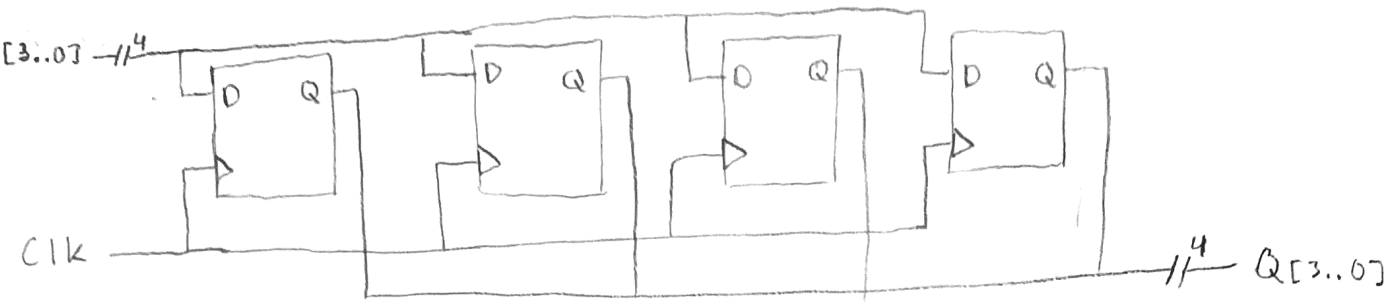
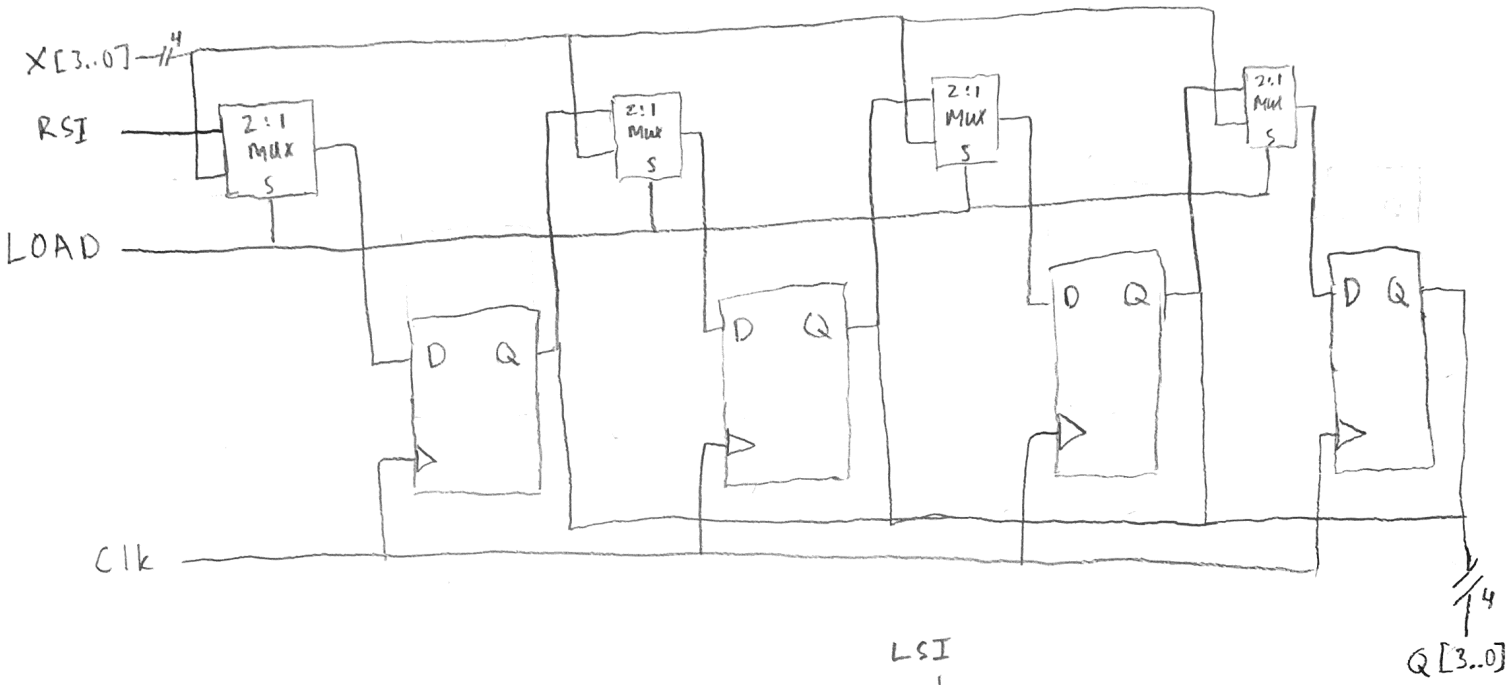
Pre-Lab 8

