CE3320 Digital Circuits
HW 7
Registers and FSM
Due: Tuesday 10/24/2023

10 points

3.39 Using the process for designing a controller, convert the FSM of Figure 3.109 to a controller, implementing the controller using a state register and logic gates.

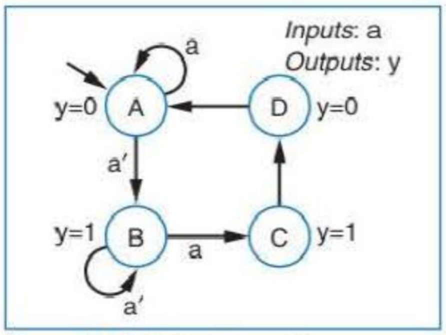

Figure 3.109 FSM example.

3.40 Using the process for designing a controller, convert the FSM of Figure 3.110 to a controller, implementing the controller using a state register and logic gates.

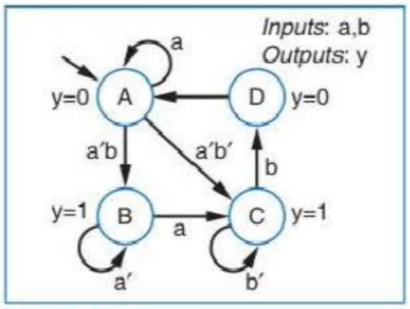

Figure 3.110 FSM example.

3.41 Using the process for designing a controller, convert the FSM you created for Exercise 3.24 to a controller, implementing the controller using a state register and logic gates.

3.43 Using the process for designing a controller, convert the FSM you created for Exercise 3.30 to a controller, implementing the controller using a state register and logic gates.

3.43b Using the process for designing a controller, convert the FSM we created for “digital lock” to a controller, implementing the controller using a state register and logic gates.

3.43c Using the process for designing a controller, convert the FSM we created for “road work sign” to a controller, implementing the controller using a state register and logic gates.

3.44 Using the process for designing a controller, convert the FSM in Figure 3.111 to a controller, stopping once you have created the truth table. Note: your truth table will be quite large, having 32 rows -- you might therefore want to use a computer tool, like a word processor or spreadsheet, to draw the table.

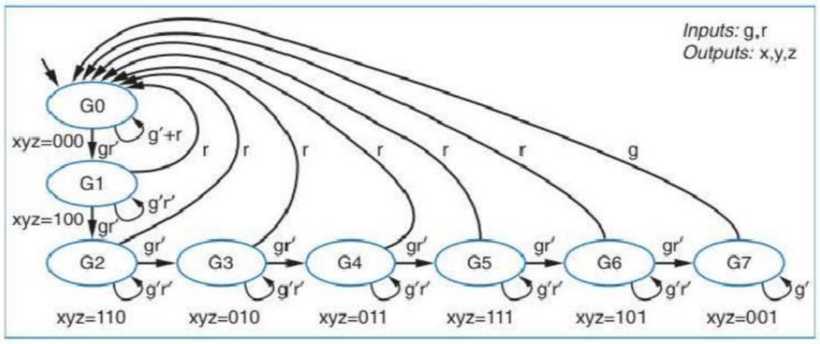

Figure 3.111 FSM example.

3.45 Create an FSM that has an input X and an output Y. Whenever X changes from 0 to 1, Y should become 1 for five clock cycles and then return to 0 -- even if X is still 1. Using the process for designing a controller, convert the FSM to a controller, stopping once you have created the truth table.

3.46 The FSM in Figure 3.112 has two problems: one state has non-exclusive transitions, and another state has incomplete transitions. By ORing and ANDing the conditions for each state’s transitions, prove that these problems exist. Then, fix these problems by refining the FSM, taking your best guess as to what was the FSM creator’s intent.

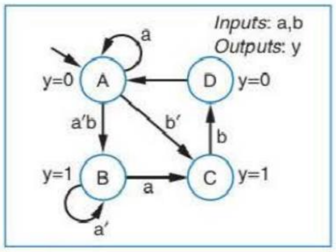

Figure 3.112 FSM example.

3.48 Reverse engineer the behavior of the sequential circuit shown in Figure 3.113.

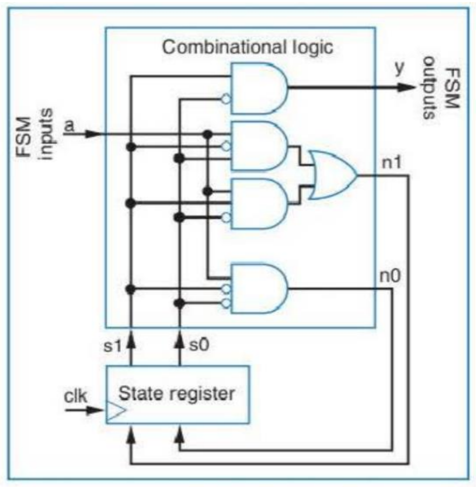

Figure 3.113 A sequential circuit to be reverse engineered.

3.48b Reverse engineer the behavior of the sequential circuit shown in Figure 3.113b.

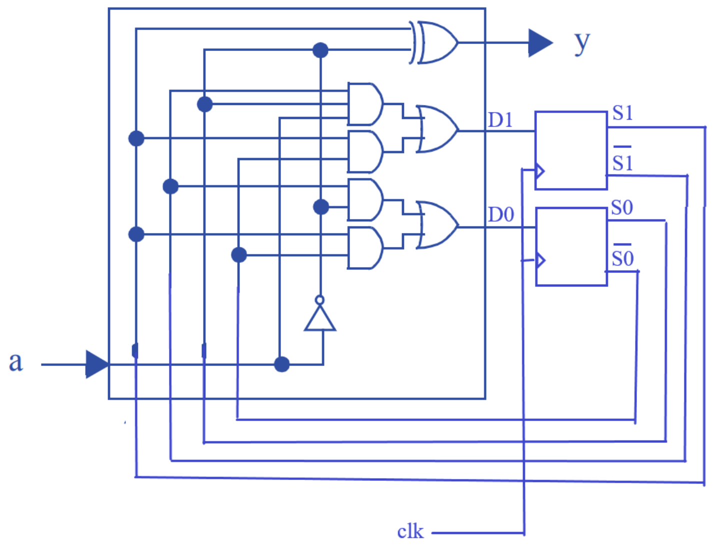

Figure 3.113b A sequential circuit to be reversed engineered

3.50 Design a controller with a 4-bit state register that gets synchronously initialized to state 1010 when an input reset is set to 1.

3.52 Draw a timing diagram for three clock cycles of the sequence generator controller of Figure 3.68 assuming that AND gates have a delay of 2 ns and inverters (including inversion bubbles) have a delay of 1 ns. The timing diagram should show the incorrect outputs that appear temporarily due to glitching. Then, introduce registered outputs to the controller using flip-flops at the outputs, and show a new timing diagram, which should no longer have glitches (but the output may be shifted in time). Let’s assume the delay of an XOR gate is the same as for an AND gate. Unregistered Output: