**Worksheet of the student**

Date of Performance: 3/4/2015

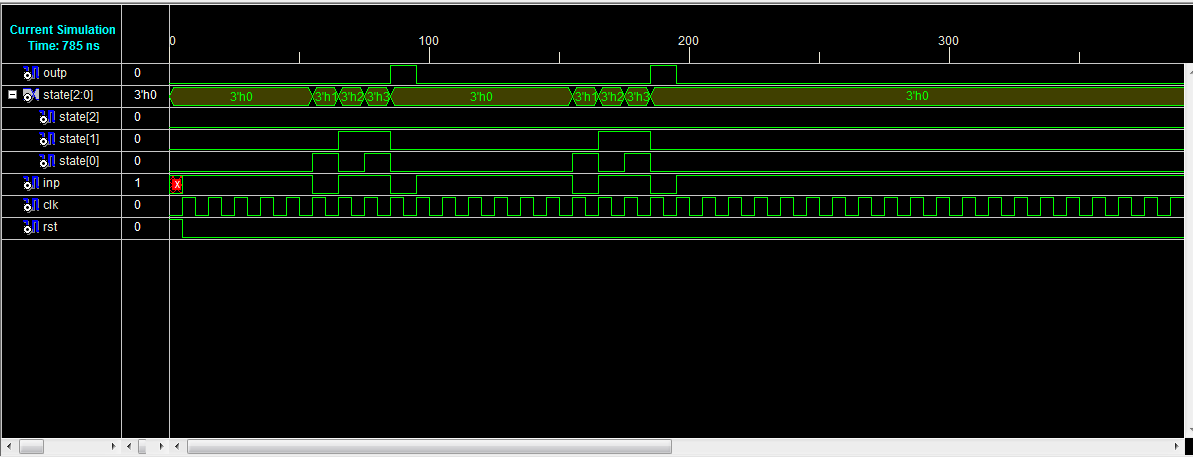
Registration No: 11206693

Exp.No: 10

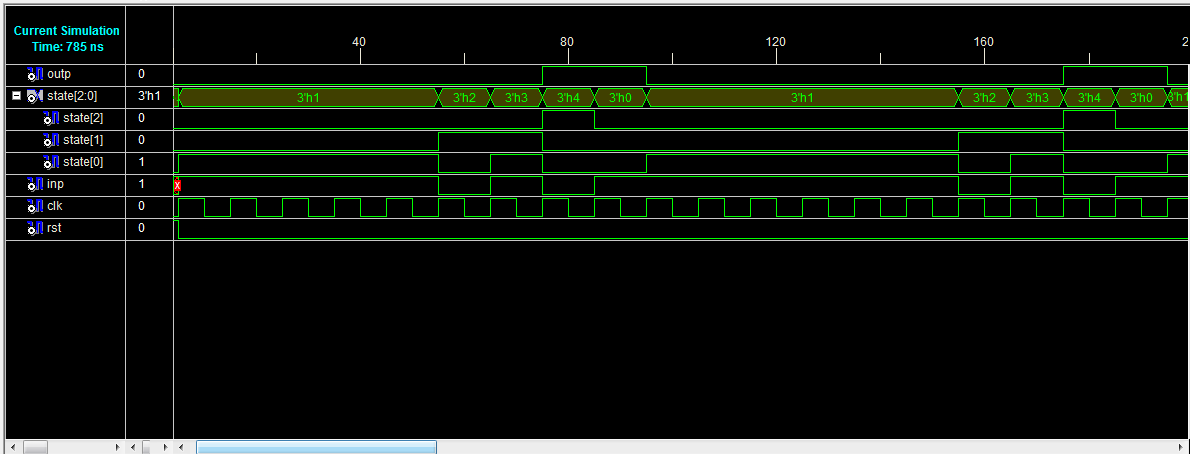
Roll No: E1206 A15

**Aim: P10(Implementation of Sequence Detector to detect 0110/10101 as melay and moore in verilog using Xilinx)**

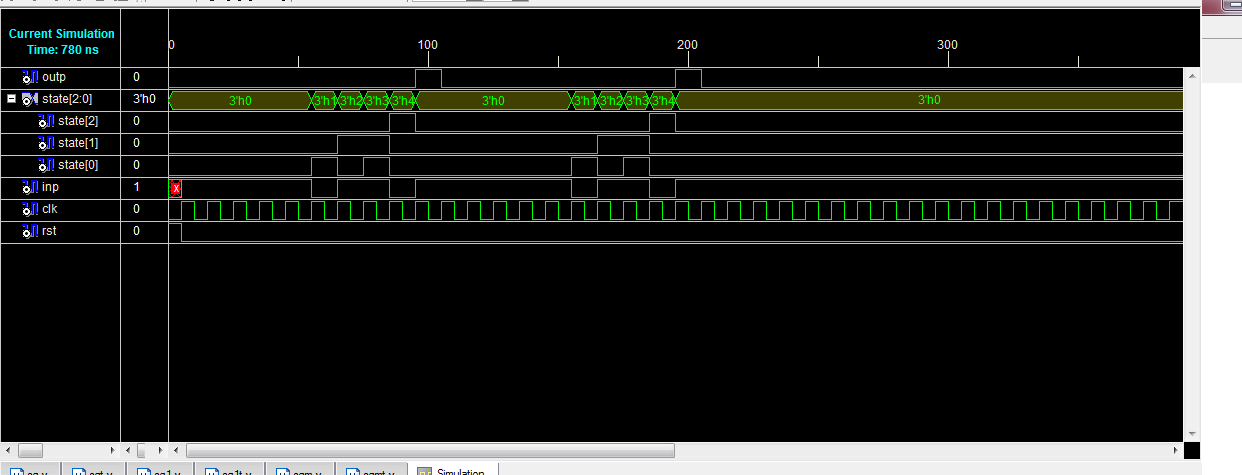
