HW 2 - Code Conversion

Jonathan Schwartz

Section 003L

TA: Conner Rickermann

Date submitted: 3/4/2021

**Theory**

For this lab, the goal was to convert from POSTNET code to XS3 code for their respective binary representations of the decimal numbers 0-9. To do this, the first step was to write out a 5-variable k-map for each term, (D, C, B, A, weighted 8, 4, 2, 1). The k-maps allow for a toroidal shape and with a 5-variable one, you can layer two of them (the bit above would be identical to the bit below so the shape is still 3-D). The goal of the k-map, which represents single-bit flips as adjacent squares, is to allow for an optimally small number of logic gates required for the same output as a less-efficient circuit would produce.

XS3 code is similar to standard base-two binary, however, it represents the values three more than the binary counterparts in an “excess 3” form. The POSTNET code is an encoding system with weights 7,4,2,1,0 for the V, W, X, Y, Z bits, and every decimal number 0-9 has 2 ones and 3 zeroes, whereas, in base-two binary, the number of each true/false can vary.

**Tables**

**POSTNET XS3**

| V | W | X | Y | Z |  | D | C | B | A |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 0 | 0 |  | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 1 |  | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |  | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |  | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |  | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 |  | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 |  | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |  | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 |  | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 |  | 1 | 1 | 0 | 0 |

**K-maps**

D

| V |  |  |  |  |  | V' |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| WX (down) \ YZ (right) | 0,0 | 0,1 | 1,1 | 1,0 |  | WX (down) \ YZ (right) | 0,0 | 0,1 | 1,1 | 1,0 |
| 0,0 | X | 1 | X | 1 |  | 0,0 | X | X | 0 | X |
| 0,1 | 1 | X | X | X |  | 0,1 | X | 0 | X | 0 |
| 1,1 | X | X | X | X |  | 1,1 | 1 | X | X | X |
| 1,0 | 0 | X | X | X |  | 1,0 | X | 0 | X | 1 |

C

| V | 0,0 | 0,1 | 1,1 | 1,0 |  | V' | 0,0 | 0,1 | 1,1 | 1,0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0,0 | X | 0 | X | 0 |  | 0,0 | X | X | 1 | X |
| 0,1 | 1 | X | X | X |  | 0,1 | X | 1 | X | 1 |
| 1,1 | X | X | X | X |  | 1,1 | 0 | X | X | X |
| 1,0 | 0 | X | X | X |  | 1,0 | X | 1 | X | 0 |

B

| V | 0,0 | 0,1 | 1,1 | 1,0 |  | V' | 0,0 | 0,1 | 1,1 | 1,0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0,0 | X | 1 | X | 1 |  | 0,0 | X | X | 0 | X |
| 0,1 | 0 | X | X | X |  | 0,1 | X | 0 | X | 1 |
| 1,1 | X | X | X | X |  | 1,1 | 0 | X | X | X |
| 1,0 | 1 | X | X | X |  | 1,0 | X | 1 | X | 0 |

A

| V | 0,0 | 0,1 | 1,1 | 1,0 |  | V' | 0,0 | 0,1 | 1,1 | 1,0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0,0 | X | 0 | X | 1 |  | 0,0 | X | X | 0 | X |
| 0,1 | 0 | X | X | X |  | 0,1 | X | 1 | X | 0 |
| 1,1 | X | X | X | X |  | 1,1 | 1 | X | X | X |
| 1,0 | 1 | X | X | X |  | 1,0 | X | 1 | X | 0 |

**Equations**

Based on these K-maps, with the sections utilized color-coded (each individual color is a gate or the overlapping of two gates), we can derive equations that function and produce the desired outcome with minimal logic gates: D = VW' + V'WZ', C = V’Z + W’X, B = X’Y’ + W’YZ’, and A = VW + VY + V'Y'.

**Home Area Code**

We write each number in the 5-digit area-code as a 5-digit POSTNET binary representation, so, for my home area code of 06117, we write 0 as 11000, 6 as 01100, each 1 as 00011, and 7 as 10001, so when all these are put together, it is written as “1100001100000110001110001.”

**Questions**

How many non-valid input codes are there?

Any POSTNET input that gives a single 1, or more than 2 ones, will be invalid, so this equation can be written with combinatorics as follows:



This equation results in the number of invalid outputs being 22.

Would your design be simpler if there was no zero input? Does your design conflict with your answer?

The design would be slightly simpler because the “A” term’s K-map would be reduced to fewer gates, and the “D” term’s K-map would be reduced as well because the entire V 4x4 K-map could be included as a term. The other K-maps would be difficult to reduce further, but the K-maps could reduce. The number of chips on the board may stay the same, but the number of logic gates would reduce, so the circuit would simplify.