

ES 4 Problem Set 2 solutions

After solving the problems, look at the solutions posted on the course website and categorize your work for each problem on the following scale:

- ● Completely correct
- ◐ Nearly correct, but made a small mathematical or copying error
- ◑ Solved part of the problem correctly
- ◒ Started some work in the right direction
- ○ Incorrect, or didn't even know where to start on the problem
- Include a question mark (?) in addition to one of the above symbols if you don't feel like you understand the question or the solution well enough to make a definite judgement.

Problems with an asterisk (*) are optional.

1.a	1.b	1.c*	1.d	1.e*	2	3.a	3.b	3.c*	3.d*

What questions do you have about these concepts and skills?

What things are you uncertain about (even if you don't have a specific question)?

Approximately how long did it take you to complete the homework?

How long did you take going over the solutions and writing this reflection?

Turn in this self-assessment sheet on Gradescope. You do not need to turn in anything else, although we're happy to look at your work if you have questions!

Part 1: Logic minimization

Minimize the truth tables and logic functions below.

(a) $\bar{A}BC + \bar{A}B\bar{C} + \bar{A}C\bar{D} + AB\bar{C} + BCD$

Solution:

		<i>CD</i>			
		00	01	11	10
<i>AB</i>	00	0	0	0	1
	01	1	1	1	1
	11	1	1	1	0
	10	0	0	0	0

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

(b) $\bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D} + AB\bar{C}D + ABC\bar{D} + ABCD$

Solution:

		<i>CD</i>			
		00	01	11	10
<i>AB</i>	00	1	0	0	1
	01	0	1	1	0
	11	0	1	1	1
	10	1	0	1	0

$$BD + \bar{A}\bar{B}\bar{D} + \bar{B}\bar{C}\bar{D} + ABC + ACD$$

We made some progress, but the final answer is still gross. This is the sort of thing where a multiplexer or LUT really shines, because it can implement *any* 4-input truth table with the same logic. Trying to do this with individual gates would be frustrating.

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	0
(c) 0	1	1	X
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	X

Solution:

In this case, we include the X's because they allow us to draw larger boxes and eliminate several variables:

		<i>BC</i>			
		00	01	11	10
<i>A</i>	0	0	1	X	0
	1	1	1	X	1

$A + C$

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>Y</i>
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	X
0	1	1	0	0
(d) 0	1	1	1	0
1	0	0	0	0
1	0	0	1	X
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	X
1	1	1	0	1
1	1	1	1	1

Solution:

Here we include the X's that help draw larger boxes, and ignore the others.

		<i>CD</i>			
		00	01	11	10
<i>AB</i>	00	1	1	0	0
	01	1	X	0	0
	11	0	X	1	1
	10	0	X	0	1

$$\overline{A}\overline{C} + ABC + AC\overline{D}$$

	A	B	C	D	Y
	0	0	0	0	1
	0	0	0	1	0
	0	0	1	0	0
	0	0	1	1	X
	0	1	0	0	1
	0	1	0	1	1
	0	1	1	0	0
(e)	0	1	1	1	1
	1	0	0	0	X
	1	0	0	1	1
	1	0	1	0	0
	1	0	1	1	1
	1	1	0	0	0
	1	1	0	1	X
	1	1	1	0	1
	1	1	1	1	0

Solution:

		CD			
		00	01	11	10
AB	00	1	0	X	0
	01	1	1	1	0
	11	0	X	0	1
	10	X	1	1	0

$$\overline{A}\overline{C}\overline{D} + \overline{A}BD + A\overline{B}D + ABC\overline{D}$$

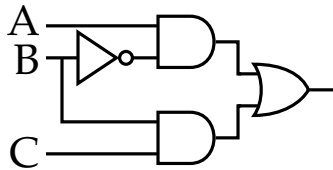
Interestingly, the X's don't allow any additional reduction of the equation!