**1-0110 melay machine**



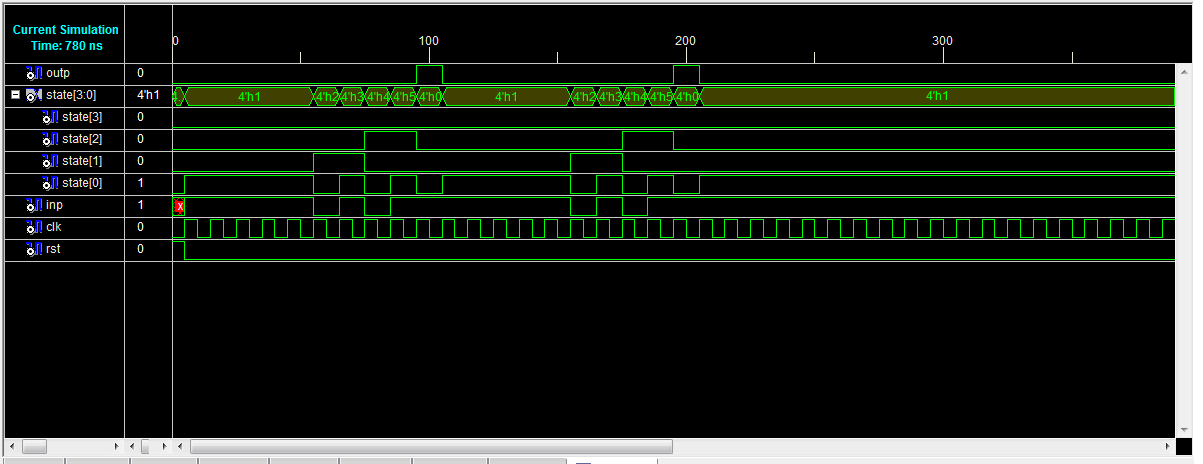
2-10101 melay machine



**3-0110 morre machine**



**4-10101 morre machine**



**Program code:**

**1-0110 melay machine**

module sq(clk,rst,inp,outp,state);

input clk, rst, inp;

output reg outp;

output [2:0] state;

reg [2:0] state;

parameter S0=0, S1=1, S2=2, S3=3,S4=4;

