EECS 151/251A Homework 3

Due Sunday, February 11th, 2018

Problem 1: Boolean Identities

(a) De Morgan's laws are useful in simplifying some boolean expressions; they are given as follows:

$$\overline{A \cdot B} \equiv \overline{A} + \overline{B}$$
$$\overline{A + B} \equiv \overline{A} \cdot \overline{B}$$

Prove these laws are true by equating truth tables derived from either side of the law.

Law 1:

	A=0	A=1
B=0	1	1
B=1	1	0

Law 2:

	A=0	A=1
B=0	1	0
B=1	0	0

(b) Use De Morgan's laws to describe how to construct:

(a) an AND gate from a NOR gate and inverters

(b) an OR gate from a NAND gate and inverters

(Hint: Invert both sides of each De Morgan law).

AND Gate: $A \cdot B = \overline{\overline{A} + \overline{B}} = \text{NOR}(\overline{A}, \overline{B})$

OR Gate: $A + B = \overline{\overline{A} \cdot \overline{B}} = \text{NAND}(\overline{A}, \overline{B})$

(c) Construct a NOR gate in terms of NAND gates and inverters, and construct a NAND gate in terms of NOR gates and inverters

$$NOR(A, B) = \overline{A + B} = \overline{A} \cdot \overline{B}$$
$$\overline{NOR(A, B)} = \overline{\overline{A} \cdot \overline{B}}$$
$$\overline{NOR(A, B)} = NAND(\overline{A}, \overline{B})$$
$$NOR(A, B) = \overline{NAND(\overline{A}, \overline{B})}$$

$$\begin{aligned} NAND(A,B) &= \overline{A \cdot B} = \overline{A} + \overline{B} \\ \overline{NAND(A,B)} &= \overline{\overline{A} + \overline{B}} \\ \overline{NAND(A,B)} &= NOR(\overline{A}, \overline{B}) \\ NAND(A,B) &= \overline{NOR(\overline{A}, \overline{B})} \end{aligned}$$

(d) Manipulate the XOR boolean expression $XOR(A,B) = (A+B) \cdot (\overline{A} + \overline{B})$ to construct it using only NAND gates and inverters.

$$\begin{aligned} XOR(A,B) &= (A+B) \cdot (\overline{A} + \overline{B}) \\ XOR(A,B) &= NOR(\overline{NAND(\overline{A}, \overline{B})}, \overline{NAND(A,B)}) \\ XOR(A,B) &= \overline{NAND(NAND(\overline{A}, \overline{B}), NAND(A,B))} \end{aligned}$$

Problem 2: Logic Simplification

Take this truth table consisting of 4 input variables and 1 output.

A	B	C	D	Out
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	x
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

where 'x' means don't care.

(a) Write a sum of products boolean function directly from the truth table

$$Out = \overline{A}\cdot\overline{B}\cdot\overline{C}\cdot\overline{D} + \overline{A}\cdot\overline{B}\cdot C\cdot\overline{D} + \overline{A}\cdot\overline{B}\cdot C\cdot D + \overline{A}\cdot B\cdot C\cdot\overline{D} + \overline{A}\cdot B\cdot C\cdot D + A\cdot\overline{B}\cdot\overline{C}\cdot\overline{D}$$

You could omit the second to last term, $\overline{A}\cdot B\cdot C\cdot D$, if you didn't include the don't-care value. You can also simplify this equation in a straightforward manner by combining common terms until you reach this form:

$$Out = \overline{B}\cdot\overline{D}\cdot\overline{A} + \overline{B}\cdot\overline{D}\cdot\overline{C} + \overline{A}\cdot C$$
$$Out = \overline{B}\cdot\overline{D}(\overline{A}+\overline{C}) + \overline{A}\cdot C$$

At this point, you will need to make an observation that $\overline{A}$ can be dropped from the first term to obtain the simplified form.

