"Digital Logic Worked Example: Interpreting a Truth Table"*

Kevin Beale
Adam Samaniego

This work is produced by The Connexions Project and licensed under the
Creative Commons Attribution License †

Abstract

Describes how to write a truth table in sum-of-products and product-of-sums canonical forms, minimize the logic using a karnaugh map, and implement the combinational logic in a digital circuit.

1 Interpreting a Truth Table

Introduction

A truth table is a table representing some logical operation, wherein all possible inputs and their corresponding outputs are shown. In this example we will work with the following made up truth table. We will show how to interpret this table in sum-of-products and product-of-sums canonical forms, how to minimize the logic using a Karnaugh map, and finally how to implement the truth table in a digital circuit with logic gates.

Truth Table

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Table 1

*Version 1.1: Dec 15, 2010 5:34 pm -0600
†http://creativecommons.org/licenses/by/3.0/

Sum-of-Products Canonical Form (SOP)
One way to describe a truth table is with the sum-of-products canonical form. In a truth table, the inputs in each row can be represented by a unique minterm - the one that is true for each row. A minterm is simply a product of all of the inputs in a particular state. For example the minterm associated with the third row of this particular truth table is A'BC' (where x' represents the logical complement of x). The Boolean equation describing any truth table can be written as a sum of the table's true minterms. This means of description is called the "Sum-of-Products Canonical Form."

For our above truth table the SOP is: **Y = A'BC' + A'BC + ABC'**

Product-of-Sums Canonical Form (POS)
Another way to represent the Boolean equation behind a truth table is through the product-of-sums form. This form relies upon expressing a combination of inputs by maxterms, which are logical sums of all the inputs for a particular state. The "Product-of-Sums Canonical Form" is obtained by multiplying together only the maxterms associated with false outputs.

For our truth table the POS is: **Y = (A+B+C)(A+B+C')(A'+B+C)(A'+B+C')(A'+B'+C')**

Minimization using a Karnaugh Map
We can use a Karnaugh map to graphically minimize the logic necessary to describe the truth table. To do so, we create a table with combinations of inputs (ordered using a Gray Code, i.e. an ordering in which only one input bit changes from column-to-column or row-to-row) arranged across the columns and rows of the table, where the cells represent the output for the given combination of inputs. We then circle all adjacent 1's that occur in contiguous blocks containing N 1's, where N is a power of 2. From this we can determine the associated minimal Boolean expression in sum-of-products form, by expressing the circled blocks as products of inputs or their complements that describe those regions. In our case, in the row where C = 0, there is a block of two 1's where we see B = 1. Thus, we can describe this region by the product BC'.

Karnaugh Map

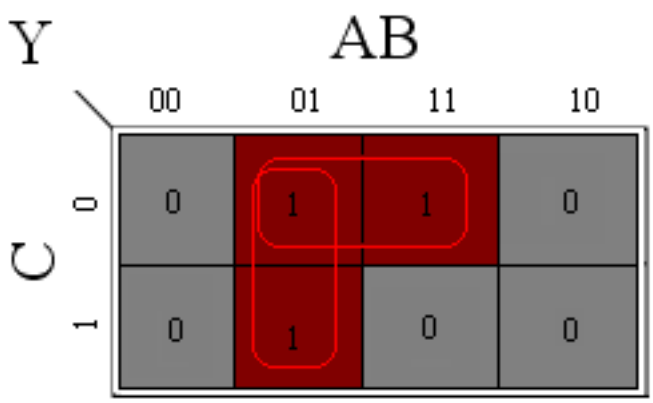

Figure 1

From the Karnaugh map the minimal equation can easily be found: **Y = BC' + A'B**

Digital Logic Circuit
Using the minimal equation found from the Karnaugh map, we can implement the table in digital logic using AND, OR, and NOT gates. Complements translate to inversion (bubbles), products correspond to AND gates, and the sum corresponds to an OR gate.

Digital Implementation

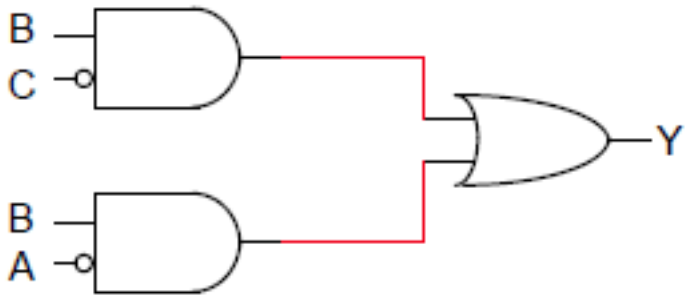

Figure 2: Circuit: Implemented!