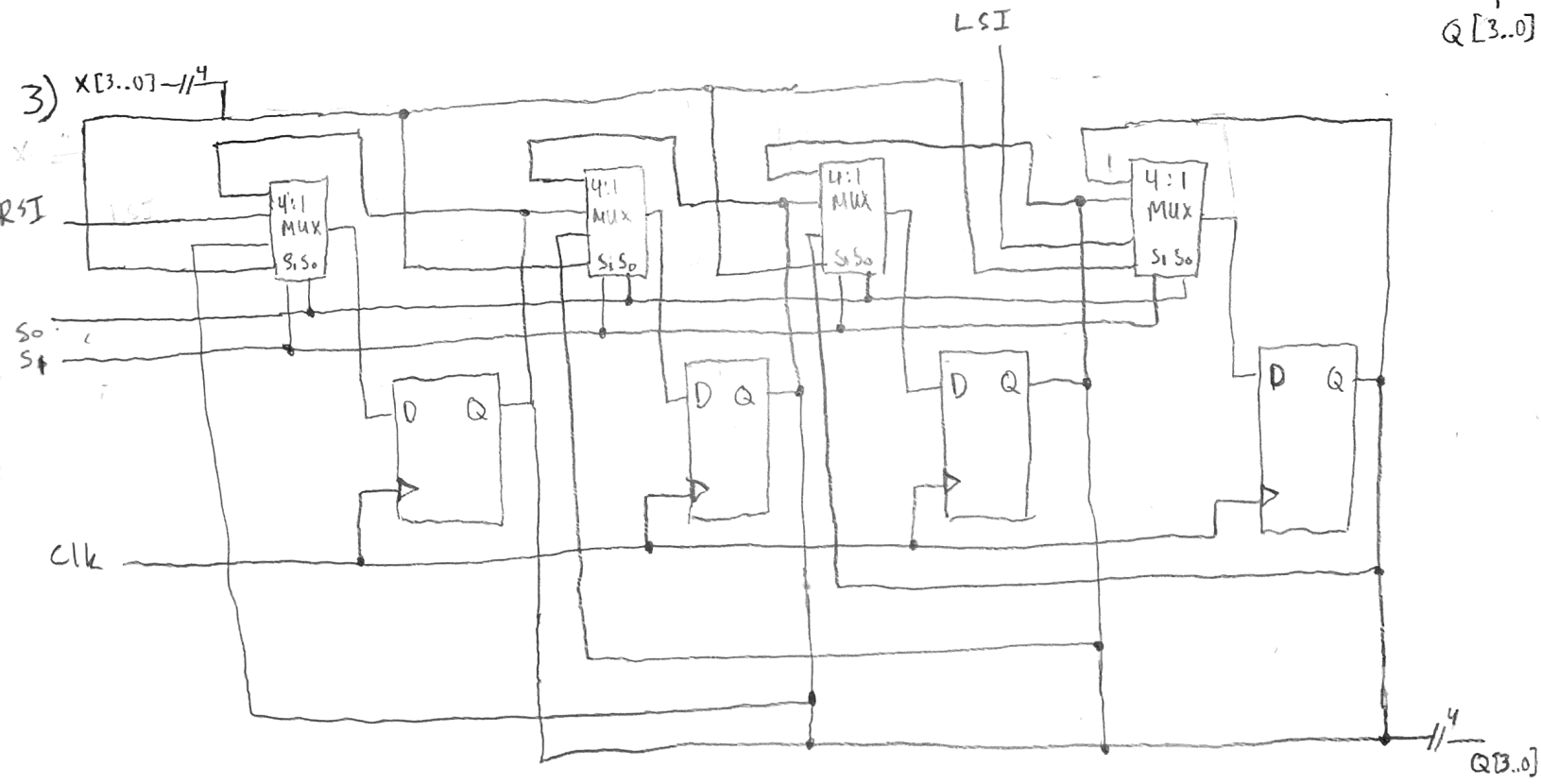
1)


X[3..0]
4
D Q
D Q
D Q
D Q
Clk
4
Q[3..0]

