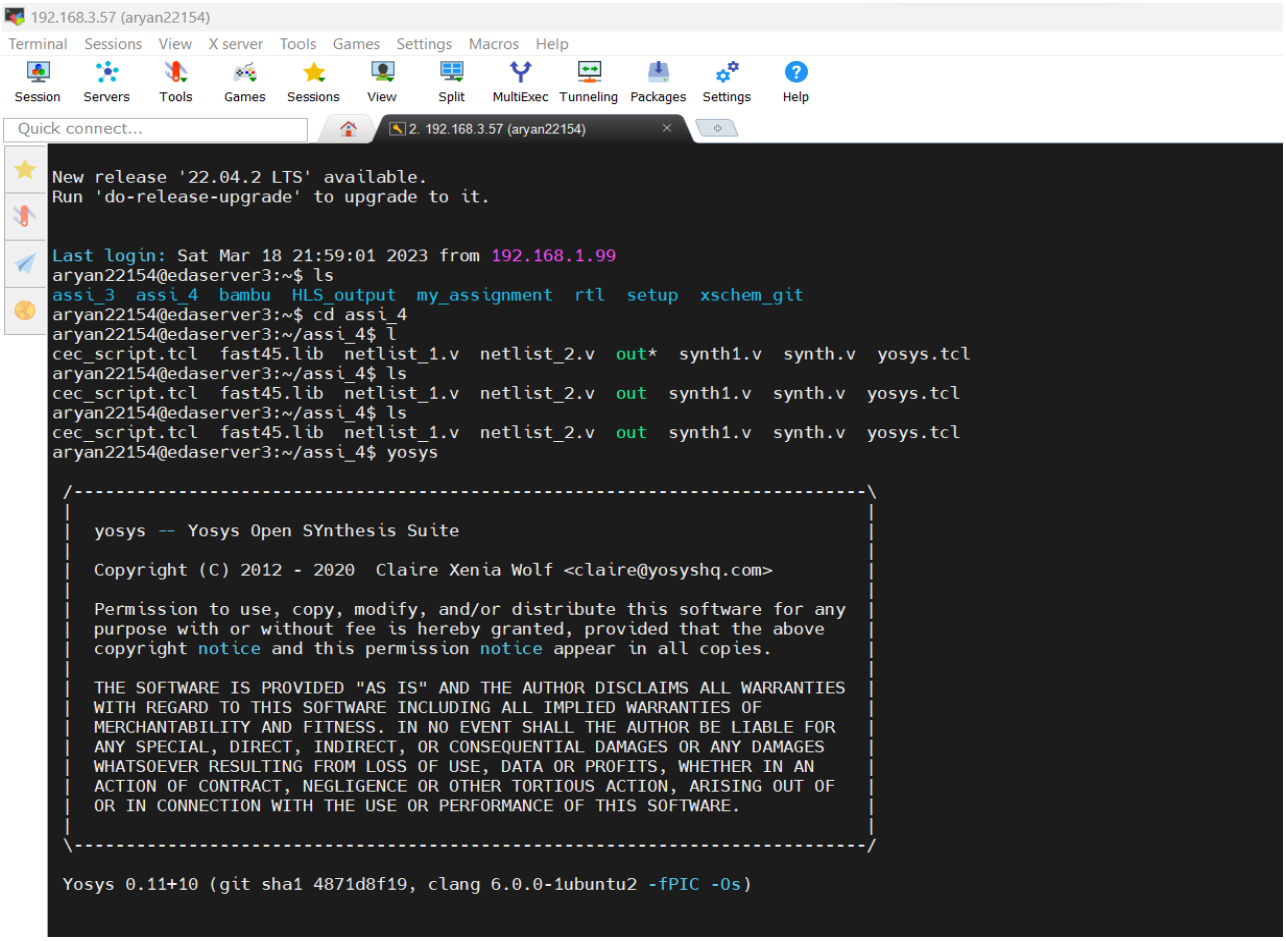
ASSIGNMENT - 4

VDF

Name – ARYAN GUPTA

ROLL NO – MT22154

Q1)- Show screenshot or snippet of log file to prove that you have indeed run the tool [If this is not given then your assignment will not be evaluated].



Q2) - Write a Verilog netlist for the two models above. Choose matching net names and instance names in the two models (though it is shown differently in the above figure).

Show the Verilog netlist in the PDF report.

ANS)-

Verilog code for model – 1

```
module inv(input a,output b);
    assign b=~a;
endmodule
module buff(input a1,output b1);
    assign b1=a1;
endmodule
module dff(input d,input cp,output reg q);
    always @(posedge cp)
    begin
        q<=d;
    end
endmodule
module nanda(input a2,input b2,output y1);
    assign y1=~(a2&b2);
endmodule
module ora(input a3,input b3,output y2);
    assign y2=(a3|b3);
endmodule
module top(input IN,input Clock,output OUT);
    wire w1,w2,w3,w4,w5,w6;
    inv I1 (.a(IN),.b(w1));
    buff I2 (.a1(IN),.b1(w2));
    dff F1 (.d(w1),.cp(Clock),.q(w3));
    dff F2 (.d(w2),.cp(Clock),.q(w4));
    nanda I3 (.a2(w3),.b2(w4),.y1(w5));
    dff F3 (.d(w5),.cp(Clock),.q(w6));
    inv I4 (.a(w6),.b(OUT));
endmodule

// module top(input IN,input Clock,output OUT);
//     wire w1,w2,w3,w4,w5,w6;
//     INV_X1 I1 (IN,w1);
//     BUFF_X1 I2 (IN,w2);
//     DFF_X1 I3 (w1,Clock,w3);
//     DFF_X1 I4 (w2,Clock,w4);
//     NAND_X1 I5 (w3,w4,w5);
//     DFF_X1 I6 (w5,Clock,w6);
//     INV_X1 I7 (w6,OUT);
// endmodule
```

Verilog Code for model -2

```
module inv(input a,output b);
    assign b=~a;
endmodule
module buff(input a1,output b1);
    assign b1=a1;
endmodule
module dff(input d,input cp,output reg q);
    always @(posedge cp)
    begin
        q<=d;
    end
endmodule
module nanda(input a2,input b2,output y1);
    assign y1=~(a2&b2);
endmodule
module ora(input a3,input b3,output y2);
    assign y2=(a3|b3);
endmodule
module top(input IN,input Clock,output OUT);
    wire w1,w2,w3,w4,w5,w6;
    inv I1 (.a(IN),.b(w1));
    buff I2 (.a1(IN),.b1(w2));
    dff F1 (.d(w1),.cp(Clock),.q(w3));
    dff F2 (.d(w2),.cp(Clock),.q(w4));
    ora I3 (.a3(w3),.b3(w4),.y2(w5));
    dff F3 (.d(w5),.cp(Clock),.q(w6));
    inv I4 (.a(w6),.b(OUT));
endmodule
// module top(input IN,input Clock,output OUT);
//     wire w1,w2,w3,w4,w5,w6;
//     INV_X1 I1 (IN,w1);
//     BUFF_X1 I2 (IN,w2);
//     DFF_X1 I3 (w1,Clock,w3);
//     DFF_X1 I4 (w2,Clock,w4);
//     OR_X1 I5 (w3,w4,w5);
//     DFF_X1 I6 (w5,Clock,w6);
//     INV_X1 I7 (w6,OUT);
// endmodule
```

Generated Netlist for Code -1

```
/* Generated by Yosys 0.11+10 (git sha1 4871d8f19, clang 6.0.0-1ubuntu2 -fPIC
-Os) */

(* src = "netlist_1.v:4.1-6.10" *)
module buff(a1, b1);
  (* src = "netlist_1.v:4.19-4.21" *)
```

```
  input a1;
  (* src = "netlist_1.v:4.29-4.31" *)
  output b1;
  assign b1 = a1;
endmodule

(* src = "netlist_1.v:7.1-12.10" *)
module dff(d, cp, q);
  wire _0_;
  (* src = "netlist_1.v:7.26-7.28" *)
  input cp;
  (* src = "netlist_1.v:7.18-7.19" *)
  input d;
  (* src = "netlist_1.v:7.40-7.41" *)
  output q;
  (* src = "netlist_1.v:8.5-11.8" *)
  DFF_X1 _1_ (
    .CK(cp),
    .D(d),
    .Q(q),
    .QN(_0_)
  );
endmodule

(* src = "netlist_1.v:1.1-3.10" *)
module inv(a, b);
  (* src = "netlist_1.v:1.18-1.19" *)
  wire _0_;
  (* src = "netlist_1.v:1.27-1.28" *)
  wire _1_;
  (* src = "netlist_1.v:1.18-1.19" *)
  input a;
  (* src = "netlist_1.v:1.27-1.28" *)
  output b;
  INV_X1 _2_ (
    .A(_0_),
    .ZN(_1_)
  );
  assign _0_ = a;
  assign b = _1_;
endmodule

(* src = "netlist_1.v:13.1-15.10" *)
module nanda(a2, b2, y1);
  (* src = "netlist_1.v:13.20-13.22" *)
  wire _0_;
  (* src = "netlist_1.v:13.29-13.31" *)
  wire _1_;
```

```
  (* src = "netlist_1.v:13.39-13.41" *)
  wire _2_;
  (* src = "netlist_1.v:14.17-14.22" *)
  wire _3_;
  (* src = "netlist_1.v:13.20-13.22" *)
  input a2;
  (* src = "netlist_1.v:13.29-13.31" *)
  input b2;
  (* src = "netlist_1.v:13.39-13.41" *)
  output y1;
  NAND2_X1 _4_ (
    .A1(_0_),
    .A2(_1_),
    .ZN(_2_)
  );
  assign y1 = _2_;
  assign _0_ = a2;
  assign _1_ = b2;
endmodule

(* top =  1  *)
(* src = "netlist_1.v:19.1-28.10" *)
module top(IN, Clock, OUT);
  (* src = "netlist_1.v:19.27-19.32" *)
  input Clock;
  (* src = "netlist_1.v:19.18-19.20" *)
  input IN;
  (* src = "netlist_1.v:19.40-19.43" *)
  output OUT;
  (* src = "netlist_1.v:20.10-20.12" *)
  wire w1;
  (* src = "netlist_1.v:20.13-20.15" *)
  wire w2;
  (* src = "netlist_1.v:20.16-20.18" *)
  wire w3;
  (* src = "netlist_1.v:20.19-20.21" *)
  wire w4;
  (* src = "netlist_1.v:20.22-20.24" *)
  wire w5;
  (* src = "netlist_1.v:20.25-20.27" *)
  wire w6;
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_1.v:23.9-23.38" *)
  dff F1 (
    .cp(Clock),
    .d(w1),
    .q(w3)
  );
```

```
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_1.v:24.9-24.38" *)
  dff F2 (
    .cp(Clock),
    .d(w2),
    .q(w4)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_1.v:26.9-26.38" *)
  dff F3 (
    .cp(Clock),
    .d(w5),
    .q(w6)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_1.v:21.9-21.27" *)
  inv I1 (
    .a(IN),
    .b(w1)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_1.v:22.10-22.30" *)
  buff I2 (
    .a1(IN),
    .b1(w2)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_1.v:25.11-25.39" *)
  nanda I3 (
    .a2(w3),
    .b2(w4),
    .y1(w5)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_1.v:27.9-27.28" *)
  inv I4 (
    .a(w6),
    .b(OUT)
  );
endmodule
```

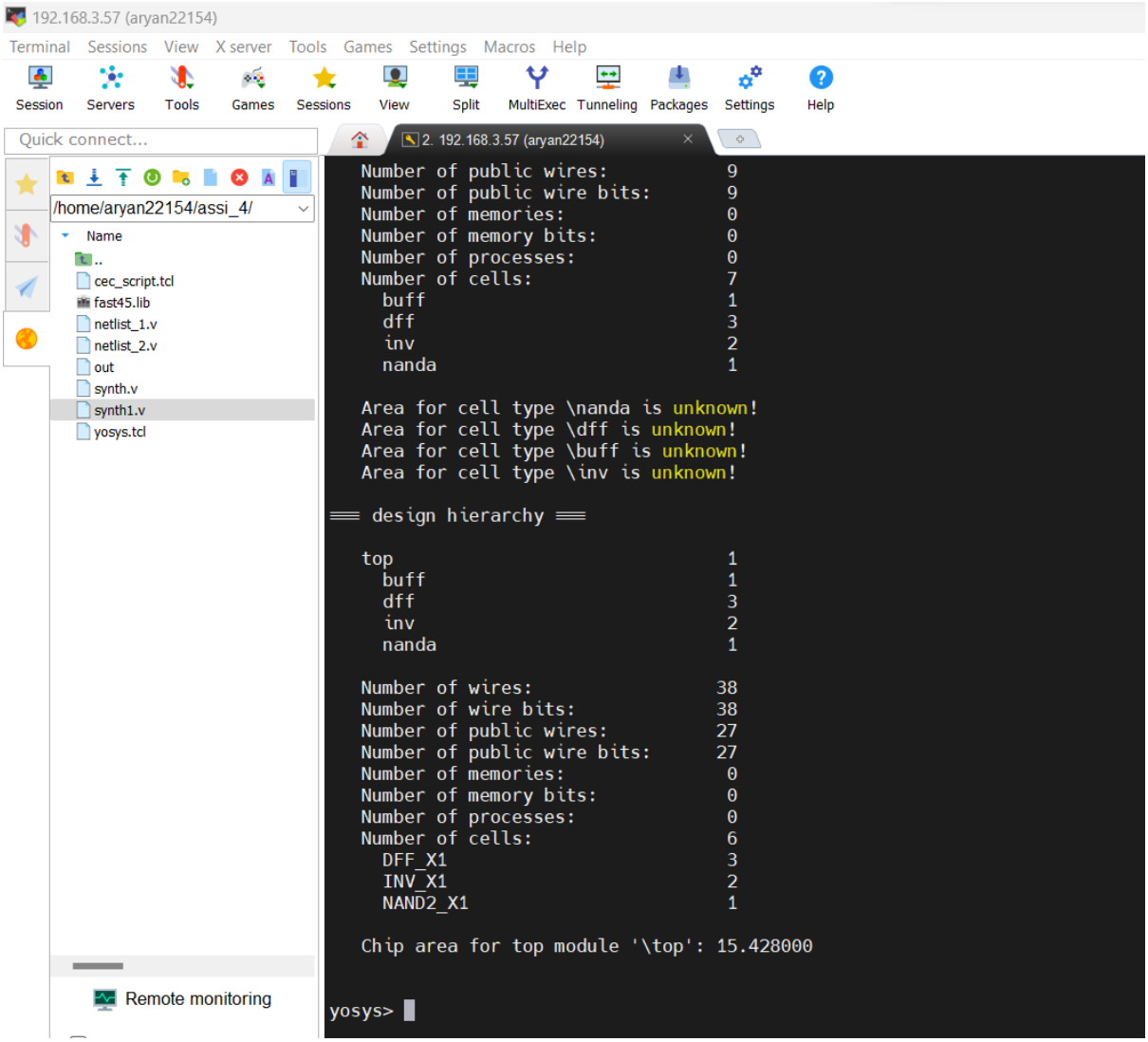
Generated Netlist for Code – 2

```
/* Generated by Yosys 0.11+10 (git sha1 4871d8f19, clang 6.0.0-1ubuntu2 -fPIC
-Os) */

(* src = "netlist_2.v:4.1-6.10" *)
module buff(a1, b1);
  (* src = "netlist_2.v:4.19-4.21" *)
  input a1;
  (* src = "netlist_2.v:4.29-4.31" *)
  output b1;
  assign b1 = a1;
endmodule

(* src = "netlist_2.v:7.1-12.10" *)
module dff(d, cp, q);
  wire _0_;
  (* src = "netlist_2.v:7.26-7.28" *)
  input cp;
  (* src = "netlist_2.v:7.18-7.19" *)
  input d;
  (* src = "netlist_2.v:7.40-7.41" *)
  output q;
  (* src = "netlist_2.v:8.5-11.8" *)
  DFF_X1 _1_ (
    .CK(cp),
    .D(d),
    .Q(q),
    .QN(_0_)
  );
endmodule

(* src = "netlist_2.v:1.1-3.10" *)
module inv(a, b);
  (* src = "netlist_2.v:1.18-1.19" *)
  wire _0_;
  (* src = "netlist_2.v:1.27-1.28" *)
  wire _1_;
  (* src = "netlist_2.v:1.18-1.19" *)
  input a;
  (* src = "netlist_2.v:1.27-1.28" *)
  output b;
  INV_X1 _2_ (
    .A(_0_),
    .ZN(_1_)
  );
  assign _0_ = a;
```