///next state logic

always @(posedge clk or posedge rst)

if(rst==1)

begin

state<=S0;

outp<=0;

$display("current state is %d",state);

$display("current output is %b",outp);

end

else

case(state)

S0: if(inp)

begin

state<=S0;

outp<=0;

$display("current state is %d",state);

$display("current output is %b",outp);

end

else

begin

state<=S1;

outp<=0;

$display("current state is %d",state);

$display("current output is %b",outp);

end

S1: if(inp)

begin

state<=S2;

outp<=0;

$display("current state is %d",state);

$display("current output is %b",outp);

end

else

begin

state<=S1;

outp<=0;

$display("current state is %d",state);

$display("current output is %b",outp);

end

S2: if(inp)

begin

state<=S3;

outp<=0;

$display("current state is %d",state);

$display("current output is %b",outp);

end

else

begin

state<=S0;

outp<=0;

$display("current state is %d",state);

$display("current output is %b",outp);

end

S3: if(inp)

begin

state<=S0;

outp<=0;

$display("current state is %d",state);

$display("current output is %b",outp);

end

else

begin

state<=S0;

outp<=1;

$display("current state is %d",state);

$display("current output is %b",outp);

end

endcase

/\*always @(state)

case(state)

S0:

outp<=0;

S1:

outp<=0;

S2:

outp<=0;

S3:

outp<=0;

S4:

outp<=1;

endcase\*/

endmodule

**2-10101 melay machine**

**module sq1(clk,rst,inp,outp,state);**

**input clk, rst, inp;**

**output reg outp;**

**output [2:0] state;**

**reg [2:0] state;**

**parameter S0=0, S1=1, S2=2, S3=3,S4=4;**

**///next state logic**

**always @(posedge clk or posedge rst)**

**if(rst==1)**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**case(state)**

**S0: if(inp)**

**begin**

**state<=S1;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S1: if(inp)**

**begin**

**state<=S1;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S2;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S2: if(inp)**

**begin**

**state<=S3;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S3: if(inp)**

**begin**

**state<=S1;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S4;**

**outp<=1;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S4: if(inp)**

**begin**

**state<=S0;**

**outp<=1;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**endcase**

**/\*always @(state)**

**case(state)**

**S0:**

**outp<=0;**

**S1:**

**outp<=0;**

**S2:**

**outp<=0;**

**S3:**

**outp<=0;**

**S4:**

**outp<=1;**

**endcase\*/**

**endmodule**

**3-0110 morre machine**

**module sqm(clk,rst,inp,outp,state);**

**input clk, rst, inp;**

**output reg outp;**

**output [2:0] state;**

**reg [2:0] state;**

**parameter S0=0, S1=1, S2=2, S3=3,S4=4;**

**///next state logic**

**always @(posedge clk or posedge rst)**

**if(rst==1)**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**case(state)**

**S0: if(inp)**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S1;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S1: if(inp)**

**begin**

**state<=S2;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S1;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S2: if(inp)**

**begin**

**state<=S3;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S3: if(inp)**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S4;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S4: if(inp)**

**begin**

**state<=S0;**

**outp<=1;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S0;**

**outp<=1;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**endcase**

**/\*always @(state)**

**case(state)**

**S0:**

**outp<=0;**

**S1:**

**outp<=0;**

**S2:**

**outp<=0;**

**S3:**

**outp<=0;**

**S4:**

**outp<=1;**

**endcase\*/**

**endmodule**

**4-10101 morre machine:**

**module sqm1(clk,rst,inp,outp,state);**

**input clk, rst, inp;**

**output reg outp;**

**output [3:0] state;**

**reg [3:0] state;**

**parameter S0=0, S1=1, S2=2, S3=3,S4=4,S5=5;**

**///next state logic**

**always @(posedge clk or posedge rst)**

**if(rst==1)**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**case(state)**

**S0: if(inp)**

**begin**

**state<=S1;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S1: if(inp)**

**begin**

**state<=S1;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S2;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S2: if(inp)**