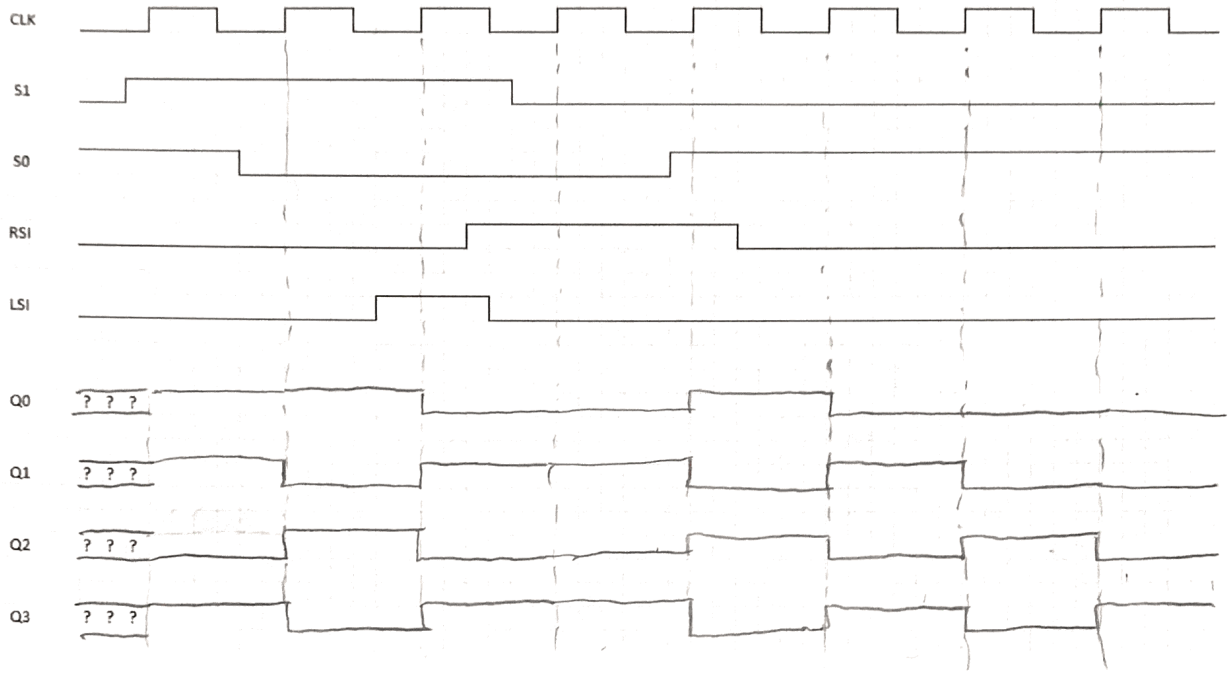
2)


X[3..0]
4
RSI
LOAD
2:1 MUX
S
Clk
4
Q[3..0]

3)


LSI
X[3..0]
4
RSI
4:1 MUX
S1 S0
S0
S1
D Q
Clk
4
Q[3..0]

(iv) Complete the timing diagram below for your universal shift register. Assume that X[2]=0 and all the other X inputs are 1. The X inputs do not change. Think of a systematic approach for testing your shift register, and write the first four steps of your test plan.



Make a state transition diagram:

(v) Draw the state transition diagram for the four-state *pong_controller1* from the state table in the controller definition section above.

(vi) State assignment: Choose binary values for each of your four states.

Submit your schematics, answers to questions, and timing diagram for the PreLab.

III. PROCEDURE

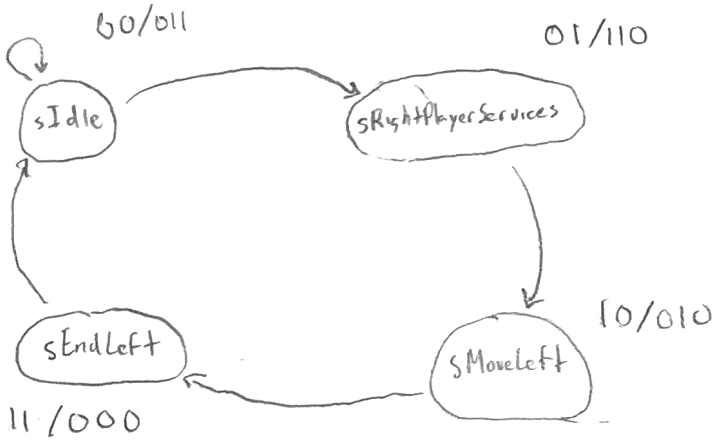
Part 1: Implement and Test a Verilog 4-bit Universal Data Shift Register, *USR4*.

1. Write a Verilog module, *USR4*, for your 4-bit universal shift register based on your pre-lab part (iii) schematic. The inputs will be X[3:0], RSI, LSI, S[1:0], and CLK. The output will be Q[3:0]. An if-else structure or a case statement can be used for the multiplexed inputs.
2. Add one instance of *USR4* to a Verilog testbench. Using the example of the testbench you used in the previous lab, instantiate a clock signal and connect it to the CLK input of your shift register.
3. Add the capability to your testbench for controlling the other inputs to shift register, and to observe the 4-bit Q output.
4. Check that the shift register operation is correct using a test plan. Try various inputs and ensure that the register is loading your data correctly. Make sure that the least significant bit of your input is also the least significant bit of the 4-bit output.

4) Testing Plan

Set the simulation up like the timing diagram from the prelab, since it covers all types of operations and loading. Check outputs to see if it matches the the Q[0..3] outputs on the timing diagram.

5)



State / LS1, S[1], S[0]

6)

State
sIdle → 00
sRightPlayerServes → 01
sMoveLeft → 10
sEndLeft → 11