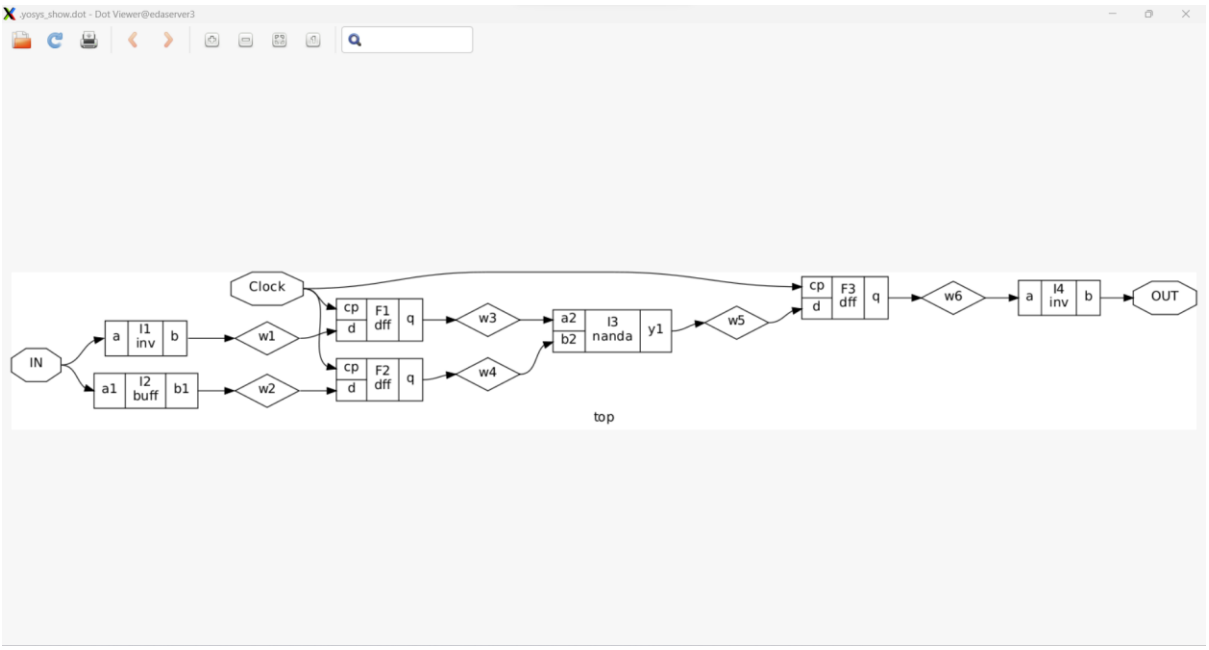
```
  assign b = _1_;
endmodule

(* src = "netlist_2.v:16.1-18.10" *)
module ora(a3, b3, y2);
  (* src = "netlist_2.v:16.18-16.20" *)
  wire _0_;
  (* src = "netlist_2.v:16.27-16.29" *)
  wire _1_;
  (* src = "netlist_2.v:16.37-16.39" *)
  wire _2_;
  (* src = "netlist_2.v:16.18-16.20" *)
  input a3;
  (* src = "netlist_2.v:16.27-16.29" *)
  input b3;
  (* src = "netlist_2.v:16.37-16.39" *)
  output y2;
  OR2_X1 _3_ (
    .A1(_0_),
    .A2(_1_),
    .ZN(_2_)
  );
  assign _0_ = a3;
  assign _1_ = b3;
  assign y2 = _2_;
endmodule

(* top =  1  *)
(* src = "netlist_2.v:19.1-28.10" *)
module top(IN, Clock, OUT);
  (* src = "netlist_2.v:19.27-19.32" *)
  input Clock;
  (* src = "netlist_2.v:19.18-19.20" *)
  input IN;
  (* src = "netlist_2.v:19.40-19.43" *)
  output OUT;
  (* src = "netlist_2.v:20.10-20.12" *)
  wire w1;
  (* src = "netlist_2.v:20.13-20.15" *)
  wire w2;
  (* src = "netlist_2.v:20.16-20.18" *)
  wire w3;
  (* src = "netlist_2.v:20.19-20.21" *)
  wire w4;
  (* src = "netlist_2.v:20.22-20.24" *)
  wire w5;
  (* src = "netlist_2.v:20.25-20.27" *)
  wire w6;
```

```
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_2.v:23.9-23.38" *)
  dff F1 (
    .cp(Clock),
    .d(w1),
    .q(w3)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_2.v:24.9-24.38" *)
  dff F2 (
    .cp(Clock),
    .d(w2),
    .q(w4)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_2.v:26.9-26.38" *)
  dff F3 (
    .cp(Clock),
    .d(w5),
    .q(w6)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_2.v:21.9-21.27" *)
  inv I1 (
    .a(IN),
    .b(w1)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_2.v:22.10-22.30" *)
  buff I2 (
    .a1(IN),
    .b1(w2)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_2.v:25.9-25.37" *)
  ora I3 (
    .a3(w3),
    .b3(w4),
    .y2(w5)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "netlist_2.v:27.9-27.28" *)
  inv I4 (
    .a(w6),
    .b(OUT)
  );
endmodule
```

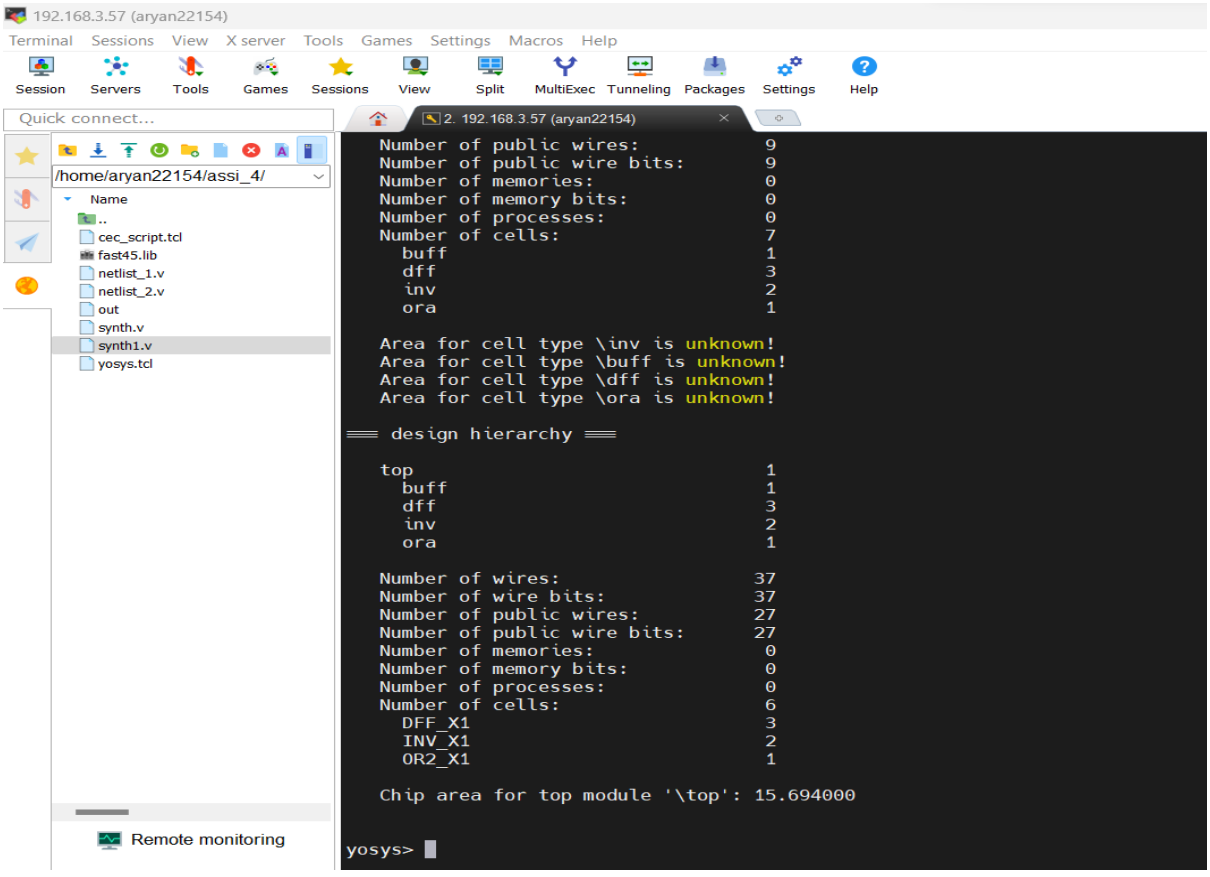
Stats for Model -1



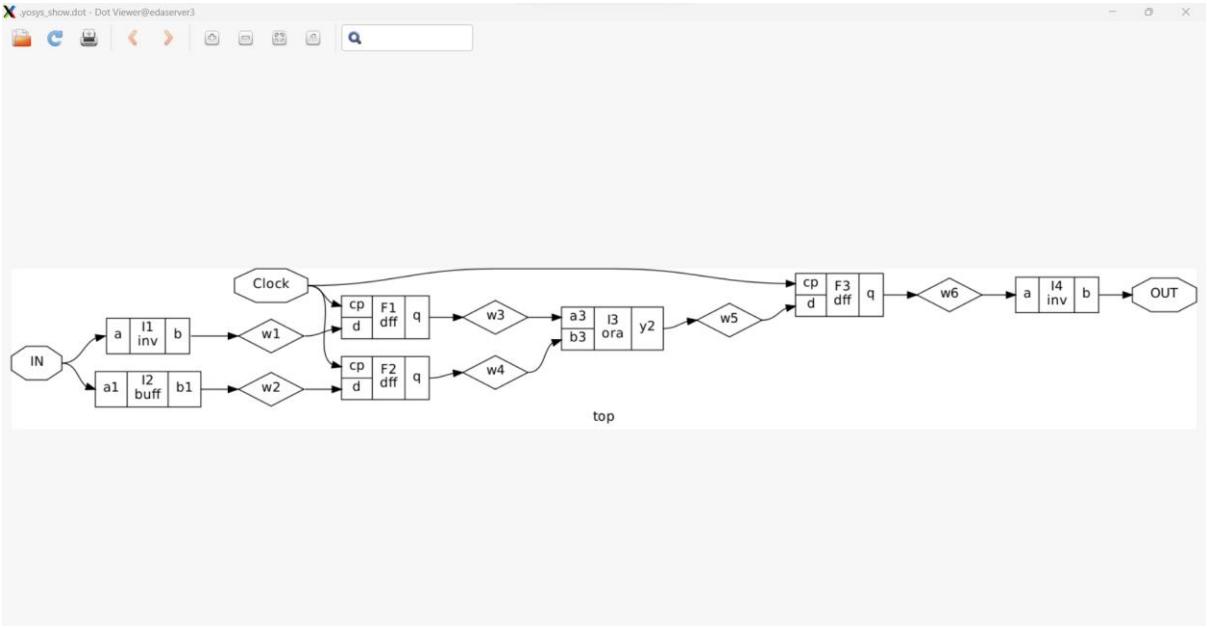
Dot Model for model -1



Stats for Model -2

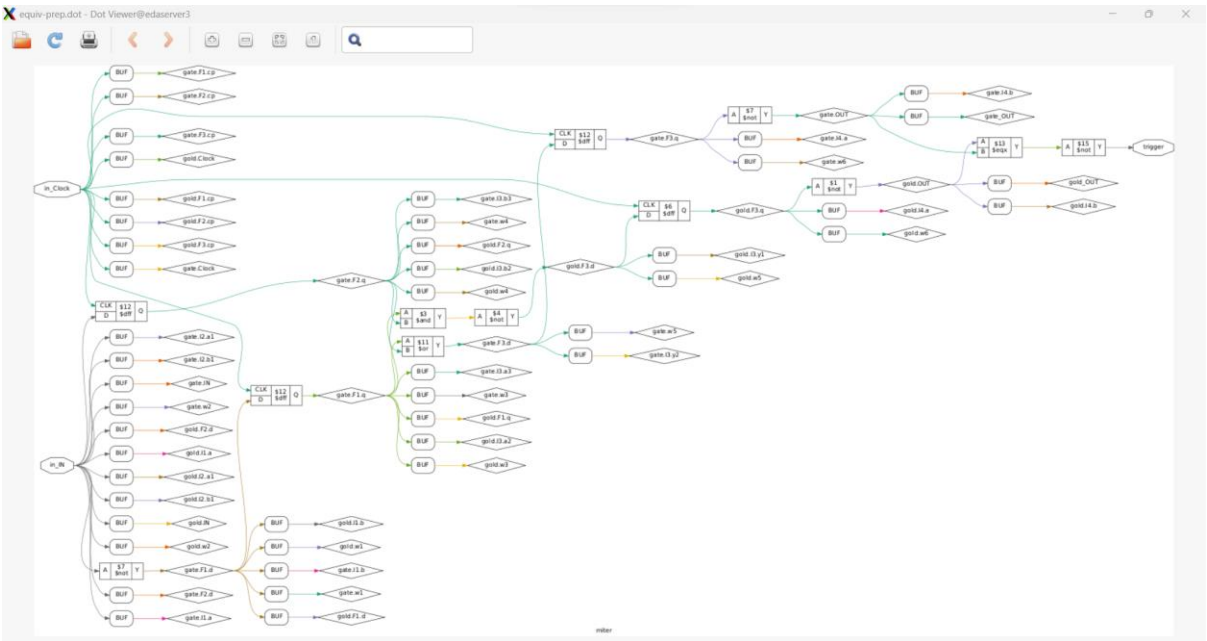


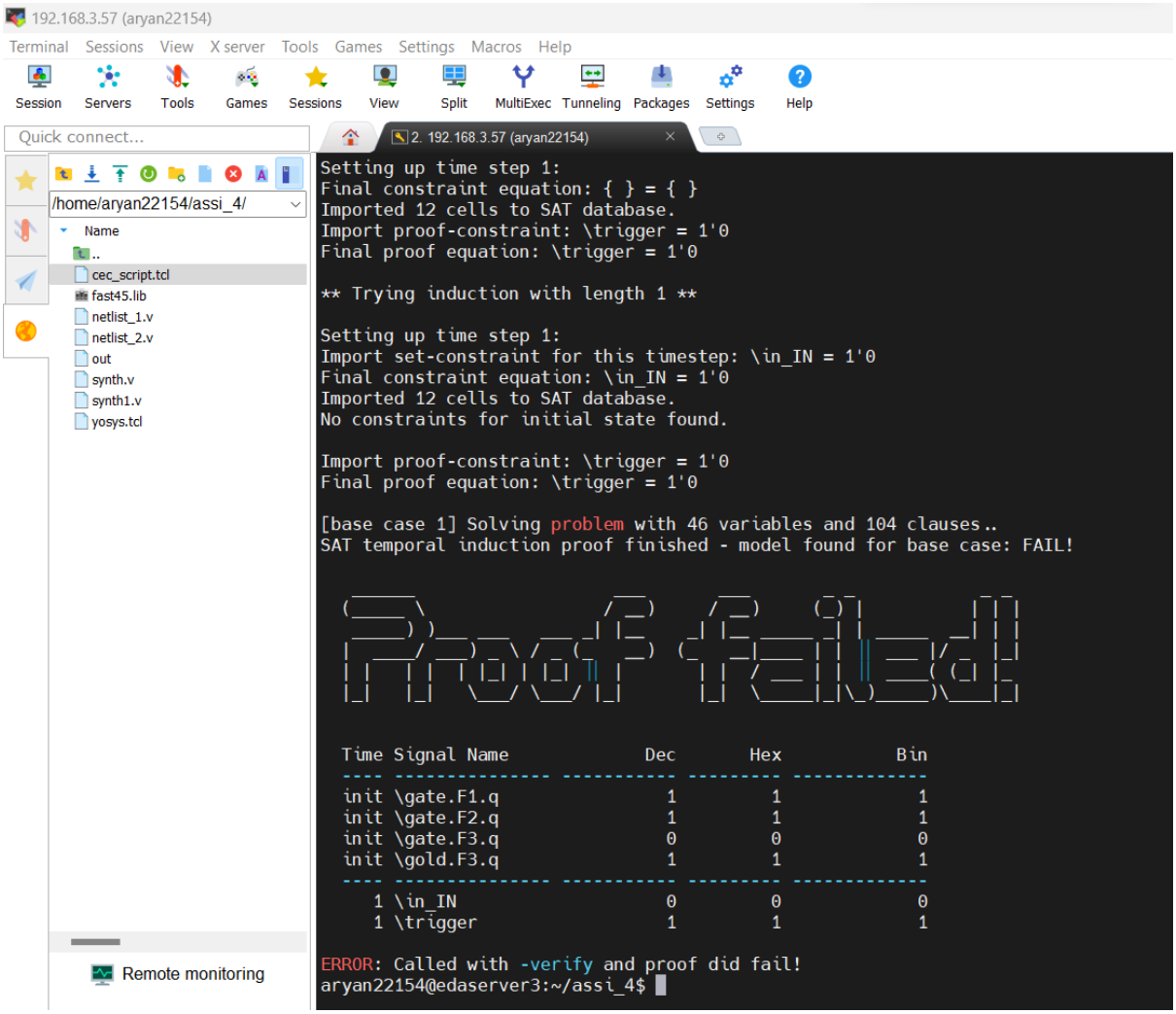
Dot Model for model -2



Library Used – fast45.lib

Q3) -Run the CEC tool for the above models and show its output. Explain the result. You can use the following commands for CEC:





