, and D are 0-1-0-0, 0-1-0-1, 0-1-1-1, 0-1-1-0, 1-1-0-0, 1-1-0-1, 1-1-1-1, and 1-1-1-0. Going through this list, we see that A is zero for some cells and one for others. This means A drops out. This is true for M and D too. That means the only variable that has the same value for all eight cells is G, and that value is one. Therefore, our first implicant is simply G. If G is ever a one, we set off the alarm, which makes sense considering our system definition.

Another way of seeing this is that this rectangle crosses all four columns, and hence, all four possible combinations of ones and zeros for M and D. That means they don't contribute to the final product. The rectangle crosses two rows of the Karnaugh map. Since these two rows are the middle two rows, they follow the rows for G equals one and cross both values of A. That means A drops out, and our final product is just G. It's also important to note that since our rectangle covers eight cells which is two to the third power, three variables should drop out.

Now for the second rectangle. The second rectangle crosses both bottom rows. Both of these rows have A equal to one, so it will be part of our product. G equals one in the third row and zero in the bottom row, so it will drop out. The second rectangle also crosses the middle two columns. D equals one for these columns, so it will be part of our product. M equals zero in the second column and one in the third, so it drops out. That gives us our product A AND D. Neither of these variables needs to be inverted

since the value they represent in all of these cells is one. If either had been zero across all the cells, they would need an inverse overbar placed over them.

The last rectangle, the one that crosses the bottom two rows and rightmost two columns, will also have two variables drop out because it is of size two to the second power. Since it covers the bottom two rows like the second rectangle did, we know that one of those variables to drop out is G. The rightmost two columns are identified with M equals one, so it remains part of the product, while D equals one for the third column and zero for the fourth, so it drops out. This gives us the simplified product A AND M. This makes the final expression for our security system X equals G OR-ed with A AND D OR-ed with A AND M. This makes a lot of sense because we want to turn on the siren if the glass is broken or if the door opens while the system is armed or if motion is detected while the system is armed. The case where both the door opens and motion is detected while the system is armed is taken care of by both of the last two products.

In our next episode, we're going to look at what happens to our Karnaugh maps when an output is undefined. For episode transcripts, worksheets, links, or other podcast notes, please visit us at intermation.com where you will also find links to our Instagram, Twitter, Facebook, and Pinterest pages. Until the next episode remember that while the scope of what makes a computer is immense, it's all just ones and zeros.