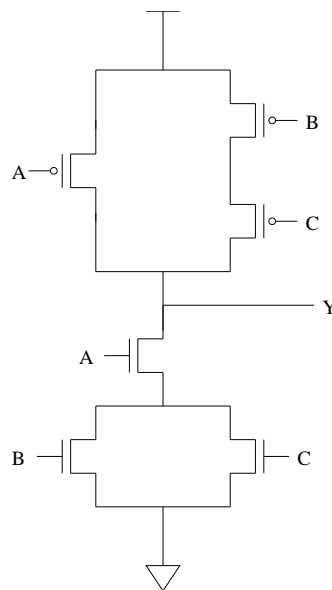


Write your answers on these pages. Additional pages may be attached (with staple) if necessary. Please ensure that your answers are legible and *show your work*. Write your name at the top of each page. Due at the *beginning of class*. Total points: 44.

1. [4 Points] **Complex Gate**. Fill in the truth table for the following complex gate:

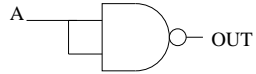


A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

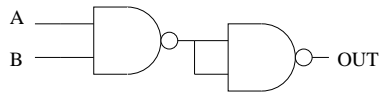
2. [8 Points] **Logical Completeness.** NAND is logically complete (*i.e.*, any logic function can be built using only NAND gates). Use the minimum number of NAND gates (and *only* NAND gates) to construct gate-level circuits that compute the following:

Answer:

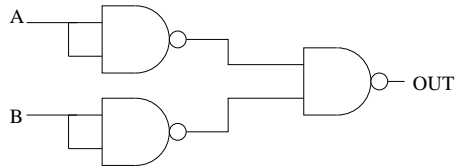
NOT



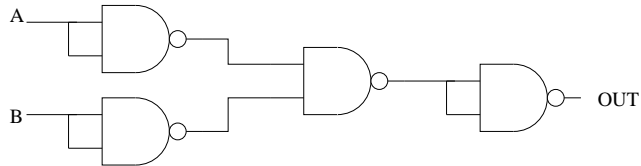
AND



OR



NOR

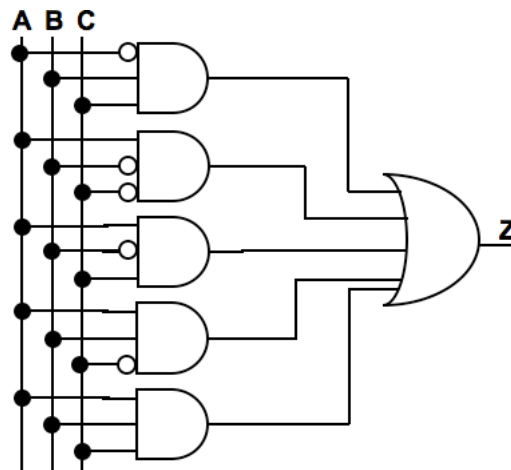


3. [18 Points ((4x4) + 2)] **Combinational Logic.** In this question you'll create four different implementations of the same logic function, Z :

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

- (a) Construct a gate-level logic circuit for Z (above) directly using a PLA.

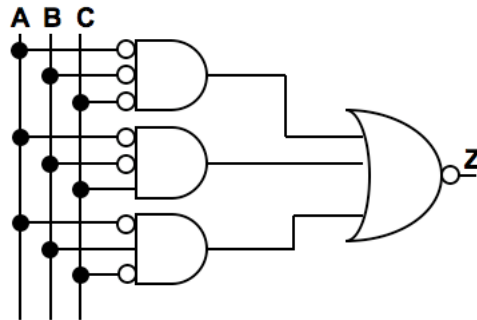
Answer:



- (b) How many transistors does your circuit require (assume that n -input NOR and NAND gates have $2n$ transistors; n -input OR and AND gates have $2n + 2$ transistors)? Hint: count only a single inverter for calculating each of \bar{A} , \bar{B} , and \bar{C} .

Answer: 58 transistors (6 transistors for NOT, 40 transistors for AND, and 12 transistors for OR)

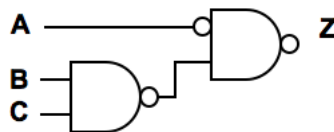
- (c) Construct a gate-level logic circuit for Z by creating a PLA for the *inverse* of Z (\bar{Z}) and replacing the OR gate with a NOR gate).



- (d) How many transistors does your second circuit require (again assuming standard n -input CMOS gates)?