**begin**

**state<=S3;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S3: if(inp)**

**begin**

**state<=S1;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S4;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S4: if(inp)**

**begin**

**state<=S5;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S0;**

**outp<=0;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**S5: if(inp)**

**begin**

**state<=S0;**

**outp<=1;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**else**

**begin**

**state<=S0;**

**outp<=1;**

**$display("current state is %d",state);**

**$display("current output is %b",outp);**

**end**

**endcase**

**/\*always @(state)**

**case(state)**

**S0:**

**outp<=0;**

**S1:**

**outp<=0;**

**S2:**

**outp<=0;**

**S3:**

**outp<=0;**

**S4:**

**outp<=1;**

**endcase\*/**

**endmodule**

Testbench:

1-0110 melay machine

////////////////////////////////////////////////////////////////////////////////

module sqt\_v;

// Inputs

reg clk;

reg rst;

reg inp;

// Outputs

wire outp;

wire [2:0] state;

// Instantiate the Unit Under Test (UUT)

sq uut (

.clk(clk),

.rst(rst),

.inp(inp),

.outp(outp),

.state(state)

);

initial

clk=0;

always

#5 clk=~clk;

initial begin

// Initialize Inputs

#0 rst = 1;

#5 rst=0;

inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=0;

#10 inp=1;

#10 inp=1;

#10 inp=0;rst= 1;rst=0;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=0;

#10 inp=1;

#10 inp=1;

#10 inp=0;rst= 1;rst=0;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#500 $stop;

end

endmodule

2-10101 melay machine:

////////////////////////////////////////////////////////////////////////////////

module sq1t\_v;

// Inputs

reg clk;

reg rst;

reg inp;

// Outputs

wire outp;

wire [2:0] state;

// Instantiate the Unit Under Test (UUT)

sq1 uut (

.clk(clk),

.rst(rst),

.inp(inp),

.outp(outp),

.state(state)

);

initial

clk=0;

always

#5 clk=~clk;

initial begin

// Initialize Inputs

#0 rst = 1;

#5 rst=0;

inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=0;

#10 inp=1;

#10 inp=0;

#10 inp=1;rst= 1;rst=0;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=0;

#10 inp=1;

#10 inp=0;

#10 inp=1;rst= 1;rst=0;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#500 $stop;

end

endmodule

3-0110 moore machine

////////////////////////////////////////////////////////////////////////////////

module sqmt\_v;

// Inputs

reg clk;

reg rst;

reg inp;

// Outputs

wire outp;

wire [2:0] state;

// Instantiate the Unit Under Test (UUT)

sqm uut (

.clk(clk),

.rst(rst),

.inp(inp),

.outp(outp),

.state(state)

);

initial

clk=0;

always

#5 clk=~clk;

initial begin

// Initialize Inputs

#0 rst = 1;

#5 rst=0;

inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=0;

#10 inp=1;

#10 inp=1;

#10 inp=0;rst= 1;rst=0;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=0;

#10 inp=1;

#10 inp=1;

#10 inp=0;rst= 1;rst=0;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#500 $stop;

end

endmodule

4-10101 morre machine:

////////////////////////////////////////////////////////////////////////////////

module sqm1t\_v;

// Inputs

reg clk;

reg rst;

reg inp;

// Outputs

wire outp;

wire [3:0] state;

// Instantiate the Unit Under Test (UUT)

sqm1 uut (

.clk(clk),

.rst(rst),

.inp(inp),

.outp(outp),

.state(state)

);

initial

clk=0;

always

#5 clk=~clk;

initial begin

// Initialize Inputs

#0 rst = 1;

#5 rst=0;

inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=0;

#10 inp=1;

#10 inp=0;

#10 inp=1;rst= 1;rst=0;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=0;

#10 inp=1;

#10 inp=0;

#10 inp=1;rst= 1;rst=0;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#10 inp=1;

#500 $stop;

end

endmodule

OUTPUT WINDOW:

This is a Full version of ISE Simulator.

