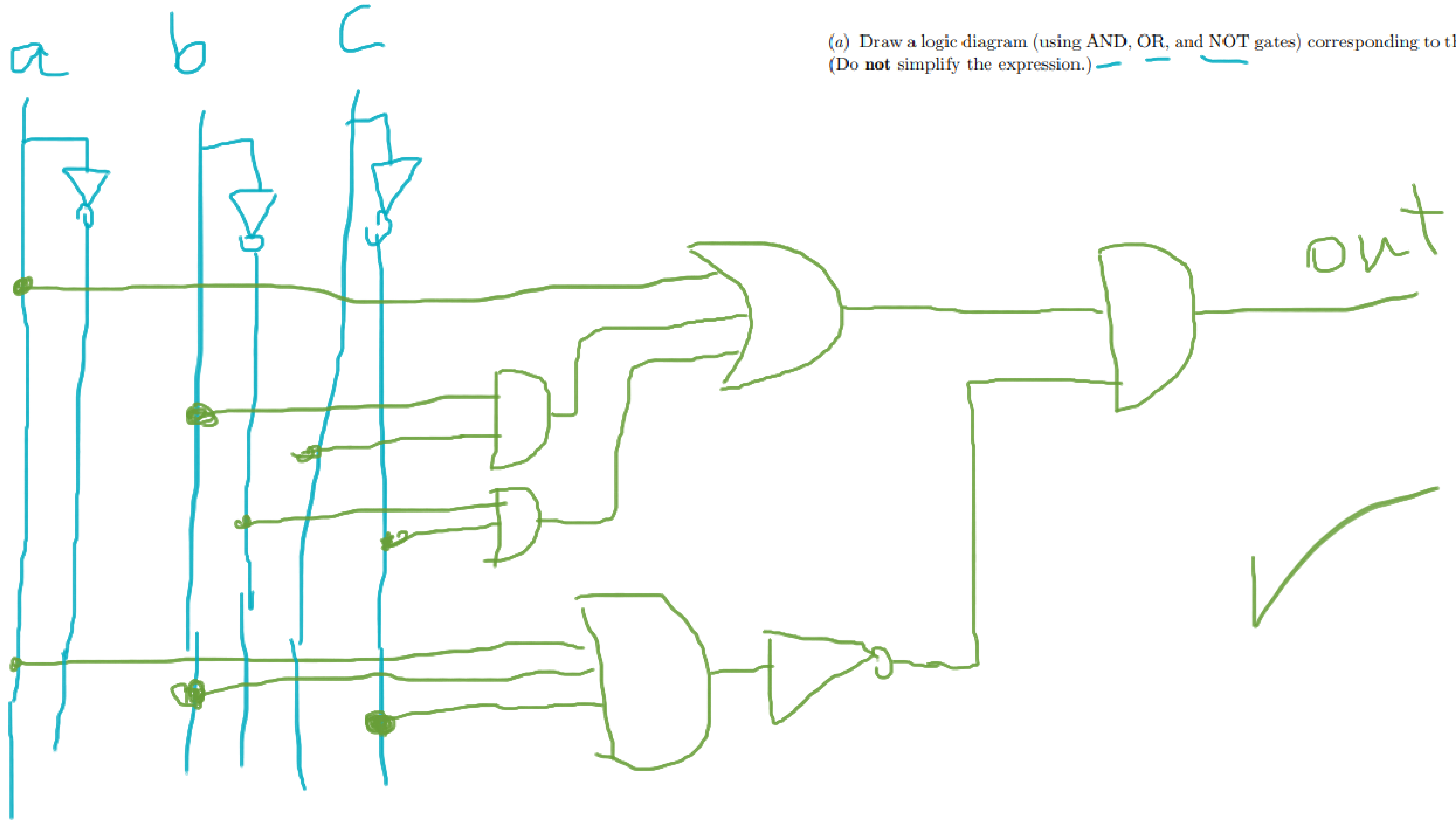
Problem 1: (22 pts) The problems below are based on the following Boolean function:

$$(a + bc + b'c')(abc')'$$

(a) Draw a logic diagram (using AND, OR, and NOT gates) corresponding to the Boolean function. (Do **not** simplify the expression.)



(*b*) Write the Boolean function in minterm canonical form. (Show a Boolean expression, not just a list of minterm numbers.) *Hint: For most people directly constructing a truth table would be easier than algebraic manipulation.*

(*c*) Write the Boolean function in maxterm canonical form.

$$(a + bc + b'c')(abc')'$$

a	b	c	out
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

$0 \to F$

$1 \to T$

SOP : Minterm

POS : Maxterms

$0 \to T$

$1 \to F$

NAND

SOP $\bar{a}\bar{b}\bar{c} + \bar{a}bc + a\bar{b}\bar{c} + a\bar{b}c + abc$

Minterm

POS $(a+b+\bar{c}) \cdot (a+\bar{b}+c) \cdot (\bar{a}+\bar{b}+c)$

NOR

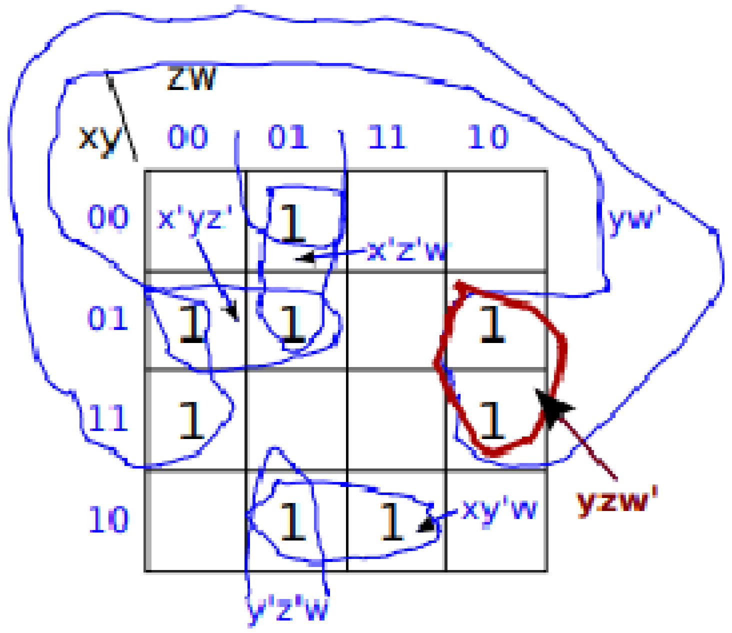
(*d*) Draw a **Karnaugh map** for the expression. (Just draw the Karnaugh map, **don't use** it to simplify the expression.)

a \ bc	00	01	11	10
0	1 (0)	(1)	1 (3)	(2)
1	1 (4)	1 (5)	1 (7)	(6)

$\overline{b}\,\overline{c} + bc + ac$

$b \odot c + ac$ #

Problem 2: (22 pts) Consider the Karnaugh map below.



