ECE 817 | William Nitsch
Lab 3 – Washing Machine FSM | Raashid Ansari
Due 3/31/15 | Jon Frey

System Design:

The original design for our washing machine included the desired eight states, individual hot and cold water temperature control and included robustness features such as detecting a loss of power (and restarting the system) and detecting the opening and closing of the door, controlling the water output and resuming the current state afterwards. While this design simulated properly, Design Vision was not able to synthesize it for unknown reasons (we spent many hours trying to solve this problem). Since this was our intended design, a state diagram has been included along with an explanation of the design. In the diagram, P is power, I is the increment signal, D is the door (0: closed, 1: open), and Return is the state to return to when the door is closed again. In this system, when a door is opened the system turns off the water for both hot and cold and then assumes the IDLE state until the door is closed. The detection of the door is asynchronous.

The second state diagram is the diagram of the system that was implemented. It is the same system without the detection of the door opening and closing. With each positive clock edge, the system increments or decrements states in a synchronous manner. In both systems, power loss detection and subsequent assumption of the OFF state is asynchronous. Upon being powered on, the system enters the IDLE state and then increments with respect to the increment signal on each positive clock edge.

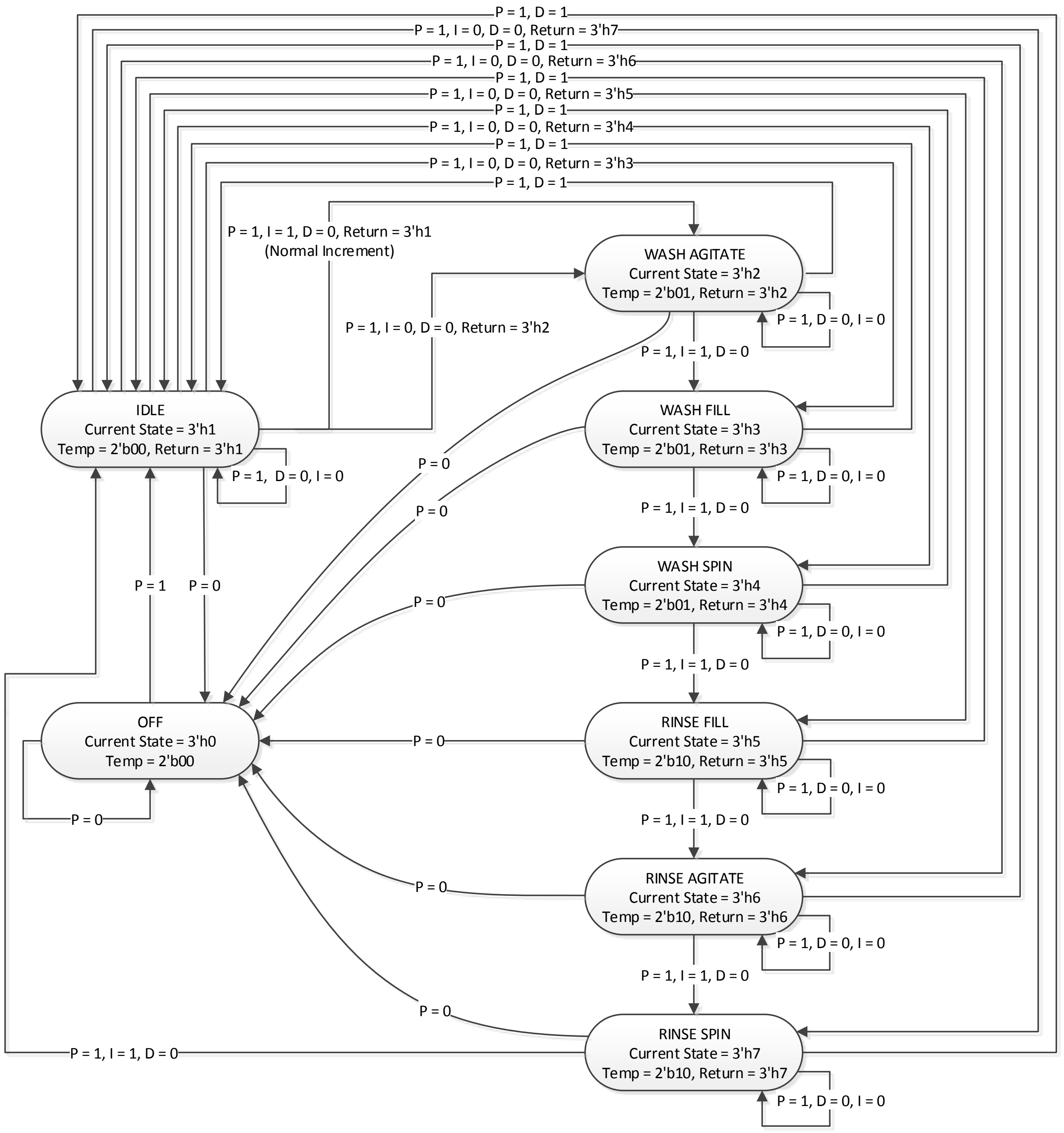


Figure 1: Originally Intended Design

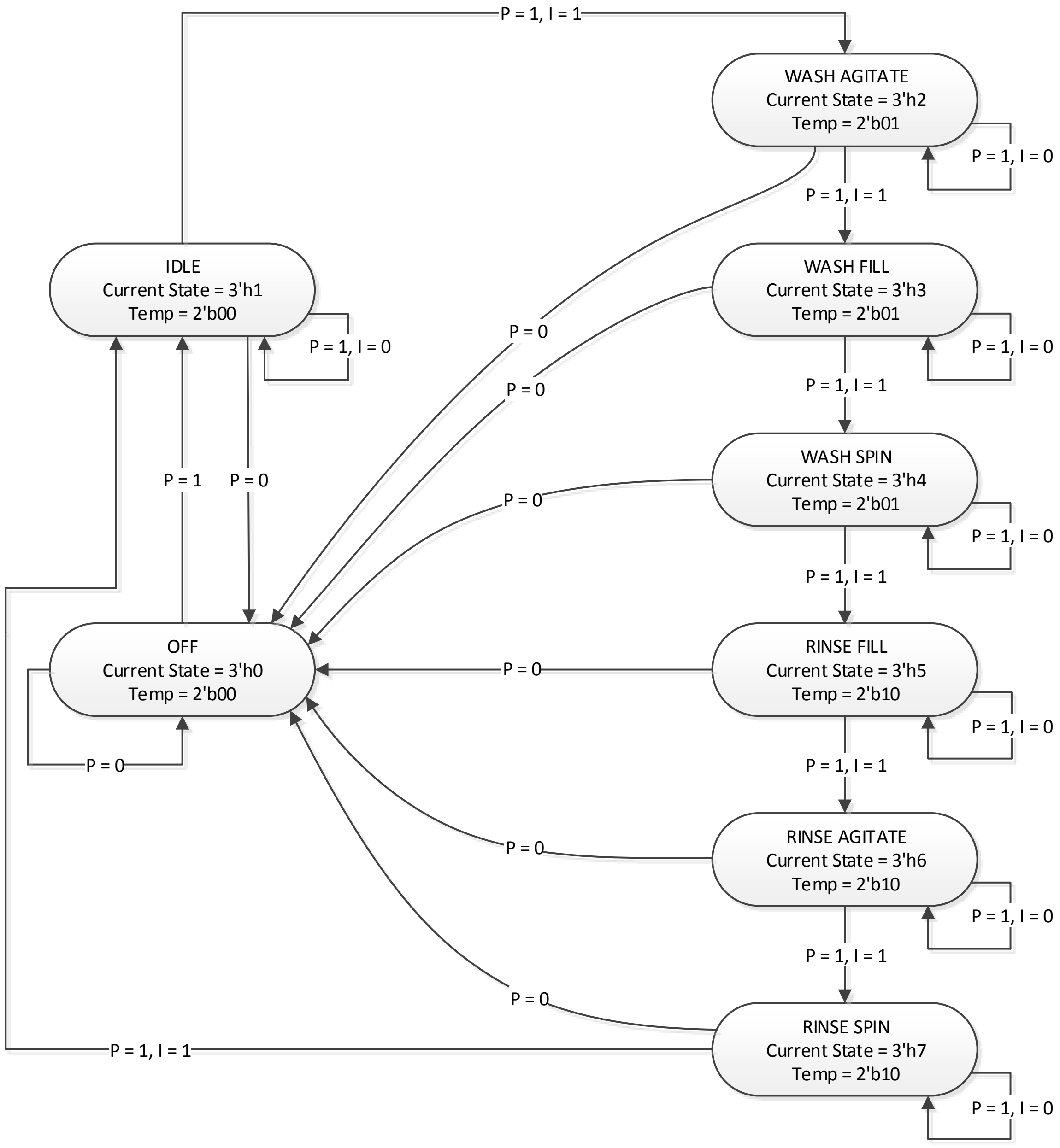


Figure 2: Implemented Design

Source Code Justification:

In the design of the source code, it was chosen that a case statement be the preference for the design. A FSM-structure was chosen for the design. It was chosen that based on the amount of states, a 3 bit signal would be enough to represent the state. There were a total of 8 states in our design. There is 1 “off” state for when the power is off, 1 “idle” state for when the machine is not running, or incrementing, and 6 states for the washing machine. We originally wanted to add two scenarios that we could handle, which are if the machine asynchronously loses power, or if the door asynchronously opens. We were successful in implementing both signals in simulation. Unfortunately, Design Vision had several issues with our design, and we had written 8-9 different design, but only could synthesize the design with the asynchronous power exception being handled. The glitches pictured are mostly due to switching FSM states and as the bits are switching, the state is read out wrong due to the propagation delay.

Source Code:

```
`timescale 1ns/1ns
module WashingMachine8(clkorig, finalwater, finalfinalstate,power,clk);

input  power, clkorig;
output reg[2:0]  finalfinalstate;
reg[2:0] currentstate;
reg increment;
output wire clk;
// MSB = hot water, LSB = cold water
output reg[1:0]  finalwater;
reg[1:0] currentwater;

//Off = 000
//Idle = 001
//Wash_fill = 010
//Wash_agitate = 011
//Wash_spin = 100
//Rinse_fill = 101
//Rinse_agitate = 110
//Rinse_spin = 111

    localparam Off = 3'h0;
    localparam Idle = 3'h1;
    localparam Wash_fill = 3'h2;
    localparam Wash_agitate = 3'h3;
    localparam Wash_spin = 3'h4;
    localparam Rinse_fill = 3'h5;
    localparam Rinse_agitate = 3'h6;
    localparam Rinse_spin = 3'h7;


assign clk = clkorig&power;
```

