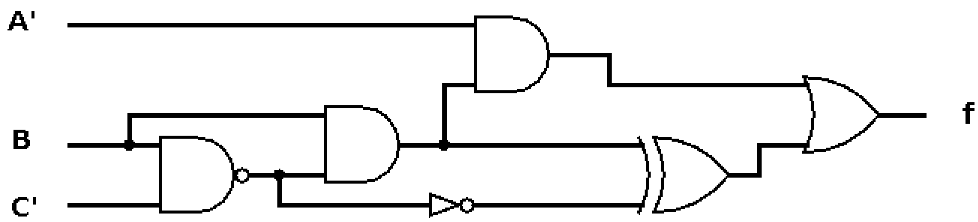
You must show all your work! Answers without supporting work will not be given credit.

All problems are inspired by our *Introduction to Logic Design* 3rd *Edition* text. Any submissions which are not clear and use a respectable amount of space will NOT be considered.

This assignment is worth 3.0 points.

Name: Damian Sclafani

1. Consider the following circuit and provide analysis as indicated. (0.4 Points)



Give your answers in terms of Δ.

(a) How long until the circuit is stable if complemented variables are available?

Answer: A' = 2Δ, B = 5Δ, C' = 5Δ

(b) How long until the circuit is stable if complemented variables are not available?

Answer: A' = 3Δ, B = 4Δ, C' = 5Δ

2. Create a full 2-bit two's complement encoder. (0.5 Points)

   Your circuit should consist of three functions of three input:

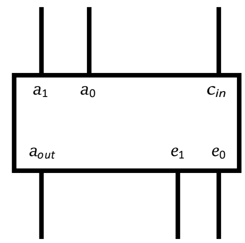
   a_1a_0 are two bits to be complemented,

