ECE 5722 Homework 2

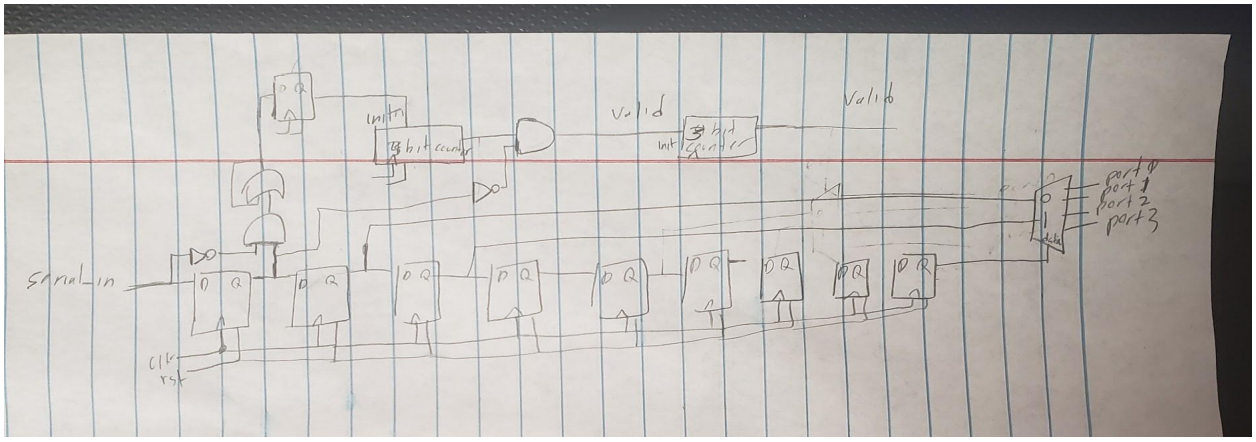
By Bruce Huynh

9/17/2021

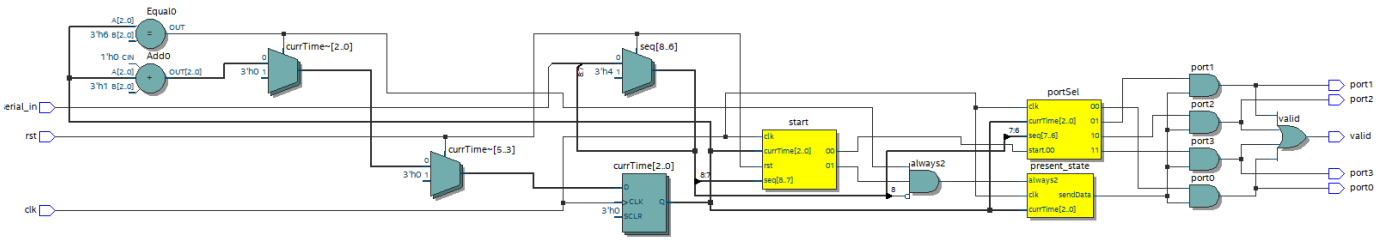
Introduction:

In this homework assignment, we were tasked with making a serial demultiplexer. This demultiplexer would start when the serial input goes from 1 to 0. The two bits after that determine the port that the serial data will exit out of. The serial data will get separated by 0's and a separation bit of 1 will terminate any port outputs.

A.) Schematic Datapath:

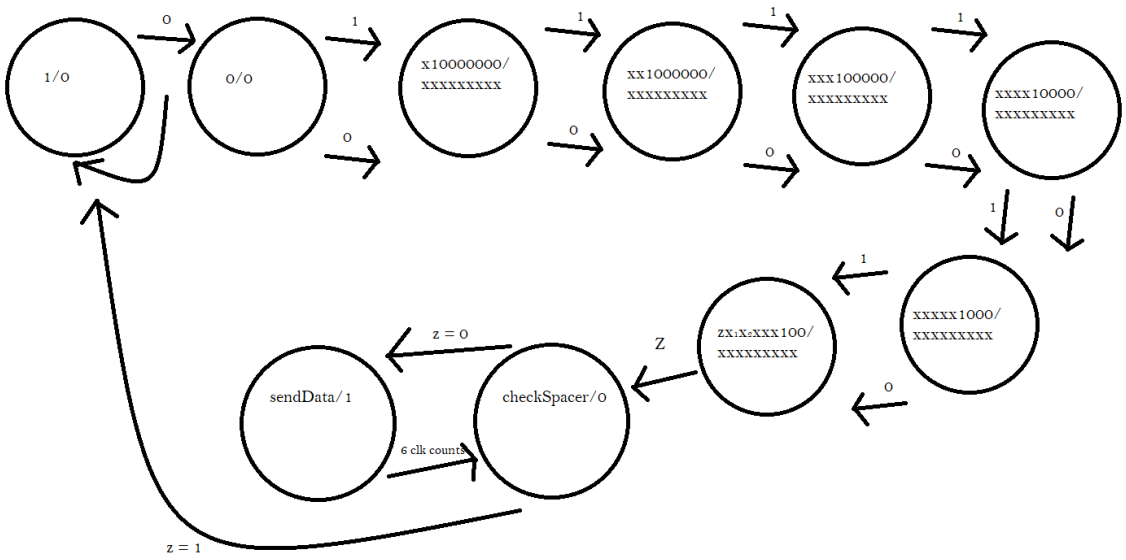


For the datapath of this assignment, shift registers are essential to its creation. Since our module is working with serial inputs, we need these registers to shift our information everytime a new input is received. Depending on the location of the register, certain actions will be performed with them like the first and second register being used to determine if the system starts.



Here is the datapath once simulated with the Verilog code.

B.) Controller diagram and code



For this state diagram, the main two states that matter are the starting states and the sending data states. The six states in the middle that contain the X's don't matter to us as they go to the same destination whether or not they produce a 1 or a zero. We only take a look at the serial input once six clock cycles have passed. If the first bit of the serial input is a 1, we send the signal back to the beginning to look for another falling edge to start receiving serial input. If the first bit of the serial input is a 1 at the end of six clock cycles, we look at the next two bits to see which port the data will be sent to and serial data from the serial input will be sent to that port for at least 6 clock cycles until the next port can be determined.

Starting code:

```
 always @ (posedge clk or posedge rst) begin
        seq <= seq >> 1;
        seq[8] <= serial_in;
        if (rst == 1'b1)
            seq <= 9'b100000000;
        else
                        if (seq[8] == 1 & currTime == 3'b110 & start ==
1)
                          start <= 0;
                else
```

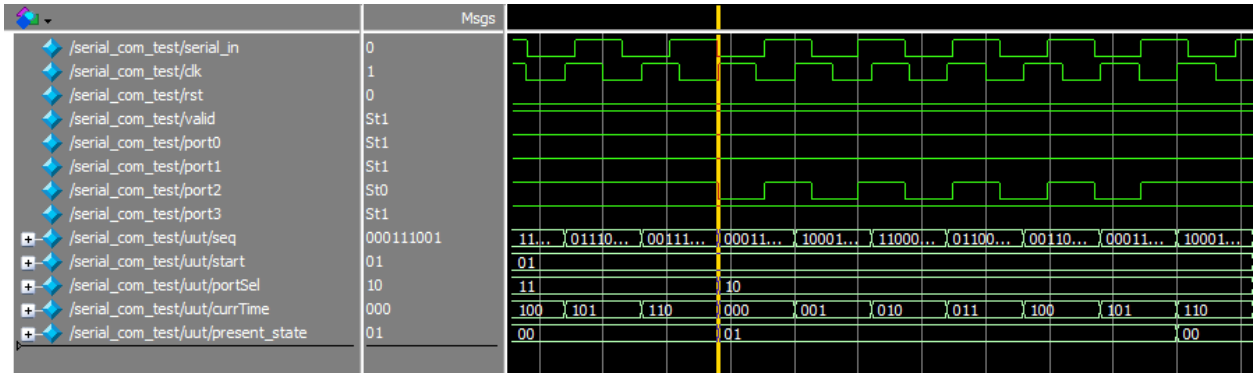
```
                    if (seq[8] == 1 & seq[7] == 0)
                                            start <= 1;
    end
```

