CS150 Discussion 3, Spring 2013

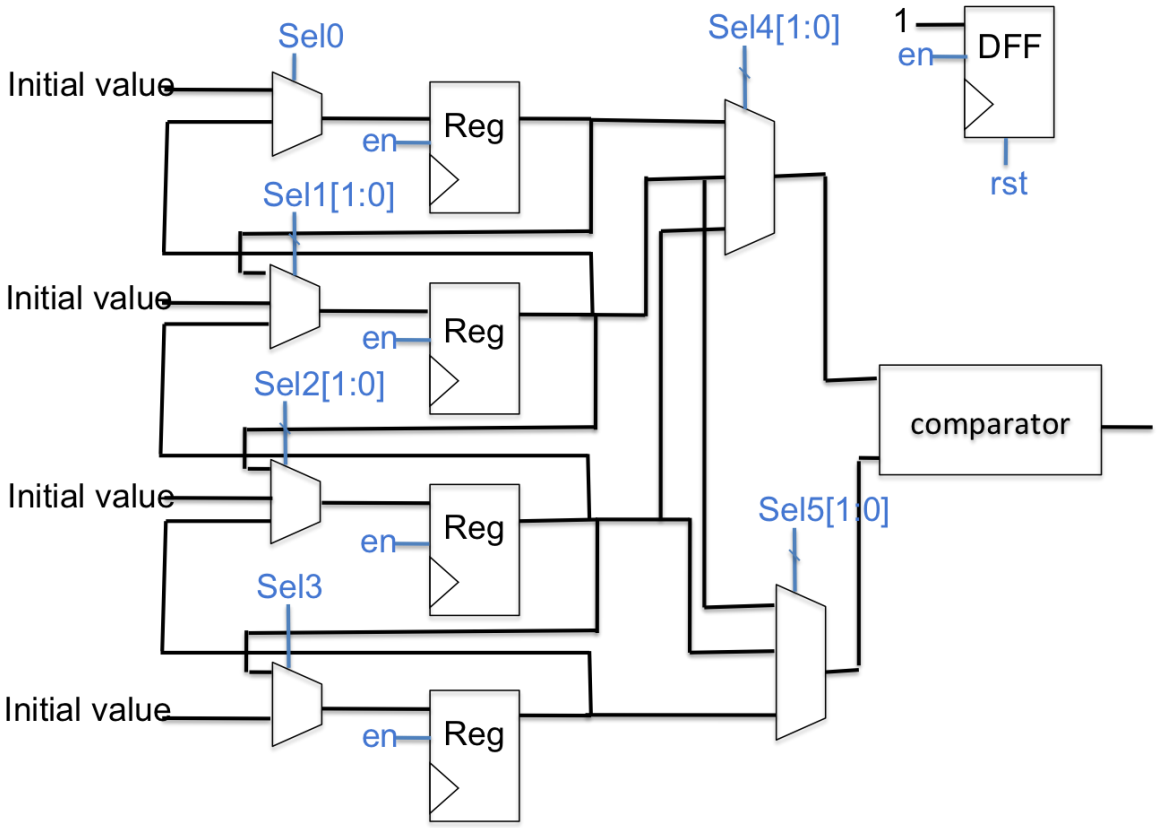
1. You are given a 3 input LUT with some functionality, write a testbench to generate the truth table so that you can find out what function it is implementing. You can use $display function in Verilog.

```
module testLUT;
        reg[2:0] count = 3'b000;
        // Inputs
        wire a,b,c;
        assign a = count[0];
        assign b = count[1];
        assign c = count[2];
        // Outputs
        wire d;
        // Instantiate the Unit Under Test (UUT)
        LUT uut (
                .a(a),
                .b(b),
                .c(c),
                .d(d)
        );
        initial
                begin
                #10;
                repeat(8)
                        begin
                                $display( "%b %b %b    %b\n", a,b,c,d);
                                #2 count = count + 1'b1;
                                #2;
                        end

                $stop;
                end
endmodule
```

2. You are given four registers each holding a number, you are to design a circuit, which does bubble sort on this set of numbers. The registers have an enable signal, which when set to 0, will ensure the register keeps its old value at the active clock edge.

a) Draw the datapath, which would allow bubble sort to be performed on these 4 numbers. Label and control signals you would use. You are allowed only one comparator, remember to allow initial loading of the values.




