Ain Shams University
Faculty of Engineering

جامعة عين شمس
كلية الهندسة

CSE215

Electronic Design Automation

Name:

Mohamed Ibrahim El Ghamry

ID:

15P6019

Email:

15P6019@eng.asu.edu.eg

mohamed.elghamry97@gmail.com

Project:

Vending Machine

https://github.com/Ghamry97/Vending-Machine

Junior

Computer Engineering and Software Systems

Introduction

Vending Machine is an automated machine that provides items like drink and juice. The logic of this machine is implemented behaviorally using VHDL code and ModelSim.

Vending Machine Features:

- Sells soft drink and juice.
- The price of the drink is **1.25LE**.
- The machine only accepts 1LE, 0.5LE and 0.25LE.
- The user first enters the money, then selects either a **soft drink** or **juice**.
- The machine returns the change if any.

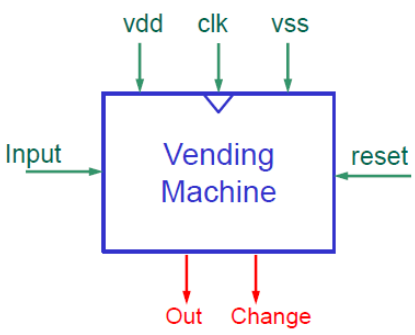
In this project we are building a simple frame decoder chip. There are 4 parts of this project:

- **Part 1 – High-Level Design**: We've implemented the behavioral VHDL code (FSM system description) and tested it with a testbench to check that the machine behaves correctly (verification).
- **Part 2 – Low-Level Synthesis, and Part 3 – Design for test (DFT):** We've synthesized our FSM with SYF tool using different state encodings, Boolean optimized the network with BOOM tool, Library mapped the network with BOOG tool to obtain a structural view, Netlist optimized the BOOG output with LOON tool, Netlist visualized the LOON output using XSCH tool, Netlist Checked using formal verification with FLABEH and PROOF tools, Delay simulated the obtained encoded code with ModelSim, and finally Scan-Path Insertion(DFT) was done with SCAPIN tool and simulated/verified with ModelSim.
- **Part 4 – Placement and Routing:** This part is about Physical Synthesis, but before starting physical synthesis, we must first construct the chip floorplanning. We've launched placement to our SCAPIN tool output file (.vst) with OCP tool, then we've routed the placed cells with NERO tool, then we started Netlist Extraction with COUGAR tool and we made the Netlist Comparison with LVX tool. After that we made the Design-rule Checking with DRUC tool. Finally, the generated symbolic layout is converted to a real technology using a cif output format for the real layout with the S2R tool (Symbolic-to-Real Conversion) and checked the real view with DREAL real layout viewer.

Details

Input	**(Encoding)**
0.25 LE	000
0.5 LE	001
1.0 LE	010
Soft Drink	011
Juice	100

States	
S0	Idle
S1	Got 0.25 LE
S2	Got 0.5 LE
S3	Got 0.75 LE
S4	Got 1.0 LE
S5	Got 1.25 LE
S6	Got 1.5 LE
S7	Got 1.75 LE
S8	Got 2.0 LE



Output	**(Encoding)**
Nothing	00
Soft Drink	01
Juice	10

Change	**(Encoding)**
No Change	00
0.25 LE	01
0.5 LE	10
0.75 LE	11

FSM Type: Mealy
Transitions: Input/OutputChange

Ex:	Input: 000 == 0.25 LE
000/0000	Output: 00 == Nothing
	Change: 00 == No Change