Outputing for 6 clock cycles:

```
always @ (posedge clk) begin



          case (present_state)
               sendData: if (currTime == 3'b101) present_state <=
checkSpacer;
                                        else present_state <= sendData;
               checkSpacer: if (seq[8] == 0 & currTime == 3'b110 &
start == 1) present_state <= sendData;
                                            else present_state <=
checkSpacer;
          endcase


     end
```

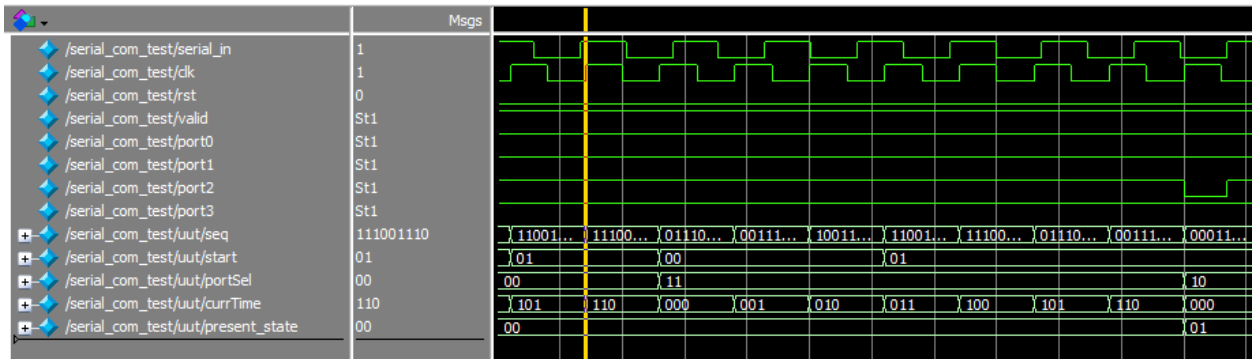
C.) Testbench

Test 1 - Fixed Pattern (serial input switches every 37 ps):



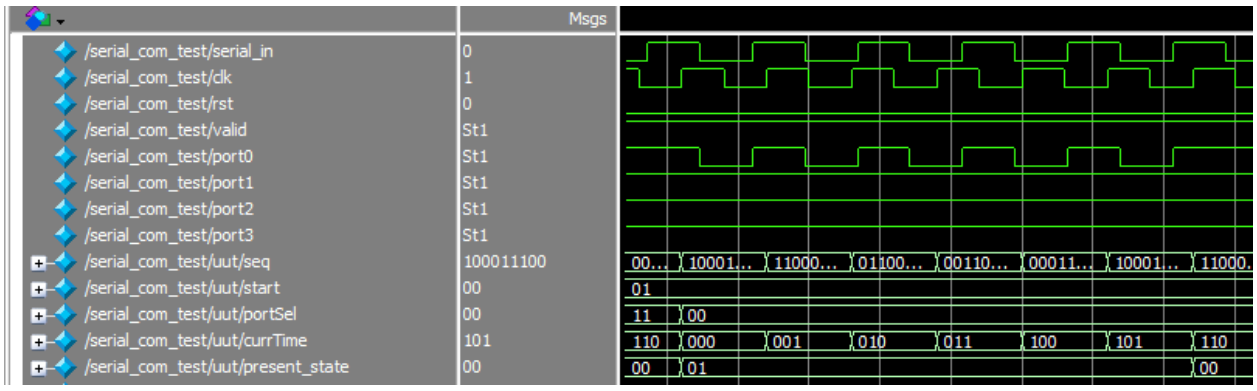
When the start signal is reached, the designated port will take output the serial information for at least six clock cycles. You can see above that portSel, the port selecting variable chooses port3 when its value is 11. The signal also stops transmitting afterwards because of the 1 in the 9th bit.