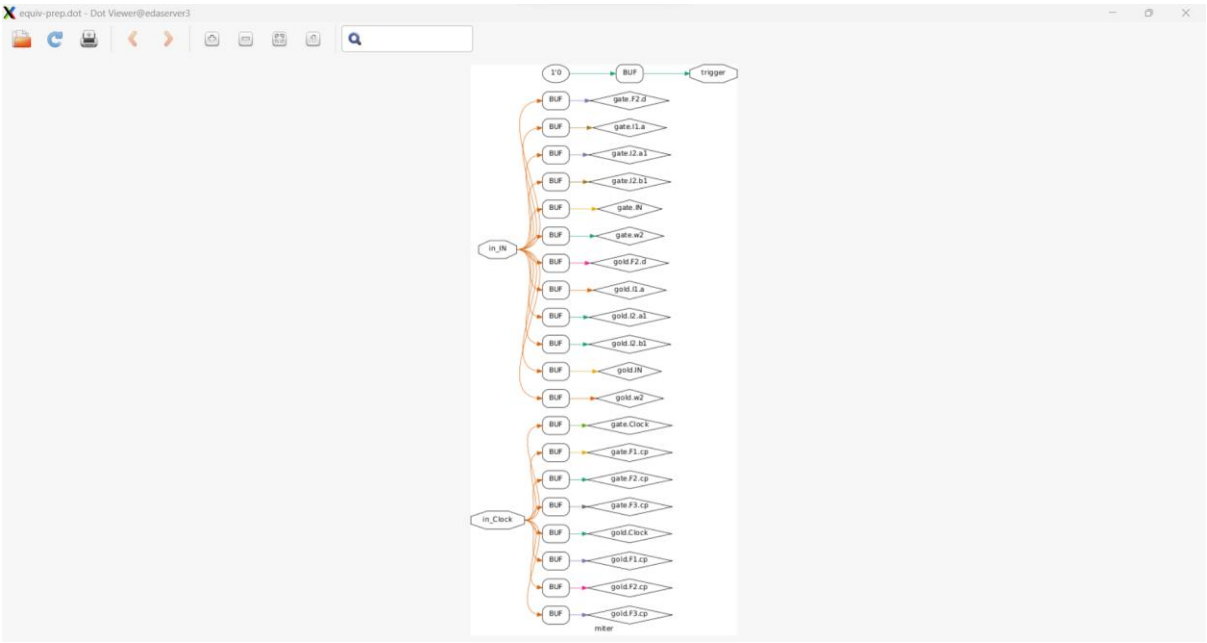
Equivalence checking takes two designs and ascertains if they have the same functionality.

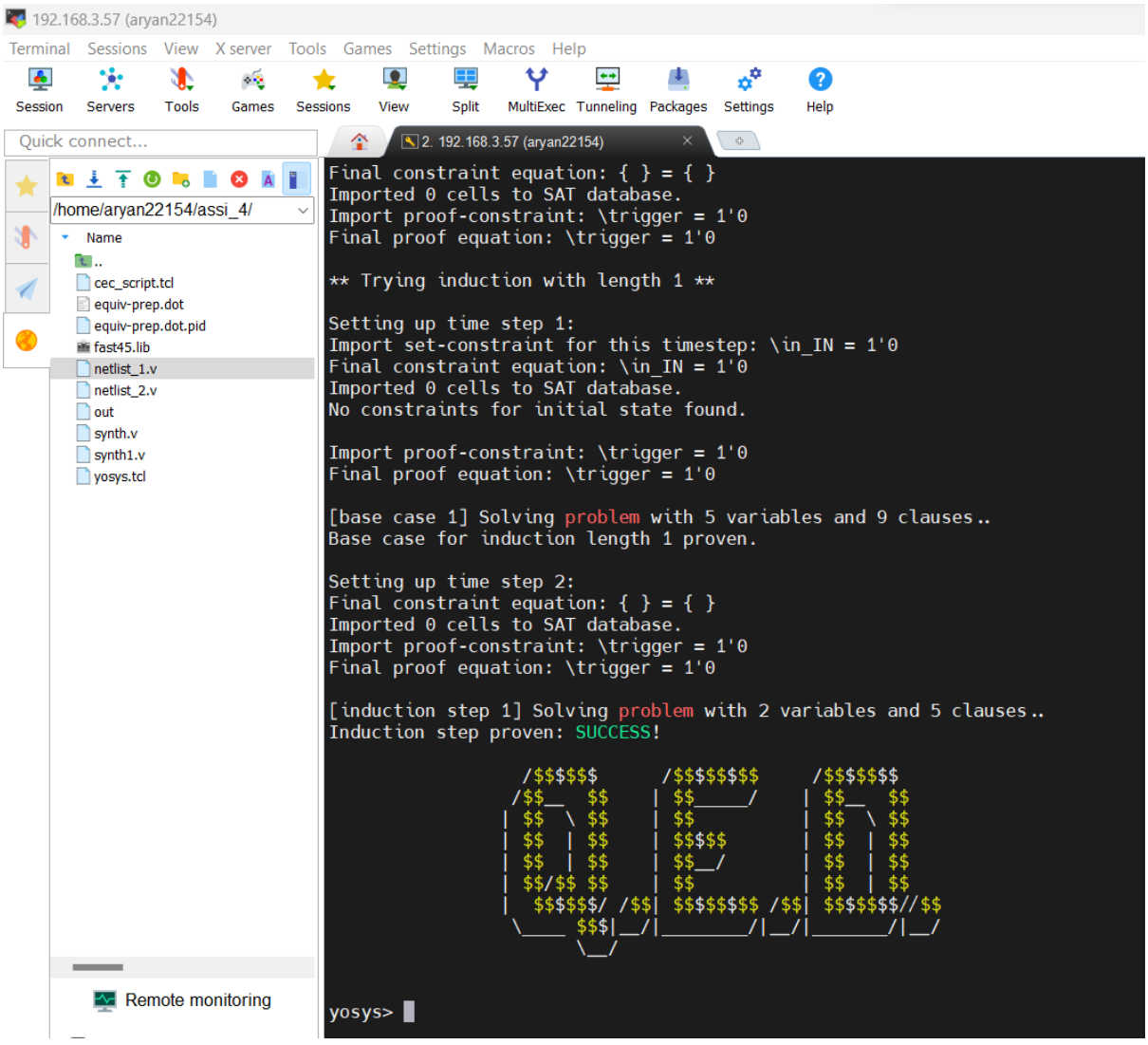
This is important in the design flow where the RTL design, usually described in Verilog or VHDL, is modified by tools and manual methods as it goes through physical implementation.

As I can see, there is a difference in the OR and NAND in the model so they have the different functionality. So proof is failed .

Q4)- Change the OR gate to NAND gate in Model-2. Run the CEC tool for the above models

and show its output. Explain the result.


equiv-prep.dot - Dot Viewer@edaserver3
1'0
BUF
trigger
gate.F2.d
gate.I1.a
gate.I2.a1
gate.I2.b1
gate.IN
gate.w2
in_IN
gold.F2.d
gold.I1.a
gold.I2.a1
gold.I2.b1
gold.IN
gold.w2
gate.Clock
gate.F1.cp
gate.F2.cp
gate.F3.cp
in_Clock
gold.Clock
gold.F1.cp
gold.F2.cp
gold.F3.cp
miter

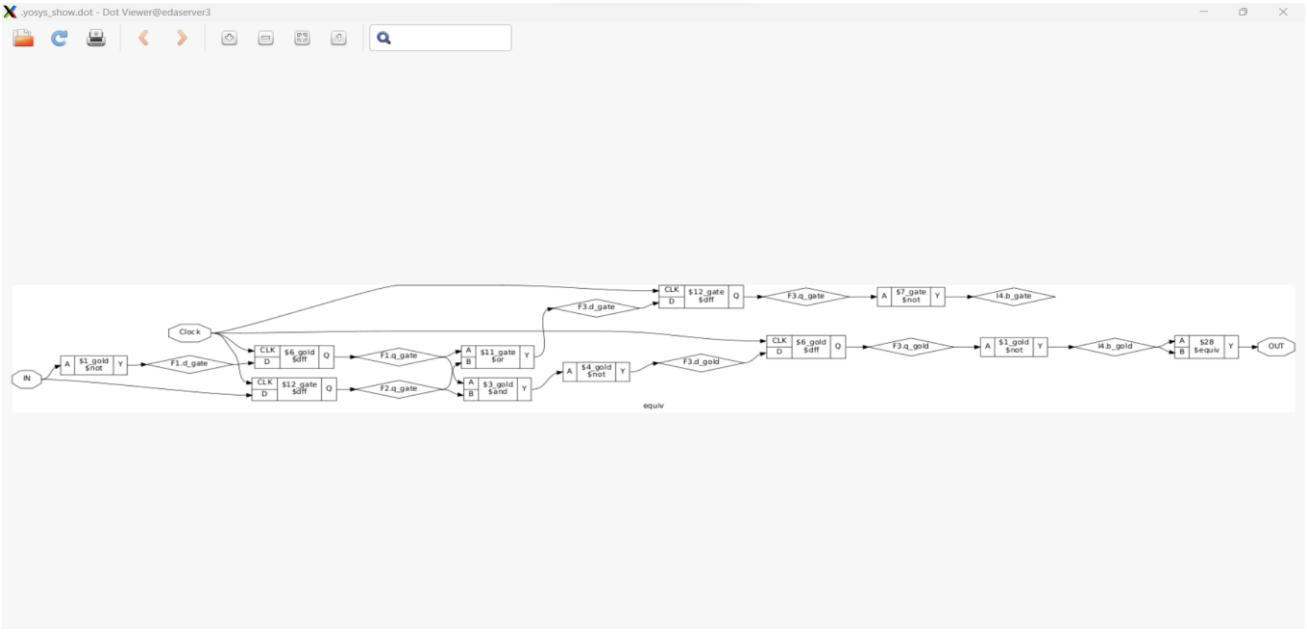

192.168.3.57 (aryan22154)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
2. 192.168.3.57 (aryan22154)
/home/aryan22154/assi_4/
Name
..
cec_script.tcl
equiv-prep.dot
equiv-prep.dot.pid
fast45.lib
netlist_1.v
netlist_2.v
out
synth.v
synth1.v
yosys.tcl
Remote monitoring
Final constraint equation: { } = { }
Imported 0 cells to SAT database.
Import proof-constraint: \trigger = 1'0
Final proof equation: \trigger = 1'0
** Trying induction with length 1 **
Setting up time step 1:
Import set-constraint for this timestep: \in_IN = 1'0
Final constraint equation: \in_IN = 1'0
Imported 0 cells to SAT database.
No constraints for initial state found.
Import proof-constraint: \trigger = 1'0
Final proof equation: \trigger = 1'0
[base case 1] Solving problem with 5 variables and 9 clauses..
Base case for induction length 1 proven.
Setting up time step 2:
Final constraint equation: { } = { }
Imported 0 cells to SAT database.
Import proof-constraint: \trigger = 1'0
Final proof equation: \trigger = 1'0
[induction step 1] Solving problem with 2 variables and 5 clauses..
Induction step proven: SUCCESS!
yosys>

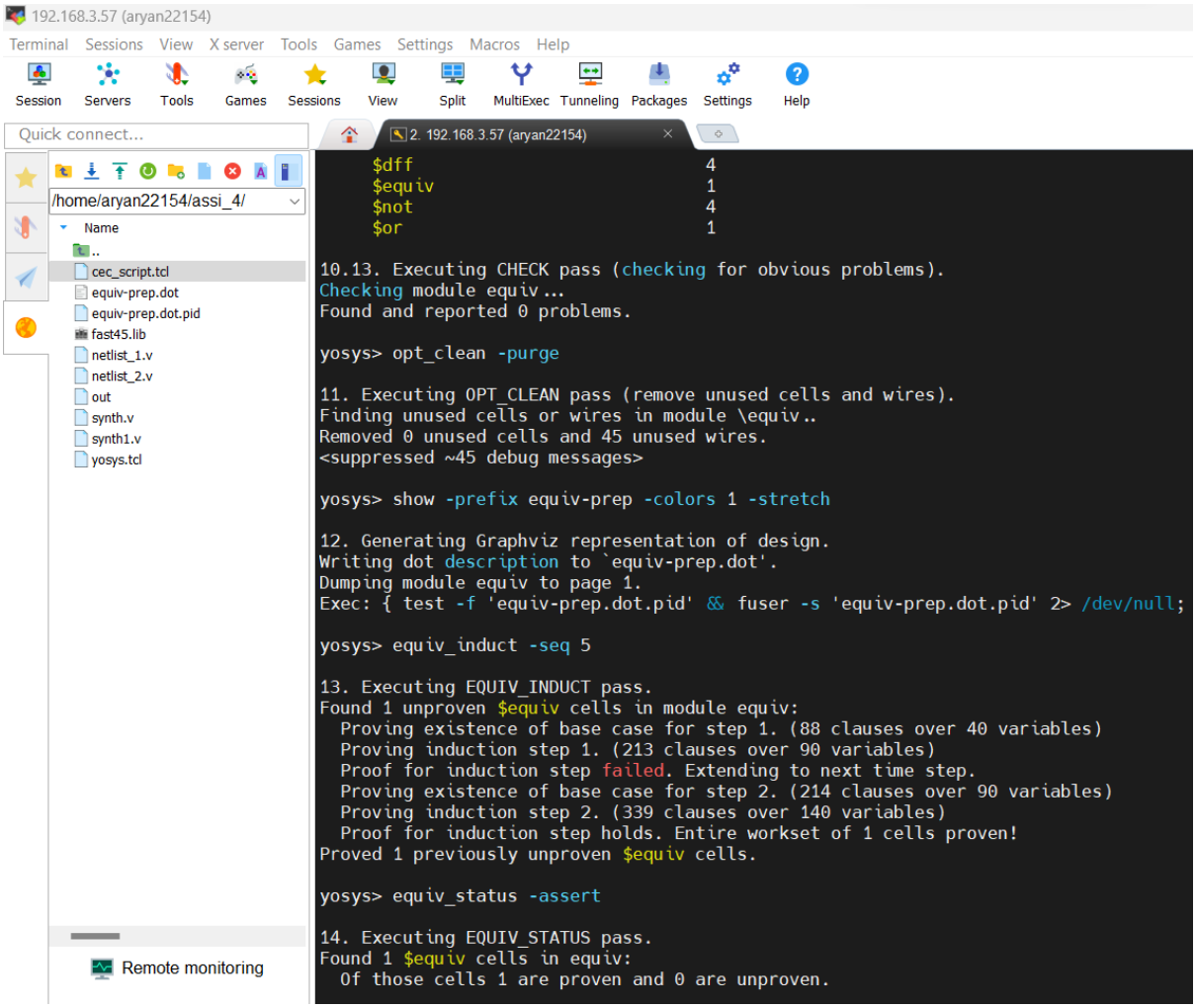
Equivalence checking takes two designs and ascertains if they have the same functionality.