Results and Snapshots

Finite state machine

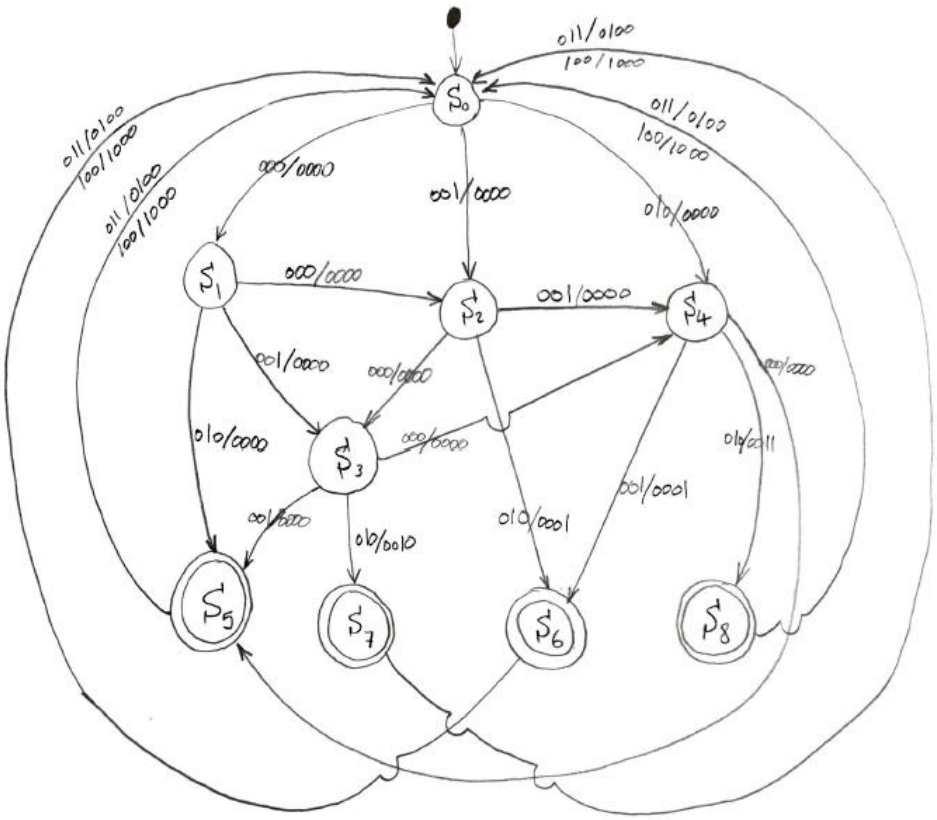


Fig (1): Finite state machine

Test bench test strategy tables

Term	**Meaning**
input <= "000"; ASSERT change = "00" and output = "00"	Current state: S0 Input: 0.25 LE Output: Nothing Change: No Change Next state: S1
input <= "000"; ASSERT change = "00" and output = "00"	Current state: S1 Input: 0.25 LE Output: Nothing Change: No Change Next state: S2

input <= "000"; ASSERT change = "00" and output = "00"	Current state: S2 Input: 0.25 LE Output: Nothing Change: No Change Next state: S3
input <= "000"; ASSERT change = "00" and output = "00"	Current state: S3 Input: 0.25 LE Output: Nothing Change: No Change Next state: S4
input <= "000"; ASSERT change = "00" and output = "00"	Current state: S4 Input: 0.25 LE Output: Nothing Change: No Change Next state: S5
input <= "011"; ASSERT change = "00" and output = "01"	Current state: S5 Input: Soft Drink Output: Soft Drink Change: No Change Next state: S0
input <= "001"; ASSERT change = "00" and output = "00"	Current state: S0 Input: 0.5 LE Output: Nothing Change: No Change Next state: S2
input <= "010"; ASSERT change = "00" and output = "00"	Current state: S2 Input: 1.0 LE Output: Nothing Change: No Change Next state: S6
input <= "100"; ASSERT change = "01" and output = "10"	Current state: S6 Input: Juice Output: Juice Change: 0.25 LE Next state: S0