Test 2 - Having a 1 in the 9th bit:

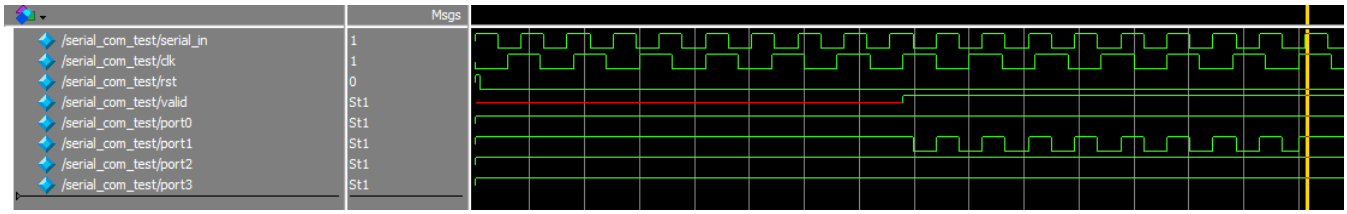


When there is a 1 in the 9th bit, the ports will not display any useful information (which in our case is a constant output of 1)

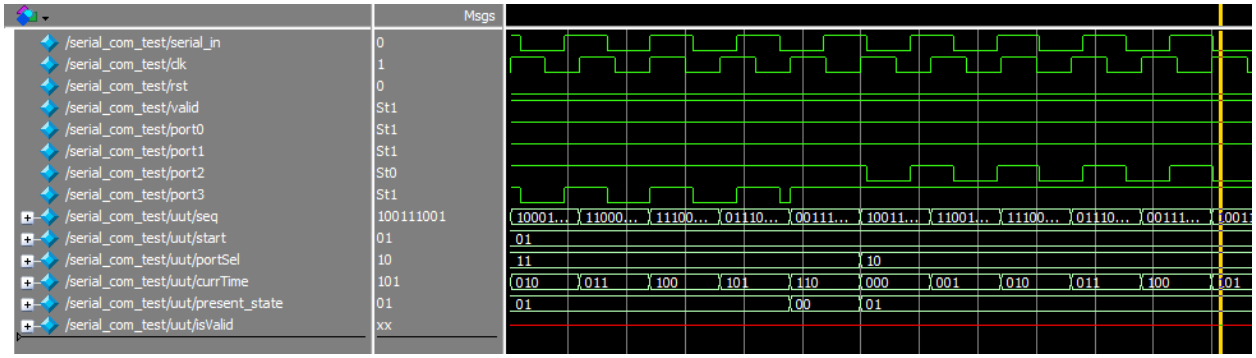
Test 4 - Testing demultiplexer



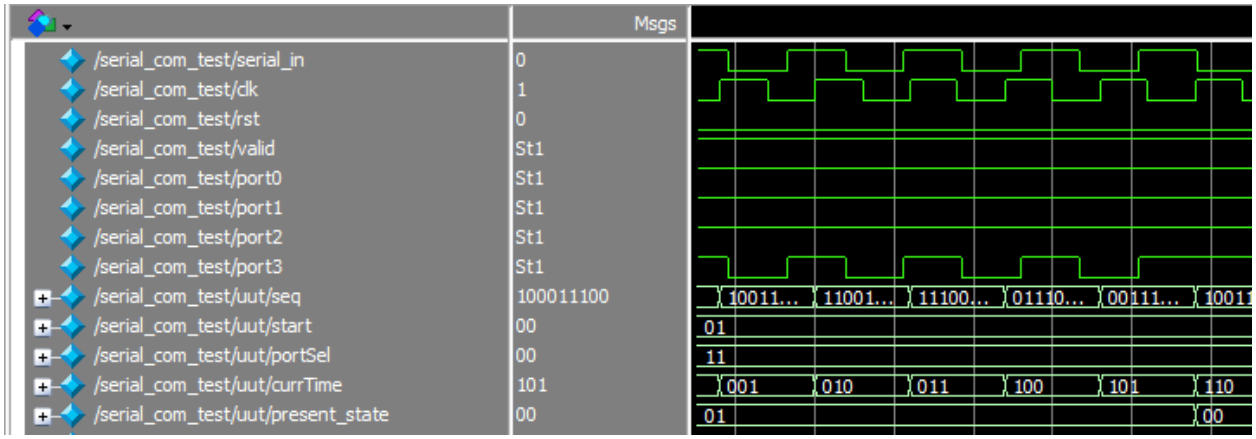
Selecting port0.