This is important in the design flow where the RTL design, usually described in Verilog or VHDL, is modified by tools and manual methods as it goes through physical implementation.

In this the OR is change to the NAND . So the functionality is get same.

Q5)- Use the netlists shown in the figure in the previous page. Run sequential equivalence checking and show its output. Explain the result. You can use the following commands for sequential equivalence checking:





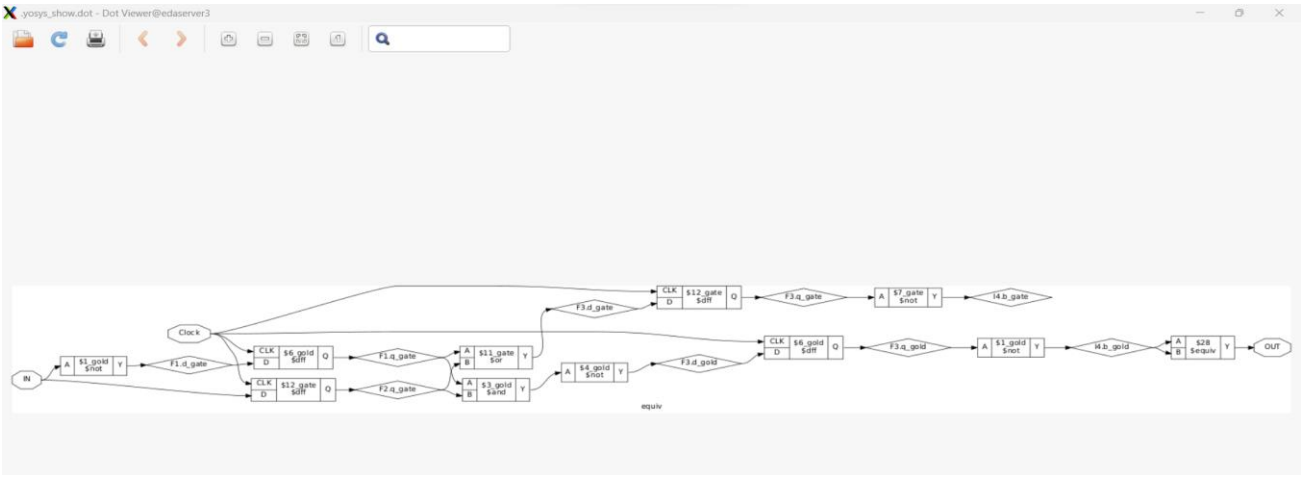
Sequential equivalence checking is a verification process.

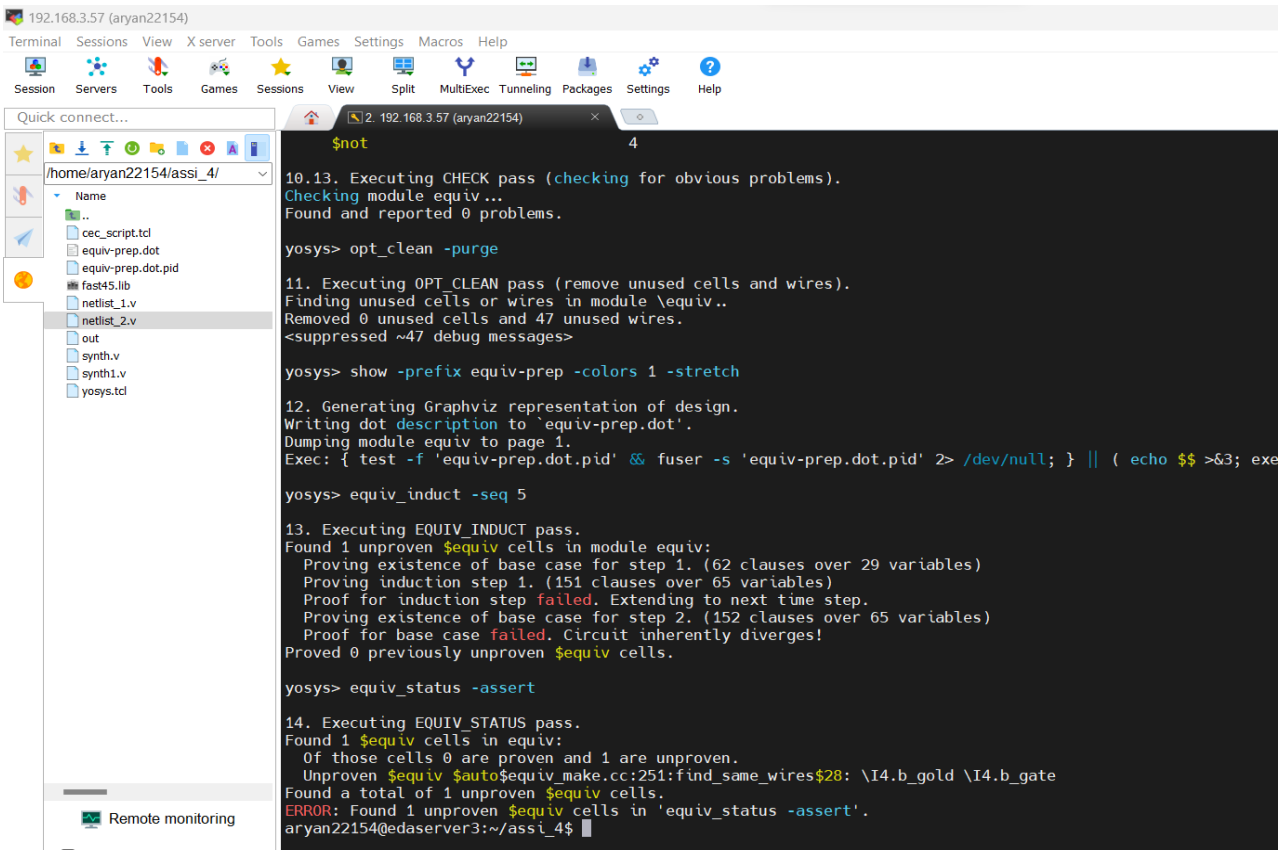
It takes a validated RTL model, considers the specification, and, compares it against a sequentially modified derivation of the model that is considered the implementation.

As I can see in this Equiv_Status is pass and found 1 cells are proven and 0 are unproven.

Q6)- Change the BUF in both the models with INV (inverters) [keep other entities as in Q-5].

Run sequential equivalence checking and show its output. Explain the result.





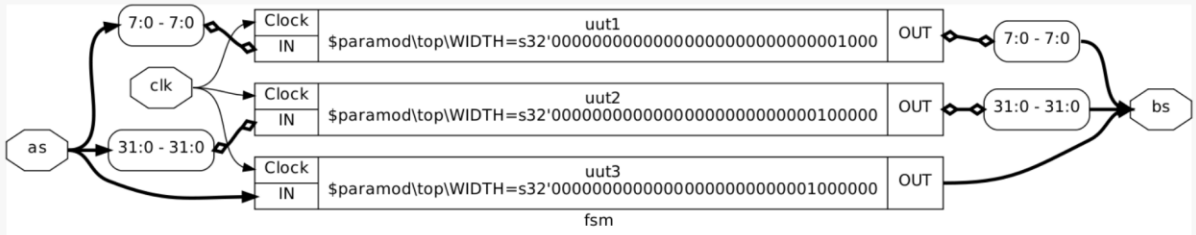
Sequential equivalence checking is a verification process.

It takes a validated RTL model, considers the specification, and, compares it against a sequentially modified derivation of the model that is considered the implementation.

As I can see in this Equiv_Status is pass and found 0 cells are proven and 1 are unproven.