ModelSim Simulation

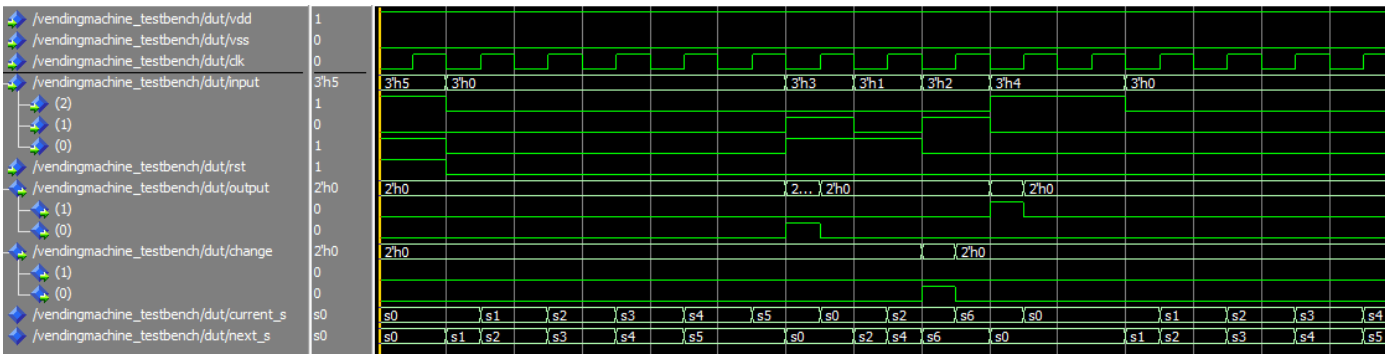


Fig (2): Simulation wave output of the original behavioral vhdl code

XFSM Tool Output

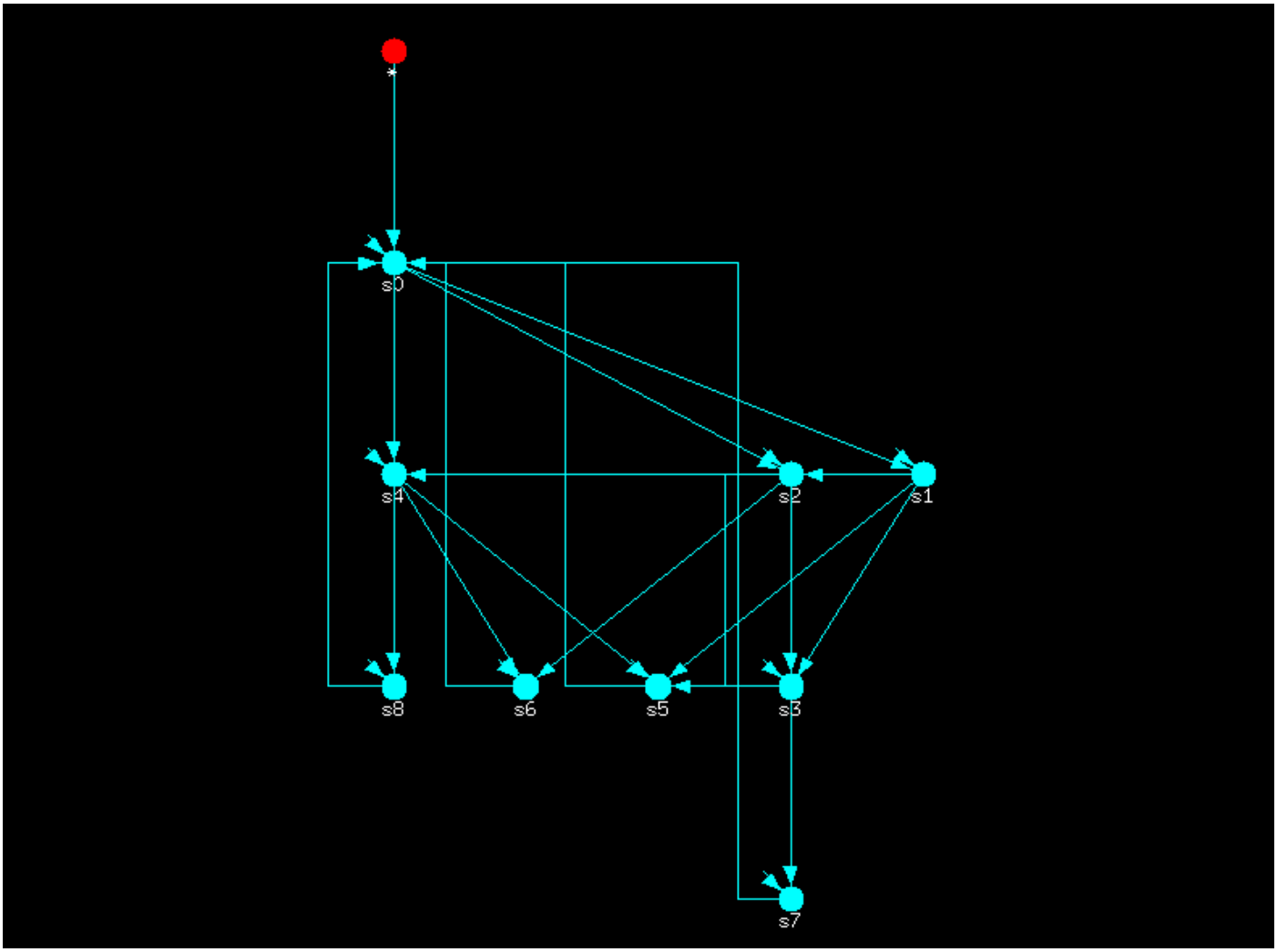


Fig (3): Alliance xfsm tool output of the original behavioral vhdl code

Boolean Network Optimization [BOOM] Comparisons

Encoding	Initial cost			Optimization parameters	Final cost			Literals after factorization
	Surface	Depth	Literals		Surface	Depth	Literals	
a	355500	13	245	Algorithm: simulated annealing Keep aux: no Area: 50 % Delay: 50 % Level: 0	155500	8	137	108
j	358750	13	249		155500	8	159	90
m	362000	13	251		200750	8	176	75
o	332500	9	223		158500	5	155	68
r	364000	14	225		195250	8	175	50

Netlist Optimization [LOON] Comparisons

Encoding	Critical path delay	Area (with over-cell routing)	Best Delay	Best Area
a	2958 ps	129000 lamda²		✓
j	3234 ps	140000 lamda²		
m	2776 ps	144500 lamda²		
o	2412 ps	152250 lamda²	✓	
r	2775 ps	151500 lamda²		

I would prefer to go with the o encoding, as the machine need to be as fast as possible in corresponding to a user input.