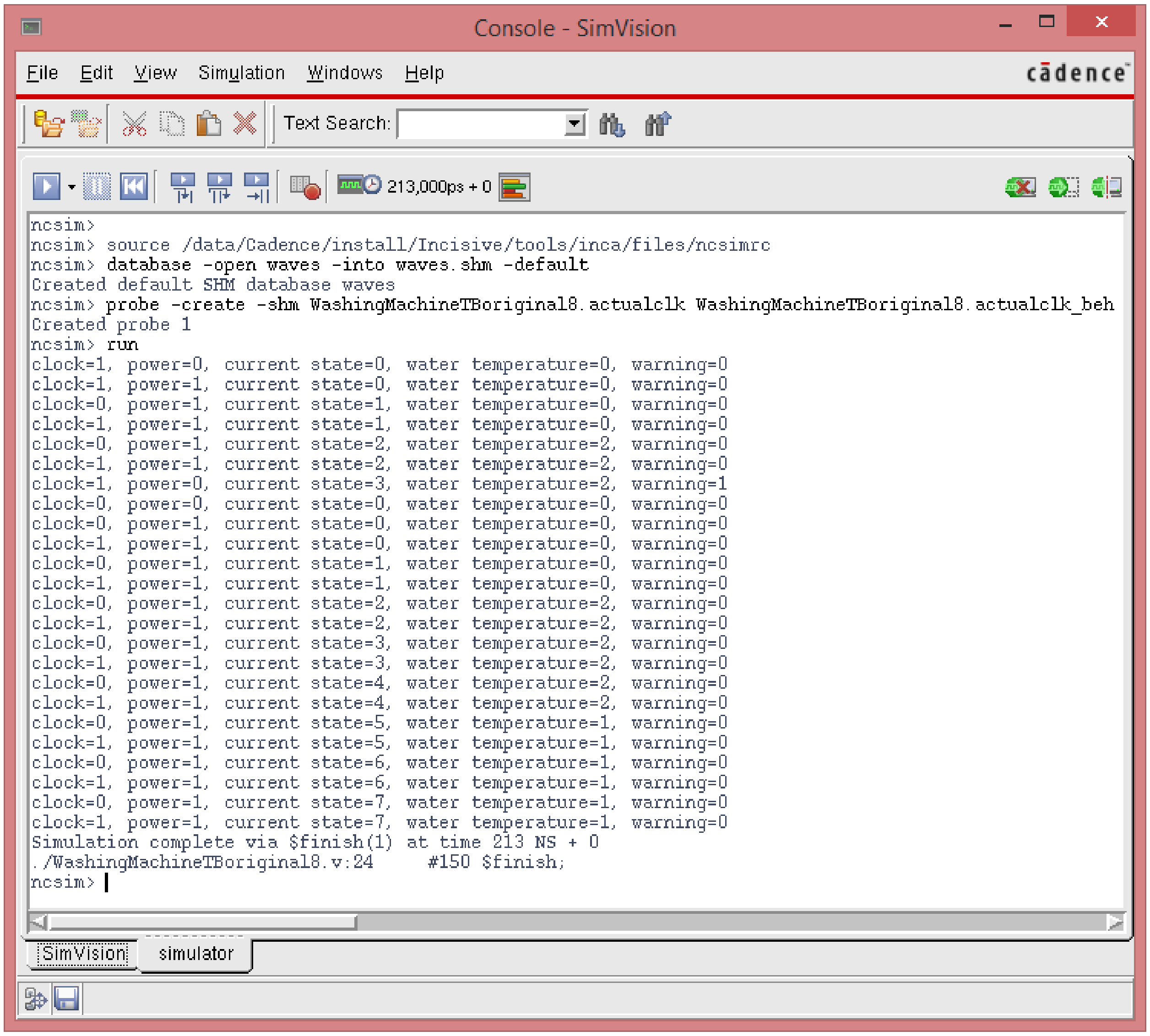
```
always@(posedge clk or negedge power)// or negedge power or posedge door or
posedge (~door))
  begin
  if(~power)
  begin
  increment <= 1'b0;
  finalfinalstate <= Off;
  finalwater <= 2'h0;
  end

  else
  begin
      case (increment)
            1'b1:
                begin
                    case (finalfinalstate)
                      Off: begin currentstate = Idle; currentwater = 2'h0;
end
                        Idle: begin currentstate = Wash_fill; currentwater =
2'h2; end
                        Wash_fill: begin currentstate = Wash_agitate;
currentwater = 2'h2; end
                        Wash_agitate: begin currentstate = Wash_spin;
currentwater = 2'h2; end
                        Wash_spin: begin currentstate = Rinse_fill;
currentwater = 2'h1; end
                        Rinse_fill: begin currentstate = Rinse_agitate;
currentwater = 2'h1; end
                        Rinse_agitate: begin currentstate = Rinse_spin;
currentwater = 2'h1; end
                        Rinse_spin: begin currentstate = Idle; currentwater =
2'h0; end
                        default: begin currentstate = Idle; currentwater =
2'h0; end
                    endcase
                end
            1'b0:
                begin
                    case (finalfinalstate)
                      Off: begin currentstate = Idle; currentwater = 2'h0;
end
                        Idle: begin currentstate = Idle; currentwater = 2'h0;
end
                        Wash_fill: begin currentstate = Wash_fill;
currentwater = 2'h2; end
                        Wash_agitate: begin currentstate = Wash_agitate;
currentwater = 2'h2; end
                        Wash_spin: begin currentstate = Wash_spin;
currentwater = 2'h2; end
                        Rinse_fill: begin currentstate = Rinse_fill;
currentwater = 2'h1; end
                        Rinse_agitate: begin currentstate = Rinse_agitate;
currentwater = 2'h1; end
                        Rinse_spin: begin currentstate = Rinse_spin;
currentwater = 2'h1; end
                        default: begin currentstate = Idle; currentwater =
2'h0; end
```

```
                    endcase
                end
        endcase
        finalfinalstate <= currentstate;
        finalwater <= currentwater;
  increment <= 1'b1;
  end

end

endmodule
```

Testbench Justification:

The testbench was designed to test whether the circuit works as per the original design. We test that if the power is off, does the clock still go on. We also test if the state of the machine goes to zero (Off state) when the power goes out. Another test case was whether the water temperature is correct for according to the states that the machine is in. If any of the above conditions are not met, the warning signal would go high to make the error prominent. A screenshot for the same can be seen below.