Q7)- Write a simple Verilog module that has a parameter named WIDTH and the size of some ports depend on the parameter WIDTH. Instantiate this module at the top level three times with different parameter values for WIDTH. Report the synthesized netlist and explain the elaboration result.

```
module inv #( parameter WIDTH = 64) (input [WIDTH-1:0]a,output [WIDTH-1:0] b);
    assign b=~a;
endmodule
module buff #( parameter WIDTH = 64) (input [WIDTH-1:0]a1,output [WIDTH-
1:0]b1);
    assign b1=a1;
endmodule
module dff #( parameter WIDTH = 64) (input [WIDTH-1:0] d,input cp,output reg
[WIDTH-1:0] q);
    always @(posedge cp)
    begin
        q<=d;
    end
endmodule
module nanda #( parameter WIDTH = 64) (input [WIDTH-1:0] a2,input [WIDTH-1:0]
b2,output [WIDTH-1:0] y1);
    assign y1=~(a2&b2);
endmodule
module ora #( parameter WIDTH = 64) (input [WIDTH-1:0] a3,input [WIDTH-1:0]
b3,output [WIDTH-1:0] y2);
    assign y2=(a3|b3);
endmodule
module top #( parameter WIDTH = 64) (input [WIDTH-1:0] IN,input Clock,output
[WIDTH-1:0] OUT);
    wire [WIDTH-1:0] w1,w2,w3,w4,w5,w6;
    inv I1 (.a(IN),.b(w1));
    buff I2 (.a1(IN),.b1(w2));
    dff F1 (.d(w1),.cp(Clock),.q(w3));
    dff F2 (.d(w2),.cp(Clock),.q(w4));
    nanda I3 (.a2(w3),.b2(w4),.y1(w5));
    dff F3 (.d(w5),.cp(Clock),.q(w6));
    inv I4 (.a(w6),.b(OUT));
endmodule
module fsm (input [63:0] as,input clk,output [63:0] bs);
    top #(.WIDTH(8)) uut1(.IN(as[7:0]),.Clock(clk),.OUT(bs[7:0]));
    top #(.WIDTH(32)) uut2(.IN(as[31:0]),.Clock(clk),.OUT(bs[31:0]));
    top #(.WIDTH(64)) uut3(.IN(as[63:0]),.Clock(clk),.OUT(bs[63:0]));
endmodule
```




```
/* Generated by Yosys 0.11+10 (git sha1 4871d8f19, clang 6.0.0-1ubuntu2 -fPIC
-Os) */

(* dynports =  1  *)
(* hdlname = "\\top" *)
(* src = "net1.v:19.1-28.10" *)
module \$paramod\top\WIDTH=s32'00000000000000000000000000001000 (IN, Clock,
OUT);
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
49 50 51 52 53 54 55" *)
  wire [55:0] _0_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
49 50 51 52 53 54 55" *)
  wire [55:0] _1_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
49 50 51 52 53 54 55" *)
  wire [55:0] _2_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
49 50 51 52 53 54 55" *)
  wire [55:0] _3_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
49 50 51 52 53 54 55" *)
  wire [55:0] _4_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
49 50 51 52 53 54 55" *)
  wire [55:0] _5_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
49 50 51 52 53 54 55" *)
  wire [55:0] _6_;
  (* src = "net1.v:19.65-19.70" *)
```

```
input Clock;
(* src = "net1.v:19.56-19.58" *)
input [7:0] IN;
(* src = "net1.v:19.90-19.93" *)
output [7:0] OUT;
(* src = "net1.v:20.22-20.24" *)
wire [7:0] w1;
(* src = "net1.v:20.25-20.27" *)
wire [7:0] w2;
(* src = "net1.v:20.28-20.30" *)
wire [7:0] w3;
(* src = "net1.v:20.31-20.33" *)
wire [7:0] w4;
(* src = "net1.v:20.34-20.36" *)
wire [7:0] w5;
(* src = "net1.v:20.37-20.39" *)
wire [7:0] w6;
(* module_not_derived = 32'd1 *)
(* src = "net1.v:23.9-23.38" *)
dff F1 (
  .cp(Clock),
  .d({ 56'h00000000000000, w1 }),
  .q({ _0_, w3 })
);
(* module_not_derived = 32'd1 *)
(* src = "net1.v:24.9-24.38" *)
dff F2 (
  .cp(Clock),
  .d({ 56'h00000000000000, w2 }),
  .q({ _6_, w4 })
);
(* module_not_derived = 32'd1 *)
(* src = "net1.v:26.9-26.38" *)
dff F3 (
  .cp(Clock),
  .d({ 56'h00000000000000, w5 }),
  .q({ _4_, w6 })
);
(* module_not_derived = 32'd1 *)
(* src = "net1.v:21.9-21.27" *)
inv I1 (
  .a({ 56'h00000000000000, IN }),
  .b({ _2_, w1 })
);
(* module_not_derived = 32'd1 *)
(* src = "net1.v:22.10-22.30" *)
buff I2 (
  .a1({ 56'h00000000000000, IN }),
```

```
    .b1({ _1_, w2 })
  );
  (* module_not_derived = 32'd1 *)
  (* src = "net1.v:25.11-25.39" *)
  nanda I3 (
    .a2({ 56'h00000000000000, w3 }),
    .b2({ 56'h00000000000000, w4 }),
    .y1({ _5_, w5 })
  );
  (* module_not_derived = 32'd1 *)
  (* src = "net1.v:27.9-27.28" *)
  inv I4 (
    .a({ 56'h00000000000000, w6 }),
    .b({ _3_, OUT })
  );
endmodule

(* dynports =  1  *)
(* hdlname = "\\top" *)
(* src = "net1.v:19.1-28.10" *)
module \$paramod\top\WIDTH=s32'00000000000000000000000000100000 (IN, Clock,
OUT);
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31" *)
  wire [31:0] _0_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31" *)
  wire [31:0] _1_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31" *)
  wire [31:0] _2_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31" *)
  wire [31:0] _3_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31" *)
  wire [31:0] _4_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31" *)
  wire [31:0] _5_;
  (* unused_bits = "0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31" *)
  wire [31:0] _6_;
  (* src = "net1.v:19.65-19.70" *)
  input Clock;
  (* src = "net1.v:19.56-19.58" *)
  input [31:0] IN;
  (* src = "net1.v:19.90-19.93" *)
```

```
output [31:0] OUT;
(* src = "net1.v:20.22-20.24" *)
wire [31:0] w1;
(* src = "net1.v:20.25-20.27" *)
wire [31:0] w2;
(* src = "net1.v:20.28-20.30" *)
wire [31:0] w3;
(* src = "net1.v:20.31-20.33" *)
wire [31:0] w4;
(* src = "net1.v:20.34-20.36" *)
wire [31:0] w5;
(* src = "net1.v:20.37-20.39" *)
wire [31:0] w6;
(* module_not_derived = 32'd1 *)
(* src = "net1.v:23.9-23.38" *)
dff F1 (
  .cp(Clock),
  .d({ 32'h00000000, w1 }),
  .q({ _4_, w3 })
);
(* module_not_derived = 32'd1 *)
(* src = "net1.v:24.9-24.38" *)
dff F2 (
  .cp(Clock),
  .d({ 32'h00000000, w2 }),
  .q({ _3_, w4 })
);
(* module_not_derived = 32'd1 *)
(* src = "net1.v:26.9-26.38" *)
dff F3 (
  .cp(Clock),
  .d({ 32'h00000000, w5 }),
  .q({ _1_, w6 })
);
(* module_not_derived = 32'd1 *)
(* src = "net1.v:21.9-21.27" *)
inv I1 (
  .a({ 32'h00000000, IN }),
  .b({ _6_, w1 })
);
(* module_not_derived = 32'd1 *)
(* src = "net1.v:22.10-22.30" *)
buff I2 (
  .a1({ 32'h00000000, IN }),
  .b1({ _5_, w2 })
);
(* module_not_derived = 32'd1 *)
(* src = "net1.v:25.11-25.39" *)
```

```
  nanda I3 (
    .a2({ 32'h00000000, w3 }),
    .b2({ 32'h00000000, w4 }),
    .y1({ _2_, w5 })
  );
  (* module_not_derived = 32'd1 *)
  (* src = "net1.v:27.9-27.28" *)
  inv I4 (
    .a({ 32'h00000000, w6 }),
    .b({ _0_, OUT })
  );
endmodule

(* dynports =  1  *)
(* hdlname = "\\top" *)
(* src = "net1.v:19.1-28.10" *)
module \$paramod\top\WIDTH=s32'00000000000000000000000001000000 (IN, Clock,
OUT);
  (* src = "net1.v:19.65-19.70" *)
  input Clock;
  (* src = "net1.v:19.56-19.58" *)
  input [63:0] IN;
  (* src = "net1.v:19.90-19.93" *)
  output [63:0] OUT;
  (* src = "net1.v:20.22-20.24" *)
  wire [63:0] w1;
  (* src = "net1.v:20.25-20.27" *)
  wire [63:0] w2;
  (* src = "net1.v:20.28-20.30" *)
  wire [63:0] w3;
  (* src = "net1.v:20.31-20.33" *)
  wire [63:0] w4;
  (* src = "net1.v:20.34-20.36" *)
  wire [63:0] w5;
  (* src = "net1.v:20.37-20.39" *)
  wire [63:0] w6;
  (* module_not_derived = 32'd1 *)
  (* src = "net1.v:23.9-23.38" *)
  dff F1 (
    .cp(Clock),
    .d(w1),
    .q(w3)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "net1.v:24.9-24.38" *)
  dff F2 (
    .cp(Clock),
    .d(w2),
```

```
    .q(w4)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "net1.v:26.9-26.38" *)
  dff F3 (
    .cp(Clock),
    .d(w5),
    .q(w6)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "net1.v:21.9-21.27" *)
  inv I1 (
    .a(IN),
    .b(w1)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "net1.v:22.10-22.30" *)
  buff I2 (
    .a1(IN),
    .b1(w2)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "net1.v:25.11-25.39" *)
  nanda I3 (
    .a2(w3),
    .b2(w4),
    .y1(w5)
  );
  (* module_not_derived = 32'd1 *)
  (* src = "net1.v:27.9-27.28" *)
  inv I4 (
    .a(w6),
    .b(OUT)
  );
endmodule

(* dynports =  1  *)
(* src = "net1.v:4.1-6.10" *)
module buff(a1, b1);
  (* src = "net1.v:4.56-4.58" *)
  input [63:0] a1;
  (* src = "net1.v:4.77-4.79" *)
  output [63:0] b1;
  assign b1 = a1;
endmodule

(* dynports =  1  *)
(* src = "net1.v:7.1-12.10" *)
```

```
module dff(d, cp, q);
  wire _000_;
  wire _001_;
  wire _002_;
  wire _003_;
  wire _004_;
  wire _005_;
  wire _006_;
  wire _007_;
  wire _008_;
  wire _009_;
  wire _010_;
  wire _011_;
  wire _012_;
  wire _013_;
  wire _014_;
  wire _015_;
  wire _016_;
  wire _017_;
  wire _018_;
  wire _019_;
  wire _020_;
  wire _021_;
  wire _022_;
  wire _023_;
  wire _024_;
  wire _025_;
  wire _026_;
  wire _027_;
  wire _028_;
  wire _029_;
  wire _030_;
  wire _031_;
  wire _032_;
  wire _033_;
  wire _034_;
  wire _035_;
  wire _036_;
  wire _037_;
  wire _038_;
  wire _039_;
  wire _040_;
  wire _041_;
  wire _042_;
  wire _043_;
  wire _044_;
  wire _045_;
  wire _046_;
```

```
wire _047_;
wire _048_;
wire _049_;
wire _050_;
wire _051_;
wire _052_;
wire _053_;
wire _054_;
wire _055_;
wire _056_;
wire _057_;
wire _058_;
wire _059_;
wire _060_;
wire _061_;
wire _062_;
wire _063_;
(* src = "net1.v:7.64-7.66" *)
input cp;
(* src = "net1.v:7.56-7.57" *)
input [63:0] d;
(* src = "net1.v:7.90-7.91" *)
output [63:0] q;
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _064_ (
  .CK(cp),
  .D(d[0]),
  .Q(q[0]),
  .QN(_001_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _065_ (
  .CK(cp),
  .D(d[1]),
  .Q(q[1]),
  .QN(_002_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _066_ (
  .CK(cp),
  .D(d[2]),
  .Q(q[2]),
  .QN(_003_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _067_ (
  .CK(cp),
  .D(d[3]),
```

```
  .Q(q[3]),
  .QN(_004_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _068_ (
  .CK(cp),
  .D(d[4]),
  .Q(q[4]),
  .QN(_005_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _069_ (
  .CK(cp),
  .D(d[5]),
  .Q(q[5]),
  .QN(_006_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _070_ (
  .CK(cp),
  .D(d[6]),
  .Q(q[6]),
  .QN(_007_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _071_ (
  .CK(cp),
  .D(d[7]),
  .Q(q[7]),
  .QN(_008_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _072_ (
  .CK(cp),
  .D(d[8]),
  .Q(q[8]),
  .QN(_009_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _073_ (
  .CK(cp),
  .D(d[9]),
  .Q(q[9]),
  .QN(_010_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _074_ (
  .CK(cp),
```

```
  .D(d[10]),
  .Q(q[10]),
  .QN(_011_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _075_ (
  .CK(cp),
  .D(d[11]),
  .Q(q[11]),
  .QN(_012_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _076_ (
  .CK(cp),
  .D(d[12]),
  .Q(q[12]),
  .QN(_013_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _077_ (
  .CK(cp),
  .D(d[13]),
  .Q(q[13]),
  .QN(_014_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _078_ (
  .CK(cp),
  .D(d[14]),
  .Q(q[14]),
  .QN(_015_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _079_ (
  .CK(cp),
  .D(d[15]),
  .Q(q[15]),
  .QN(_016_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _080_ (
  .CK(cp),
  .D(d[16]),
  .Q(q[16]),
  .QN(_017_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _081_ (
```

```
  .CK(cp),
  .D(d[17]),
  .Q(q[17]),
  .QN(_018_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _082_ (
  .CK(cp),
  .D(d[18]),
  .Q(q[18]),
  .QN(_019_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _083_ (
  .CK(cp),
  .D(d[19]),
  .Q(q[19]),
  .QN(_020_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _084_ (
  .CK(cp),
  .D(d[20]),
  .Q(q[20]),
  .QN(_021_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _085_ (
  .CK(cp),
  .D(d[21]),
  .Q(q[21]),
  .QN(_022_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _086_ (
  .CK(cp),
  .D(d[22]),
  .Q(q[22]),
  .QN(_023_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _087_ (
  .CK(cp),
  .D(d[23]),
  .Q(q[23]),
  .QN(_024_)
);
(* src = "net1.v:8.5-11.8" *)
```

```
  DFF_X1 _088_ (
    .CK(cp),
    .D(d[24]),
    .Q(q[24]),
    .QN(_025_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _089_ (
    .CK(cp),
    .D(d[25]),
    .Q(q[25]),
    .QN(_026_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _090_ (
    .CK(cp),
    .D(d[26]),
    .Q(q[26]),
    .QN(_027_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _091_ (
    .CK(cp),
    .D(d[27]),
    .Q(q[27]),
    .QN(_028_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _092_ (
    .CK(cp),
    .D(d[28]),
    .Q(q[28]),
    .QN(_029_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _093_ (
    .CK(cp),
    .D(d[29]),
    .Q(q[29]),
    .QN(_030_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _094_ (
    .CK(cp),
    .D(d[30]),
    .Q(q[30]),
    .QN(_031_)
  );
```

```
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _095_ (
  .CK(cp),
  .D(d[31]),
  .Q(q[31]),
  .QN(_032_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _096_ (
  .CK(cp),
  .D(d[32]),
  .Q(q[32]),
  .QN(_033_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _097_ (
  .CK(cp),
  .D(d[33]),
  .Q(q[33]),
  .QN(_034_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _098_ (
  .CK(cp),
  .D(d[34]),
  .Q(q[34]),
  .QN(_035_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _099_ (
  .CK(cp),
  .D(d[35]),
  .Q(q[35]),
  .QN(_036_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _100_ (
  .CK(cp),
  .D(d[36]),
  .Q(q[36]),
  .QN(_037_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _101_ (
  .CK(cp),
  .D(d[37]),
  .Q(q[37]),
  .QN(_038_)
```

```
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _102_ (
  .CK(cp),
  .D(d[38]),
  .Q(q[38]),
  .QN(_039_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _103_ (
  .CK(cp),
  .D(d[39]),
  .Q(q[39]),
  .QN(_040_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _104_ (
  .CK(cp),
  .D(d[40]),
  .Q(q[40]),
  .QN(_041_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _105_ (
  .CK(cp),
  .D(d[41]),
  .Q(q[41]),
  .QN(_042_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _106_ (
