시험문제 verilog 손코딩(배운거에서)

또는 schematic 보고 손코딩

총 7문제

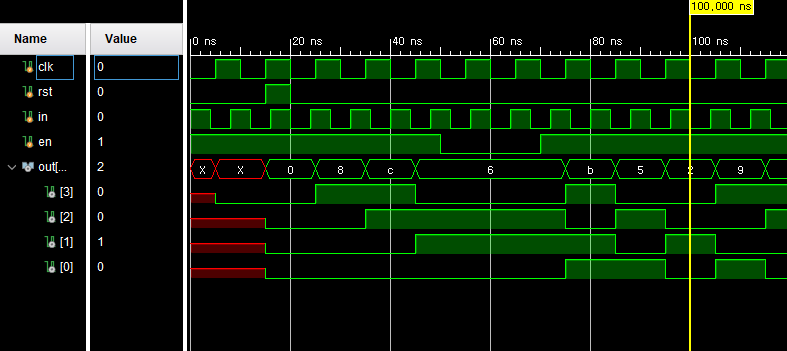
Simulation 내용 x

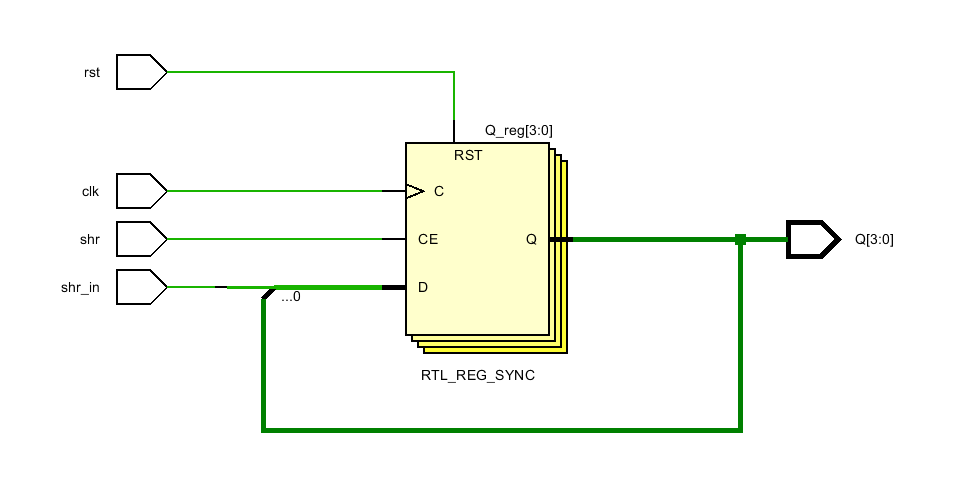
k-map, truth/state table 그림 위주

input, output, timescale 주어짐. Wire, reg 알아서 추가해 사용

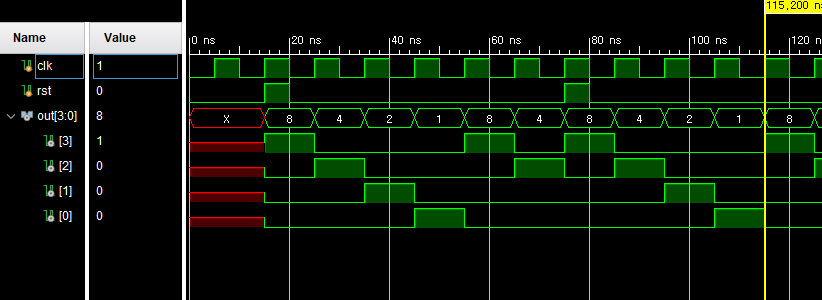
보고서는 시험 당일 제출

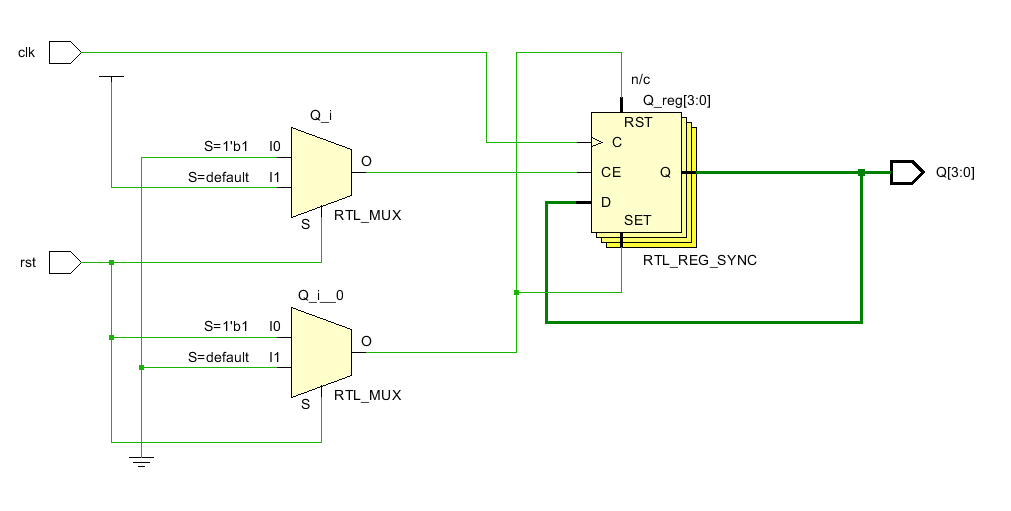
Shift



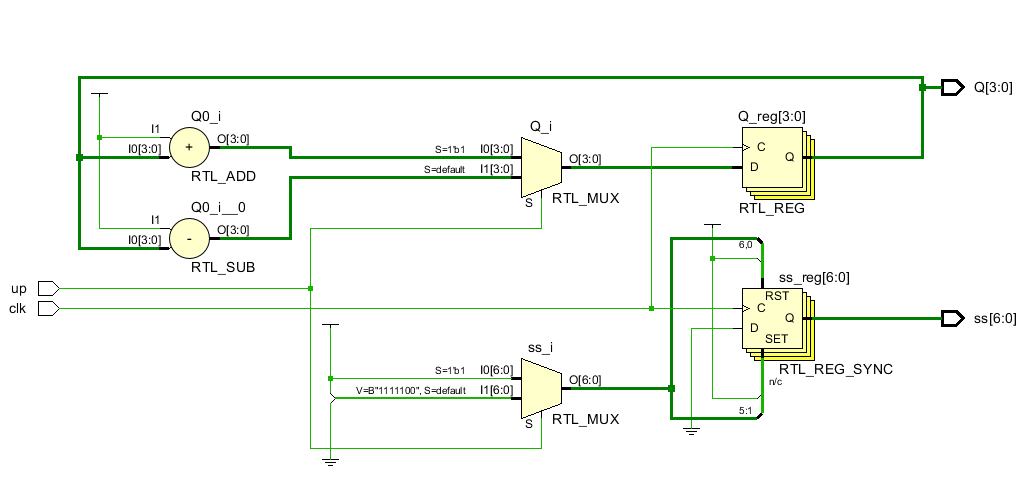


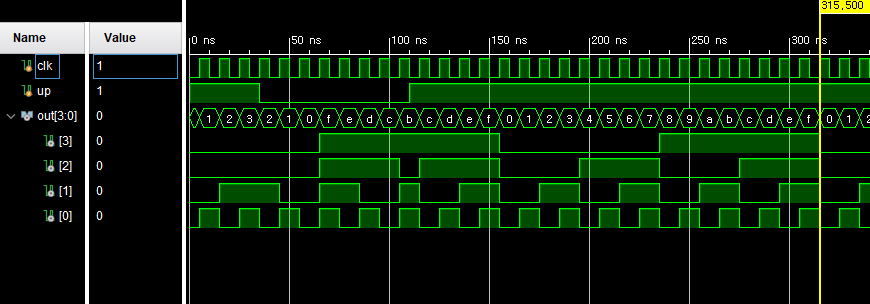
Rc





Udc





`timescale 1ns / 1ps

module shr\_sim;

reg clk, rst, in, en;

wire[3:0] out;

shift\_register shr(

.shr(en),

.rst(rst),

.shr\_in(in),

.clk(clk),

.Q(out)

);

initial en = 1'b1;

initial clk = 1'b0;

initial rst = 1'b0;

initial in = 1'b1;

always clk = #5 ~clk;

always in = #4 ~in;

initial begin

#15

assign rst = 1'b1;

#5

assign rst = 1'b0;

#30

assign en = 1'b0;

#20

assign en = 1'b1;

$finish;

end

endmodule

module rc\_sim;

reg clk, rst;

wire[3:0] out;

ring\_counter rc(

.clk(clk),

.rst(rst),

.Q(out)

);

initial clk = 1'b0;

initial rst = 1'b0;

always clk = #5 ~clk;

initial begin

#15

assign rst = 1'b1;

#5

assign rst = 1'b0;

#55

assign rst = 1'b1;

#5

assign rst = 1'b0;

#1600

$finish;

end