(a) Write in the row and column numbers.

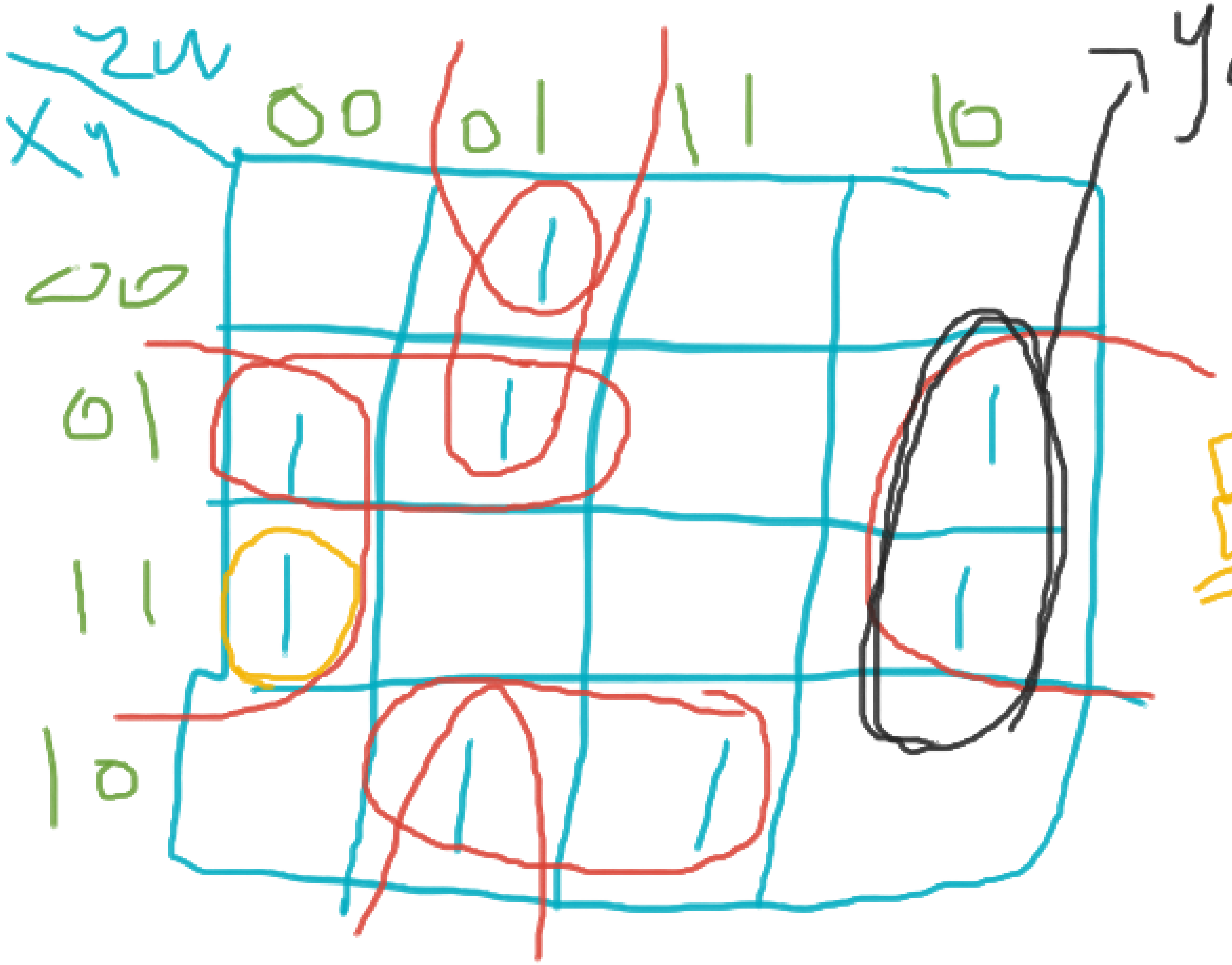
(b) List all of the prime implicants both on the Karnaugh map above, and as a list below.

(c) In the list of prime implicants above, write an "E" next to each *essential* prime implicant.

(d) Provide an example of an implicant that's neither a prime implicant, nor a minterm. Circle this implicant and show the corresponding Boolean expression. *Grading Note: The original wording of the checkbox item below was slightly different in the original exam.*

(e) Based on the Karnaugh map show a minimum-cost expression for this logic function.

~~xywz~~



$yz\bar{w}$

- It's not a prime implicant because it's not minimal; you can further reduce it to A'C.
- It's not a minterm because it doesn't represent a single minterm from the given set.

$\bar{x}\bar{z}w$, $\bar{y}\bar{z}w$, $\bar{x}y\bar{z}$

E $y\bar{w}$, $x\bar{y}w$ E

xy \ zw	00	01	11	10
00		1		
01	1	1		1
11	1			1
10		1	1	

$$y\bar{w} + \bar{x}\bar{z}w + x\bar{y}w$$

Problem 3: (22 pts) Consider the Boolean function below:

$$ab' + b'c + a'bc'$$

(*a*) Use a 3×8 decoder plus whatever logic gates are needed to implement this function.