XSCH Tool Output

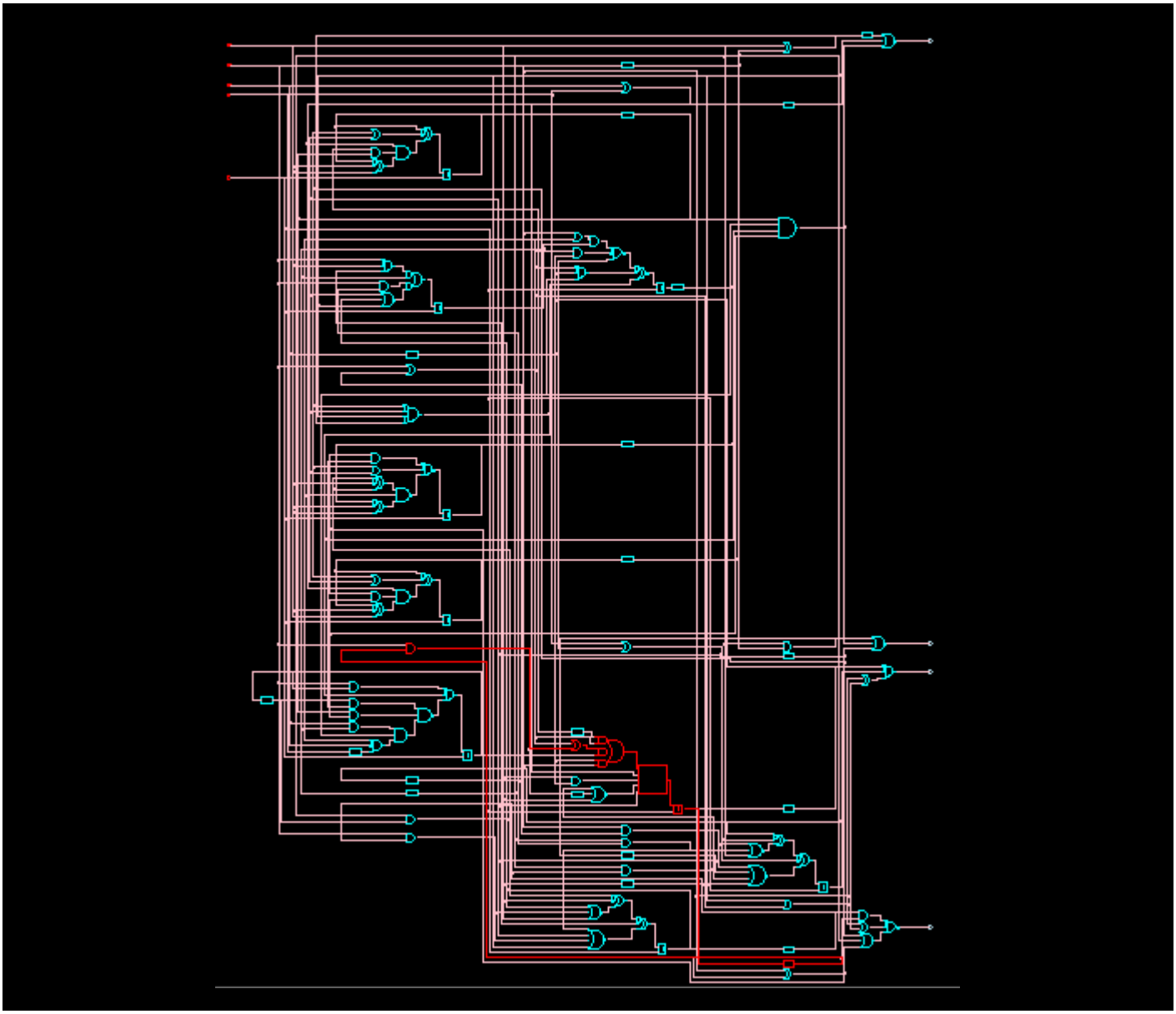
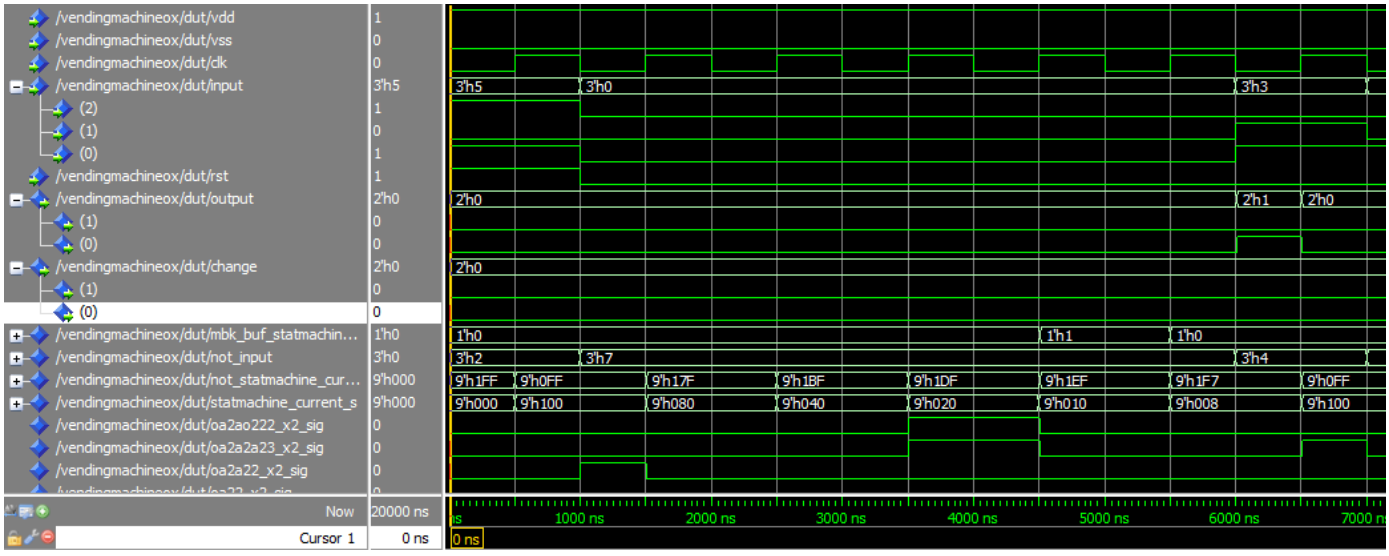