We test the machine at 4 major test points mentioned above to check if the machine responds correctly to the asynchronous power signal. We think these are the test cases necessary to be checked for robustness according to the design.



```
ncsim>
ncsim> source /data/Cadence/install/Incisive/tools/inca/files/ncsimrc
ncsim> database -open waves -into waves.shm -default
Created default SHM database waves
ncsim> probe -create -shm WashingMachineTBoriginal8.actualclk WashingMachineTBoriginal8.actualclk_beh
Created probe 1
ncsim> run
clock=1, power=0, current state=0, water temperature=0, warning=0
clock=1, power=1, current state=0, water temperature=0, warning=0
clock=0, power=1, current state=1, water temperature=0, warning=0
clock=1, power=1, current state=1, water temperature=0, warning=0
clock=0, power=1, current state=2, water temperature=2, warning=0
clock=1, power=1, current state=2, water temperature=2, warning=0
clock=1, power=0, current state=3, water temperature=2, warning=1
clock=0, power=0, current state=0, water temperature=0, warning=0
clock=0, power=1, current state=0, water temperature=0, warning=0
clock=1, power=1, current state=0, water temperature=0, warning=0
clock=0, power=1, current state=1, water temperature=0, warning=0
clock=1, power=1, current state=1, water temperature=0, warning=0
clock=0, power=1, current state=2, water temperature=2, warning=0
clock=1, power=1, current state=2, water temperature=2, warning=0
clock=0, power=1, current state=3, water temperature=2, warning=0
clock=1, power=1, current state=3, water temperature=2, warning=0
clock=0, power=1, current state=4, water temperature=2, warning=0
clock=1, power=1, current state=4, water temperature=2, warning=0
clock=0, power=1, current state=5, water temperature=1, warning=0
clock=1, power=1, current state=5, water temperature=1, warning=0
clock=0, power=1, current state=6, water temperature=1, warning=0
clock=1, power=1, current state=6, water temperature=1, warning=0
clock=0, power=1, current state=7, water temperature=1, warning=0
clock=1, power=1, current state=7, water temperature=1, warning=0
Simulation complete via $finish(1) at time 213 NS + 0
./WashingMachineTBoriginal8.v:24     #150 $finish;
ncsim>
```

Figure 3: Output of the Test Bench

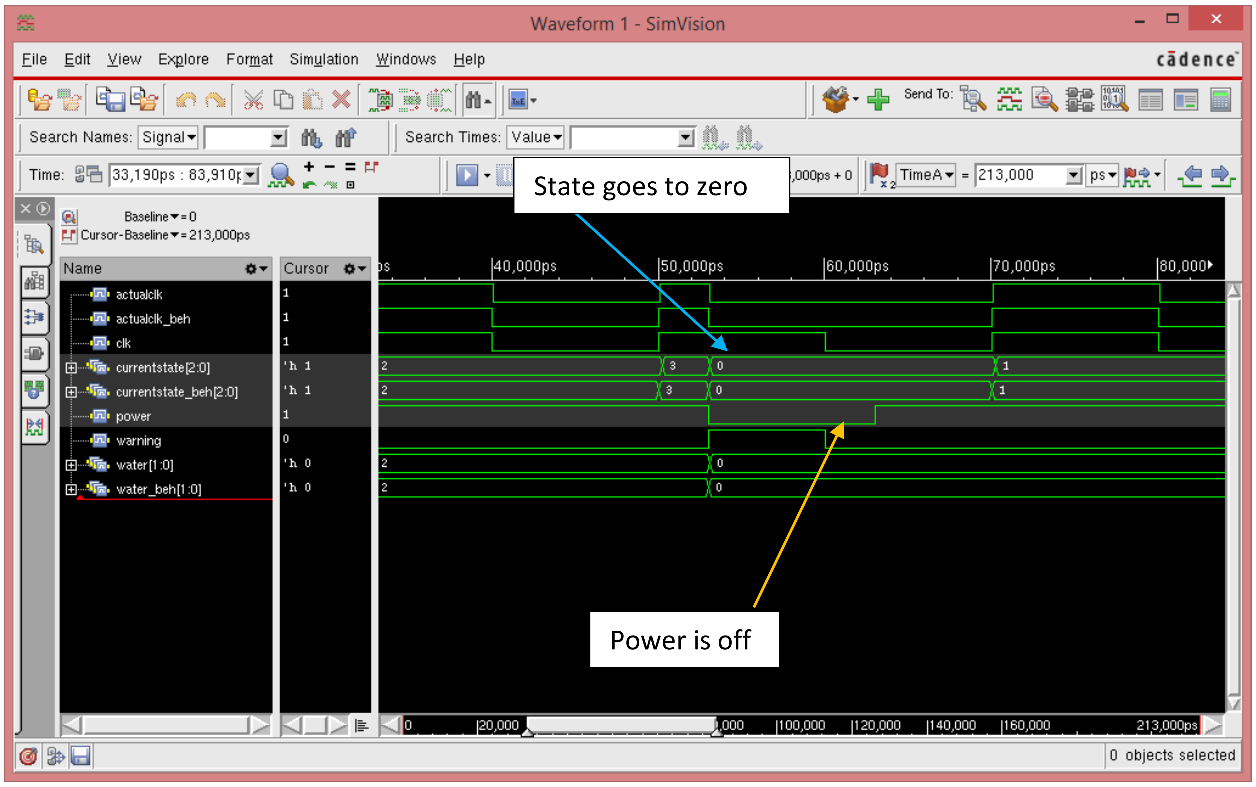
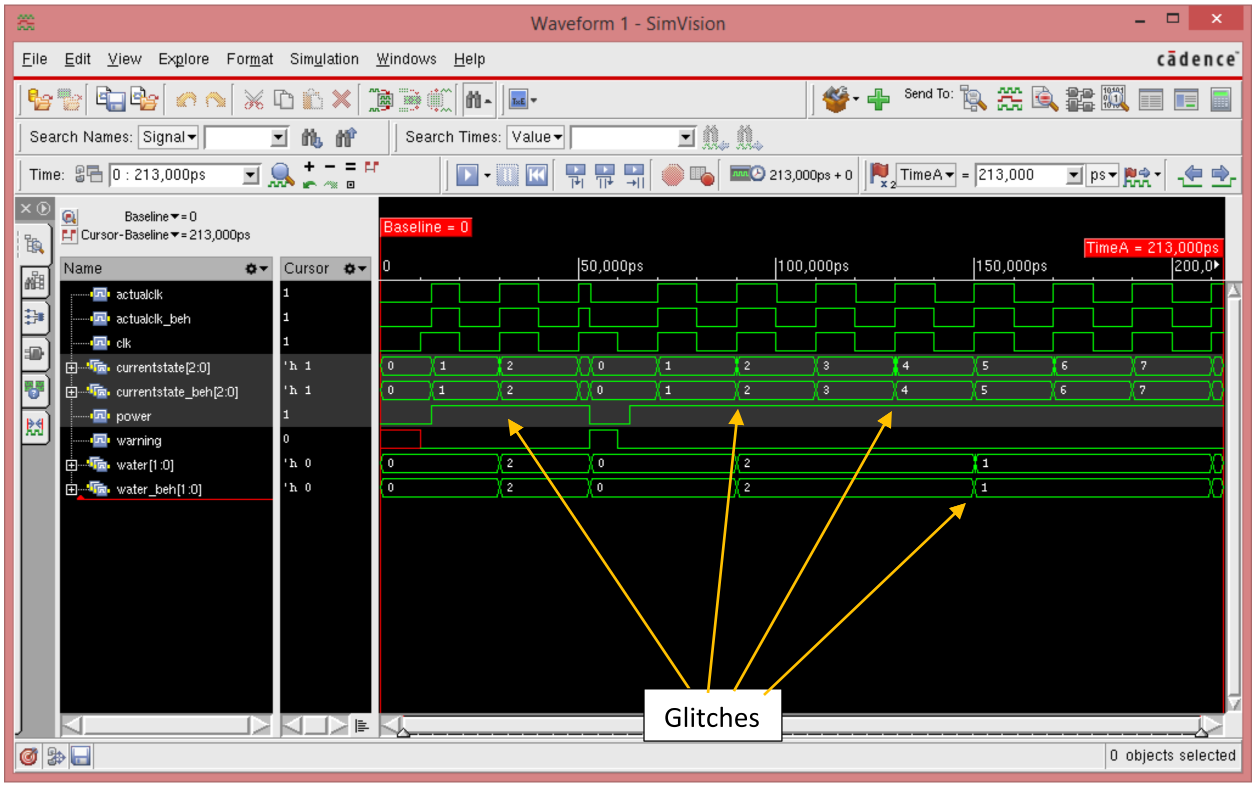