Simulator is doing circuit initialization process.

Finished circuit initialization process.

current state is x

current output is x

current state is 0

current output is 0

current state is 1

current output is 0

current state is 1

current output is 0

current state is 1

current output is 0

current state is 1

current output is 0

current state is 1

current output is 0

current state is 2

current output is 0

current state is 3

current output is 0

current state is 4

current output is 0

current state is 5

current output is 0

current state is 0

current output is 1

current state is 1

current output is 0

current state is 1

current output is 0

current state is 1

current output is 0

current state is 1

current output is 0

current state is 1

current output is 0

current state is 2

current output is 0

current state is 3

current output is 0

current state is 4

current output is 0

current state is 5

current output is 0

current state is 0

current output is 1

current state is 1

current output is 0

current state is 1

current output is 0

current state is 1

current output is 0

current state is 1

current output is 0

current state is 1

current output is 0

**Console window output**:

Started : "Check Syntax".

Running vlogcomp

Compiling project file "sqm1t\_v\_stx.prj"

Compiling verilog file "C:/Xilinx92i/sequence/sqm1.v" in library isim\_temp

Module <sqm1> compiled

Parsing C:/Xilinx92i/sequence/sqm1.v: 0.02

Compiling verilog file "C:/Xilinx92i/sequence/sqm1t.v" in library isim\_temp

Module <sqm1t\_v> compiled

Parsing C:/Xilinx92i/sequence/sqm1t.v: 0.00

Compiling verilog file "C:/Xilinx92i/verilog/src/glbl.v" in library isim\_temp

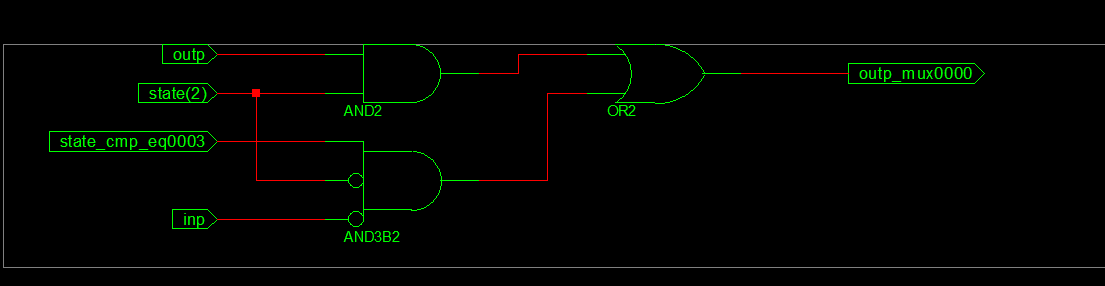
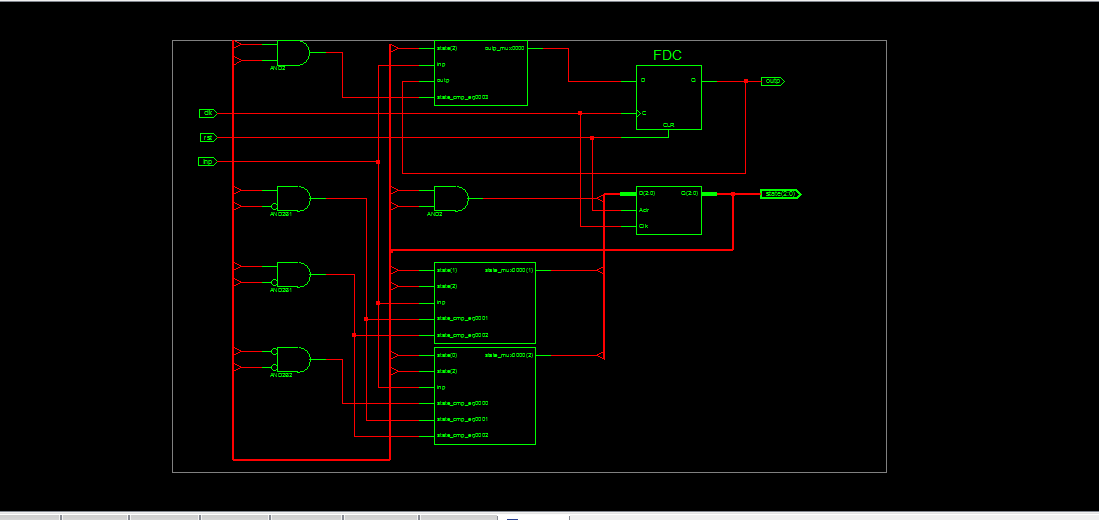
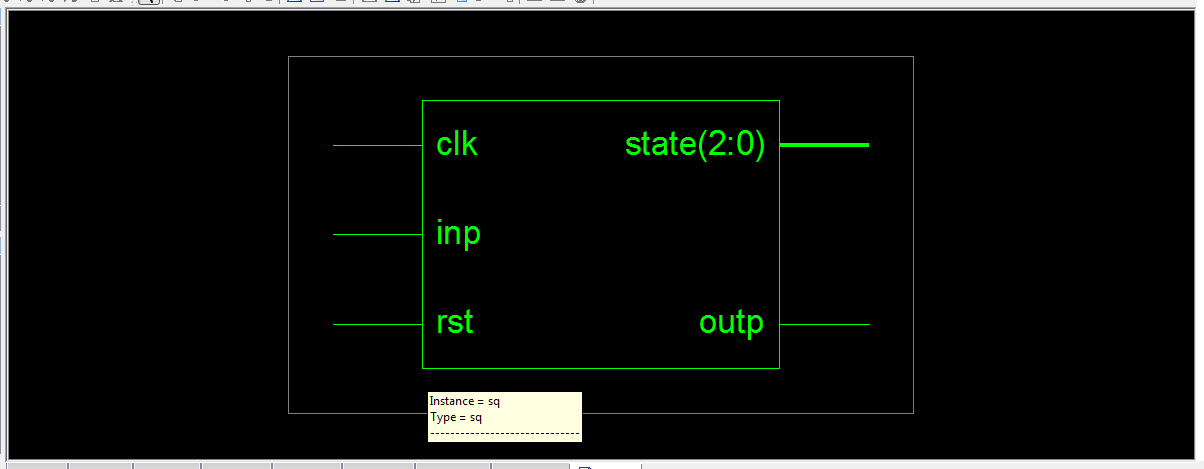
Module <glbl> compiled

Parsing C:/Xilinx92i/verilog/src/glbl.v: 0.01

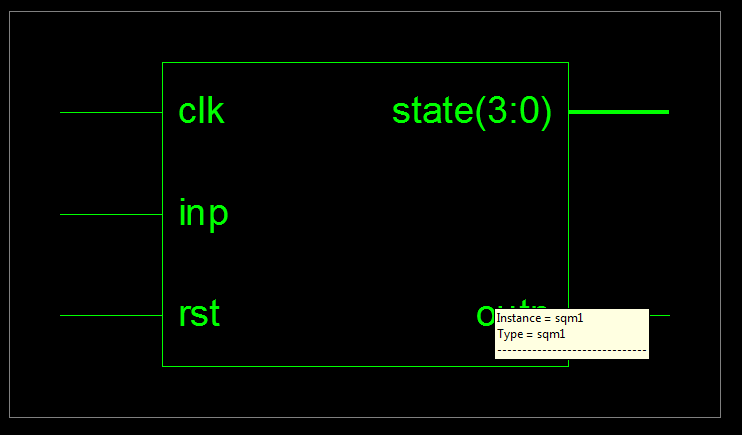
Process "Check Syntax" completed successfully

