Karnaugh Maps

Levi Seibert

Karnaugh Maps

CS309L

Assignment 0

Fall 2019

Abstract:

This research essay provides a brief overview of Karnaugh Maps, created by Maurice Karnaugh in 1953. It explains the general format of a K-Map and its use in circuit design. Details are provided on how to draw a Karnaugh Map. It then discusses and explains several ways of filling out a Karnaugh map, such as by using a truth table, a Boolean equation, or a truth table. It also explains how cells are grouped together and the rules that must be followed in order to simply the original equation.

A Karnaugh Map is a graphical method used in digital logic to minimize the size of circuits ("K-maps (Karnaugh Maps), n.d.). The Karnaugh Map (often referred to as a K-Map), groups expressions with common factors, thus eliminating unnecessary variables (University of Surrey, "Karnaugh Maps", n.d.). Karnaugh Maps, developed by Maurice Karnaugh in 1953, make use of humans' innate pattern-recognizing capabilities to make Boolean expressions easier to understand and simplify ("Karnaugh Map", 23 Sep. 2019). They are like truth tables, but instead of the various variables being listed in columns, and their results in rows, the K-Map arranges the variables along two axes in a grid-like form (Weisstein & Stuart, n.d.). A K-Map with two variables will have four squares or possible combinations of variables; one with three variables will have eight, one with four will have 16, and so on, such that for *n* = the number of variables, there are 2^n squares. Above each column and row that is formed, a corresponding variable or pair of variables is assigned in such a way that all variable possibilities can be produced. The variables may be split up in such a way that only one bit changes between adjacent squares. For example, a three-variable Karnaugh map may have 8 squares that are arranged in two rows of four columns. The first row may be marked with the variable A' (meaning the Boolean expression has a 0 in the A position) and the second row with A (meaning that the Boolean expression has a 1 in the A position). Above the first column would be the combination B'C', above the second would be B'C, the third BC, and finally the fourth BC'.

Once the Karnaugh Map has been drawn with proper squares and column/row headings, it can then be filled in to solve the Boolean function. The cells of the K-Map are ordered in Gray Code, which is a way of encoding numbers such that each adjacent number only has one digit that differs by one (Weisstein, n.d.). In binary, the gray code counts as follows: 0, 1, 11, 10, 110, 111, 101, 100, 1100, 1101, and 1111 (Mano & Ciletti, 2013, p. 24). Based on these facts, there are a couple of ways of filling out a K-Map. The most common way of solving a K-Map is to consider the actual Boolean function. First, you must convert the function into sum-of-products canonical form. Then you should mark each minterm with a 1 in its equivalent cell on the K-Map. For example, the function, F(A, B, C) = A'BC + ABC, has two minterms. The first, A'BC, is equivalent to the Karnaugh Map cell in the A' row and the BC column. Likewise, the second, ABC, is equivalent to the A row and BC column. You would mark a 1 in both of these cells.

Another way of filling out a K-Map is by using a Boolean sum, where each number listed in the sum is equivalent to the gray code number of the cells. This Boolean sum can be derived from a truth table by considering every minterm that produces an output and numbering them. Using this method, once you discover the minterms, mark a 1 in the K-Map cell with the equivalent number. For example, if F (A, B, C) = ∑(1, 4, 7), you would place a 1 in the three cells marked 1, 4, and 7 (of course, using the gray code numbering system).

Once each square (cell) on the K-Map has been marked for the function, begin to group the 1's, in order to reduce the function into its sum of product form, also called ANDed terms ORed together. There are 8 basic rules for group the cells together: don't include any zeros in any groups; do not group in diagonals; each group must have 2^nnumber of cells; the goal is to make as large groups as possible, each 1 marked on the K-Map must be included in a group; overlapping groups is permitted; you are allowed to wrap groups around the map (opposite sides); and try to produce the fewest number of groups possible (University of Surrey, "Karnaugh Maps - Rules of Simplification", n.d.).

Once all possible groups have been found, you can then write out the equivalent Boolean expression. This is done by observing the variables used in each group and writing down those that remain constant in each grouping. For example, if a group has the cells ABC, and ABC' marked, you will have a two cell group that yield yield the Boolean expression AB. Perform this procedure for each group on the K-Map and then OR all the Boolean expressions together. Once this process has been completed, you will have the simplified version of the original Boolean equation.

Works Cited

Intrator, N. (n.d.). K - maps (Karnaugh Maps). Retrieved October 4, 2019, from https://www.cs.tau.ac.il/~nin/Courses/mivne98/design/kmaps.html

Karnaugh map. (2019, September 23). Retrieved October 5, 2019, from https://en.wikipedia.org/wiki/Karnaugh_map#Karnaugh_map..

Mano, M. M., & Ciletti, M. D. (2019). *Digital design: with a introduction to the Verilog Hdl, Vhdl, and SystemVerilog*. Harlow, UK: Pearson.

University of Surrey. (n.d.). Karnaugh Maps. Retrieved October 4, 2019, from http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/karnaugh.html.

University of Surrey. (n.d.). Karnaugh Maps - Rules of Simplification. Retrieved October 15, 2019, from http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/karrules.html.

Weisstein, E., & Wilson, S. (n.d.). Karnaugh Map. Retrieved October 4, 2019, from http://mathworld.wolfram.com/KarnaughMap.html.

Weisstein, E. (n.d.). Gray Code. Retrieved October 15, 2019, from http://mathworld.wolfram.com/GrayCode.html.