Shift Register

Verilog Code

module shif(shr,rst,shr\_in,clk,Q);

input shr , shr\_in;

input clk,rst;

output [3:0] Q;

reg [3:0] Q;

always @(posedge clk) begin

if(rst == 1)

R<=4'b0000;

else if (shr ==1) begin

R[3] <=shr\_in; R[2] <= R[3];

R[1] <= R[2]; R[0] <= R[1];

end

end

//assign Q =R;

endmodule

Simulation

module inv\_sim;

reg clk;

reg rst;

reg shr,shr\_in;

wire [3:0] out;

inv aoi(shr,rst,shr\_in,clk,out);

always #5 clk = ~clk;

always #3 shr = ~shr;

always #20 shr\_in = ~shr\_in;

initial begin

shr <= 0;

shr\_in <= 1;

clk <= 0;

rst <= 0;

#20 rst <=0;

#80 rst <=1;

#50 rst <=0;

end

endmodule

Ring Counter

Verilog Code

module ring(clk,clr,Q);

input clk;

input clr;

output [3:0] Q;

reg [3:0] Q;

always @(posedge clk or posedge clr)

begin

if(clr==1)

Q<=1;

else

begin

Q[3] <= Q[0];

Q[2:0] <= Q[3:1];

end

end

endmodule

Simulation

`timescale 1ns / 1ps

module inv\_sim;

//x,clk,preset,clear,outz

reg clk,clr;

wire [3:0] Q;

inv aoi(clk,clr,Q);

always #5 clk = ~clk;

initial begin

clk <= 0;

clr <= 0;

#20 clr <=0;

#80 clr <=1;

#50 clr <=0;

end

Endmodule

**UP/DOWN COUNTER**

`timescale 1ns / 1ps

module inv(clk,up,clr,Q,x,seg);

input clk,up;

input clr;

output [3:0] Q;

output x;

output [6:0] seg;

reg [3:0] Q;

reg [6:0] seg;

always @(posedge clk or posedge clr)

begin

if(~clr)

begin

Q <=0;

seg = 7'b1111111;

end

else if(up)

begin

seg = 7'b0111110;

Q=Q+1;

end

else

begin

seg = 7'b0111101;

Q=Q-1;

end

end

assign x = 1;

endmodule

`timescale 1ns / 1ps

module inv\_sim;

reg clk,clr,up;

wire [3:0] Q;

wire x;

wire [6:0] seg;

inv aoi(clk,up,clr,Q,x,seg);

always #5 clk = ~clk;

always #20 up = ~up;

initial begin

up<=1;

clk <= 0;

clr <= 0;

#20 clr <=0;

#80 clr <=1;

#40 clr <=0;

#20 clr <=1;

#30 clr <=0;

end

endmodule