Selecting port1.

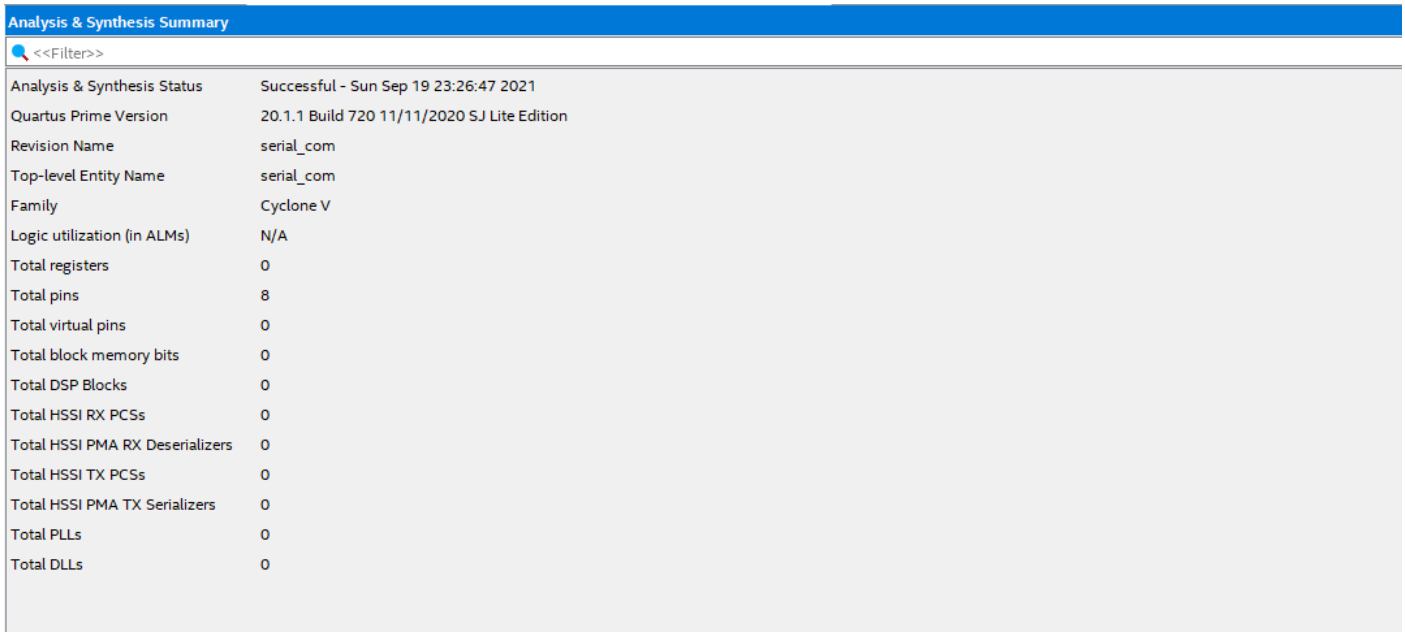


Selecting port2.



Selecting port3.

D.) Quartus synthesis



Analysis & Synthesis Summary

<<Filter>>

Analysis & Synthesis Status	Successful - Sun Sep 19 23:26:47 2021
Quartus Prime Version	20.1.1 Build 720 11/11/2020 SJ Lite Edition
Revision Name	serial_com
Top-level Entity Name	serial_com
Family	Cyclone V
Logic utilization (in ALMs)	N/A
Total registers	0
Total pins	8
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0

Conclusion:

In this homework assignment, we learned how to work with serial inputs and allowed us to have more diverse knowledge on state machines.