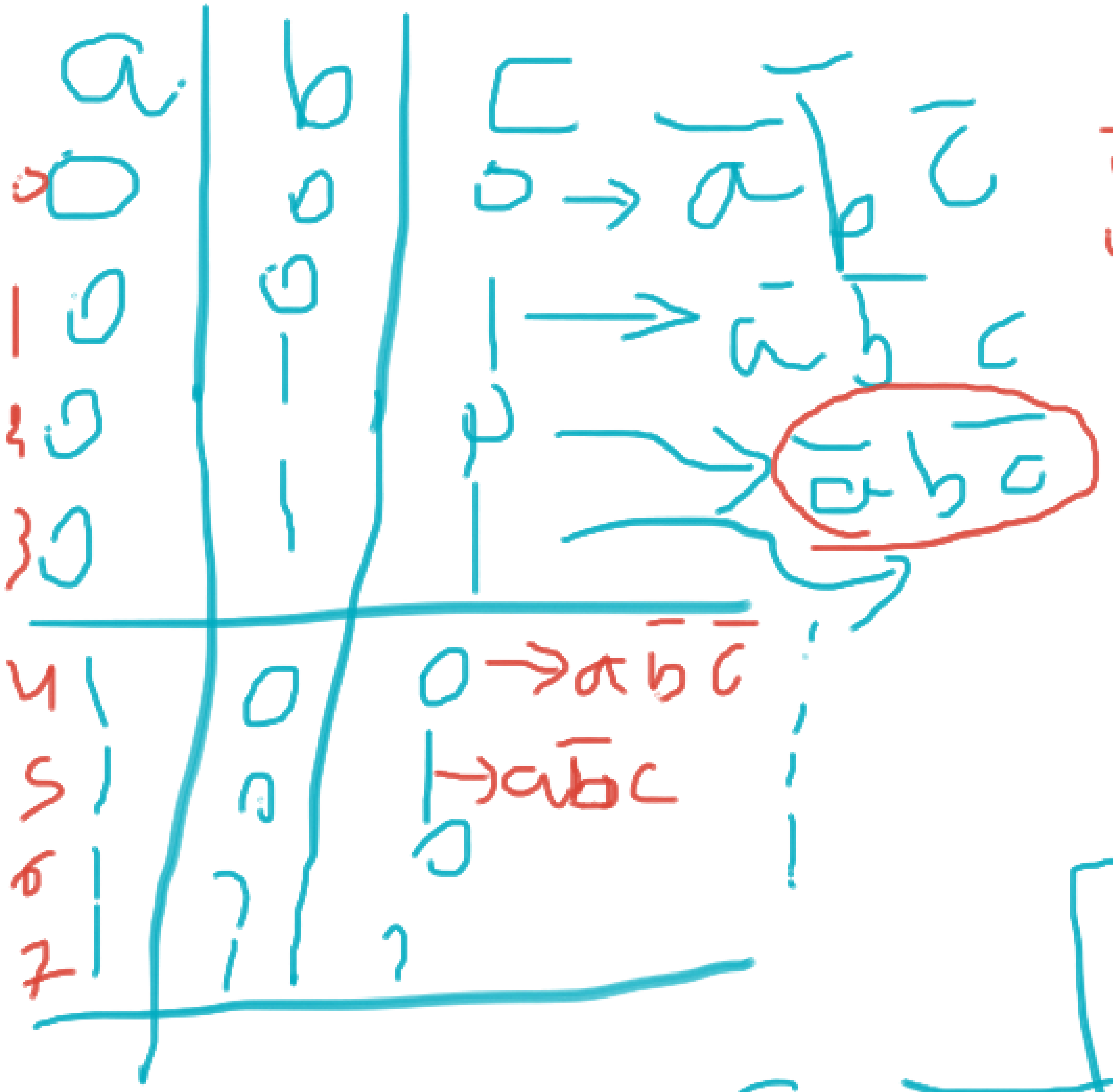
$\bar{b}c < \begin{matrix} \bar{a}\bar{b}c \\ a\bar{b}c \end{matrix}$

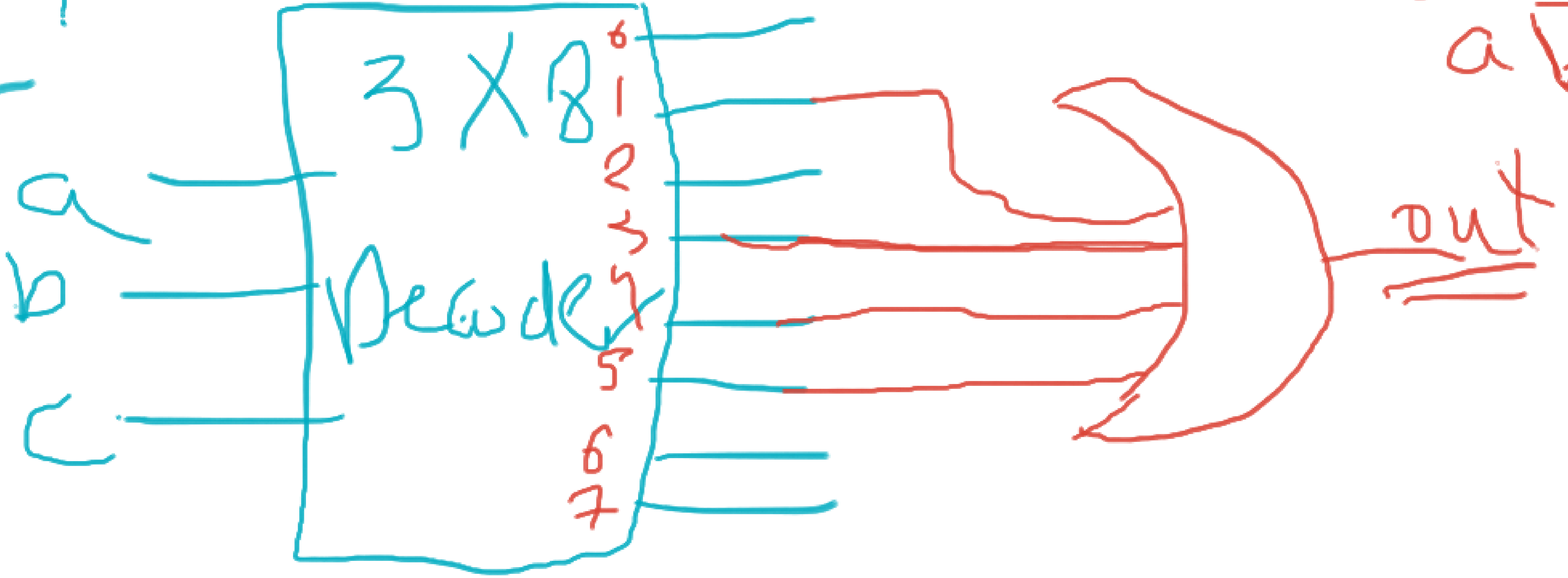
$\boxed{a\bar{b}} < \begin{matrix} a\bar{b}\bar{c} \checkmark \\ a\bar{b}c \checkmark \end{matrix}$

$a\bar{b}c + a\bar{b}\bar{c}$

$a\bar{b}(c + \bar{c})$

$a\bar{b}$





(b) Use an 8-input multiplexer to implement this function.