   c_{in} is a signal to add 1 (if 1),

   $e_1(a_1, a_0, c_{in})$,

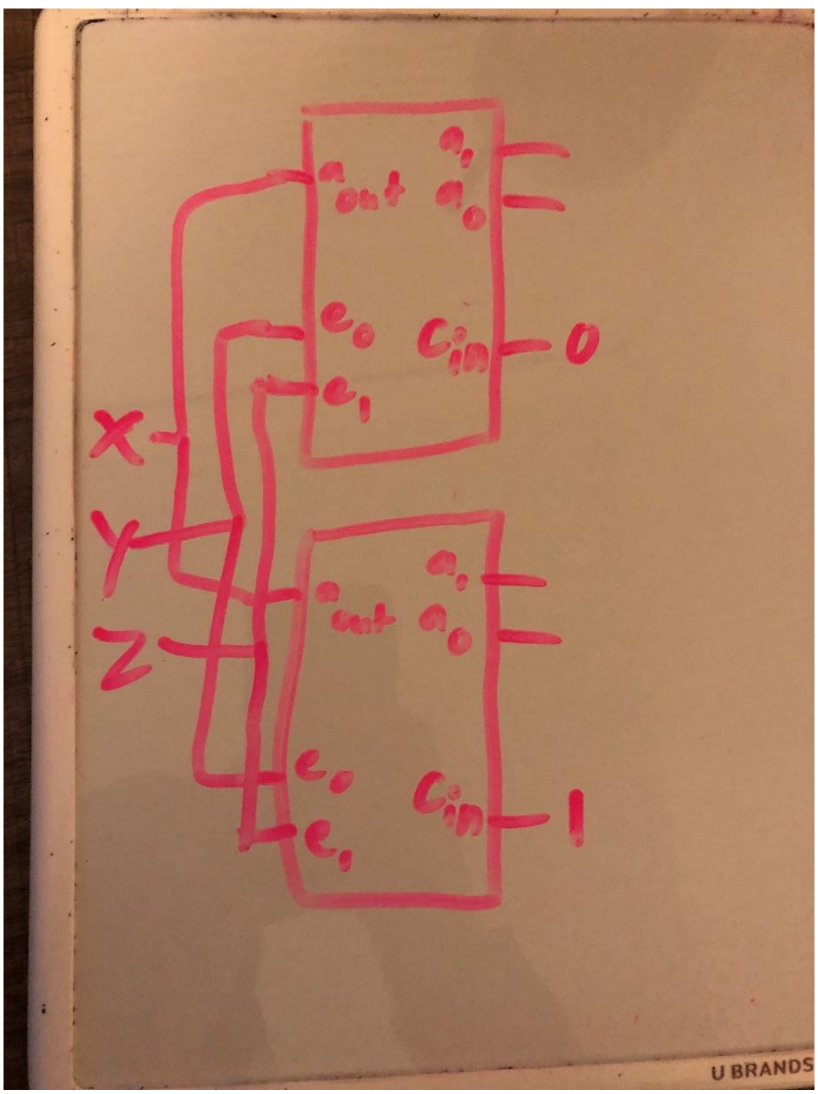
   $e_0(a_1, a_0, c_{in})$, and

   $c_{out}(a_1, a_0, c_{in})$



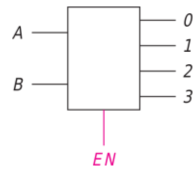
Full 2-bit Two's Complement Encoder.

   Provide minimized equations for the three functions AND a block diagram for a full 4-bit two's complement encoder using two "Full 2-bit Two's Complement Encoder" above. Note that for any iterative circuit, c_{in} of the lowest two bits ($2^n \ldots 2^1 2^0$) must be hard-coded with input 1 to "flip the bits and add 1."



Cont.

3. Using three of the following 2-to-4 active-high enabled, active-high output binary decoders, build a 3-to-8 active-high output binary decoder. (0.5 Points)



A 2-to-4 active-high enabled, active-high output binary decoder.

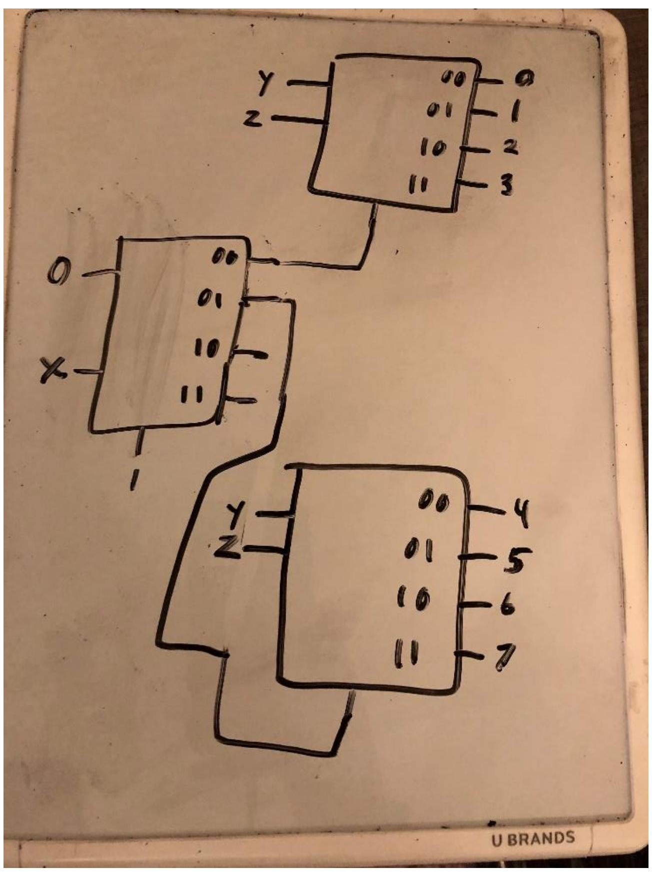
Your decoder should have inputs *x, y, z* and output lines 0*,* 1*,* 2*,* 3*,* 4*,* 5*,* 6*,* 7:

x, y, z inputs are select lines for output,

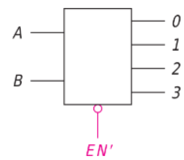
Your decoder's eight output should be ascending, top-to-bottom,

Your decoder does not require an enable, and

You may assume you can access a low or high signal (0 or 1) as needed.

