SC1005 Digital Logic
Tutorial 10

Q1. Design a finite state machine that has a single input and single output. It outputs a 1 from the second consecutive high input, and only then outputs a zero after the second consecutive low input. Hence, two consecutive 1 inputs, get a high output, that stays until two consecutive low inputs are received.

(a) Implement the finite state machine in Verilog using a combinational always block for the state transition logic.

(b) Redo the implementation using only assign statements for the state transition logic.

(c) Show how the state machine would respond to the following sequence of inputs:

0,1,0,1,1,0,1,1,0,0,0,1,0,1,1

```
module consec(input b,
  input clk, rst,
  output x);

parameter start = 2'b00,intermOne = 2'b01,last = 2'b11, intermTwo = 2'b10;

reg [1:0] nst,st;

assign x = (s1&s0) | (b&s1) | (b&s0) ;

always@(posedge clk)
begin
 if(rst)
  st<=start;
 else
  st<=nst;
end

always@*
begin
 nst = st;
 case(st)
  start:if(b) nst = intermOne;
     else nst = start;
  intermOne: if(b) nst = last;
     else nst = start;
  last: if(b) nst = last;
       else nst = intermTwo;
  intermTwo: if(b) nst = last;
     else nst = start;
  default: nst = start;
 endcase

end

endmodule
```

```
assign n[1] = (s[0]&a) | (s[1]&~s[0]);

assign n[0] = (~s[1]&~s[0]&a) | (s[1]&~s[0]&~a);

assign x = s[1];
```

b)

```
module consec(input b,
  input clk, rst,
  output x);

wire n1,n0;
reg s1,s0;

assign x = (s1&s0) | (b&s1) | (b&s0);
assign n1 = x;
assign n0 = b;

always@(posedge clk)
begin
 if(rst)begin
  s1<=1'b0;
  s0<=1'b0;
 end else begin
  s1<= n1;
  s0<= n0;
 end
end

endmodule
```