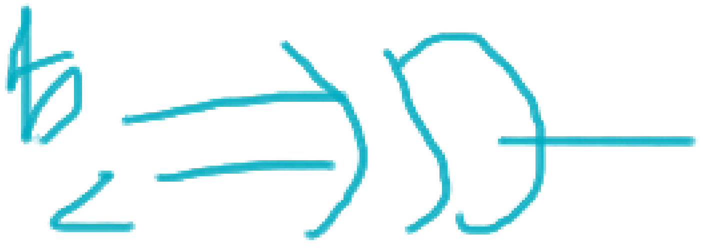
a	b	c	o	
0	0	0	0	0
0	0	1	1	1
0	1	0	1	2
0	1	1	0	3
1	0	0	1	4
1	0	1	1	5
1	1	0	0	6
1	1	1	0	7

$a\bar{b} + \bar{b}c + \bar{a}b\bar{c}$

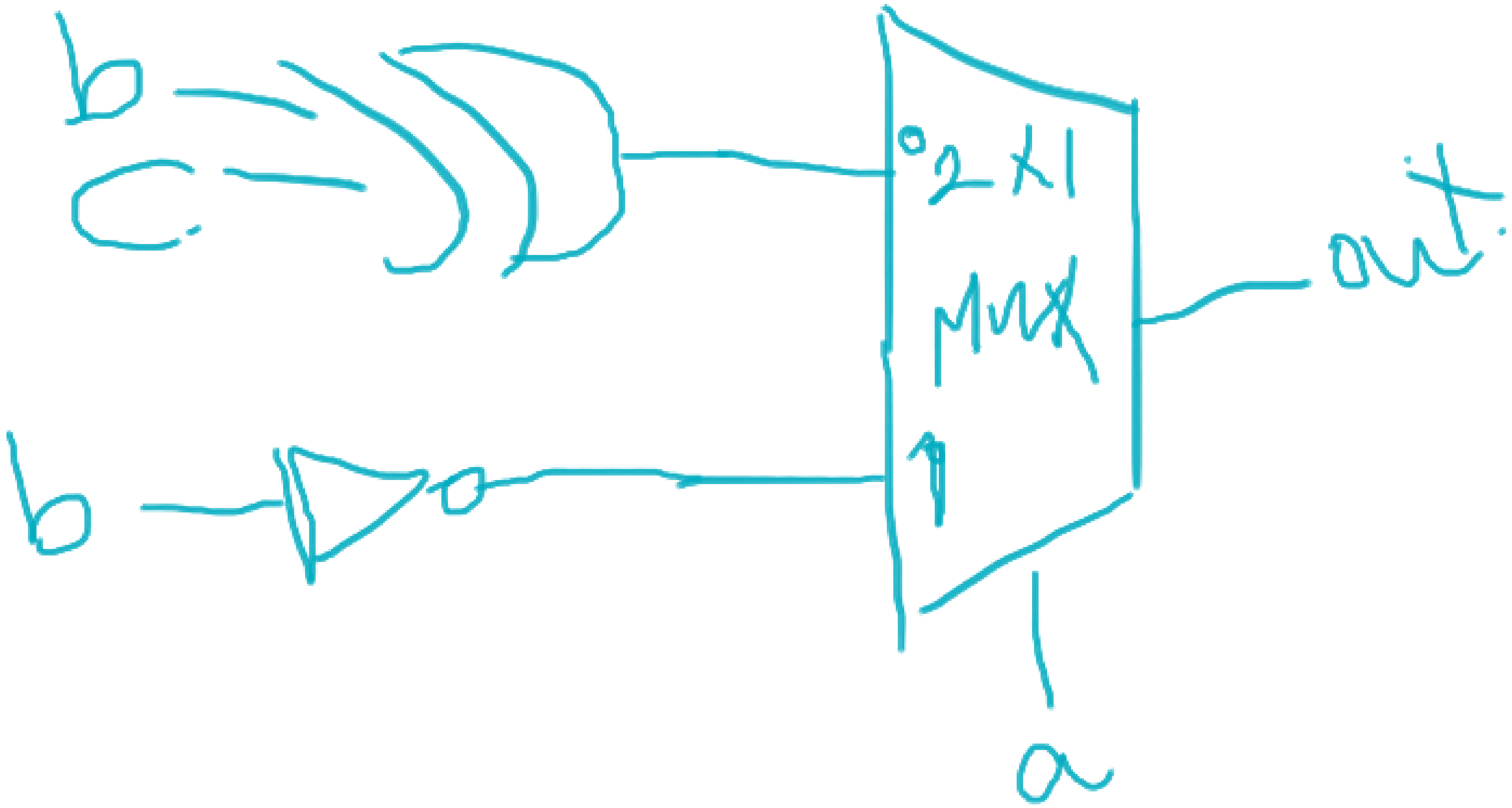
(c) Use a **multiplexer** and **additional** logic, **including** possibly exclusive-or gates, **to implement** this function by performing a Shannon expansion with respect to a (use a as the multiplexer control input). *Hint: it might be easier to eyeball a truth table than to do this by algebraic manipulation.*

