Lab 1: Boolean Logic and Combinational Circuits

Review Questions (optional, not to be handed in)

- Convert the function \( X = A \land ((B \land C) \lor (B' \land C')) \) into sum-of-products form, into product-of-sums form, into a truth table, and into three circuits based on the original equation and the sum-of-products and product-of-sums forms you produced.

- Design a 4-bit equality comparator.

Part I: Written Homework

1. Convert the function \( X = A' \lor A \land ((A' \land B) \lor (B' \land C))' \) into the following (remember that \( \land \) has higher precedence than \( \lor \), so you can’t just replace the initial \( A' \lor A \) with true):
   a) sum-of-products form
   b) product-of-sums form
   c) a truth table (note: you can use this to check your answers to Parts a and b above)
   d) a two-level NAND or NOR gate circuit implementation for Parts a and for Part b

2. Show that if \( a \land b = a \land c \) and \( a \lor b = a \lor c \) then \( b = c \). You may prove it algebraically or by truth-tables, but explain your work clearly. Hint: take cases on \( a \).

3. Show that the binary operator \( \Theta \) (defined below) is universal (e.g., by constructing a known universal gate).
   \[ F \Theta F = F \quad F \Theta T = T \quad T \Theta F = F \quad T \Theta T = F \]

Part II: Circuit Design

(Note that this is based on Exercise 3.23 from Harris and Harris, but requires only low-level combinational design, not sequential design).

Design a combinational circuit for a vending machine with a “simultaneous” coin input system: coins are loaded into a tray which is then pushed in. Some washing machines and dryers use this system, but those require a specific set of coins — the machine you are designing should accept any combination of coins and produce the right change.

To keep things simple, we’ll assume that everything from the machine is the same price: $0.30. The coin tray has one slot for a quarter and three slots for dimes (for extra credit, allow up to six slots for nickels). Each slot controls one of the four inputs to your circuit \((q, d_1, d_2, d_3)\), so for example if there is a quarter in the quarter slot and one dime in the middle slot, then \( q = T, d_1 = F, d_2 = T, \) and \( d_3 = F \).
The goal of your circuit is to determine whether or not enough money has been entered and control the number of nickels (up to five) that should be returned as change. For the sample input above \( q = T, d_1 = F, d_2 = T, \) and \( d_3 = F \), enough money has been entered and one nickel should be returned. In contrast, for \( q = F, d_1 = F, d_2 = T, \) and \( d_3 = T \), there is not enough money, and four nickels should be returned.

Your circuit should have a one-bit boolean output \( V \) (for “vend”) that will be true iff enough money was entered, and three wires \( n_2 \ n_1 \ n_0 \) that together constitute a positive binary number giving the number of nickels (use a “wire merge” to join these into one three-bit line on your circuit and connect it to the numeric output display).

Design steps:

1. Make yourself a table of the 16 four-bit possible inputs.

2. (Optional) write equations for the four output variables.

3. Draw your circuit based on the equations or table above. You may wish to draw a rough draft on paper and then enter the circuit into tkgate, but this is not required.

Your final submitted work should be well-organized on the screen and include comments to make clear your design process and the relationship of the circuit to the design.