$$Out = \overline{A}\cdot C + \overline{B}\cdot\overline{C}\cdot\overline{D}$$

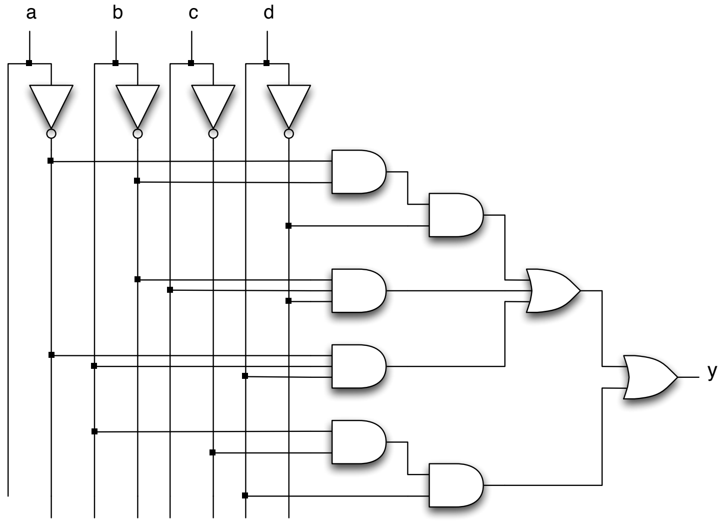
(b) Use a 4 variable Karnaugh Map to derive a simplified boolean function from this truth table.

	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	1	x	0	0
10	1	1	0	0

$$Out = \overline{A}\cdot C + \overline{B}\cdot\overline{C}\cdot\overline{D}$$

Problem 3: Representations of Combinational Logic

For the following circuit,



1. Write a Boolean equation that represents the function of the circuit.

$$y = \overline{a} \cdot \overline{b} \cdot \overline{d} + \overline{b} \cdot c \cdot \overline{d} + \overline{a} \cdot b \cdot d + b \cdot \overline{c} \cdot d$$

2. Draw and fill in the truth table.

a	b	c	d	y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

3. Write the sum-of-products canonical form for this function.

$$y = \overline{a} \cdot \overline{b} \cdot \overline{c} \cdot \overline{d} + \overline{a} \cdot \overline{b} \cdot c \cdot \overline{d} + \overline{a} \cdot b \cdot \overline{c} \cdot d + \overline{a} \cdot b \cdot c \cdot d + a \cdot \overline{b} \cdot c \cdot \overline{d} + a \cdot b \cdot \overline{c} \cdot d$$

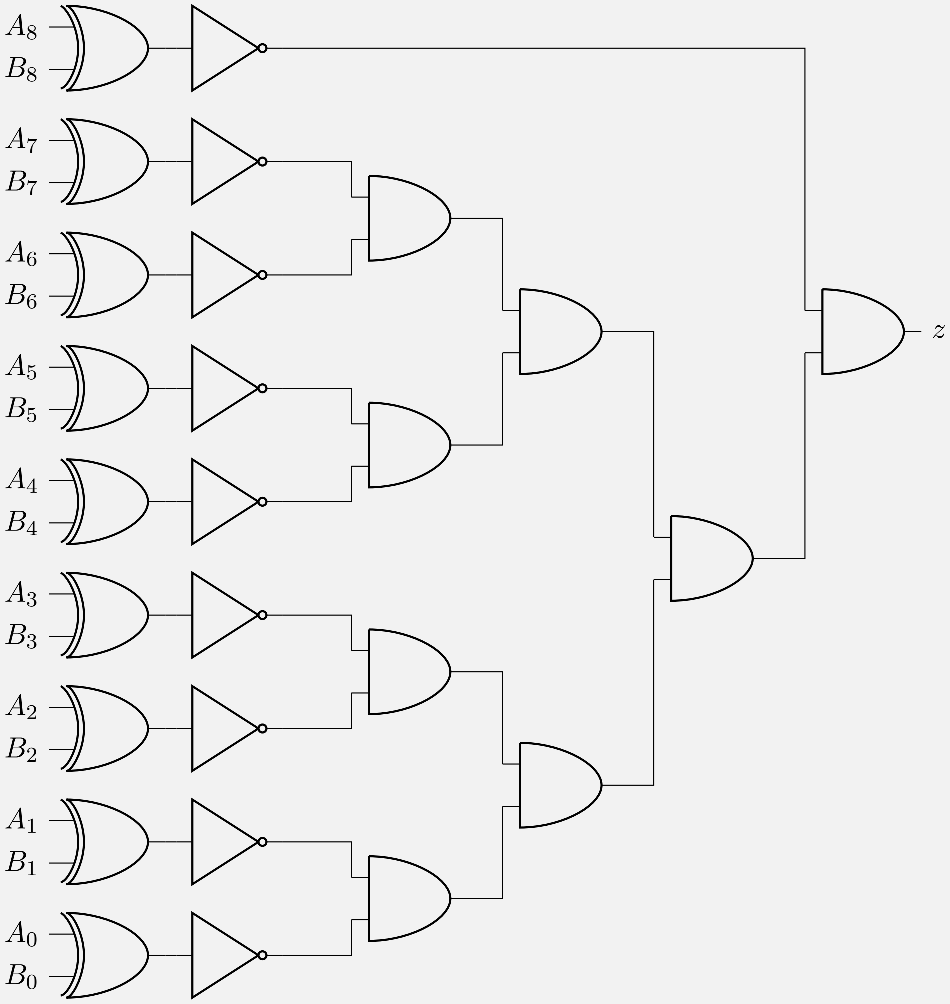
Problem 4: Combinational Logic

Consider the design of a combinational logic circuit that compares two 9-bit integers for equivalence (an "equal comparator"). The circuit takes 2 9-bit integers A, and B, and outputs 1 on **z** iff A≡B.