$$ab' + b'c + \cancel{a'bc'}$$

$\underline{a=0}$: $\bar{b}c + b\bar{c}$

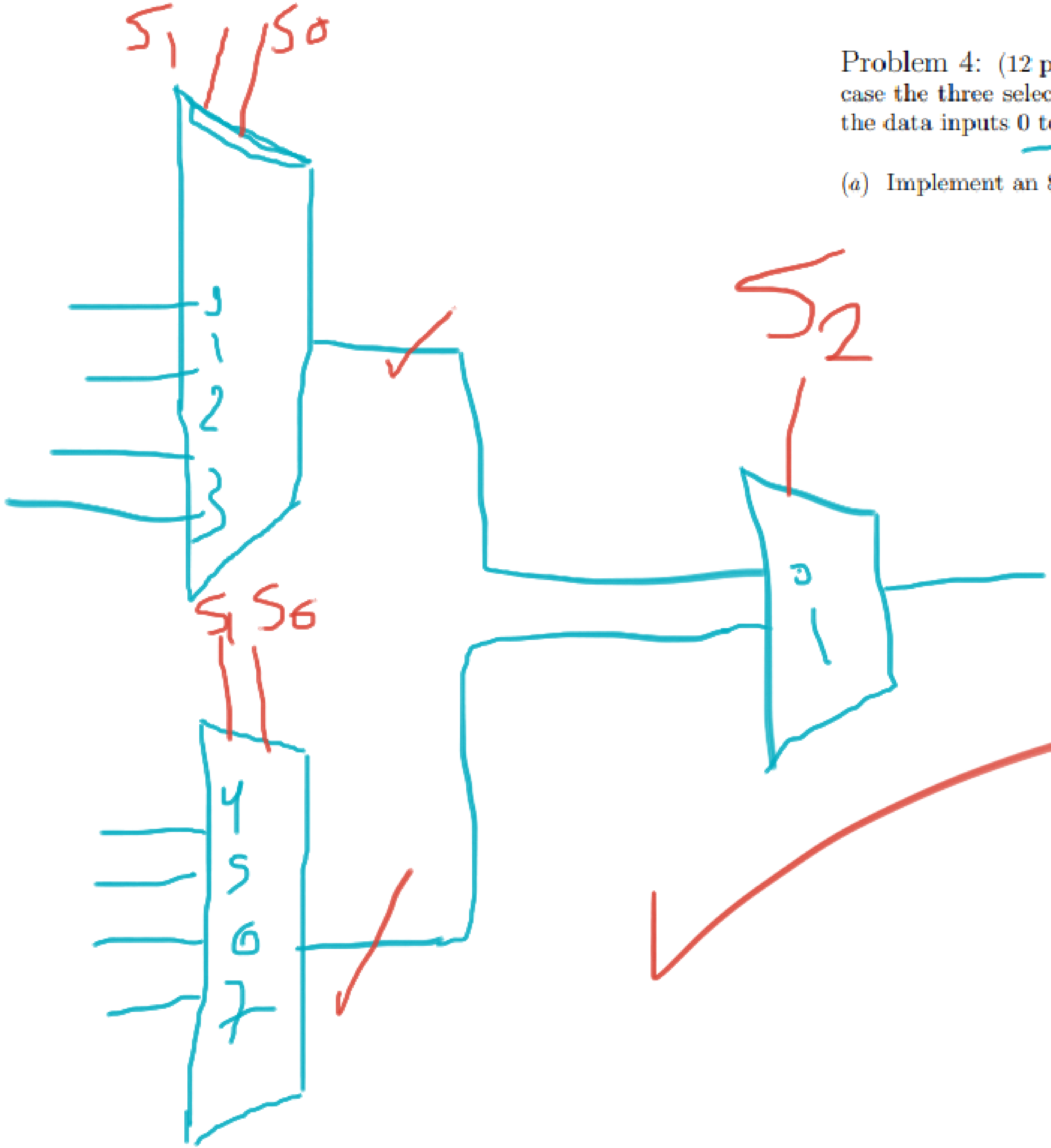


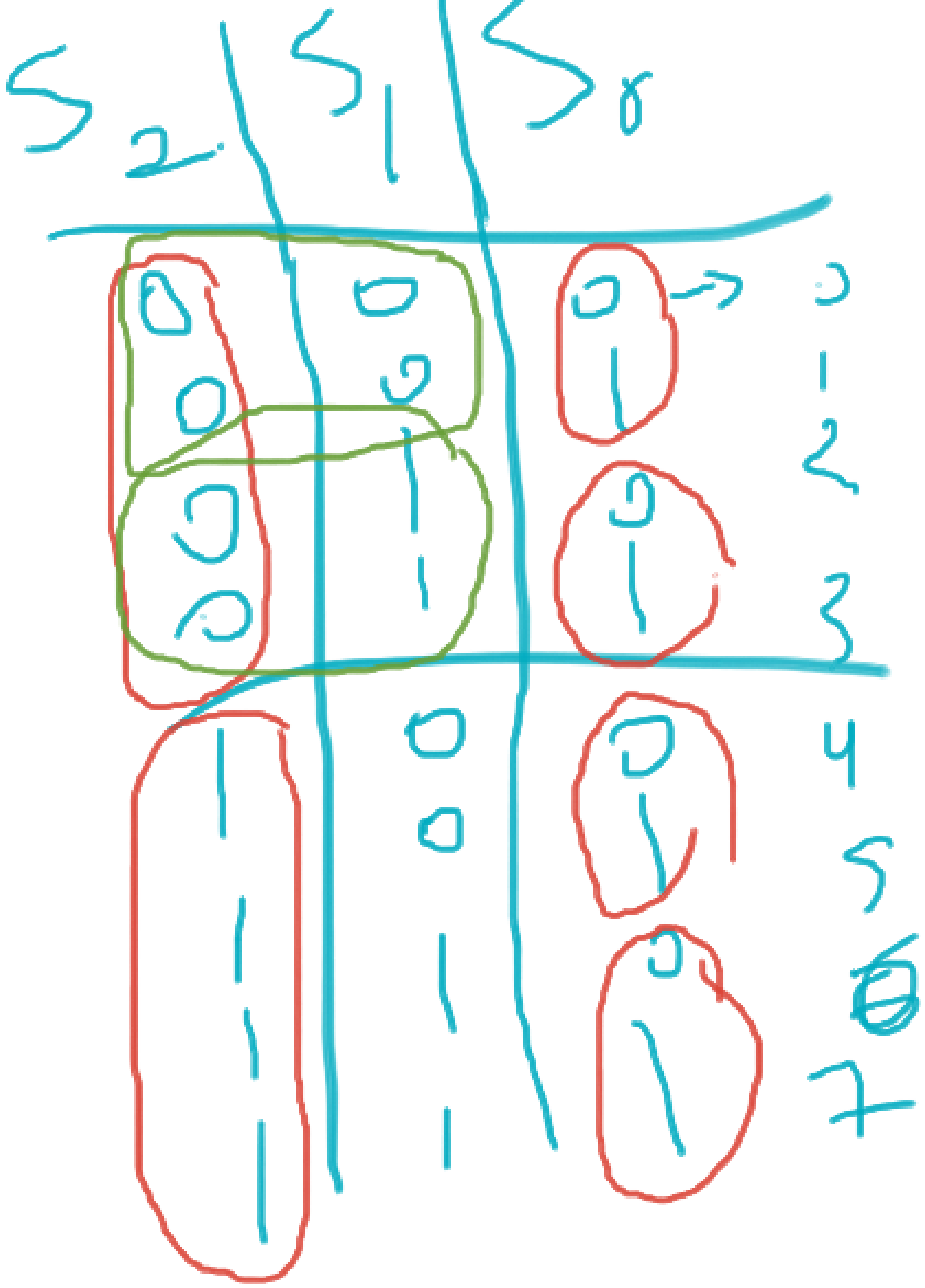
$\underline{a=1}$: $\bar{b} + \bar{b}c = \bar{b}(\cancel{1+c}) = \boxed{\bar{b}}$



