EECS 151/251A Homework 4

Due Monday, Feb 25th, 2019

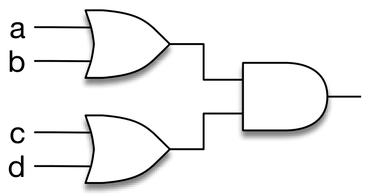
Problem 1: Correct JohnW's lecture mistake! [5 pts]

For the "BCD Incrementer Example" presented in lecture 6, derive the minimized logic equations for the outputs, expressed in sum-of-products (SOP) form.

Problem 2: Bubble Pushing [14 pts]

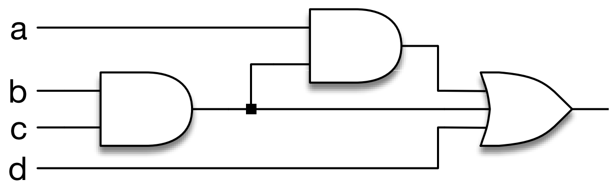
Two-level OR/AND circuits can be implemented as two-level NOR/NOR circuits.

(a) For the example below, using the "bubble pushing" technique, show the series of steps that you would take to convert the circuit to use only NORs. [4pts]



(b) Now convert the OR/AND circuit to use only NANDs (and possibly inverters). [5pts]

(c) Convert the following circuit to all NANDs (and possibly inverters). [5pts]



Problem 3: SOP, POS, and Minimum Expressions [9pts]

Consider the function f defined with the truth-table below:

x_0	x_1	x_2	x_3	f
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
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	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

(a) Write out the sum-of-products (SOP) and product-of-sums (POS) canonical forms. [3pts]

(b) Derive the minimized expressions for f and for f' in SOP and POS forms (four answers required here). Hint: use a K-map. [6pts]

Problem 4: Simplification with Boolean Algebra [6pts]

For the following function g defined with the truth-table below, use algebraic manipulation to derive its minimized form.

x_0	x_1	x_2	g
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Problem 5: Building a Better Saturating Incrementer [10pts]

Consider the design of a 3-bit binary unsigned incrementer with saturating output. Saturating in this case means that if the input is the maximum representable value (in this case 111_2), then it is passed through unchanged. In Verilog we might try defining the incrementer with:

```
assign y = (x == 3'b111) ? 3'b111 : x + 1;
```

A naive logic synthesizer might turn this into a comparitor, a multiplexor, and an adder. Your job is to derive a simpler circuit by hand.

Show your derivation of logic equations for the 3-output bits, y[0], y[1], y[2].

Problem 6: Writing a FSM in Verilog [8pts]

Write the behavior Verilog description for the Combinational Lock FSM example presented in lecture 7.

Problem 7: Designing an FSM [11pts]

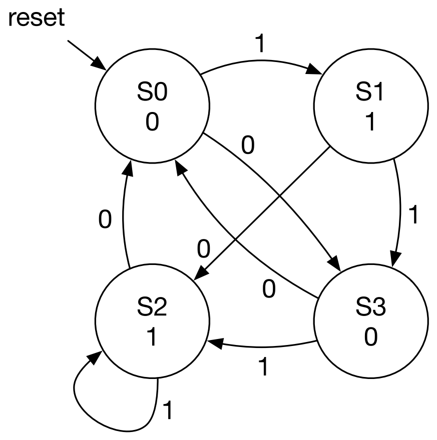
Consider the design of a Moore-style FSM with a 1-bit input (x), and a 1-bit output (y). The FSM accepts inputs one bit at a time and outputs a 0 until it sees the sequence 0101 (with no other bit values in between). After seeing the second 1 in the sequence it outputs a 1 and then starts over.

(a) Draw the state transition diagram. Label all nodes and arcs. [3pts]

(b) Derive the next state logic and the output logic equations. [8pts]

Problem 8: Implementing an FSM [13pts]

Consider the state transition diagram shown below:

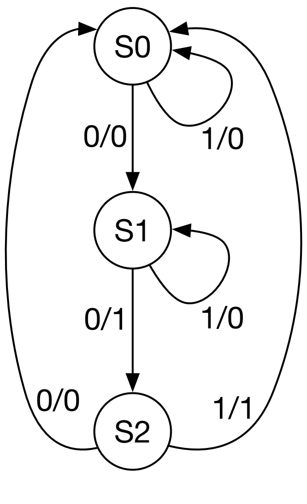


(a) Write the Verilog behavior description for the corresponding circuit. [5pts]

(b) Draw the circuit diagram for one-hot encoded implementation. (You may assume that flip-flops have both "set" and "reset" inputs, but you must label which one you would use for each flip-flop.) [8pts]

Problem 9: Converting from Mealy to Moore Style [6pts]

Convert the Mealy-style state transition diagram to a Moore-style.


S0
0/0
1/0
S1
0/1
1/0
0/0
1/1
S2