  .CK(cp),
  .D(d[42]),
  .Q(q[42]),
  .QN(_043_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _107_ (
  .CK(cp),
  .D(d[43]),
  .Q(q[43]),
  .QN(_044_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _108_ (
  .CK(cp),
  .D(d[44]),
  .Q(q[44]),
```

```
  .QN(_045_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _109_ (
  .CK(cp),
  .D(d[45]),
  .Q(q[45]),
  .QN(_046_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _110_ (
  .CK(cp),
  .D(d[46]),
  .Q(q[46]),
  .QN(_047_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _111_ (
  .CK(cp),
  .D(d[47]),
  .Q(q[47]),
  .QN(_048_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _112_ (
  .CK(cp),
  .D(d[48]),
  .Q(q[48]),
  .QN(_049_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _113_ (
  .CK(cp),
  .D(d[49]),
  .Q(q[49]),
  .QN(_050_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _114_ (
  .CK(cp),
  .D(d[50]),
  .Q(q[50]),
  .QN(_051_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _115_ (
  .CK(cp),
  .D(d[51]),
```

```
  .Q(q[51]),
  .QN(_052_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _116_ (
  .CK(cp),
  .D(d[52]),
  .Q(q[52]),
  .QN(_053_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _117_ (
  .CK(cp),
  .D(d[53]),
  .Q(q[53]),
  .QN(_054_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _118_ (
  .CK(cp),
  .D(d[54]),
  .Q(q[54]),
  .QN(_055_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _119_ (
  .CK(cp),
  .D(d[55]),
  .Q(q[55]),
  .QN(_056_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _120_ (
  .CK(cp),
  .D(d[56]),
  .Q(q[56]),
  .QN(_057_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _121_ (
  .CK(cp),
  .D(d[57]),
  .Q(q[57]),
  .QN(_058_)
);
(* src = "net1.v:8.5-11.8" *)
DFF_X1 _122_ (
  .CK(cp),
```

```
    .D(d[58]),
    .Q(q[58]),
    .QN(_059_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _123_ (
    .CK(cp),
    .D(d[59]),
    .Q(q[59]),
    .QN(_060_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _124_ (
    .CK(cp),
    .D(d[60]),
    .Q(q[60]),
    .QN(_061_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _125_ (
    .CK(cp),
    .D(d[61]),
    .Q(q[61]),
    .QN(_062_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _126_ (
    .CK(cp),
    .D(d[62]),
    .Q(q[62]),
    .QN(_063_)
  );
  (* src = "net1.v:8.5-11.8" *)
  DFF_X1 _127_ (
    .CK(cp),
    .D(d[63]),
    .Q(q[63]),
    .QN(_000_)
  );
endmodule

(* top =  1  *)
(* src = "net1.v:29.1-33.10" *)
module fsm(as, clk, bs);
  (* src = "net1.v:29.26-29.28" *)
  input [63:0] as;
  (* src = "net1.v:29.53-29.55" *)
  output [63:0] bs;
```

```
  (* src = "net1.v:29.35-29.38" *)
  input clk;
  (* src = "net1.v:30.22-30.66" *)
  \$paramod\top\WIDTH=s32'00000000000000000000000000001000  uut1 (
    .Clock(clk),
    .IN(as[7:0]),
    .OUT(bs[7:0])
  );
  (* src = "net1.v:31.23-31.69" *)
  \$paramod\top\WIDTH=s32'00000000000000000000000000100000  uut2 (
    .Clock(clk),
    .IN(as[31:0]),
    .OUT(bs[31:0])
  );
  (* src = "net1.v:32.23-32.69" *)
  \$paramod\top\WIDTH=s32'00000000000000000000000001000000  uut3 (
    .Clock(clk),
    .IN(as),
    .OUT(bs)
  );
endmodule

(* dynports =  1  *)
(* src = "net1.v:1.1-3.10" *)
module inv(a, b);
  (* src = "net1.v:1.55-1.56" *)
  wire _000_;
  (* src = "net1.v:1.55-1.56" *)
  wire _001_;
  (* src = "net1.v:1.55-1.56" *)
  wire _002_;
  (* src = "net1.v:1.55-1.56" *)
  wire _003_;
  (* src = "net1.v:1.55-1.56" *)
  wire _004_;
  (* src = "net1.v:1.55-1.56" *)
  wire _005_;
  (* src = "net1.v:1.55-1.56" *)
  wire _006_;
  (* src = "net1.v:1.55-1.56" *)
  wire _007_;
  (* src = "net1.v:1.55-1.56" *)
  wire _008_;
  (* src = "net1.v:1.55-1.56" *)
  wire _009_;
  (* src = "net1.v:1.55-1.56" *)
  wire _010_;
  (* src = "net1.v:1.55-1.56" *)
```

```
wire _011_;
(* src = "net1.v:1.55-1.56" *)
wire _012_;
(* src = "net1.v:1.55-1.56" *)
wire _013_;
(* src = "net1.v:1.55-1.56" *)
wire _014_;
(* src = "net1.v:1.55-1.56" *)
wire _015_;
(* src = "net1.v:1.55-1.56" *)
wire _016_;
(* src = "net1.v:1.55-1.56" *)
wire _017_;
(* src = "net1.v:1.55-1.56" *)
wire _018_;
(* src = "net1.v:1.55-1.56" *)
wire _019_;
(* src = "net1.v:1.55-1.56" *)
wire _020_;
(* src = "net1.v:1.55-1.56" *)
wire _021_;
(* src = "net1.v:1.55-1.56" *)
wire _022_;
(* src = "net1.v:1.55-1.56" *)
wire _023_;
(* src = "net1.v:1.55-1.56" *)
wire _024_;
(* src = "net1.v:1.55-1.56" *)
wire _025_;
(* src = "net1.v:1.55-1.56" *)
wire _026_;
(* src = "net1.v:1.55-1.56" *)
wire _027_;
(* src = "net1.v:1.55-1.56" *)
wire _028_;
(* src = "net1.v:1.55-1.56" *)
wire _029_;
(* src = "net1.v:1.55-1.56" *)
wire _030_;
(* src = "net1.v:1.55-1.56" *)
wire _031_;
(* src = "net1.v:1.55-1.56" *)
wire _032_;
(* src = "net1.v:1.55-1.56" *)
wire _033_;
(* src = "net1.v:1.55-1.56" *)
wire _034_;
(* src = "net1.v:1.55-1.56" *)
```

```
wire _035_;
(* src = "net1.v:1.55-1.56" *)
wire _036_;
(* src = "net1.v:1.55-1.56" *)
wire _037_;
(* src = "net1.v:1.55-1.56" *)
wire _038_;
(* src = "net1.v:1.55-1.56" *)
wire _039_;
(* src = "net1.v:1.55-1.56" *)
wire _040_;
(* src = "net1.v:1.55-1.56" *)
wire _041_;
(* src = "net1.v:1.55-1.56" *)
wire _042_;
(* src = "net1.v:1.55-1.56" *)
wire _043_;
(* src = "net1.v:1.55-1.56" *)
wire _044_;
(* src = "net1.v:1.55-1.56" *)
wire _045_;
(* src = "net1.v:1.55-1.56" *)
wire _046_;
(* src = "net1.v:1.55-1.56" *)
wire _047_;
(* src = "net1.v:1.55-1.56" *)
wire _048_;
(* src = "net1.v:1.55-1.56" *)
wire _049_;
(* src = "net1.v:1.55-1.56" *)
wire _050_;
(* src = "net1.v:1.55-1.56" *)
wire _051_;
(* src = "net1.v:1.55-1.56" *)
wire _052_;
(* src = "net1.v:1.55-1.56" *)
wire _053_;
(* src = "net1.v:1.55-1.56" *)
wire _054_;
(* src = "net1.v:1.55-1.56" *)
wire _055_;
(* src = "net1.v:1.55-1.56" *)
wire _056_;
(* src = "net1.v:1.55-1.56" *)
wire _057_;
(* src = "net1.v:1.55-1.56" *)
wire _058_;
(* src = "net1.v:1.55-1.56" *)
```

```
wire _059_;
(* src = "net1.v:1.55-1.56" *)
wire _060_;
(* src = "net1.v:1.55-1.56" *)
wire _061_;
(* src = "net1.v:1.55-1.56" *)
wire _062_;
(* src = "net1.v:1.55-1.56" *)
wire _063_;
(* src = "net1.v:1.76-1.77" *)
wire _064_;
(* src = "net1.v:1.76-1.77" *)
wire _065_;
(* src = "net1.v:1.76-1.77" *)
wire _066_;
(* src = "net1.v:1.76-1.77" *)
wire _067_;
(* src = "net1.v:1.76-1.77" *)
wire _068_;
(* src = "net1.v:1.76-1.77" *)
wire _069_;
(* src = "net1.v:1.76-1.77" *)
wire _070_;
(* src = "net1.v:1.76-1.77" *)
wire _071_;
(* src = "net1.v:1.76-1.77" *)
wire _072_;
(* src = "net1.v:1.76-1.77" *)
wire _073_;
(* src = "net1.v:1.76-1.77" *)
wire _074_;
(* src = "net1.v:1.76-1.77" *)
wire _075_;
(* src = "net1.v:1.76-1.77" *)
wire _076_;
(* src = "net1.v:1.76-1.77" *)
wire _077_;
(* src = "net1.v:1.76-1.77" *)
wire _078_;
(* src = "net1.v:1.76-1.77" *)
wire _079_;
(* src = "net1.v:1.76-1.77" *)
wire _080_;
(* src = "net1.v:1.76-1.77" *)
wire _081_;
(* src = "net1.v:1.76-1.77" *)
wire _082_;
(* src = "net1.v:1.76-1.77" *)
```

```
wire _083_;
(* src = "net1.v:1.76-1.77" *)
wire _084_;
(* src = "net1.v:1.76-1.77" *)
wire _085_;
(* src = "net1.v:1.76-1.77" *)
wire _086_;
(* src = "net1.v:1.76-1.77" *)
wire _087_;
(* src = "net1.v:1.76-1.77" *)
wire _088_;
(* src = "net1.v:1.76-1.77" *)
wire _089_;
(* src = "net1.v:1.76-1.77" *)
wire _090_;
(* src = "net1.v:1.76-1.77" *)
wire _091_;
(* src = "net1.v:1.76-1.77" *)
wire _092_;
(* src = "net1.v:1.76-1.77" *)
wire _093_;
(* src = "net1.v:1.76-1.77" *)
wire _094_;
(* src = "net1.v:1.76-1.77" *)
wire _095_;
(* src = "net1.v:1.76-1.77" *)
wire _096_;
(* src = "net1.v:1.76-1.77" *)
wire _097_;
(* src = "net1.v:1.76-1.77" *)
wire _098_;
(* src = "net1.v:1.76-1.77" *)
wire _099_;
(* src = "net1.v:1.76-1.77" *)
wire _100_;
(* src = "net1.v:1.76-1.77" *)
wire _101_;
(* src = "net1.v:1.76-1.77" *)
wire _102_;
(* src = "net1.v:1.76-1.77" *)
wire _103_;
(* src = "net1.v:1.76-1.77" *)
wire _104_;
(* src = "net1.v:1.76-1.77" *)
wire _105_;
(* src = "net1.v:1.76-1.77" *)
wire _106_;
(* src = "net1.v:1.76-1.77" *)
```

```
wire _107_;
(* src = "net1.v:1.76-1.77" *)
wire _108_;
(* src = "net1.v:1.76-1.77" *)
wire _109_;
(* src = "net1.v:1.76-1.77" *)
wire _110_;
(* src = "net1.v:1.76-1.77" *)
wire _111_;
(* src = "net1.v:1.76-1.77" *)
wire _112_;
(* src = "net1.v:1.76-1.77" *)
wire _113_;
(* src = "net1.v:1.76-1.77" *)
wire _114_;
(* src = "net1.v:1.76-1.77" *)
wire _115_;
(* src = "net1.v:1.76-1.77" *)
wire _116_;
(* src = "net1.v:1.76-1.77" *)
wire _117_;
(* src = "net1.v:1.76-1.77" *)
wire _118_;
(* src = "net1.v:1.76-1.77" *)
wire _119_;
(* src = "net1.v:1.76-1.77" *)
wire _120_;
(* src = "net1.v:1.76-1.77" *)
wire _121_;
(* src = "net1.v:1.76-1.77" *)
wire _122_;
(* src = "net1.v:1.76-1.77" *)
wire _123_;
(* src = "net1.v:1.76-1.77" *)
wire _124_;
(* src = "net1.v:1.76-1.77" *)
wire _125_;
(* src = "net1.v:1.76-1.77" *)
wire _126_;
(* src = "net1.v:1.76-1.77" *)
wire _127_;
(* src = "net1.v:1.55-1.56" *)
input [63:0] a;
(* src = "net1.v:1.76-1.77" *)
output [63:0] b;
INV_X1 _128_ (
  .A(_000_),
  .ZN(_064_)
```

```
);
INV_X1 _129_ (
  .A(_011_),
  .ZN(_075_)
);
INV_X1 _130_ (
  .A(_022_),
  .ZN(_086_)
);
INV_X1 _131_ (
  .A(_033_),
  .ZN(_097_)
);
INV_X1 _132_ (
  .A(_044_),
  .ZN(_108_)
);
INV_X1 _133_ (
  .A(_055_),
  .ZN(_119_)
);
INV_X1 _134_ (
  .A(_060_),
  .ZN(_124_)
);
INV_X1 _135_ (
  .A(_061_),
  .ZN(_125_)
);
INV_X1 _136_ (
  .A(_062_),
  .ZN(_126_)
);
INV_X1 _137_ (
  .A(_063_),
  .ZN(_127_)
);
INV_X1 _138_ (
  .A(_001_),
  .ZN(_065_)
);
INV_X1 _139_ (
  .A(_002_),
  .ZN(_066_)
);
INV_X1 _140_ (
  .A(_003_),
  .ZN(_067_)
```

```
);
INV_X1 _141_ (
  .A(_004_),
  .ZN(_068_)
);
INV_X1 _142_ (
  .A(_005_),
  .ZN(_069_)
);
INV_X1 _143_ (
  .A(_006_),
  .ZN(_070_)
);
INV_X1 _144_ (
  .A(_007_),
  .ZN(_071_)
);
INV_X1 _145_ (
  .A(_008_),
  .ZN(_072_)
);
INV_X1 _146_ (
  .A(_009_),
  .ZN(_073_)
);
INV_X1 _147_ (
  .A(_010_),
  .ZN(_074_)
);
INV_X1 _148_ (
  .A(_012_),
  .ZN(_076_)
);
INV_X1 _149_ (
  .A(_013_),
  .ZN(_077_)
);
INV_X1 _150_ (
  .A(_014_),
  .ZN(_078_)
);
INV_X1 _151_ (
  .A(_015_),
  .ZN(_079_)
);
INV_X1 _152_ (
  .A(_016_),
  .ZN(_080_)
```

```
);
INV_X1 _153_ (
  .A(_017_),
  .ZN(_081_)
);
INV_X1 _154_ (
  .A(_018_),
  .ZN(_082_)
);
INV_X1 _155_ (
  .A(_019_),
  .ZN(_083_)
);
INV_X1 _156_ (
  .A(_020_),
  .ZN(_084_)
);
INV_X1 _157_ (
  .A(_021_),
  .ZN(_085_)
);
INV_X1 _158_ (
  .A(_023_),
  .ZN(_087_)
);
INV_X1 _159_ (
  .A(_024_),
  .ZN(_088_)
);
INV_X1 _160_ (
  .A(_025_),
  .ZN(_089_)
);
INV_X1 _161_ (
  .A(_026_),
  .ZN(_090_)
);
INV_X1 _162_ (
  .A(_027_),
  .ZN(_091_)
);
INV_X1 _163_ (
  .A(_028_),
  .ZN(_092_)
);
INV_X1 _164_ (
  .A(_029_),
  .ZN(_093_)
```

```
);
INV_X1 _165_ (
  .A(_030_),
  .ZN(_094_)
);
INV_X1 _166_ (
  .A(_031_),
  .ZN(_095_)
);
INV_X1 _167_ (
  .A(_032_),
  .ZN(_096_)
);
INV_X1 _168_ (
  .A(_034_),
  .ZN(_098_)
);
INV_X1 _169_ (
  .A(_035_),
  .ZN(_099_)
);
INV_X1 _170_ (
  .A(_036_),
  .ZN(_100_)
);
INV_X1 _171_ (
  .A(_037_),
  .ZN(_101_)
);
INV_X1 _172_ (
  .A(_038_),
  .ZN(_102_)
);
INV_X1 _173_ (
  .A(_039_),
  .ZN(_103_)
);
INV_X1 _174_ (
  .A(_040_),
  .ZN(_104_)
);
INV_X1 _175_ (
  .A(_041_),
  .ZN(_105_)
);
INV_X1 _176_ (
  .A(_042_),
  .ZN(_106_)
```

```
);
INV_X1 _177_ (
  .A(_043_),
  .ZN(_107_)
);
INV_X1 _178_ (
  .A(_045_),
  .ZN(_109_)
);
INV_X1 _179_ (
  .A(_046_),
  .ZN(_110_)
);
INV_X1 _180_ (
  .A(_047_),
  .ZN(_111_)
);
INV_X1 _181_ (
  .A(_048_),
  .ZN(_112_)
);
INV_X1 _182_ (
  .A(_049_),
  .ZN(_113_)
);
INV_X1 _183_ (
  .A(_050_),
  .ZN(_114_)
);
INV_X1 _184_ (
  .A(_051_),
  .ZN(_115_)
);
INV_X1 _185_ (
  .A(_052_),
  .ZN(_116_)
);
INV_X1 _186_ (
  .A(_053_),
  .ZN(_117_)
);
INV_X1 _187_ (
  .A(_054_),
  .ZN(_118_)
);
INV_X1 _188_ (
  .A(_056_),
  .ZN(_120_)
```

```
);
INV_X1 _189_ (
  .A(_057_),
  .ZN(_121_)
);
INV_X1 _190_ (
  .A(_058_),
  .ZN(_122_)
);
INV_X1 _191_ (
  .A(_059_),
  .ZN(_123_)
);
assign _000_ = a[0];
assign b[0] = _064_;
assign _011_ = a[1];
assign b[1] = _075_;
assign _022_ = a[2];
assign b[2] = _086_;
assign _033_ = a[3];
assign b[3] = _097_;
assign _044_ = a[4];
assign b[4] = _108_;
assign _055_ = a[5];
assign b[5] = _119_;
assign _060_ = a[6];
assign b[6] = _124_;
assign _061_ = a[7];
assign b[7] = _125_;
assign _062_ = a[8];
assign b[8] = _126_;
assign _063_ = a[9];
assign b[9] = _127_;
assign _001_ = a[10];
assign b[10] = _065_;
assign _002_ = a[11];
assign b[11] = _066_;
assign _003_ = a[12];
assign b[12] = _067_;
assign _004_ = a[13];
assign b[13] = _068_;
assign _005_ = a[14];
assign b[14] = _069_;
assign _006_ = a[15];
assign b[15] = _070_;
assign _007_ = a[16];
assign b[16] = _071_;
assign _008_ = a[17];
```

```
assign b[17] = _072_;
assign _009_ = a[18];
assign b[18] = _073_;
assign _010_ = a[19];
assign b[19] = _074_;
assign _012_ = a[20];
assign b[20] = _076_;
assign _013_ = a[21];
assign b[21] = _077_;
assign _014_ = a[22];
assign b[22] = _078_;
assign _015_ = a[23];
assign b[23] = _079_;
assign _016_ = a[24];
assign b[24] = _080_;
assign _017_ = a[25];
assign b[25] = _081_;
assign _018_ = a[26];
assign b[26] = _082_;
assign _019_ = a[27];
assign b[27] = _083_;
assign _020_ = a[28];
assign b[28] = _084_;
assign _021_ = a[29];
assign b[29] = _085_;
assign _023_ = a[30];
assign b[30] = _087_;
assign _024_ = a[31];
assign b[31] = _088_;
assign _025_ = a[32];
assign b[32] = _089_;
assign _026_ = a[33];
assign b[33] = _090_;
assign _027_ = a[34];
assign b[34] = _091_;
assign _028_ = a[35];
assign b[35] = _092_;
assign _029_ = a[36];
assign b[36] = _093_;
assign _030_ = a[37];
assign b[37] = _094_;
assign _031_ = a[38];
assign b[38] = _095_;
assign _032_ = a[39];
assign b[39] = _096_;
assign _034_ = a[40];
assign b[40] = _098_;
assign _035_ = a[41];
```

```
  assign b[41] = _099_;
  assign _036_ = a[42];
  assign b[42] = _100_;
  assign _037_ = a[43];
  assign b[43] = _101_;