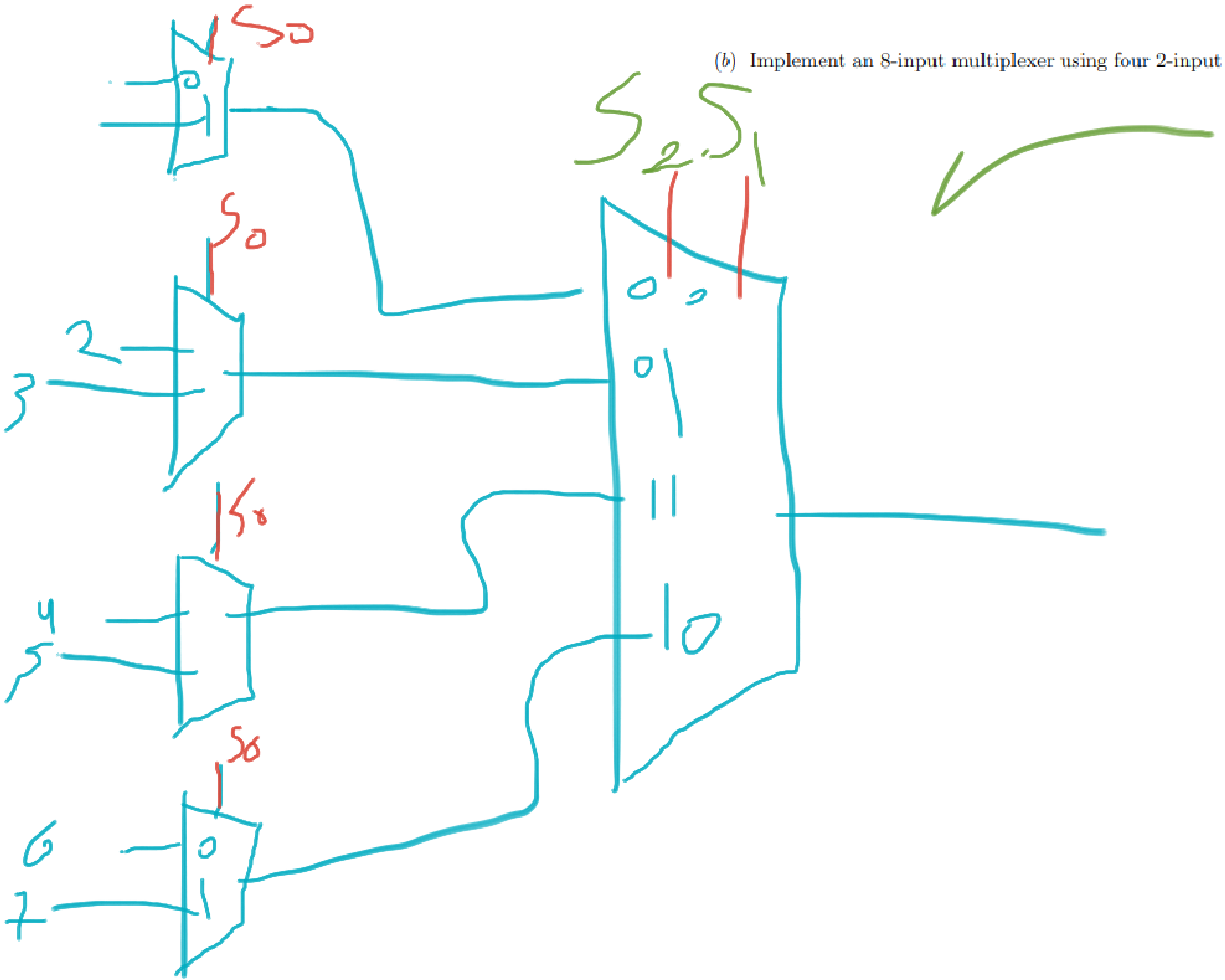
Problem 4: (12 pts) Show how to implement the 8-input multiplexers described below. In each case the three select input bits should be labeled s_2, s_1, s_0, with s_0 being least significant. Label the data inputs 0 to 7.

(a) Implement an 8-input multiplexer using two 4-input multiplexers and a 2-input multiplexer.