Answer: 36 transistors (6 transistors for NOT, 24 transistors for AND, and 6 transistors for NOR)

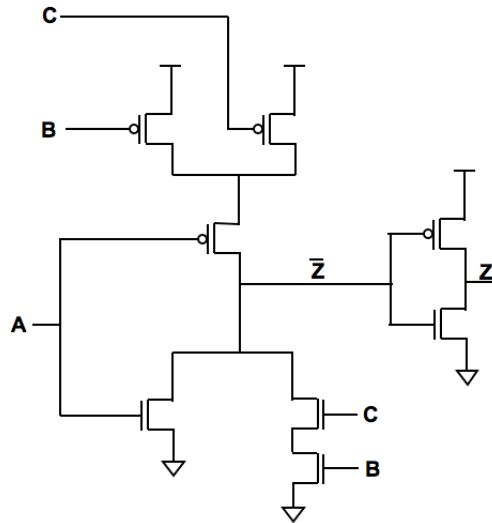
- (e) Construct a “minimal” gate-level logic circuit for Z (minimal in terms of transistors) using the standard CMOS gates (and, or, not, nand, nor).



- (f) How many transistors does your third circuit require (again, assuming standard n -input CMOS gates)?

Answer: 10 transistors (4 transistors for each NAND and 2 more transistors for an inverter)

- (g) Construct a “minimal” CMOS transistor-level circuit for Z (minimal in terms of transistors). Hint: first construct a single complex gate for \bar{Z} .



- (h) How many transistors does your transistor-level circuit use?

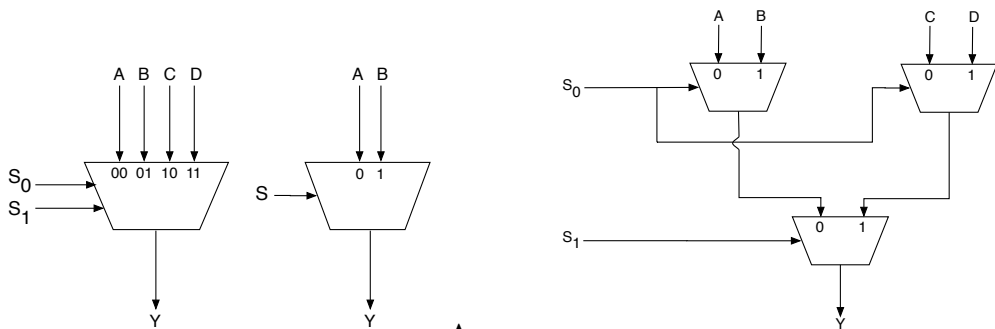
Answer: 8 transistors

- (i) What can you learn from comparing the number of transistors used by these four different implementations of the same logic function?

Answer:

There are several things one could learn from the above four designs. First, the number of transistors in a PLA be reduce by simply inverting the function if the number of 1s in the output is greater than the number of 0s. Second, a custom circuit for a logic function is can be significantly more efficient than a PLA. Third, a transistor-level design can be even more efficient. The general trend is designing at a more detailed level can result in more efficient circuits, at the cost of giving up the automatic PLA (it takes more intellectual effort to design the more efficient circuits). We were looking for perhaps two things you may have learned; just writing "the number of transistors goes down" is not a very good answer because it doesn't explain what can be learned.

4. [5 Points] **Multiplexers.** A one-bit 4-to-1 mux (below, left) selects among four inputs using two selector lines. Specifically, the mux outputs A when the selectors have the value 00 (with S_1 being the high-order bit), B when they have the value 01, C when they have the value 10, and D when they have the value 11. Construct a one-bit 4-to-1 mux (*i.e.*, build a circuit with the behavior of a one-bit 4-to-1 mux) using only one-bit 2-to-1 muxes (below, right). Be sure to label the inputs (A, B, C, D, S_0 , and S_1) and output (Y).



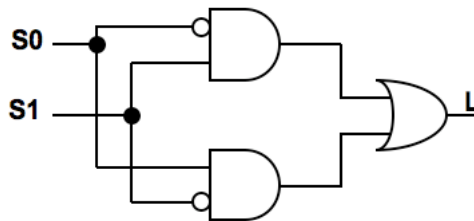
Answer:

5. [8 Points (3+4+1)] **Logic Design.** Your task is to design the logic to control a overhead light in a room based on the position of two light switches. Because the switches are at two different entrances to the room, either switch should be able to change the state (off/on) of the light independently. If both switches are in the “down” position (represented by a zero), the light must be off (represented by a zero). No matter what position the switches are in or the current state of the light, flipping either switch must change the state of the light.

(a) Complete the truth table for the circuit:

S_0	S_1	L
down (0)	down (0)	off (0)
down (0)	up (1)	on (1)
up (1)	down (0)	on (1)
up (1)	up (1)	off (0)

(b) Create a gate-level circuit (using and, or, not, nor, nand) for the logic function:



(c) What is the common name for this function?

Answer: XOR