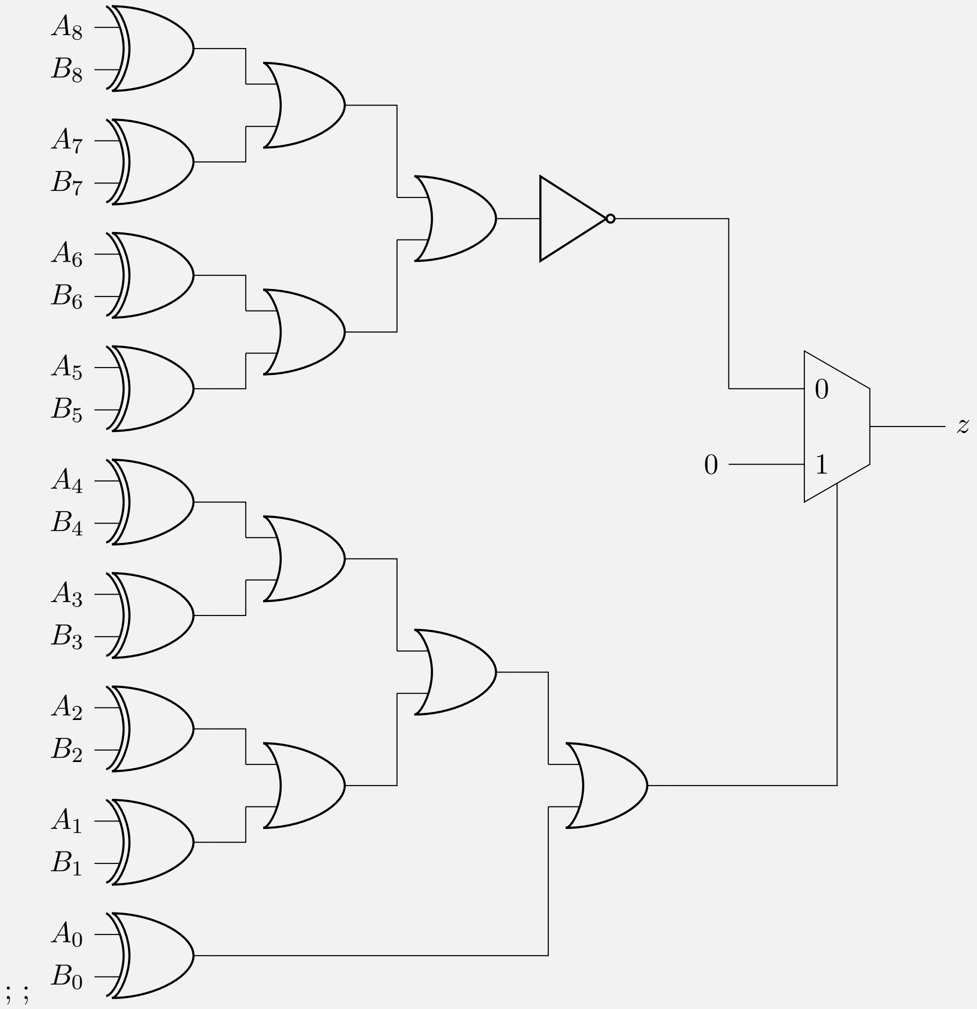
Your solution can include any combination of: inverters, 2-input AND gates, 2-input OR gates, 2 to 1 multiplexors, and 2-input exclusive-OR gates. Your goal is to minimize the delay through the circuit. You can assume that the delay through each of these components is the same.

In the space below neatly draw the circuit diagram for your equal comparator. Label all inputs and outputs.

The following circuit is one solution. You could also swap the XORs and their inverters with straight XOR gates, and the ANDs with ORs, then invert the output with a final inverter. That would take 8 fewer components! But since it won't change the delay through the circuit, it's equivalent.