  assign _038_ = a[44];
  assign b[44] = _102_;
  assign _039_ = a[45];
  assign b[45] = _103_;
  assign _040_ = a[46];
  assign b[46] = _104_;
  assign _041_ = a[47];
  assign b[47] = _105_;
  assign _042_ = a[48];
  assign b[48] = _106_;
  assign _043_ = a[49];
  assign b[49] = _107_;
  assign _045_ = a[50];
  assign b[50] = _109_;
  assign _046_ = a[51];
  assign b[51] = _110_;
  assign _047_ = a[52];
  assign b[52] = _111_;
  assign _048_ = a[53];
  assign b[53] = _112_;
  assign _049_ = a[54];
  assign b[54] = _113_;
  assign _050_ = a[55];
  assign b[55] = _114_;
  assign _051_ = a[56];
  assign b[56] = _115_;
  assign _052_ = a[57];
  assign b[57] = _116_;
  assign _053_ = a[58];
  assign b[58] = _117_;
  assign _054_ = a[59];
  assign b[59] = _118_;
  assign _056_ = a[60];
  assign b[60] = _120_;
  assign _057_ = a[61];
  assign b[61] = _121_;
  assign _058_ = a[62];
  assign b[62] = _122_;
  assign _059_ = a[63];
  assign b[63] = _123_;
endmodule

(* dynports =  1  *)
```

```
(* src = "net1.v:13.1-15.10" *)
module nanda(a2, b2, y1);
  (* src = "net1.v:13.58-13.60" *)
  wire _000_;
  (* src = "net1.v:13.58-13.60" *)
  wire _001_;
  (* src = "net1.v:13.58-13.60" *)
  wire _002_;
  (* src = "net1.v:13.58-13.60" *)
  wire _003_;
  (* src = "net1.v:13.58-13.60" *)
  wire _004_;
  (* src = "net1.v:13.58-13.60" *)
  wire _005_;
  (* src = "net1.v:13.58-13.60" *)
  wire _006_;
  (* src = "net1.v:13.58-13.60" *)
  wire _007_;
  (* src = "net1.v:13.58-13.60" *)
  wire _008_;
  (* src = "net1.v:13.58-13.60" *)
  wire _009_;
  (* src = "net1.v:13.58-13.60" *)
  wire _010_;
  (* src = "net1.v:13.58-13.60" *)
  wire _011_;
  (* src = "net1.v:13.58-13.60" *)
  wire _012_;
  (* src = "net1.v:13.58-13.60" *)
  wire _013_;
  (* src = "net1.v:13.58-13.60" *)
  wire _014_;
  (* src = "net1.v:13.58-13.60" *)
  wire _015_;
  (* src = "net1.v:13.58-13.60" *)
  wire _016_;
  (* src = "net1.v:13.58-13.60" *)
  wire _017_;
  (* src = "net1.v:13.58-13.60" *)
  wire _018_;
  (* src = "net1.v:13.58-13.60" *)
  wire _019_;
  (* src = "net1.v:13.58-13.60" *)
  wire _020_;
  (* src = "net1.v:13.58-13.60" *)
  wire _021_;
  (* src = "net1.v:13.58-13.60" *)
  wire _022_;
```

```
(* src = "net1.v:13.58-13.60" *)
wire _023_;
(* src = "net1.v:13.58-13.60" *)
wire _024_;
(* src = "net1.v:13.58-13.60" *)
wire _025_;
(* src = "net1.v:13.58-13.60" *)
wire _026_;
(* src = "net1.v:13.58-13.60" *)
wire _027_;
(* src = "net1.v:13.58-13.60" *)
wire _028_;
(* src = "net1.v:13.58-13.60" *)
wire _029_;
(* src = "net1.v:13.58-13.60" *)
wire _030_;
(* src = "net1.v:13.58-13.60" *)
wire _031_;
(* src = "net1.v:13.58-13.60" *)
wire _032_;
(* src = "net1.v:13.58-13.60" *)
wire _033_;
(* src = "net1.v:13.58-13.60" *)
wire _034_;
(* src = "net1.v:13.58-13.60" *)
wire _035_;
(* src = "net1.v:13.58-13.60" *)
wire _036_;
(* src = "net1.v:13.58-13.60" *)
wire _037_;
(* src = "net1.v:13.58-13.60" *)
wire _038_;
(* src = "net1.v:13.58-13.60" *)
wire _039_;
(* src = "net1.v:13.58-13.60" *)
wire _040_;
(* src = "net1.v:13.58-13.60" *)
wire _041_;
(* src = "net1.v:13.58-13.60" *)
wire _042_;
(* src = "net1.v:13.58-13.60" *)
wire _043_;
(* src = "net1.v:13.58-13.60" *)
wire _044_;
(* src = "net1.v:13.58-13.60" *)
wire _045_;
(* src = "net1.v:13.58-13.60" *)
wire _046_;
```

```
(* src = "net1.v:13.58-13.60" *)
wire _047_;
(* src = "net1.v:13.58-13.60" *)
wire _048_;
(* src = "net1.v:13.58-13.60" *)
wire _049_;
(* src = "net1.v:13.58-13.60" *)
wire _050_;
(* src = "net1.v:13.58-13.60" *)
wire _051_;
(* src = "net1.v:13.58-13.60" *)
wire _052_;
(* src = "net1.v:13.58-13.60" *)
wire _053_;
(* src = "net1.v:13.58-13.60" *)
wire _054_;
(* src = "net1.v:13.58-13.60" *)
wire _055_;
(* src = "net1.v:13.58-13.60" *)
wire _056_;
(* src = "net1.v:13.58-13.60" *)
wire _057_;
(* src = "net1.v:13.58-13.60" *)
wire _058_;
(* src = "net1.v:13.58-13.60" *)
wire _059_;
(* src = "net1.v:13.58-13.60" *)
wire _060_;
(* src = "net1.v:13.58-13.60" *)
wire _061_;
(* src = "net1.v:13.58-13.60" *)
wire _062_;
(* src = "net1.v:13.58-13.60" *)
wire _063_;
(* src = "net1.v:13.79-13.81" *)
wire _064_;
(* src = "net1.v:13.79-13.81" *)
wire _065_;
(* src = "net1.v:13.79-13.81" *)
wire _066_;
(* src = "net1.v:13.79-13.81" *)
wire _067_;
(* src = "net1.v:13.79-13.81" *)
wire _068_;
(* src = "net1.v:13.79-13.81" *)
wire _069_;
(* src = "net1.v:13.79-13.81" *)
wire _070_;
```

```
(* src = "net1.v:13.79-13.81" *)
wire _071_;
(* src = "net1.v:13.79-13.81" *)
wire _072_;
(* src = "net1.v:13.79-13.81" *)
wire _073_;
(* src = "net1.v:13.79-13.81" *)
wire _074_;
(* src = "net1.v:13.79-13.81" *)
wire _075_;
(* src = "net1.v:13.79-13.81" *)
wire _076_;
(* src = "net1.v:13.79-13.81" *)
wire _077_;
(* src = "net1.v:13.79-13.81" *)
wire _078_;
(* src = "net1.v:13.79-13.81" *)
wire _079_;
(* src = "net1.v:13.79-13.81" *)
wire _080_;
(* src = "net1.v:13.79-13.81" *)
wire _081_;
(* src = "net1.v:13.79-13.81" *)
wire _082_;
(* src = "net1.v:13.79-13.81" *)
wire _083_;
(* src = "net1.v:13.79-13.81" *)
wire _084_;
(* src = "net1.v:13.79-13.81" *)
wire _085_;
(* src = "net1.v:13.79-13.81" *)
wire _086_;
(* src = "net1.v:13.79-13.81" *)
wire _087_;
(* src = "net1.v:13.79-13.81" *)
wire _088_;
(* src = "net1.v:13.79-13.81" *)
wire _089_;
(* src = "net1.v:13.79-13.81" *)
wire _090_;
(* src = "net1.v:13.79-13.81" *)
wire _091_;
(* src = "net1.v:13.79-13.81" *)
wire _092_;
(* src = "net1.v:13.79-13.81" *)
wire _093_;
(* src = "net1.v:13.79-13.81" *)
wire _094_;
```

```
(* src = "net1.v:13.79-13.81" *)
wire _095_;
(* src = "net1.v:13.79-13.81" *)
wire _096_;
(* src = "net1.v:13.79-13.81" *)
wire _097_;
(* src = "net1.v:13.79-13.81" *)
wire _098_;
(* src = "net1.v:13.79-13.81" *)
wire _099_;
(* src = "net1.v:13.79-13.81" *)
wire _100_;
(* src = "net1.v:13.79-13.81" *)
wire _101_;
(* src = "net1.v:13.79-13.81" *)
wire _102_;
(* src = "net1.v:13.79-13.81" *)
wire _103_;
(* src = "net1.v:13.79-13.81" *)
wire _104_;
(* src = "net1.v:13.79-13.81" *)
wire _105_;
(* src = "net1.v:13.79-13.81" *)
wire _106_;
(* src = "net1.v:13.79-13.81" *)
wire _107_;
(* src = "net1.v:13.79-13.81" *)
wire _108_;
(* src = "net1.v:13.79-13.81" *)
wire _109_;
(* src = "net1.v:13.79-13.81" *)
wire _110_;
(* src = "net1.v:13.79-13.81" *)
wire _111_;
(* src = "net1.v:13.79-13.81" *)
wire _112_;
(* src = "net1.v:13.79-13.81" *)
wire _113_;
(* src = "net1.v:13.79-13.81" *)
wire _114_;
(* src = "net1.v:13.79-13.81" *)
wire _115_;
(* src = "net1.v:13.79-13.81" *)
wire _116_;
(* src = "net1.v:13.79-13.81" *)
wire _117_;
(* src = "net1.v:13.79-13.81" *)
wire _118_;
```

```
(* src = "net1.v:13.79-13.81" *)
wire _119_;
(* src = "net1.v:13.79-13.81" *)
wire _120_;
(* src = "net1.v:13.79-13.81" *)
wire _121_;
(* src = "net1.v:13.79-13.81" *)
wire _122_;
(* src = "net1.v:13.79-13.81" *)
wire _123_;
(* src = "net1.v:13.79-13.81" *)
wire _124_;
(* src = "net1.v:13.79-13.81" *)
wire _125_;
(* src = "net1.v:13.79-13.81" *)
wire _126_;
(* src = "net1.v:13.79-13.81" *)
wire _127_;
(* src = "net1.v:13.101-13.103" *)
wire _128_;
(* src = "net1.v:13.101-13.103" *)
wire _129_;
(* src = "net1.v:13.101-13.103" *)
wire _130_;
(* src = "net1.v:13.101-13.103" *)
wire _131_;
(* src = "net1.v:13.101-13.103" *)
wire _132_;
(* src = "net1.v:13.101-13.103" *)
wire _133_;
(* src = "net1.v:13.101-13.103" *)
wire _134_;
(* src = "net1.v:13.101-13.103" *)
wire _135_;
(* src = "net1.v:13.101-13.103" *)
wire _136_;
(* src = "net1.v:13.101-13.103" *)
wire _137_;
(* src = "net1.v:13.101-13.103" *)
wire _138_;
(* src = "net1.v:13.101-13.103" *)
wire _139_;
(* src = "net1.v:13.101-13.103" *)
wire _140_;
(* src = "net1.v:13.101-13.103" *)
wire _141_;
(* src = "net1.v:13.101-13.103" *)
wire _142_;
```

```
(* src = "net1.v:13.101-13.103" *)
wire _143_;
(* src = "net1.v:13.101-13.103" *)
wire _144_;
(* src = "net1.v:13.101-13.103" *)
wire _145_;
(* src = "net1.v:13.101-13.103" *)
wire _146_;
(* src = "net1.v:13.101-13.103" *)
wire _147_;
(* src = "net1.v:13.101-13.103" *)
wire _148_;
(* src = "net1.v:13.101-13.103" *)
wire _149_;
(* src = "net1.v:13.101-13.103" *)
wire _150_;
(* src = "net1.v:13.101-13.103" *)
wire _151_;
(* src = "net1.v:13.101-13.103" *)
wire _152_;
(* src = "net1.v:13.101-13.103" *)
wire _153_;
(* src = "net1.v:13.101-13.103" *)
wire _154_;
(* src = "net1.v:13.101-13.103" *)
wire _155_;
(* src = "net1.v:13.101-13.103" *)
wire _156_;
(* src = "net1.v:13.101-13.103" *)
wire _157_;
(* src = "net1.v:13.101-13.103" *)
wire _158_;
(* src = "net1.v:13.101-13.103" *)
wire _159_;
(* src = "net1.v:13.101-13.103" *)
wire _160_;
(* src = "net1.v:13.101-13.103" *)
wire _161_;
(* src = "net1.v:13.101-13.103" *)
wire _162_;
(* src = "net1.v:13.101-13.103" *)
wire _163_;
(* src = "net1.v:13.101-13.103" *)
wire _164_;
(* src = "net1.v:13.101-13.103" *)
wire _165_;
(* src = "net1.v:13.101-13.103" *)
wire _166_;
```

```
(* src = "net1.v:13.101-13.103" *)
wire _167_;
(* src = "net1.v:13.101-13.103" *)
wire _168_;
(* src = "net1.v:13.101-13.103" *)
wire _169_;
(* src = "net1.v:13.101-13.103" *)
wire _170_;
(* src = "net1.v:13.101-13.103" *)
wire _171_;
(* src = "net1.v:13.101-13.103" *)
wire _172_;
(* src = "net1.v:13.101-13.103" *)
wire _173_;
(* src = "net1.v:13.101-13.103" *)
wire _174_;
(* src = "net1.v:13.101-13.103" *)
wire _175_;
(* src = "net1.v:13.101-13.103" *)
wire _176_;
(* src = "net1.v:13.101-13.103" *)
wire _177_;
(* src = "net1.v:13.101-13.103" *)
wire _178_;
(* src = "net1.v:13.101-13.103" *)
wire _179_;
(* src = "net1.v:13.101-13.103" *)
wire _180_;
(* src = "net1.v:13.101-13.103" *)
wire _181_;
(* src = "net1.v:13.101-13.103" *)
wire _182_;
(* src = "net1.v:13.101-13.103" *)
wire _183_;
(* src = "net1.v:13.101-13.103" *)
wire _184_;
(* src = "net1.v:13.101-13.103" *)
wire _185_;
(* src = "net1.v:13.101-13.103" *)
wire _186_;
(* src = "net1.v:13.101-13.103" *)
wire _187_;
(* src = "net1.v:13.101-13.103" *)
wire _188_;
(* src = "net1.v:13.101-13.103" *)
wire _189_;
(* src = "net1.v:13.101-13.103" *)
wire _190_;
```

```
(* src = "net1.v:13.101-13.103" *)
wire _191_;
(* src = "net1.v:14.17-14.22" *)
wire [63:0] _192_;
(* src = "net1.v:13.58-13.60" *)
input [63:0] a2;
(* src = "net1.v:13.79-13.81" *)
input [63:0] b2;
(* src = "net1.v:13.101-13.103" *)
output [63:0] y1;
NAND2_X1 _193_ (
  .A1(_000_),
  .A2(_064_),
  .ZN(_128_)
);
NAND2_X1 _194_ (
  .A1(_011_),
  .A2(_075_),
  .ZN(_139_)
);
NAND2_X1 _195_ (
  .A1(_022_),
  .A2(_086_),
  .ZN(_150_)
);
NAND2_X1 _196_ (
  .A1(_033_),
  .A2(_097_),
  .ZN(_161_)
);
NAND2_X1 _197_ (
  .A1(_044_),
  .A2(_108_),
  .ZN(_172_)
);
NAND2_X1 _198_ (
  .A1(_055_),
  .A2(_119_),
  .ZN(_183_)
);
NAND2_X1 _199_ (
  .A1(_060_),
  .A2(_124_),
  .ZN(_188_)
);
NAND2_X1 _200_ (
  .A1(_061_),
  .A2(_125_),
```

```
  .ZN(_189_)
);
NAND2_X1 _201_ (
  .A1(_062_),
  .A2(_126_),
  .ZN(_190_)
);
NAND2_X1 _202_ (
  .A1(_063_),
  .A2(_127_),
  .ZN(_191_)
);
NAND2_X1 _203_ (
  .A1(_001_),
  .A2(_065_),
  .ZN(_129_)
);
NAND2_X1 _204_ (
  .A1(_002_),
  .A2(_066_),
  .ZN(_130_)
);
NAND2_X1 _205_ (
  .A1(_003_),
  .A2(_067_),
  .ZN(_131_)
);
NAND2_X1 _206_ (
  .A1(_004_),
  .A2(_068_),
  .ZN(_132_)
);
NAND2_X1 _207_ (
  .A1(_005_),
  .A2(_069_),
  .ZN(_133_)
);
NAND2_X1 _208_ (
  .A1(_006_),
  .A2(_070_),
  .ZN(_134_)
);
NAND2_X1 _209_ (
  .A1(_007_),
  .A2(_071_),
  .ZN(_135_)
);
NAND2_X1 _210_ (
```

```
  .A1(_008_),
  .A2(_072_),
  .ZN(_136_)
);
NAND2_X1 _211_ (
  .A1(_009_),
  .A2(_073_),
  .ZN(_137_)
);
NAND2_X1 _212_ (
  .A1(_010_),
  .A2(_074_),
  .ZN(_138_)
);
NAND2_X1 _213_ (
  .A1(_012_),
  .A2(_076_),
  .ZN(_140_)
);
NAND2_X1 _214_ (
  .A1(_013_),
  .A2(_077_),
  .ZN(_141_)
);
NAND2_X1 _215_ (
  .A1(_014_),
  .A2(_078_),
  .ZN(_142_)
);
NAND2_X1 _216_ (
  .A1(_015_),
  .A2(_079_),
  .ZN(_143_)
);
NAND2_X1 _217_ (
  .A1(_016_),
  .A2(_080_),
  .ZN(_144_)
);
NAND2_X1 _218_ (
  .A1(_017_),
  .A2(_081_),
  .ZN(_145_)
);
NAND2_X1 _219_ (
  .A1(_018_),
  .A2(_082_),
  .ZN(_146_)
```

```
);
NAND2_X1 _220_ (
  .A1(_019_),
  .A2(_083_),
  .ZN(_147_)
);
NAND2_X1 _221_ (
  .A1(_020_),
  .A2(_084_),
  .ZN(_148_)
);
NAND2_X1 _222_ (
  .A1(_021_),
  .A2(_085_),
  .ZN(_149_)
);
NAND2_X1 _223_ (
  .A1(_023_),
  .A2(_087_),
  .ZN(_151_)
);
NAND2_X1 _224_ (
  .A1(_024_),
  .A2(_088_),
  .ZN(_152_)
);
NAND2_X1 _225_ (
  .A1(_025_),
  .A2(_089_),
  .ZN(_153_)
);
NAND2_X1 _226_ (
  .A1(_026_),
  .A2(_090_),
  .ZN(_154_)
);
NAND2_X1 _227_ (
  .A1(_027_),
  .A2(_091_),
  .ZN(_155_)
);
NAND2_X1 _228_ (
  .A1(_028_),
  .A2(_092_),
  .ZN(_156_)
);
NAND2_X1 _229_ (
  .A1(_029_),
```

```
  .A2(_093_),
  .ZN(_157_)
);
NAND2_X1 _230_ (
  .A1(_030_),
  .A2(_094_),
  .ZN(_158_)
);
NAND2_X1 _231_ (
  .A1(_031_),
  .A2(_095_),
  .ZN(_159_)
);
NAND2_X1 _232_ (
  .A1(_032_),
  .A2(_096_),
  .ZN(_160_)
);
NAND2_X1 _233_ (
  .A1(_034_),
  .A2(_098_),
  .ZN(_162_)
);
NAND2_X1 _234_ (
  .A1(_035_),
  .A2(_099_),
  .ZN(_163_)
);
NAND2_X1 _235_ (
  .A1(_036_),
  .A2(_100_),
  .ZN(_164_)
);
NAND2_X1 _236_ (
  .A1(_037_),
  .A2(_101_),
  .ZN(_165_)
);
NAND2_X1 _237_ (
  .A1(_038_),
  .A2(_102_),
  .ZN(_166_)
);
NAND2_X1 _238_ (
  .A1(_039_),
  .A2(_103_),
  .ZN(_167_)
);
```

```
 NAND2_X1 _239_ (
   .A1(_040_),
   .A2(_104_),
   .ZN(_168_)
 );
 NAND2_X1 _240_ (
   .A1(_041_),
   .A2(_105_),
   .ZN(_169_)
 );
 NAND2_X1 _241_ (
   .A1(_042_),
   .A2(_106_),
   .ZN(_170_)
 );
 NAND2_X1 _242_ (
   .A1(_043_),
   .A2(_107_),
   .ZN(_171_)
 );
 NAND2_X1 _243_ (
   .A1(_045_),
   .A2(_109_),
   .ZN(_173_)
 );
 NAND2_X1 _244_ (
   .A1(_046_),
   .A2(_110_),
   .ZN(_174_)
 );
 NAND2_X1 _245_ (
   .A1(_047_),
   .A2(_111_),
   .ZN(_175_)
 );
 NAND2_X1 _246_ (
   .A1(_048_),
   .A2(_112_),
   .ZN(_176_)
 );
 NAND2_X1 _247_ (
   .A1(_049_),
   .A2(_113_),
   .ZN(_177_)
 );
 NAND2_X1 _248_ (
   .A1(_050_),
   .A2(_114_),
```