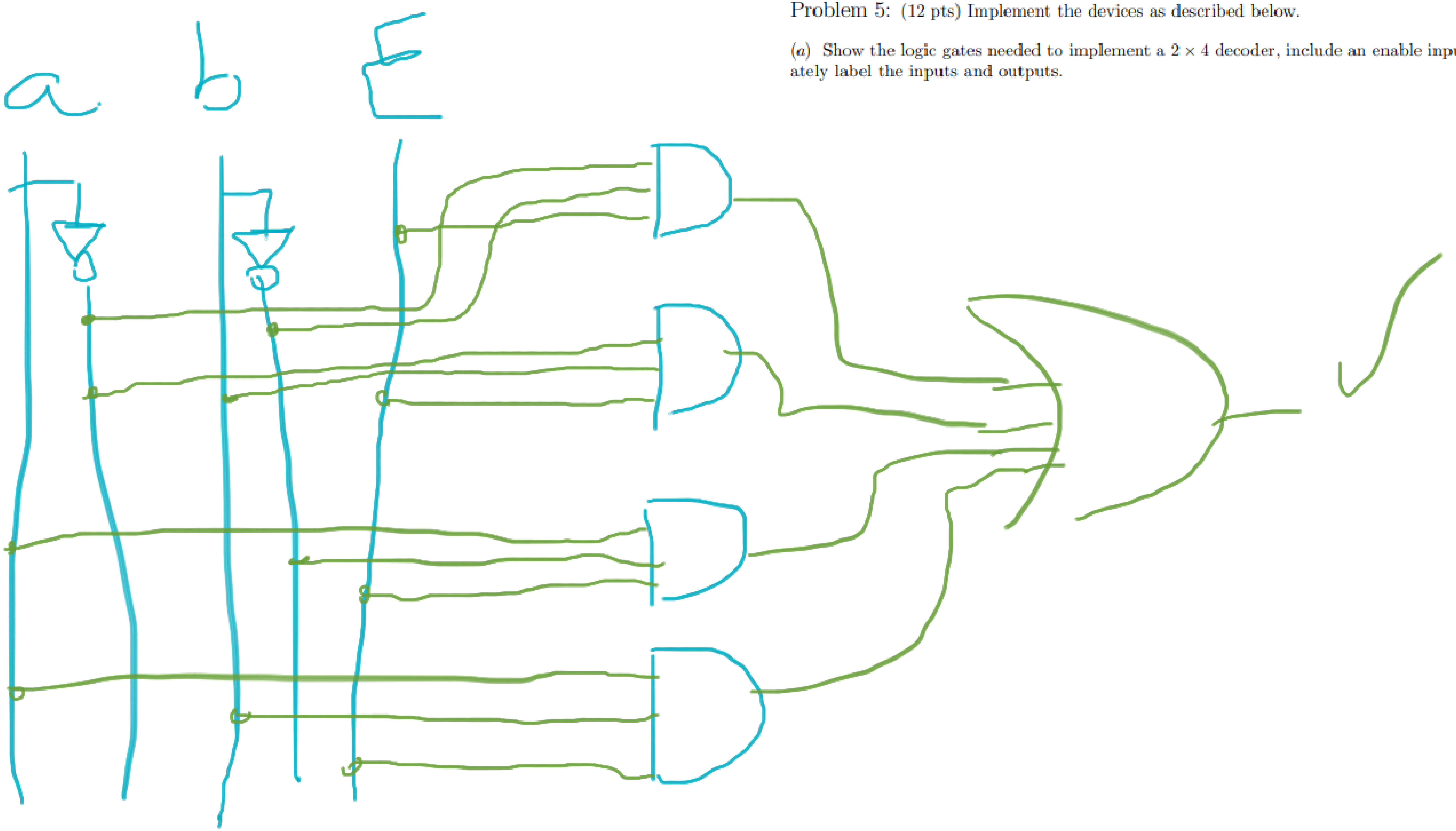


(b) Implement an 8-input multiplexer using four 2-input multiplexers and one 4-input multiplexer.

