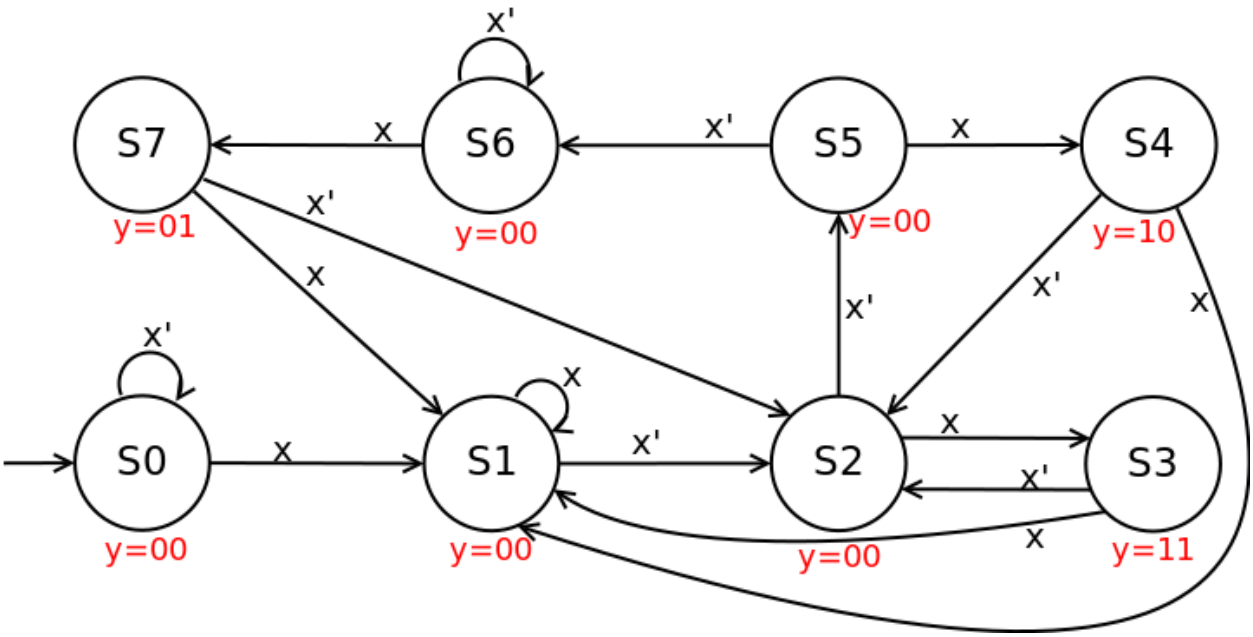
Student Names: Cemal Burak AYGÜN , Yaşar Alim TÜRKMEN
Student IDs: 2014400072 , 2014400165
Group ID: 7
Session: 1

CMPE 240 Experiment 3 Preliminary Work

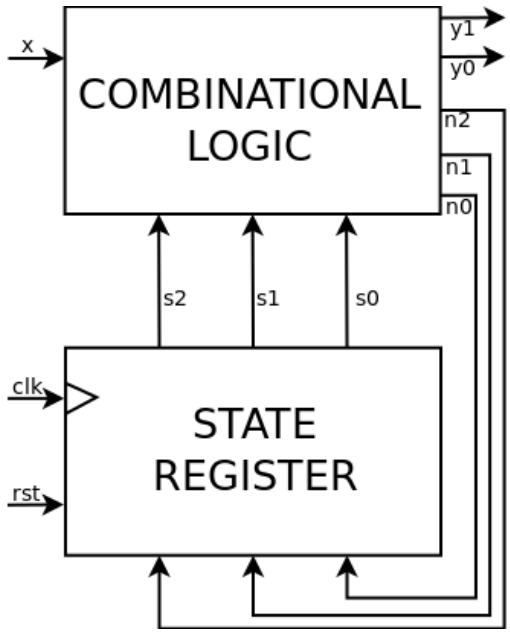
Step 1: Capture the FSM: Create and draw the finite state machine that describes the desired behavior of the controller.



NOTE: There is a reset which is not shown in the FSM above. Whenever reset is HIGH, the state becomes initial state which is S0.(Reset is synchronous to the rising edge of the clock.)

Student Names: Cemal Burak AYGÜN , Yaşar Alim TÜRKMEN
Student IDs: 2014400072 , 2014400165
Group ID: 7
Session: 1

Step 2: Create the architecture: Create and draw standard architecture by a using state register of the appropriate width and combinational logic with inputs being the state register bits and the finite state machine inputs and outputs being the next state bits and the finite state machine engine.



NOTE: Whenever reset (rst) is HIGH (1), s2s1s0 becomes 000 (the initial state in the FSM is coded as 000) on the rising edge of the clock.

Step 3: Encode the states: Assign a unique binary number to each state. Each binary number representing a state is known as an encoding. Any encoding will do as long as each state has a unique encoding.

STATE NAME	ENCODING
S(0)	000
S(1)	001
S(2)	010
S(3)	011
S(4)	100
S(5)	101
S(6)	110
S(7)	111

Student Names: Cemal Burak AYGÜN , Yaşar Alim TÜRKMEN
Student IDs: 2014400072 , 2014400165
Group ID: 7
Session: 1

Step 4: Create the state table: Create a truth table for the combinational logic such that the logic will generate the correct FSM outputs and next state signals. Ordering the inputs with state bits first makes this truth table describe the state behavior, so the table is a state table.

CURRENT STATE (s2s1s0)	EXTERNAL INPUT (x)	NEXT STATE (n2n1n0)	OUTPUT (y1y0)
000	0	000	00
000	1	001	00
001	0	010	00
001	1	001	00
010	0	101	00
010	1	011	00
011	0	010	11
011	1	001	11
100	0	010	10
100	1	001	10
101	0	110	00
101	1	100	00
110	0	110	00
110	1	111	00
111	0	010	01
111	1	001	01

Step 5: Draw the combinational logic: Implement the combinatorial logic using any method (You do not need to draw the inside circuit of multiplexers or decoders if you are using any. You can show those as blocks).

y1 = s2’s1s0 + s2s1’s0’
y0 = s1s0
n2 = x’s1s0’ + s2s1’s0 + s2s1s0’
n1 = x’s0 + x’s2 + xs1s0’
n0 = xs2’ + xs0’ + xs1 + s2’s1s0’

Student Names: Cemal Burak AYGÜN , Yaşar Alim TÜRKMEN
Student IDs: 2014400072 , 2014400165
Group ID: 7
Session: 1