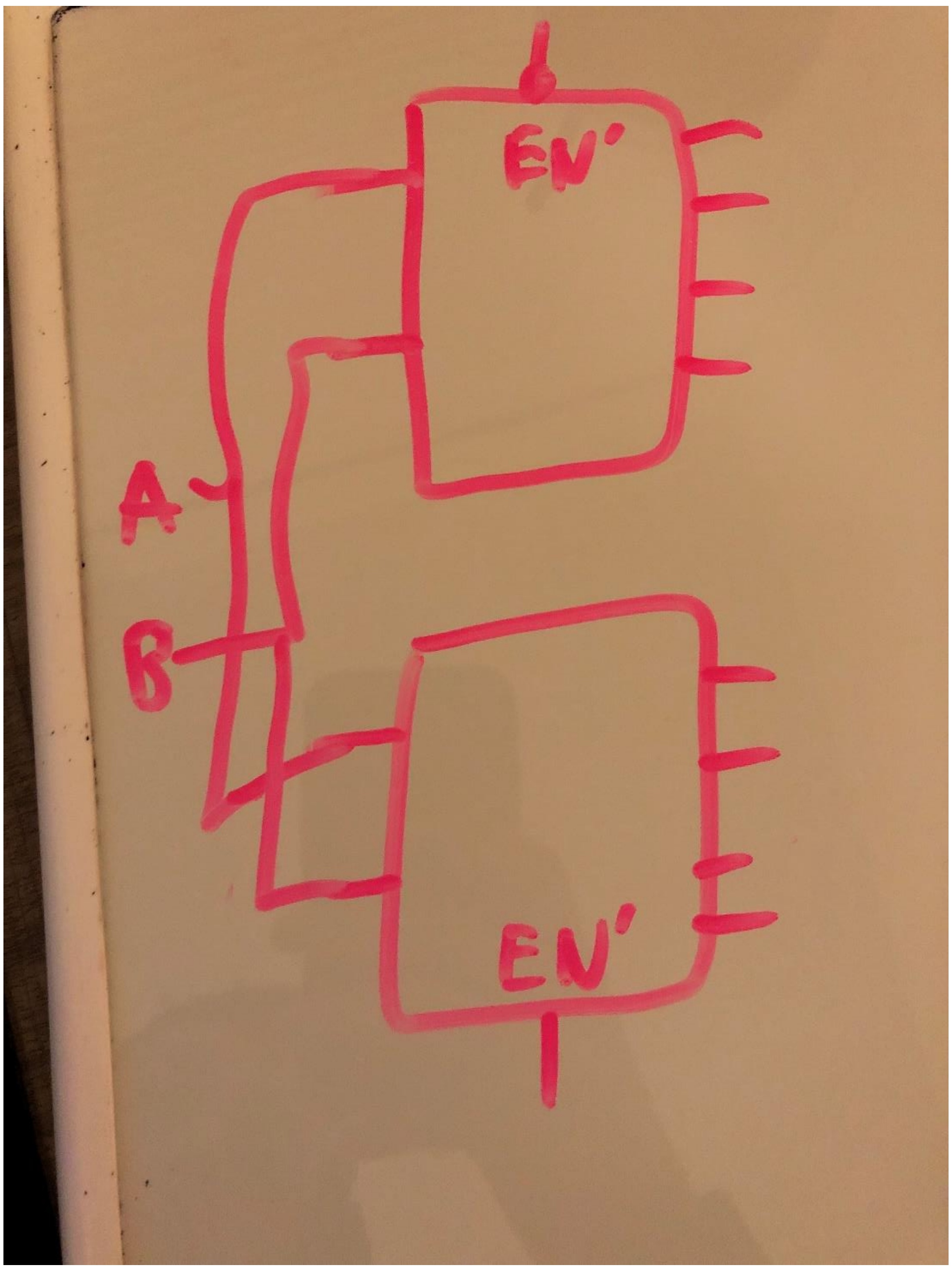


4. Using two decoders and any number of OR and NOT gates, implement a one-bit full binary adder. (0.5 Points)



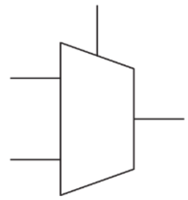
A 2-to-4 active-low enabled, active-high output binary decoder.

Your adder should have the standard inputs *a, b,* c_{in} and the standard outputs *s,* c_{out}. See Figure 1.2 for reminder and Table 1.5 for an explanation of behavior (page 10 from our text).