First, what we need is a mechanism to swap the values in two adjacent registers; this is achieved by having a mux in front of every register to select the source of the data. For example, if we want to swap the value between the top most register and the one below it, we would assert Sel0 and Sel1 such that the appropriate value is fed into the input of each register, and we turn on enable signal for these two registers, when the next clock edge comes, the swap would be completed. Of course, whether to swap the values is a decision the controller would have to make based on the output of the comparator.

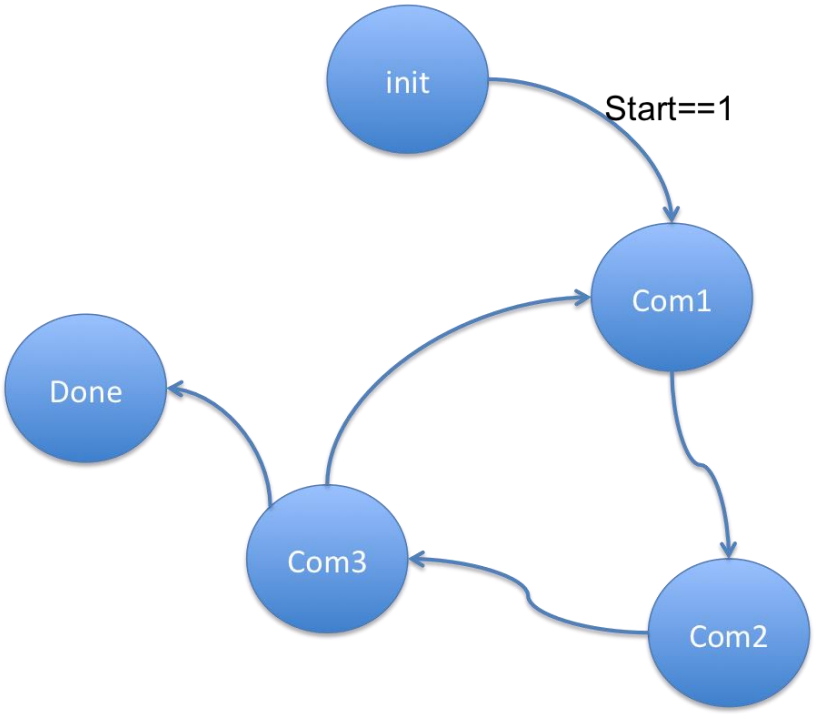
Secondly, as there is only one comparator, we would need two muxes in each of its inputs so we can compare any adjacent pair of numbers.

Besides, the stand-alone DFF is used to keep track of if a swap is performed during the last pass through the 4 values, as would be elaborated in the next part.

All the signals shown in blue are the control signals from the controller.

b) Draw a state transition diagram; clearly states what actions would be taken in each state, referring to the control signals you have in the last part. You can assume there is a "start" signal, and you should assert a "done" signal when the sorting is done.

init state: all register enabled, sel0~3 are asserted in such a way that the initial value are fed into the registers