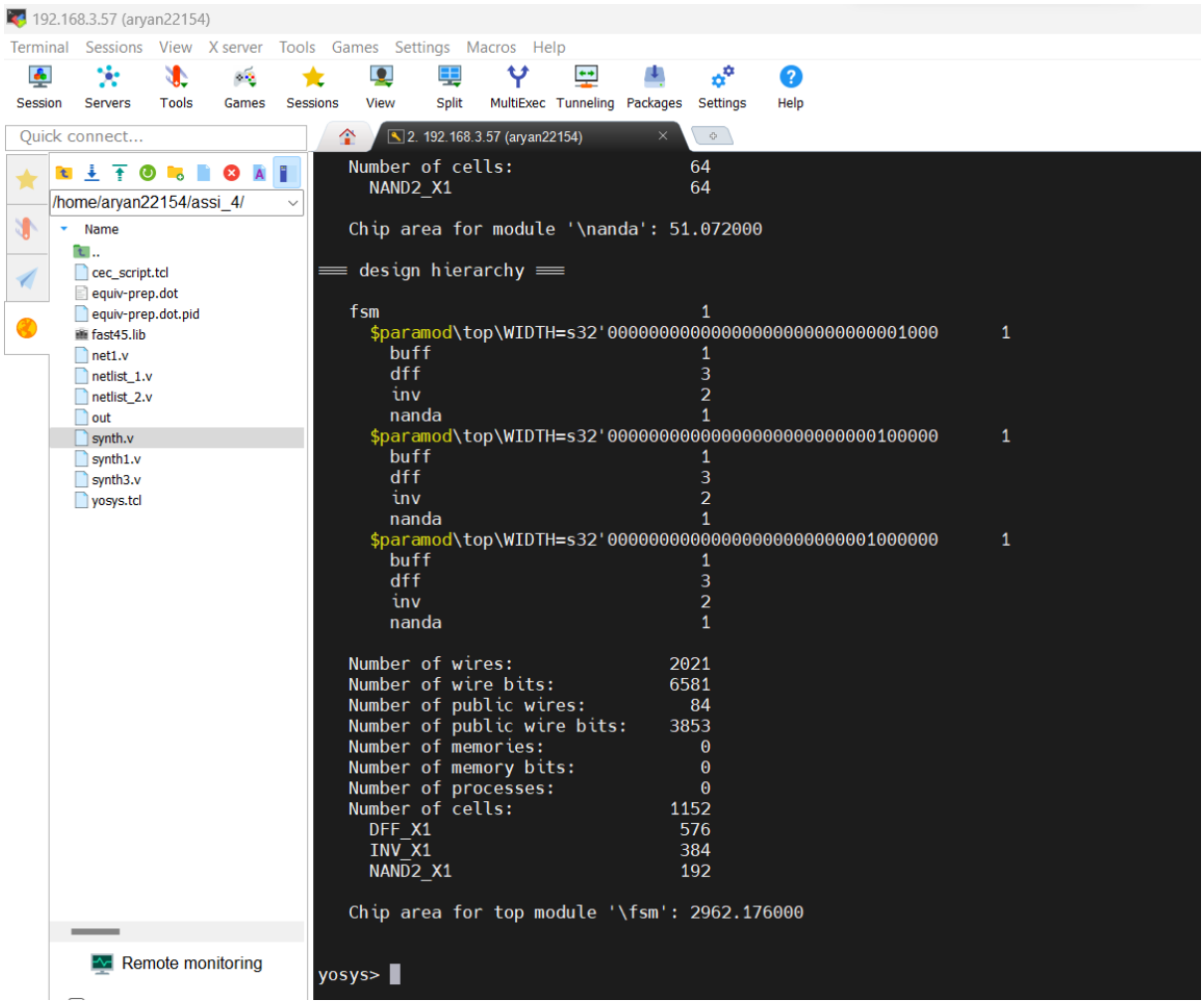
```
  .ZN(_178_)
);
NAND2_X1 _249_ (
  .A1(_051_),
  .A2(_115_),
  .ZN(_179_)
);
NAND2_X1 _250_ (
  .A1(_052_),
  .A2(_116_),
  .ZN(_180_)
);
NAND2_X1 _251_ (
  .A1(_053_),
  .A2(_117_),
  .ZN(_181_)
);
NAND2_X1 _252_ (
  .A1(_054_),
  .A2(_118_),
  .ZN(_182_)
);
NAND2_X1 _253_ (
  .A1(_056_),
  .A2(_120_),
  .ZN(_184_)
);
NAND2_X1 _254_ (
  .A1(_057_),
  .A2(_121_),
  .ZN(_185_)
);
NAND2_X1 _255_ (
  .A1(_058_),
  .A2(_122_),
  .ZN(_186_)
);
NAND2_X1 _256_ (
  .A1(_059_),
  .A2(_123_),
  .ZN(_187_)
);
assign y1[0] = _128_;
assign y1[1] = _139_;
assign y1[2] = _150_;
assign y1[3] = _161_;
assign y1[4] = _172_;
assign y1[5] = _183_;
```

```
  assign y1[6] = _188_;
  assign y1[7] = _189_;
  assign y1[8] = _190_;
  assign y1[9] = _191_;
  assign y1[10] = _129_;
  assign y1[11] = _130_;
  assign y1[12] = _131_;
  assign y1[13] = _132_;
  assign y1[14] = _133_;
  assign y1[15] = _134_;
  assign y1[16] = _135_;
  assign y1[17] = _136_;
  assign y1[18] = _137_;
  assign y1[19] = _138_;
  assign y1[20] = _140_;
  assign y1[21] = _141_;
  assign y1[22] = _142_;
  assign y1[23] = _143_;
  assign y1[24] = _144_;
  assign y1[25] = _145_;
  assign y1[26] = _146_;
  assign y1[27] = _147_;
  assign y1[28] = _148_;
  assign y1[29] = _149_;
  assign y1[30] = _151_;
  assign y1[31] = _152_;
  assign y1[32] = _153_;
  assign y1[33] = _154_;
  assign y1[34] = _155_;
  assign y1[35] = _156_;
  assign y1[36] = _157_;
  assign y1[37] = _158_;
  assign y1[38] = _159_;
  assign y1[39] = _160_;
  assign y1[40] = _162_;
  assign y1[41] = _163_;
  assign y1[42] = _164_;
  assign y1[43] = _165_;
  assign y1[44] = _166_;
  assign y1[45] = _167_;
  assign y1[46] = _168_;
  assign y1[47] = _169_;
  assign y1[48] = _170_;
  assign y1[49] = _171_;
  assign y1[50] = _173_;
  assign y1[51] = _174_;
  assign y1[52] = _175_;
  assign y1[53] = _176_;
```

```
assign y1[54] = _177_;
assign y1[55] = _178_;
assign y1[56] = _179_;
assign y1[57] = _180_;
assign y1[58] = _181_;
assign y1[59] = _182_;
assign y1[60] = _184_;
assign y1[61] = _185_;
assign y1[62] = _186_;
assign y1[63] = _187_;
assign _060_ = a2[6];
assign _124_ = b2[6];
assign _061_ = a2[7];
assign _125_ = b2[7];
assign _062_ = a2[8];
assign _126_ = b2[8];
assign _063_ = a2[9];
assign _127_ = b2[9];
assign _001_ = a2[10];
assign _065_ = b2[10];
assign _002_ = a2[11];
assign _066_ = b2[11];
assign _003_ = a2[12];
assign _067_ = b2[12];
assign _004_ = a2[13];
assign _068_ = b2[13];
assign _005_ = a2[14];
assign _069_ = b2[14];
assign _006_ = a2[15];
assign _070_ = b2[15];
assign _007_ = a2[16];
assign _071_ = b2[16];
assign _008_ = a2[17];
assign _072_ = b2[17];
assign _009_ = a2[18];
assign _073_ = b2[18];
assign _010_ = a2[19];
assign _074_ = b2[19];
assign _012_ = a2[20];
assign _076_ = b2[20];
assign _013_ = a2[21];
assign _077_ = b2[21];
assign _014_ = a2[22];
assign _078_ = b2[22];
assign _015_ = a2[23];
assign _079_ = b2[23];
assign _016_ = a2[24];
assign _080_ = b2[24];
```

```
assign _017_ = a2[25];
assign _081_ = b2[25];
assign _018_ = a2[26];
assign _082_ = b2[26];
assign _019_ = a2[27];
assign _083_ = b2[27];
assign _020_ = a2[28];
assign _084_ = b2[28];
assign _021_ = a2[29];
assign _085_ = b2[29];
assign _023_ = a2[30];
assign _087_ = b2[30];
assign _024_ = a2[31];
assign _088_ = b2[31];
assign _025_ = a2[32];
assign _089_ = b2[32];
assign _026_ = a2[33];
assign _090_ = b2[33];
assign _027_ = a2[34];
assign _091_ = b2[34];
assign _028_ = a2[35];
assign _092_ = b2[35];
assign _029_ = a2[36];
assign _093_ = b2[36];
assign _030_ = a2[37];
assign _094_ = b2[37];
assign _031_ = a2[38];
assign _095_ = b2[38];
assign _032_ = a2[39];
assign _096_ = b2[39];
assign _034_ = a2[40];
assign _098_ = b2[40];
assign _035_ = a2[41];
assign _099_ = b2[41];
assign _036_ = a2[42];
assign _100_ = b2[42];
assign _037_ = a2[43];
assign _101_ = b2[43];
assign _038_ = a2[44];
assign _102_ = b2[44];
assign _039_ = a2[45];
assign _103_ = b2[45];
assign _040_ = a2[46];
assign _104_ = b2[46];
assign _041_ = a2[47];
assign _105_ = b2[47];
assign _042_ = a2[48];
assign _106_ = b2[48];
```

```
  assign _043_ = a2[49];
  assign _107_ = b2[49];
  assign _045_ = a2[50];
  assign _109_ = b2[50];
  assign _046_ = a2[51];
  assign _110_ = b2[51];
  assign _047_ = a2[52];
  assign _111_ = b2[52];
  assign _048_ = a2[53];
  assign _112_ = b2[53];
  assign _049_ = a2[54];
  assign _113_ = b2[54];
  assign _050_ = a2[55];
  assign _114_ = b2[55];
  assign _051_ = a2[56];
  assign _115_ = b2[56];
  assign _052_ = a2[57];
  assign _116_ = b2[57];
  assign _053_ = a2[58];
  assign _117_ = b2[58];
  assign _054_ = a2[59];
  assign _118_ = b2[59];
  assign _056_ = a2[60];
  assign _120_ = b2[60];
  assign _057_ = a2[61];
  assign _121_ = b2[61];
  assign _058_ = a2[62];
  assign _122_ = b2[62];
  assign _059_ = a2[63];
  assign _123_ = b2[63];
  assign _000_ = a2[0];
  assign _064_ = b2[0];
  assign _011_ = a2[1];
  assign _075_ = b2[1];
  assign _022_ = a2[2];
  assign _086_ = b2[2];
  assign _033_ = a2[3];
  assign _097_ = b2[3];
  assign _044_ = a2[4];
  assign _108_ = b2[4];
  assign _055_ = a2[5];
  assign _119_ = b2[5];
endmodule
```



So, for making the top of variable length, the parameter WIDTH is introduced, and its size is increase , decrease by using the # width command in FSM implementation.

We can also see in the implementation part that the three top instances are created in fsm.

With the variable length of 8 Bits

Then 32 bits

And then 64 Bits

Synthesis Netlist with Library fast45.lib gives

DFF_X1 – 576

INV_X1 – 384

NAND2_X1 – 192

For this the MODEL -1 is used which is given in the question.