endmodule

module udc\_sim;

reg clk, up;

wire[3:0] out;

wire[6:0] ss;

up\_down\_counter udc(

.clk(clk),

.up(up),

.Q(out),

.ss(ss)

);

initial clk = 1'b0;

initial up = 1'b1;

always clk = #5 ~clk;

initial begin

#35

assign up = 1'b0;

#75

assign up = 1'b1;

#1600

$finish;

end

endmodule

`timescale 1ns / 1ps

module shift\_register(shr, rst, shr\_in, clk, Q);

input clk, rst, shr\_in, shr;

output[3:0] Q;

reg [3:0] Q;

always @(posedge clk) begin

if(rst) begin

Q <= 4'b0000;

end

else if(shr) begin

Q[0] <= Q[1];

Q[1] <= Q[2];

Q[2] <= Q[3];

Q[3] <= shr\_in;

end

end

endmodule

module ring\_counter(clk, rst, Q);

input clk, rst;

output [3:0] Q;

reg [3:0] Q;

always @(posedge clk) begin

if(rst)

Q = 4'b1000;

else begin

Q[0] <= Q[1];

Q[1] <= Q[2];

Q[2] <= Q[3];

Q[3] <= Q[0];

end

end

endmodule

module up\_down\_counter(up, clk, Q, ss, digit);

input up, clk;

output [3:0] Q;

output [6:0] ss;

output digit;

assign digit = 1'b1;

reg[3:0] Q;

reg[6:0] ss;

initial Q <= 4'b0000;

always @(posedge clk) begin

if(up) begin

Q <= Q + 4'b0001;

ss <= 7'b0111110;

end

else begin

Q <= Q - 4'b0001;

ss <= 7'b0111101;

end

end

endmodule