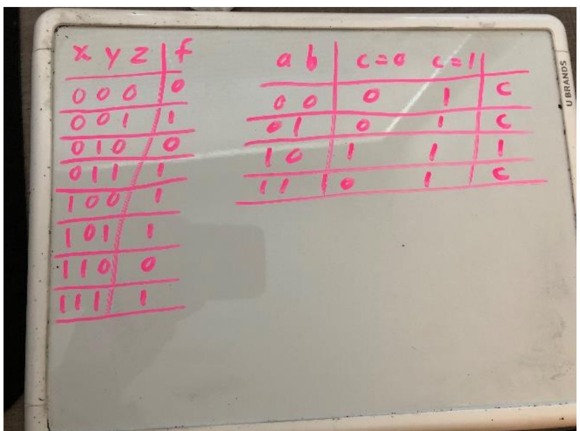
Cont.

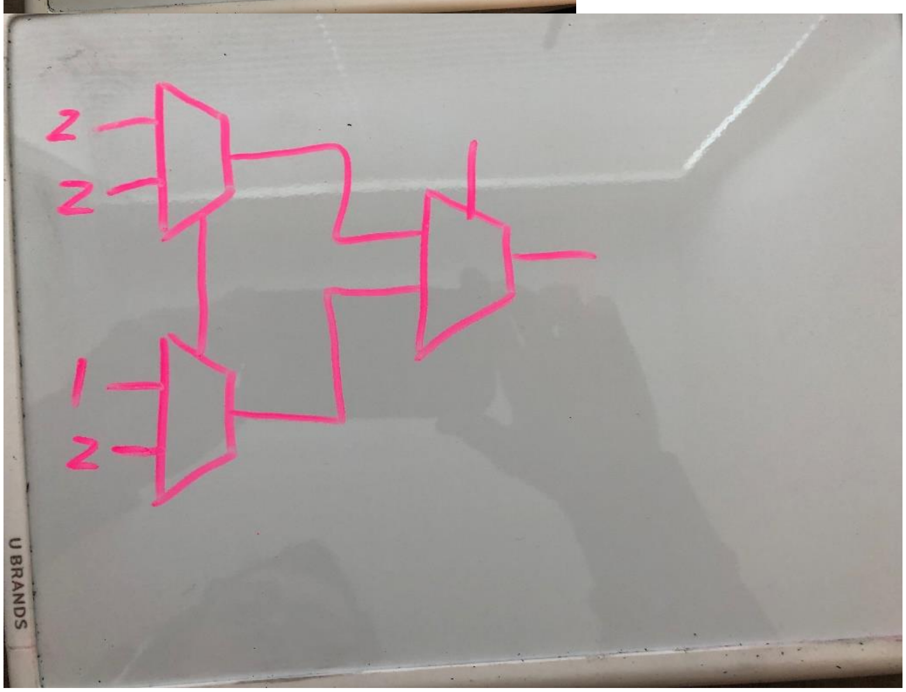
5. Using any number of two-way MUX, implement the following function. (0.5 Points)
   $f(x, y, z) = \Sigma m(1, 3, 4, 5, 7)$



A two-way MUX.

Your adder should have the standard inputs a, b, c_{in} and the standard outputs s, c_{out}. See Figure 1.2 for reminder and Table 1.5 for an explanation of behavior (page 10 from our text).





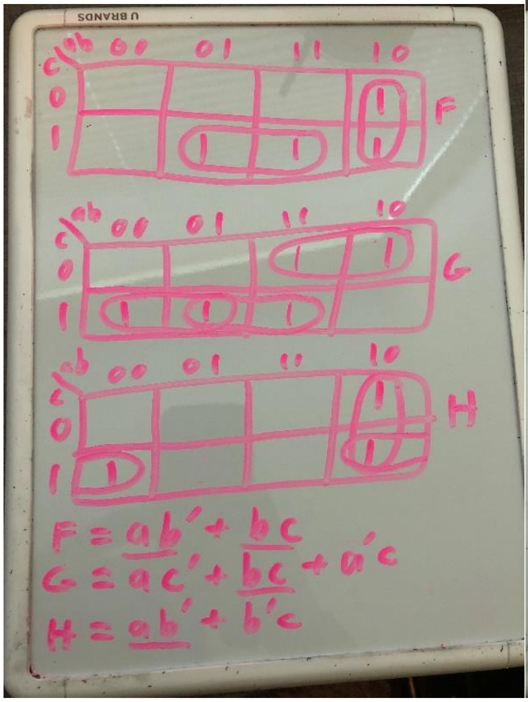
Cont.