Figure 4: Handling Power Situation

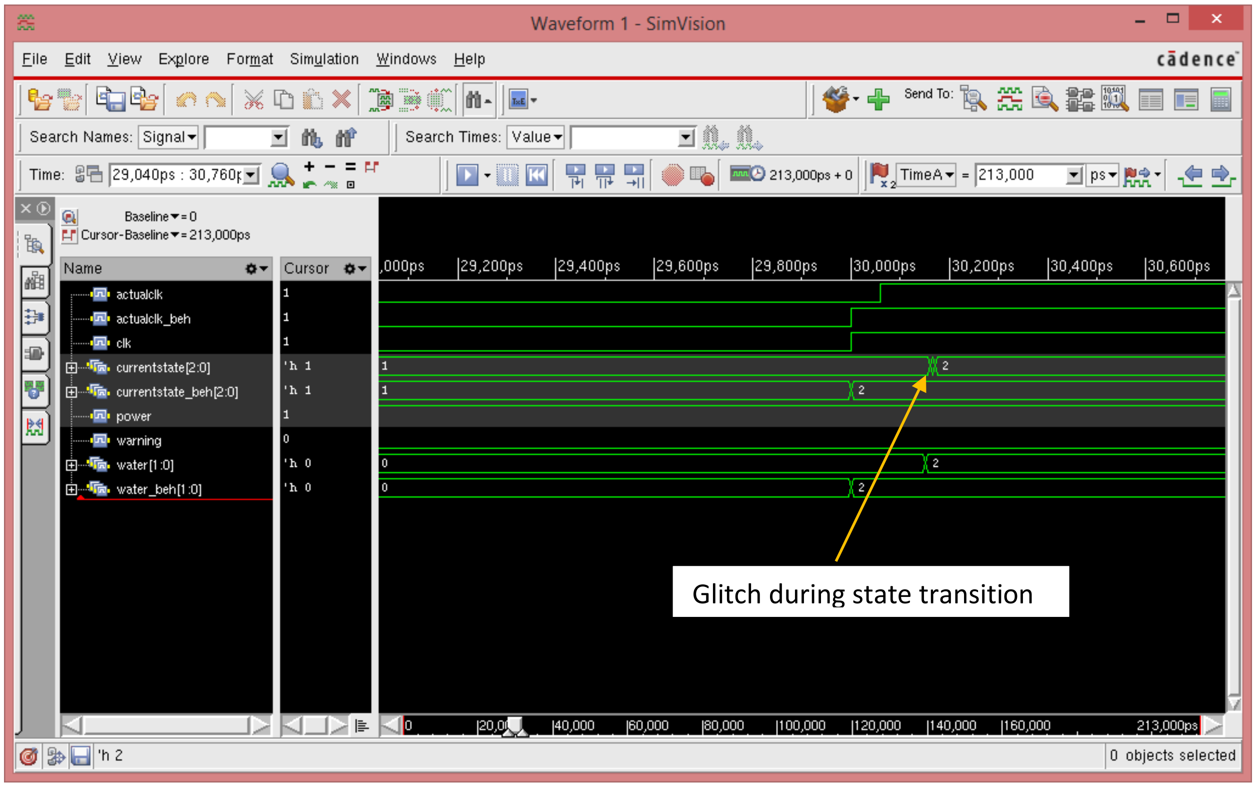
Glitches:

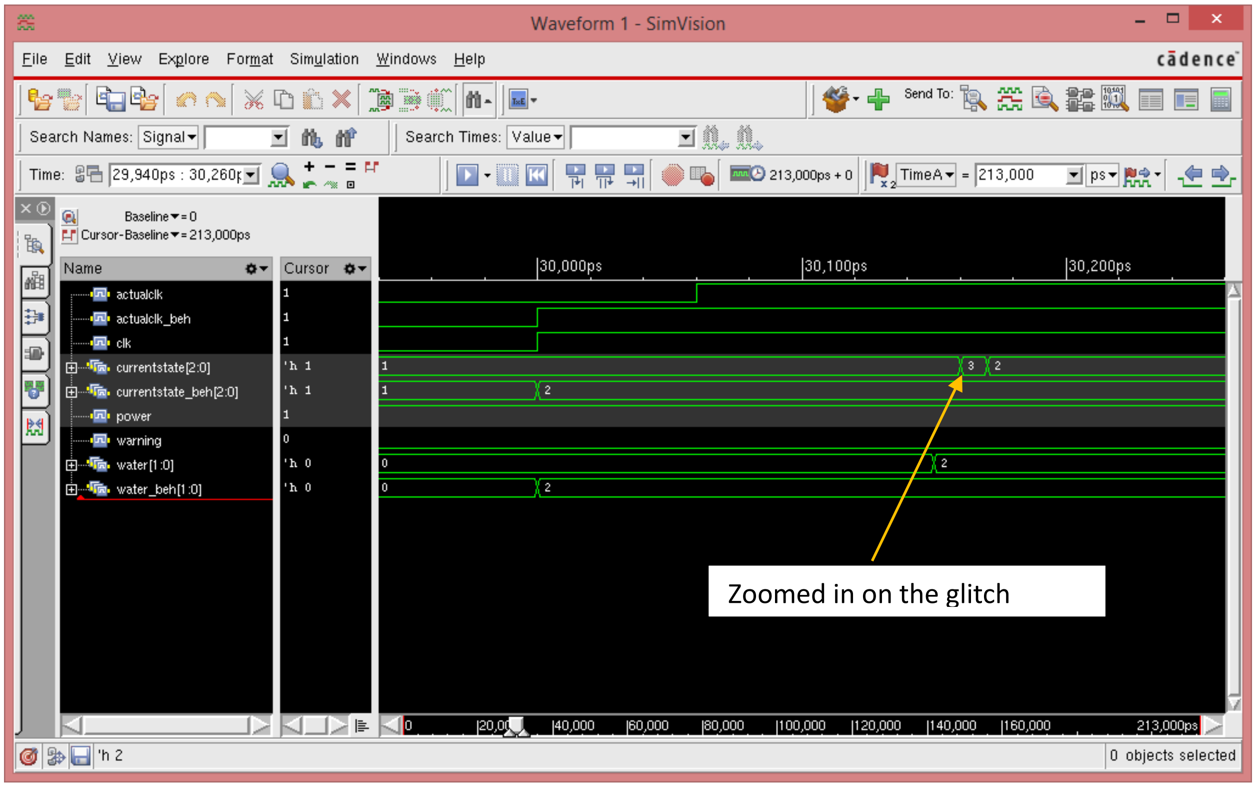


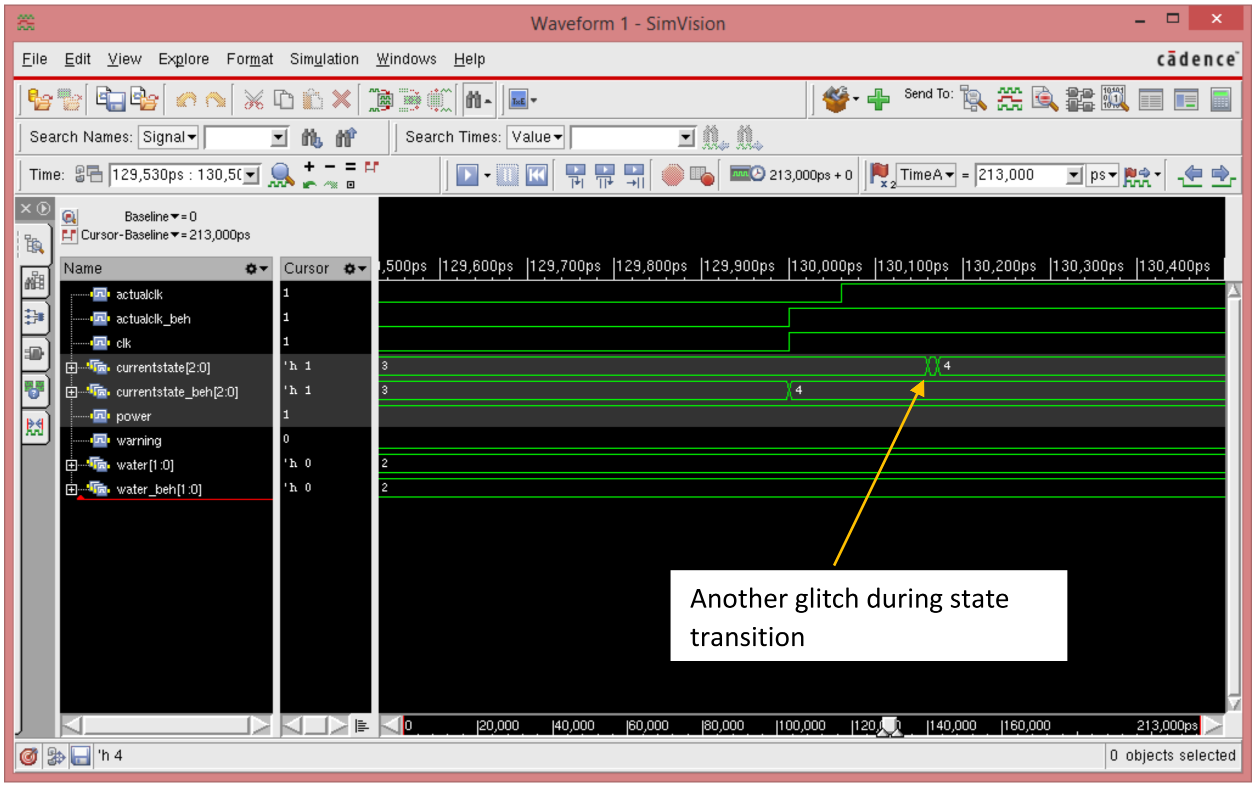
Name	Cursor
actualclk	1
actualclk_beh	1
clk	1
currentstate[2:0]	'h 1
currentstate_beh[2:0]	'h 1
power	1
warning	0