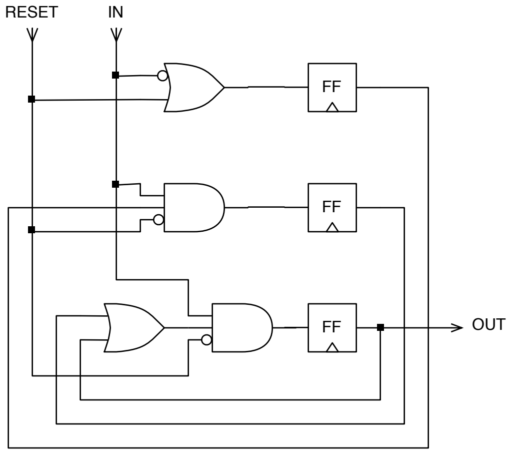


There is a way to do it with one fewer level. First, think about the naive implementation but with ORs, to isolate the use of the inverter. Then observe that this inverter, which you will inevitably use, can be put in a path parallel with the OR tree (or part thereof). You can then use the multiplexer to select the output of the operation on a subset of the bits, based on the input from the operation on a different set. Something like this:

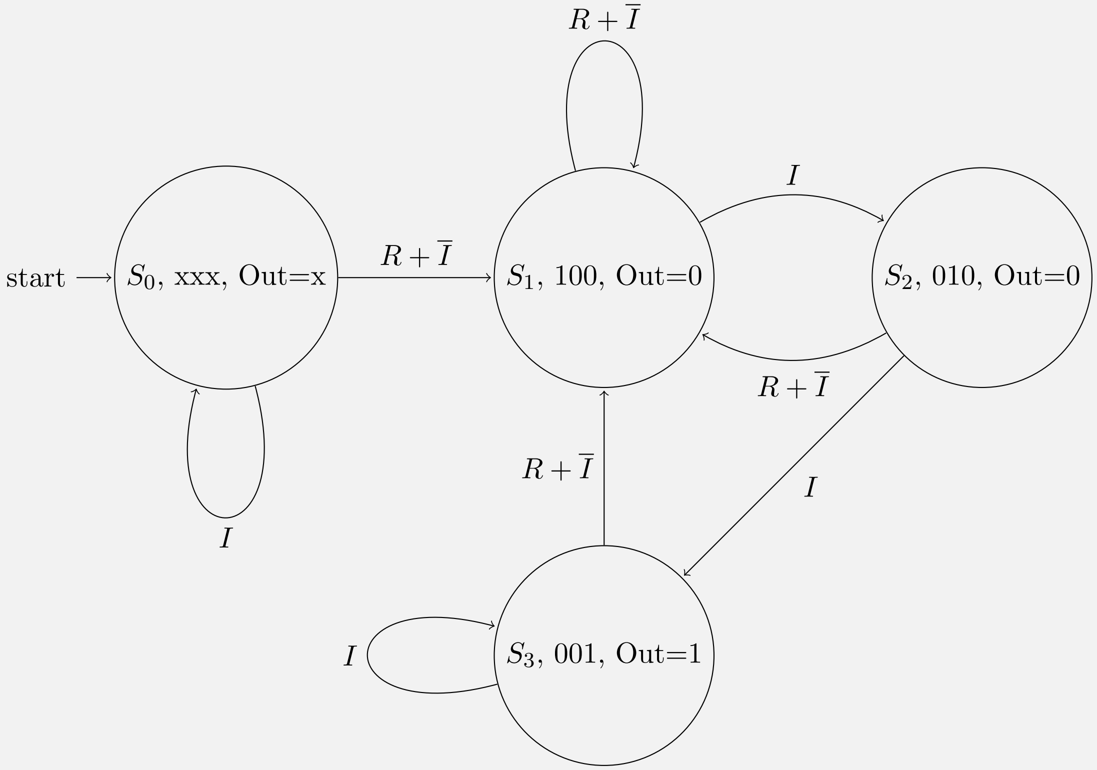


Problem 5: Finite State Machine

Draw the *state transition diagram* for the finite state machine circuit shown below.



The starting state is unknown, it is included for completeness. Only states S_1 - S_3 are reachable. Inputs have been abbreviated: I = IN, R = RESE.



Optional: What is the maximum *clock frequency* for this circuit? Assume IN comes from the OUT of an identical copy of this circuit, all logic gates have a delay of 1ns and the flip-flop setup time and clock-to-q delay are both 0.5ns.

The critical path is from the lowest flip-flop back to its own input (through two gates). The path through OUT to IN only traverses one gate. Hence

$$\begin{aligned} T_{crit} &= T_{clk-q} + 2 \times T_{pd} + T_{setup} \\ &= 0.5 + 2 \times 1 + 0.5 \\ &= 3 \text{ ns} \end{aligned}$$

Thus the maximum frequency is 333 MHz (3 s.f.).