**FSM Screen**

// Code your design here

module FSM\_screen (clk, SGP, state, next);

input clk;

input [2:0] SGP;

output [2:0] state, next;

reg [2:0] state = 2'b0;

reg [2:0] next = 2'b0;

parameter start = 3'b0;

parameter game = 3'b1;

parameter p1 = 3'b10;

parameter p2 = 3'b11;

parameter start\_space = 3'b100;

always @(posedge clk)

begin

state <= next;

end

always @(SGP or state)

begin

case (SGP[2])

1'b0: if (state == start\_space) next = start;

1'b1: if (state == start) next = game;

else if (state == p1 || state == p2) next = start\_space;

endcase

case (SGP[1:0])

2'b10: if (state == game) next = p1;

2'b11: if (state == game) next = p2;

endcase

end

Endmodule

**TEST BENCH Screen.**

// Code your testbench here

// or browse Examples

module test();

reg clk;

reg [2:0] SGP;

wire [2:0] state, next;

FSM\_screen t (clk,SGP,state, next);

initial

begin

clk <= 0;

SGP <= 0;

end

always

#5 clk = ~clk;

always

begin

#20 SGP = 3'b100;

#30 SGP = 3'b0;

#40 SGP = 3'b10;

#50 SGP = 3'b0;

#60 SGP = 3'b100;

#70 SGP = 3'b0;

#80 SGP = 3'b100;

#90 SGP = 3'b0;

#100 SGP = 3'b11;

#110 SGP = 3'b0;

#120 SGP = 3'b100;

#130 SGP = 3'b0;

end

initial

begin

$dumpfile("dump.vcd");

$dumpvars(1,test);

#1000 $finish;

end

Endmodule

**FSM Move**

// Code your design here

module FSM\_move(clk, keys, state);

input clk;

input [3:0] keys;

output [2:0] state;

reg [2:0] state = 0;

reg [2:0] next;

parameter up = 3'b0;

parameter up\_ = 3'b1;

parameter down = 3'b10;

parameter down\_ = 3'b11;

parameter left = 3'b100;

parameter left\_ = 3'b101;

parameter right = 3'b110;

parameter right\_ = 3'b111;

always @(posedge clk)

begin

state <= next;

end

always @(keys or state)

begin

case (keys)

4'b0: next[0] = 0;

4'b1: if (state == up) next = up\_;

else if (state != up\_) next = up;

4'b10: if (state == down) next = down\_;

else if (state != down\_) next = down;

4'b100: if (state == left) next = left\_;

else if (state != left\_) next = left;

4'b1000: if (state == right) next = right\_;

else if (state != right\_)next = right;

endcase

end

endmodule

module FSM\_screen (clk, SGP, state);

input clk;

input [2:0] SGP;

output [1:0] state;

reg [1:0] state = 0;

reg [1:0] next;

parameter start = 2'b0;

parameter game = 2'b1;

parameter p1 = 2'b10;

parameter p2 = 2'b11;

always @(posedge clk)

begin

state <= next;

end

always @(SGP or state)

begin

case (SGP)

3'b1XX: if (state == start) next = game;

3'bX10: if (state == game) next = p1;

3'bX11: if (state == game) next = p2;

3'b1XX: if (state == p1 || state == p2) next = start;

endcase

end

endmodule

**TestBench Move**

// Code your testbench here

// or browse Examples

module test();

reg clk;

reg [3:0] keys;

wire [2:0] state;

FSM\_move t (clk,keys,state);

initial

begin

clk <= 0;

keys <= 0;

end

always

#5 clk = ~clk;

always

begin

#20 keys = 4'b0001;

#30 keys = 0;

#40 keys = 4'b0010;

#50 keys = 0;

#60 keys = 4'b0100;

#70 keys = 0;

#80 keys = 4'b1000;

#90 keys = 0;

end

initial

begin

$dumpfile("dump.vcd");

$dumpvars(1,test);

#1000 $finish;

end

endmodule