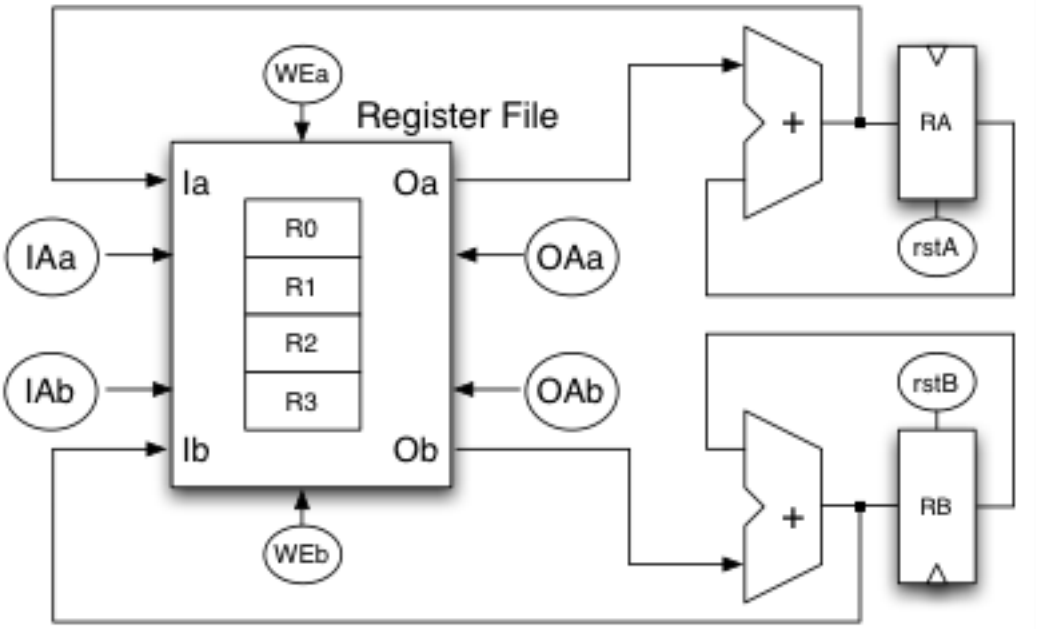

Com1 state: Sel0 select the output of the 2nd register, sel1 select the output of the top register. Sel4 select the output of the top register, select5 select the output of the 2nd register. If the comparator shows the 2nd register has greater value than the top register, the enable signal for the stand-alone DFF is asserted, and the enable signals of the top two register are high.

Com2 state: Sel1 select the output of the 3rd register, sel2 select the output of the 2nd register. Sel4 select the output of the 2nd register, select5 select the output of the 3rd register. If the comparator shows the 3rd register has greater value than the 2nd register, the enable signal for the stand-alone DFF is asserted, and the enable signals of the middle two register are high.

Com3 state: Sel2 select the output of the bottom register, sel3 select the output of the 3rd register. Sel4 select the output of the 3rd register, select5 select the output of the bottom register. If the comparator shows the bottom register has greater value than the 3rd register, the enable signals of the bottom two register are high. Also, at this state, the rst signal for the stand-alone DFF is asserted.

The arc out of Com3: to determine if next state is Com1 or Done, we look at if the stand-alone DFF is 1 and also if Com3 is doing a swap(by looking at the output of the comparator). If either of the two conditions is true, the next state is Com1, else the next state is Done.

3. Consider the datapath shown below. A controller (not shown), is used to control operation of the datapath. It sets the value of all the control signals (circled in the datapath diagram). The block labeled "Register File" stores four data registers, R0–R3, has two asychronous read ports, and two synchronous write ports. The four port addresses are IAa, IAb, OAa, and OAb. Writing to the register file is controlled by the WEa and WEb signals. The data registers at the output of the adders, RA and RB, have synchronous reset inputs, rstA and rstB, respectively.




a) Assume that the register file is initialized with registers R0~R3 holding the values a, b, c, and d, respectively. Registers RA and RB begin uninitialized. Your task is to generate the control signal sequence which will result in a in R0, a+b in R1, a + b + c in R2, and a + b + c + d in R3, and do so in the minimum number of clock cycles. Indicate your answer by filling in the table with the control signal values that the controller would generate on each clock cycle (for use on the next positive clock edge).

| cycle # | IAa | IAb | OAa | OAb | WEa | WEb | rstA | rstB |
|---|---|---|---|---|---|---|---|---|
| 1 | | | | | | | | |
| 2 | | | | | | | | |
| 3 | | | | | | | | |
| 4 | | | | | | | | |
| 5 | | | | | | | | |
| 6 | | | | | | | | |
| 7 | | | | | | | | |
| 8 | | | | | | | | |

b) Now assume that the Register File has a read access delay of 2ns, a write setup time of 1ns, and a write delay of 1ns, i.e., the written data appears in the proper register1ns after the clock edge. There is no register file bypassing. The adders have a combinational logic delay of 4ns, and the output registers have a setup time, 1ns and clock-to-q delay of 1ns, and a hold time of 1ns. Ignore wire delay and clock skew. What is maximum clock frequency for this circuit?