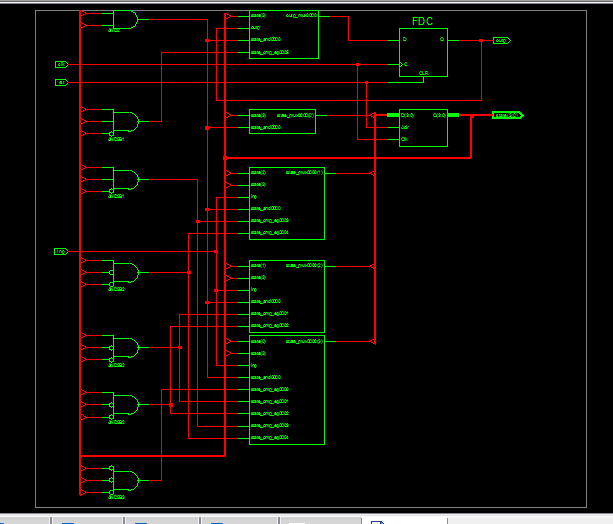
**RTL schemetaic:**

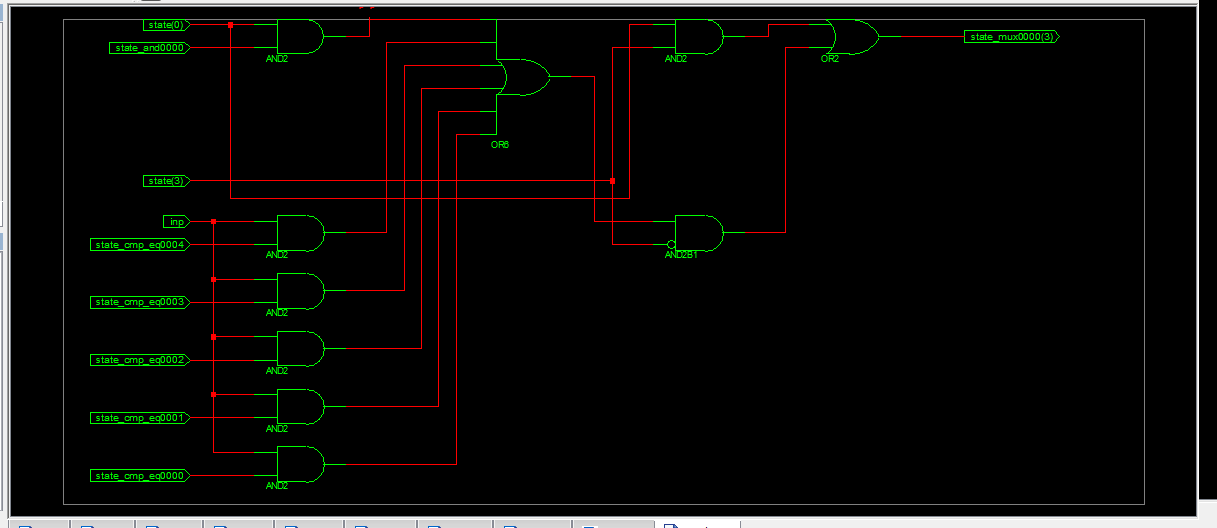
**For melay machine:**

****

For moore machine:



****

****

**Result and Discussion:**

**Melay and moore Sequence detector are succesfully implemented using verilog language and verified in xilinx.**

**Learning Outcomes (what I have learnt):**

**from this experiment we have learned how to make melay moore sequence detector using xilinx software in verilog language in behavioural modelling.**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Parameter (Scale from 1-10, 1 for very poor and 10 excellent)** | **Max. Marks** | **Marks Obtained** |
| **1** | **Understanding of the student about the procedure/apparatus.** |  | **20** |
| **2** | **Observations and analysis including learning Outcomes** |  | **20** |
| **3** | **Completion\* of experiment, Discipline and Cleanliness** |  | **10** |
| **4** | **Signature of Faculty Total marks**  **Obtained** | **Total marks obtained** |  |