water[1:0]	'h 0
water_beh[1:0]	'h 0

Figure 5: Testbench waveform comparing between behavioral vs post-synthesized circuits

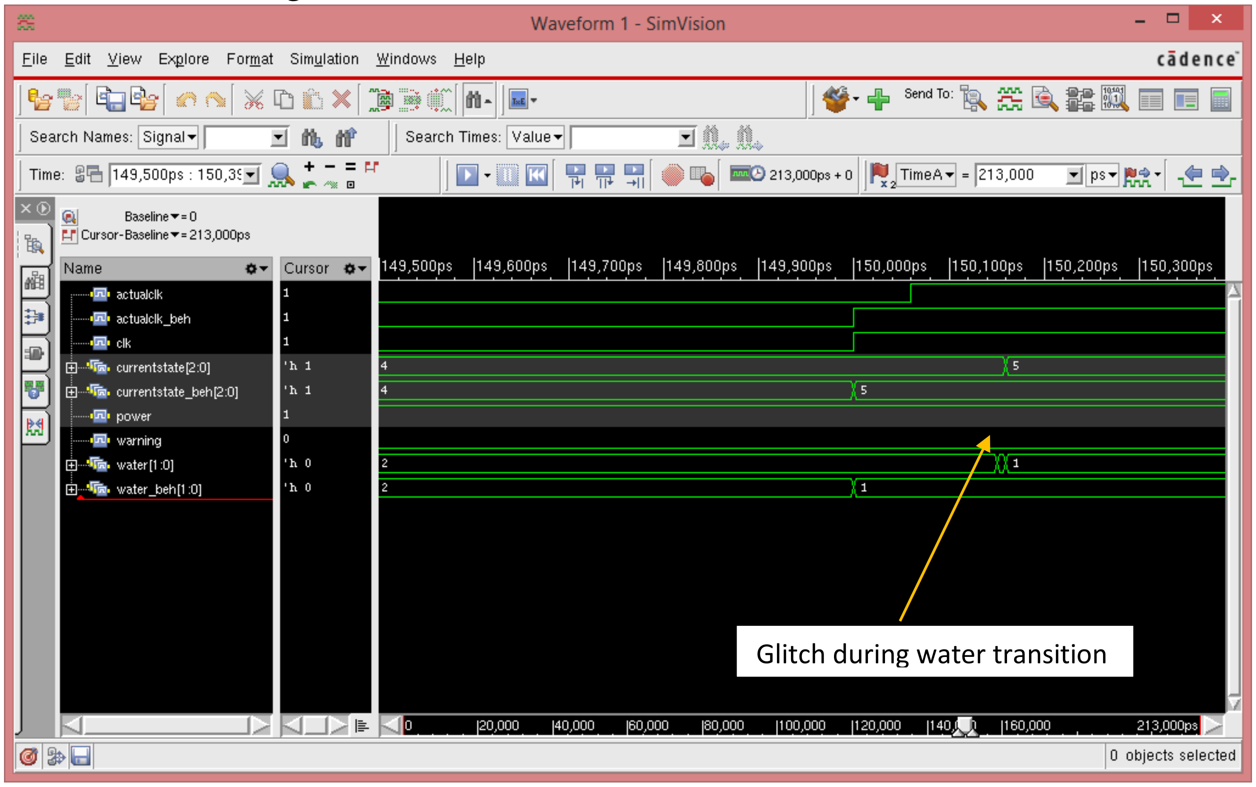
Glitches in State transition:







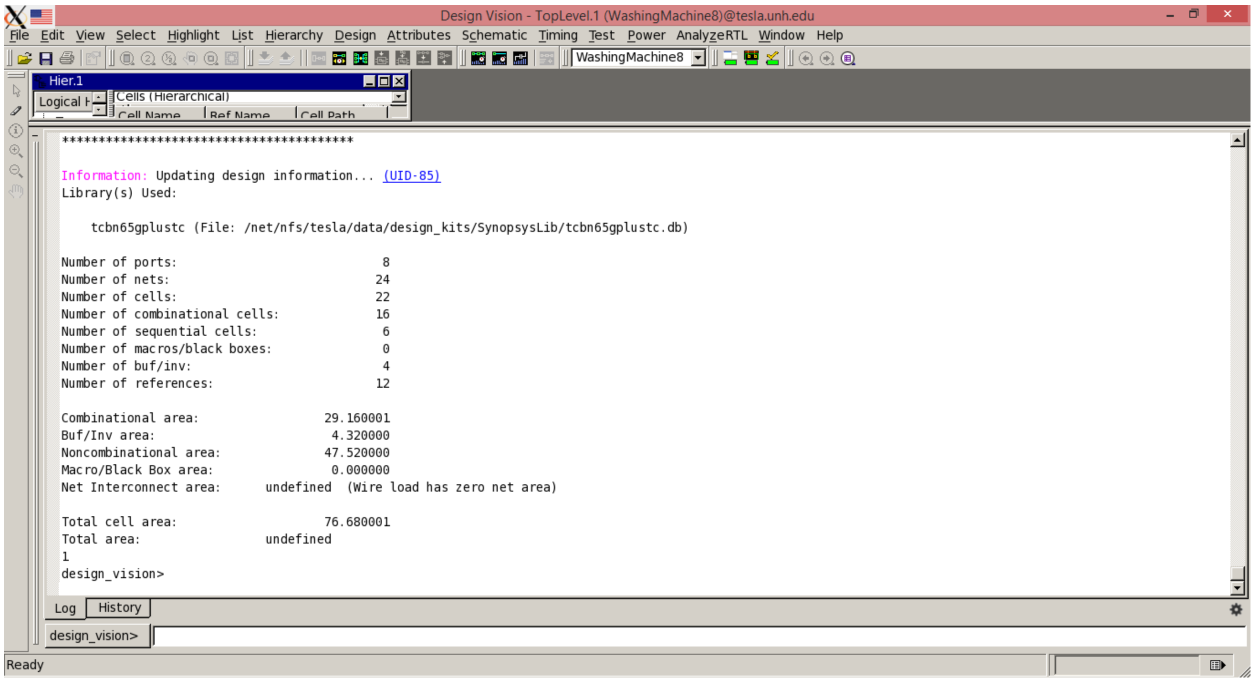
Water state transition glitch:



Testbench Source Code:

```
`timescale 1ns/1ns
module WashingMachineTBoriginal8;

  reg power;
  wire[2:0] currentstate, currentstate_beh;
  reg clk;
  wire actualclk,actualclk_beh;
  wire[1:0] water, water_beh;
  reg warning;

  WashingMachine8PostV w0(.clkorig(clk), .power(power), .finalwater(water),
.finalfinalstate(currentstate), .clk(actualclk));

  WashingMachine8 w1(.clkorig(clk), .power(power), .finalwater(water_beh),
.finalfinalstate(currentstate_beh), .clk(actualclk_beh));

  initial
  begin
    $sdf_annotate("WashingMachine8.sdf",w0,,,"MAXIMUM");

          power = 1'b0;
          clk = 1'b0;
    #13   power = 1'b1;
    #40   power = 1'b0;
    #10   power = 1'b1;
    #150 $finish;
  end

  always
  begin
    #10 clk = ~clk;
  end

  always@(clk, power)
  begin
  if(currentstate == (3'h2 || 3'h3 || 3'h4)) //Checking water temperature for
wash states
      begin
        case(water)
            2'b01:warning=1'b0;
            default:warning=1'b1;
        endcase
      end
      else
      begin
        warning=1'b0;
      end

      if(currentstate == (3'h5 || 3'h6 || 3'h7)) //Checking water temperature
for rinse states
      begin
        case(water)
            2'b10:warning=1'b0;
            default:warning=1'b1;
        endcase
      end
      else
      begin
        warning=1'b0;
      end

      if(currentstate == (3'h0 || 3'h1)) //Checking water state for Idle and 
Off states
      begin
        case(water)
            2'b00:warning=1'b0;
            default:warning=1'b1;
        endcase
      end
      else
      begin
        warning=1'b0;
      end

      //Checking clock and state when power is out
      if(power == 1'b0)
      begin
        case(actualclk)
            1'b1:warning = 1'b1;
            1'b0:warning = 1'b0;
        endcase
        case(currentstate)
            3'h0:warning = 1'b0;
            default:warning = 1'b1;
        endcase
      end
      else
        warning = 1'b0;

      $display("clock=%b, power=%b, current state=%d, water temperature=%d, 
warning=%b",clk,power,currentstate,water,warning);
    end
  endmodule
```

Reporting the area of our design:


Design Vision - TopLevel.1 (WashingMachine8)@tesla.unh.edu
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
WashingMachine8
Hier.1
Logical H
Cells (Hierarchical)
Cell Name
Ref Name
Cell Path
Log
History
design_vision>
Ready

```
****************************************

Information: Updating design information... (UID-85)
Library(s) Used:

    tcbn65gplustc (File: /net/nfs/tesla/data/design_kits/SynopsysLib/tcbn65gplustc.db)

Number of ports:                            8
Number of nets:                            24
Number of cells:                           22
Number of combinational cells:             16
Number of sequential cells:                 6
Number of macros/black boxes:               0
Number of buf/inv:                          4
Number of references:                      12

Combinational area:                 29.160001
Buf/Inv area:                        4.320000
Noncombinational area:              47.520000
Macro/Black Box area:                0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:                    76.680001
Total area:                 undefined
1
design_vision>
```