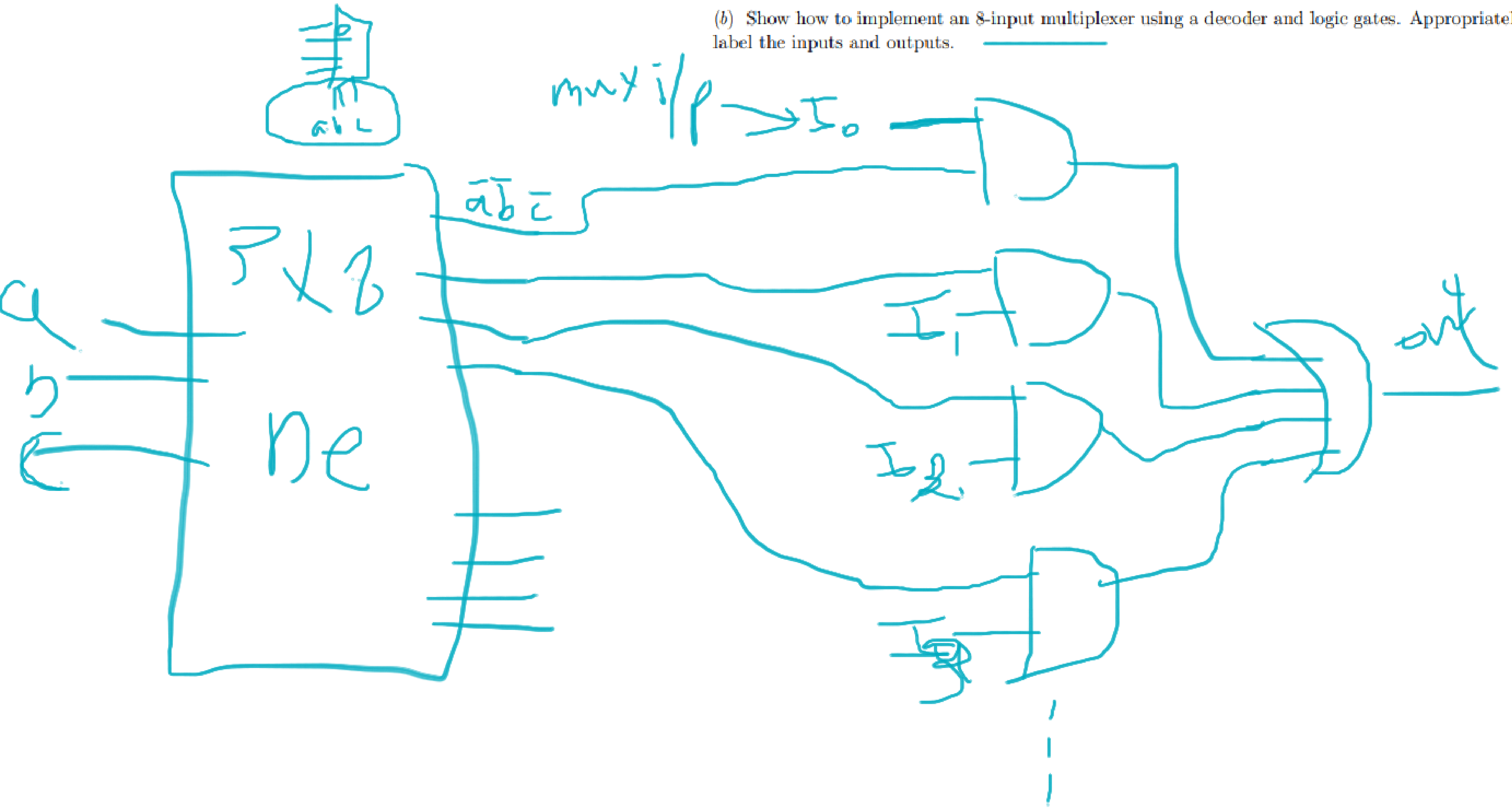


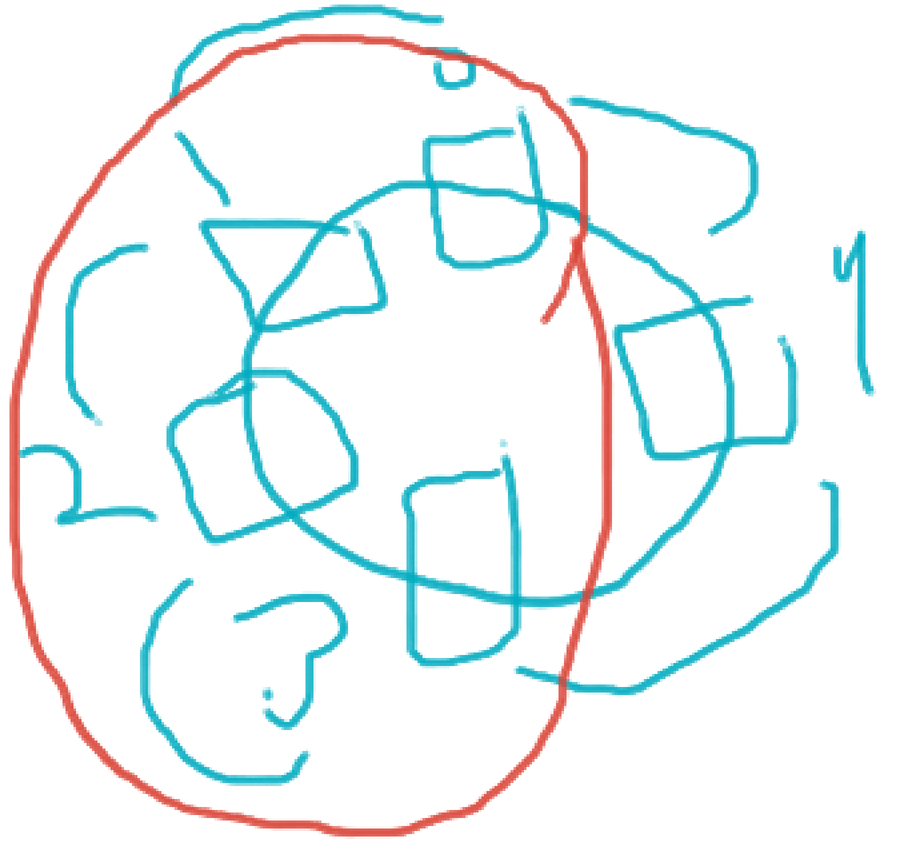
Problem 5: (12 pts) Implement the devices as described below.

(*a*) Show the logic gates needed to implement a 2×4 decoder, include an enable input. Appropriately label the inputs and outputs.



(b) Show how to implement an 8-input multiplexer using a decoder and logic gates. Appropriately label the inputs and outputs.





Problem 6: (10 pts) Answer each question below.

(*a*) Consider five seats, numbered 0 to 4, arranged in a circle and described by Boolean variables i_0 to i_4. Boolean variable i_0 is true if seat 0 is occupied and i_0 is false if the seat is not occupied (no one is sitting in the seat), likewise for i_1, i_2, i_3, and i_4.

Write a Boolean expression that's true if at least two people are sitting next to each other and at least one seat is not occupied. (Note: Just write one Boolean expression.) *Hint: This can easily be solved without a truth table.*

$$(i_0 i_1 + i_1 i_2 + i_2 i_3 + i_3 i_4 + i_4 i_5) \cdot (\overline{i_0 i_1 i_2 i_3 i_4})$$

$0123 + 1234 + 2340$

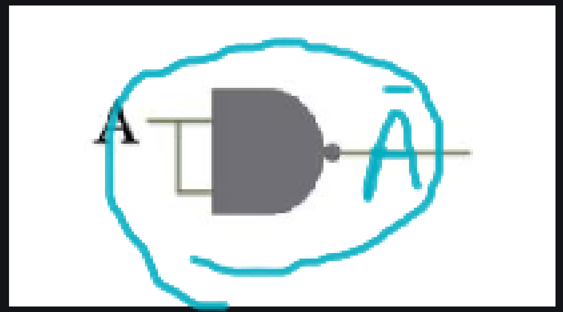
$\overline{01234}$

(*b*) The statement below is not true. Explain why and correct it.

"By implementing a sum-of-products expression using only NAND gates (in place of AND and OR gates) we expose additional opportunities for simplification."

both exp - identical

COMPLEMENT Using NAND



(AA)' = A'

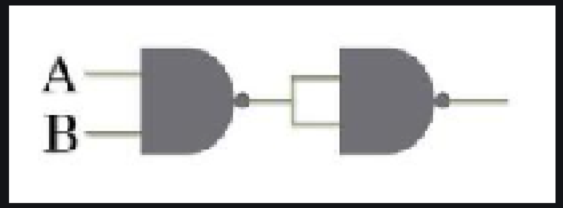
$$\overline{(A \cdot A)} \to \bar{A}$$

$$\overline{\overline{A \cdot B}} = A \cdot B$$

$$\overline{(\bar{A} \cdot \bar{B})} \to \boxed{\bar{\bar{A}} + \bar{\bar{B}}}$$

AND Using NAND

OR → NOR

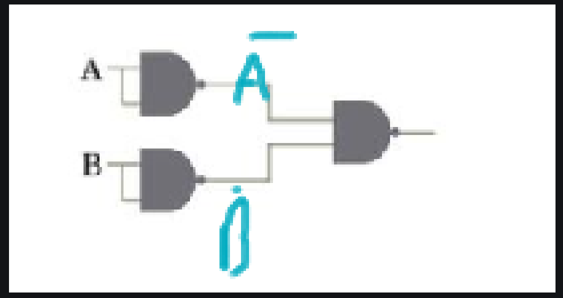


Evaluates to AB

This is quite straightforward, we wish to obtain AB but the NAND gate gives an output (AB)' so we complement the output of the NAND gate using another NAND gate to obtain ((AB)')' which is AB.

OR Using NAND

AND → NOR



Evaluates to A+B

AND

$$F_1(A,B,C,D) = \bar{A}\bar{B}C + A\bar{B}\bar{C} + A\bar{B}C + \cancel{\bar{A}B\bar{C}} + \bar{A}BC + ABC$$

b. NAND gate only.

$$\bar{A}C(\cancel{\bar{B}+B})$$

$$AC(\cancel{\bar{B}+B})$$

$$\cancel{(A+\bar{A})}\boxed{C}$$

$$C + A\bar{B}\bar{C}$$