Part 2: Gate implementation

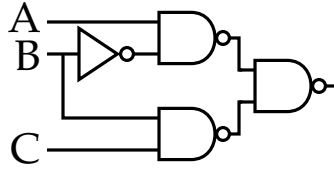
Implement the equation $Y = A\overline{B} + BC$ using only NAND gates (no inverters, just NANDs):

Solution:

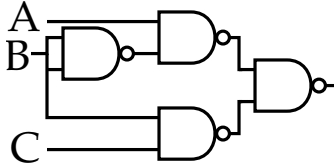
One way to accomplish this is with bubble pushing. Start by drawing the logic diagram as written:



Then we can use involution and bubble-pushing to switch the gates to NANDs:



Finally we need to generate \overline{B} , which we can do by using B as both inputs to a NAND gate:



Remember, the whole point of this is that inverting logic (NAND/NOR/NOT) is faster (lower propagation delay), smaller (fewer transistors) and uses less power than non-inverting logic. This actually has real utility when you're designing logic at the lowest level.

Part 3: Multiplexers

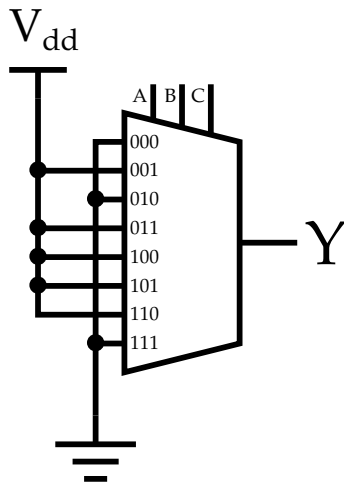
- (a) Implement the equation $Y = A\overline{C} + \overline{A}C + \overline{B}C$ using an 8:1 multiplexer.

Solution:

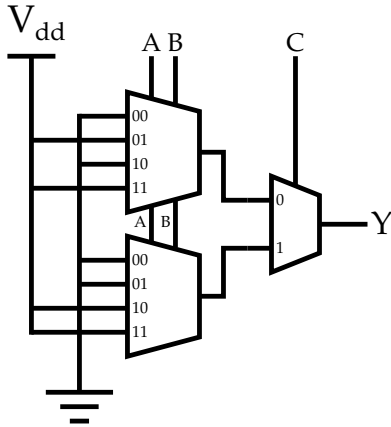
This equation isn't in canonical form, so each term represents more than one row of the truth table (and therefore more than one input to the mux.) Start by writing out the truth table.

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

From here, the multiplexer implementation is straightforward:



- (b) Write a truth table for this circuit, showing the output Y as a function of the inputs A , B , and C .



Solution:

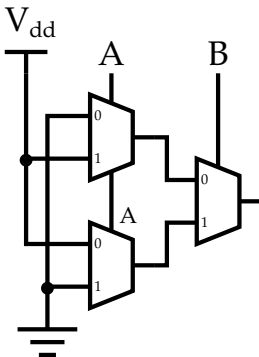
Be careful of the ordering of A , B , and C ! If you draw them in the conventional order (as below), note that you'll need to read alternately from the top and bottom 4:1 multiplexers.

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

- (c) Draw a logic diagram showing how you could implement the equation $Y = A \oplus B$ using only 2:1 multiplexers. *Hint: It might help to first draw how you could implement this with a 4:1 multiplexer.*

Solution:

One solution is to just build a 4:1 mux with three 2:1 multiplexers, and wire up the inputs to match the truth table of an XOR:



There's a way to implement $Y = A \oplus B$ with only *two* 2:1 multiplexers (and no other parts) — the solution may require some imagination and persistence, so it's ok to let the problem rattle around in your head a while!

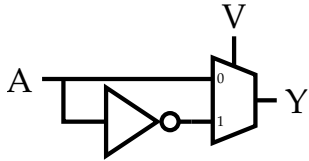
- (d) Draw a logic diagram for a circuit with the following behavior:

- When V is low, the output signal is simply the input: $Y = A$.
- When V is high, the output signal is inverted : $Y = \bar{A}$.

Write a truth table for Y in terms of A and V . What do you notice?

Solution:

Here is one possible logic implementation:



And here's the truth table:

V	A	Y
0	0	0
0	1	1
1	0	1
1	1	0

If you look at this for a while, you might notice that this is just the truth table for an XOR gate!

This is actually a helpful and common way to think about XOR: it allows one signal to control whether another is inverted or not.

Practice Problems - For review

These are selected problems from the textbook (at the end of each chapter) which may be helpful for practice and review. The answers to these problems are online at <https://booksite.elsevier.com/9780128000564/solutions.php>.

- 2.17 (simplifying equations, drawing circuits)
- 2.27 (bubble pushing)
- 2.31 (minimizing with don't-cares)
- 2.35 (writing and minimizing equations, drawing circuits)