CSE 260M - Homework 2

Always show all work for full credit.

1.1. Write a Boolean equation in sum-of-products canonical form for:

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

1.2. Write a Boolean equation in sum-of-products canonical form for:

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

1.3. Write a Boolean equation in sum-of-products canonical form for:

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

CSE 260M - Homework 2

1.4. Write a Boolean equation in sum-of-products canonical form for:

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

2.1 Determine the minimal expression minimal (in terms of AND operations / the Karnaugh map sense) for the table in 1.1.

2.2 Determine the minimal expression minimal (in terms of AND operations / the Karnaugh map sense) for the table in 1.2.

2.3 Determine the minimal expression minimal (in terms of AND operations / the Karnaugh map sense) for the table in 1.3.

3. Implement the below with only NAND gates (it's the same function in 1.2 and 2.2). Sketch the circuit (Or use JLS to create the circuit diagram)

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

CSE 260M - Homework 2

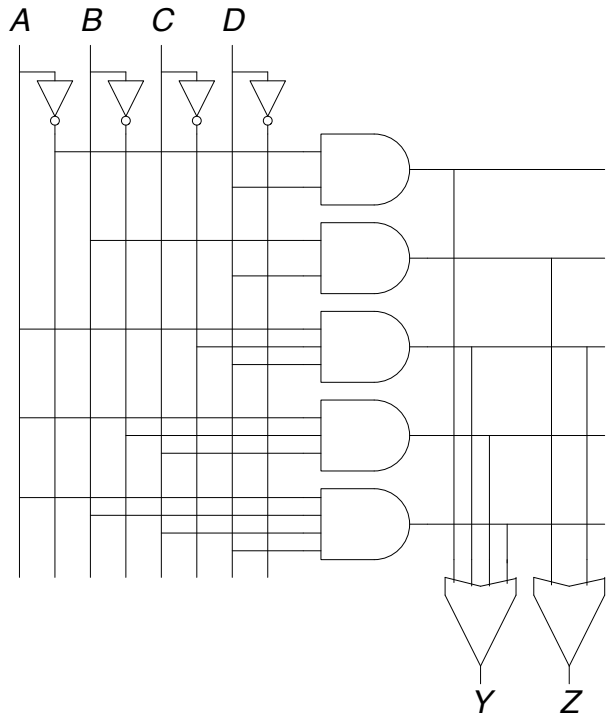
4.1 Determine a minimal (in terms of AND operations / the Karnaugh map sense) form of:

$$Y = BC + \bar{A}\bar{B}\bar{C} + B\bar{C}$$

4.2 Determine a minimal (in terms of AND operations / the Karnaugh map sense) form of:

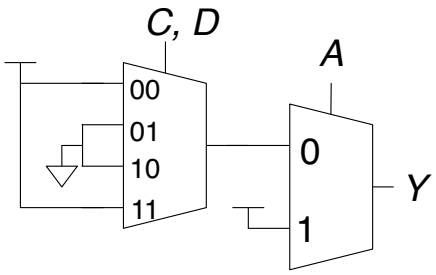
$$Y = \overline{A + \bar{A}B + \bar{A}\bar{B}} + \overline{A + \bar{B}}$$

5. Write the Boolean equation for the circuit below (no need to minimize: give the exact equation implemented in the circuit):



CSE 260M - Homework 2

6. Give the minimized (minimal AND / Karnaugh Map sense) Boolean equation for the function performed by (note that the "T" in the upper left and connecting to the bottom input of the right-most MUX represent a voltage source (that is, a logic 1) and the triangle represents a ground (that is, a logic 0):

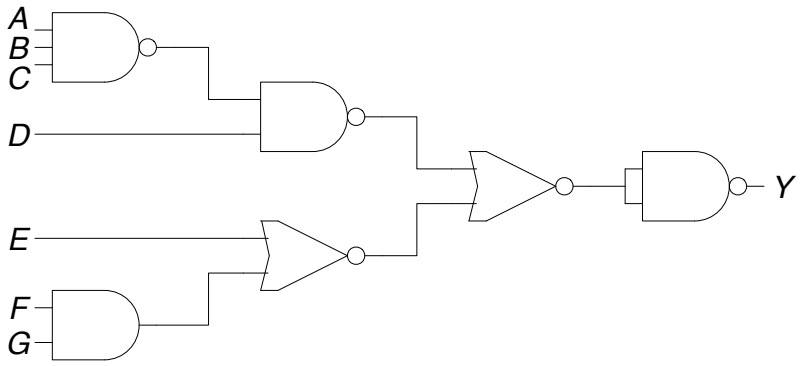


7. Use an 8-to-1 Multiplexor to implement the same function from the previous problem.

8. Assuming the propagation delays are:

Gate	t_{pd} (ps)
NOT	15
2-input NAND	20
3-input NAND	30
2-input NOR	30
3-input NOR	45
2-input AND	30
3-input AND	40
2-input OR	40
3-input OR 55	55
2-input XOR	60

Determine the propagation delay of:



CSE 260M - Homework 2

9.1 Consider adding 2-digit binary numbers, A and B, to produce the 2-bit result, S, and a carry-out, C:

$$\begin{array}{cccc} & A_1 & A_0 \\ + & B_1 & B_0 \\ C & S_1 & S_0 \end{array}$$

Develop the truth table and sum-of-products equations for adding the two, 2-bit numbers and produce the 3-bits of the result.

9.2 Download the JLS starter file from the assignment page and complete the circuit. Note that it uses 2-bit inputs for A and B. You test all possible combinations of inputs to confirm your 2-bit adder works correctly. Hints:

- Think carefully about the structure of the truth table and the terms being used for each output.
- Some product terms can be reused for different outputs.
- JLS has parts called "named wires". Named wires can be used to organize your work and re-use common terms. Named wires have two parts: the source ("name a wire" is shown when you hover over the named wire part on the menu) and connections ("connect to a named wire"). You can use more than one of the "connect to a named wire" part to have a single wire connect multiple places without having a literal wire (line) running across the diagram. If you use named wires, try to name them with a name that describes their purpose.
- Some wires and parts have already been provided. They are intended to provide one tidy approach to laying out the circuit.
- This circuit uses multi-bit inputs and separates them into their individual bits (A0, A1, etc.) for you. Test signal values can be specified in decimal rather than binary. For example, A can be 0, 1, 2, or 3. When A is 2 its individual bits, A1 and A0, are 1 and 0 respectively. The bits of S are combined into a single, 2-bit output, which doesn't include C.
- Carefully think about test cases. A signal generator has already been provided.
- DO NOT change the names of either input or output ports. The Autograder depends on them being named as-is.