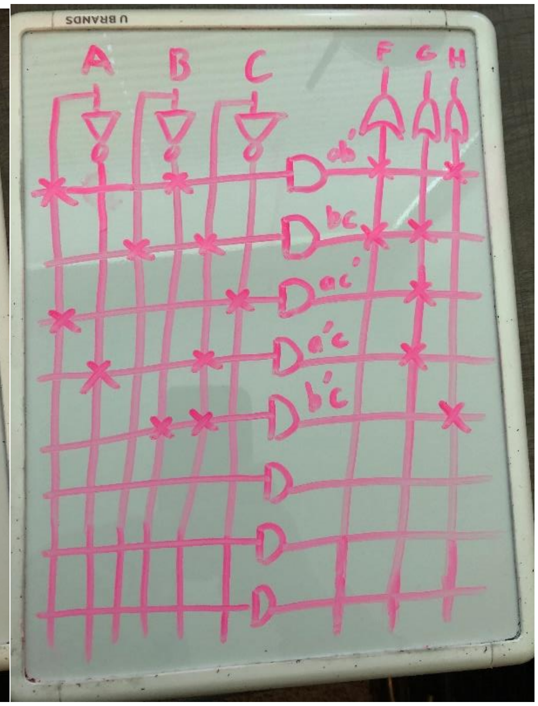
6. For the following functions, program each of a ROM, PLA, and PAL. (0.6 points)

$F(A, B, C) = \Sigma m(3, 4, 5, 7)$
$G(A, B, C) = \Sigma m(1, 3, 5, 6, 7)$
$H(A, B, C) = \Sigma m(1, 4, 5)$

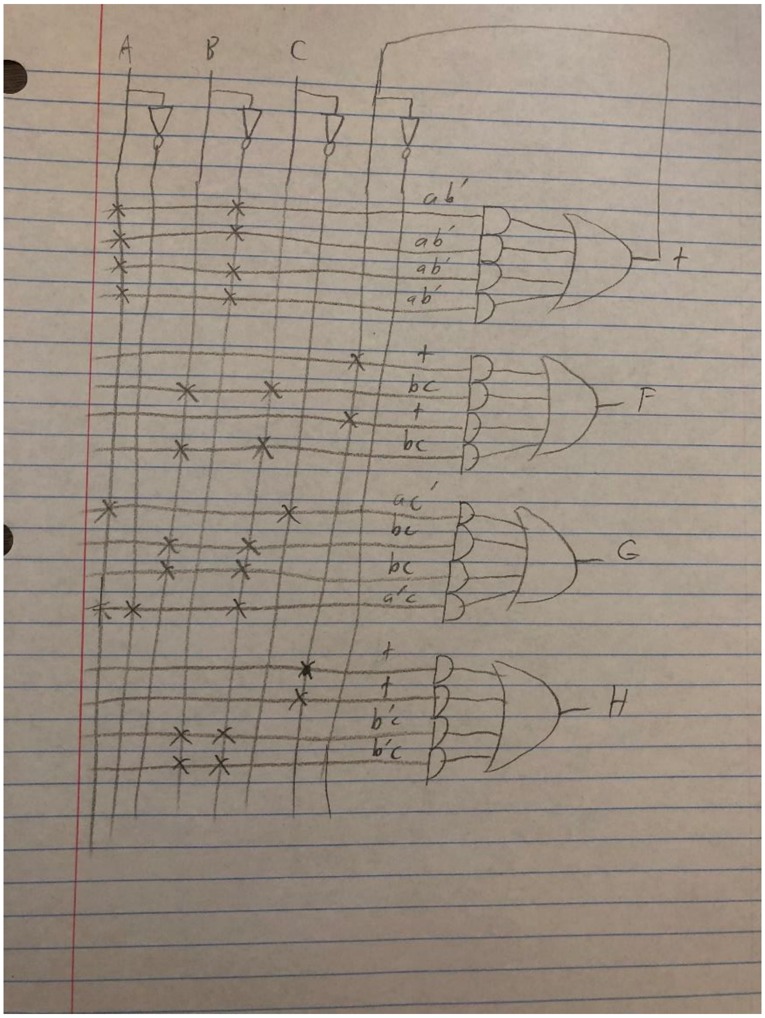
(a) Label the AND gates corresponding to minterms (0–7). You only need use three input, but your minterms must be ascending top-to-bottom.

(b) Simplify F, G, and H to a shared total of four product terms. Label each line with its product term.





(c) You must share (at least) the essential prime implicant AB'.



The End.