Fig (4): Alliance xsch tool output of the best implementation (-o encoding)

ModelSim Simulation #2



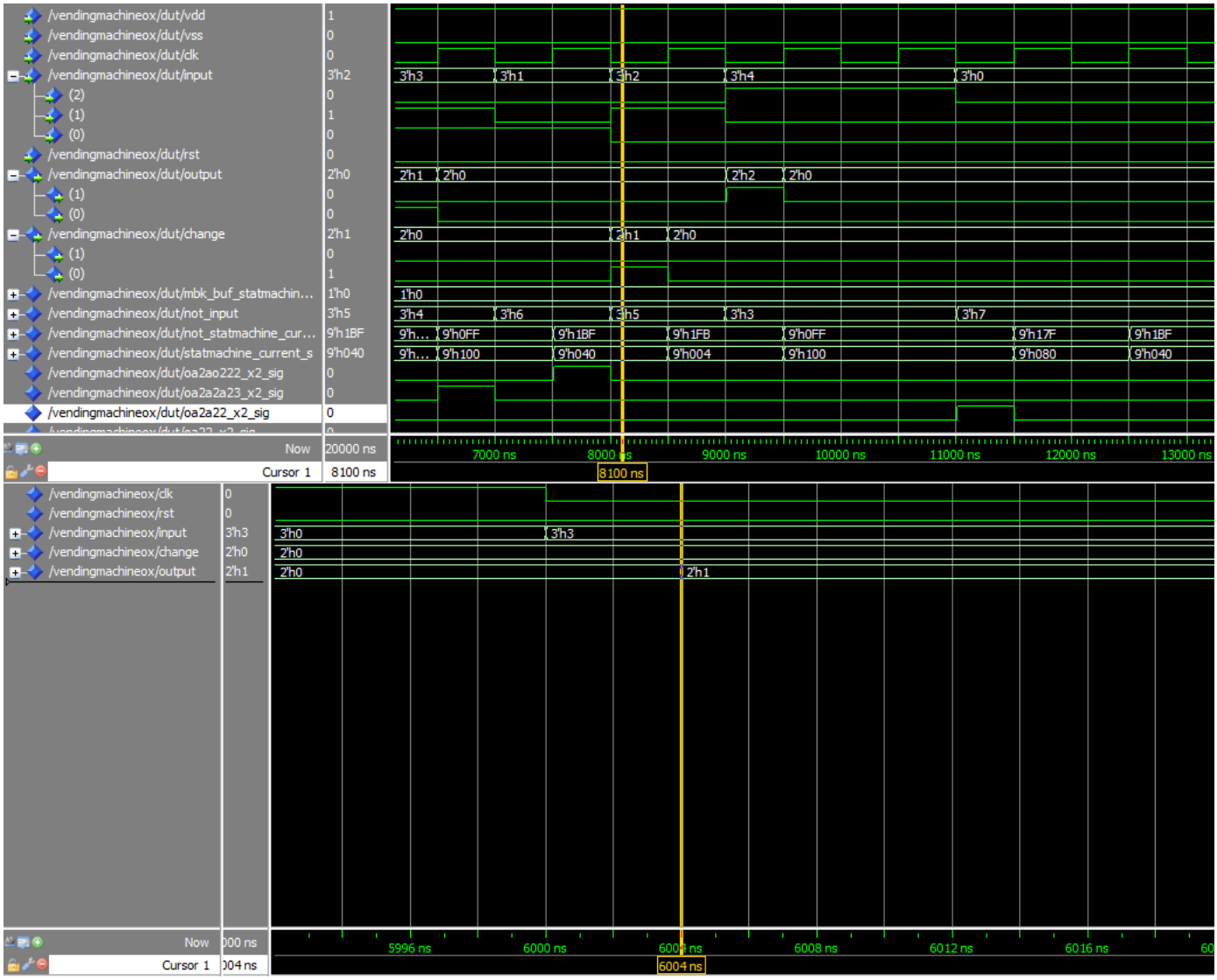
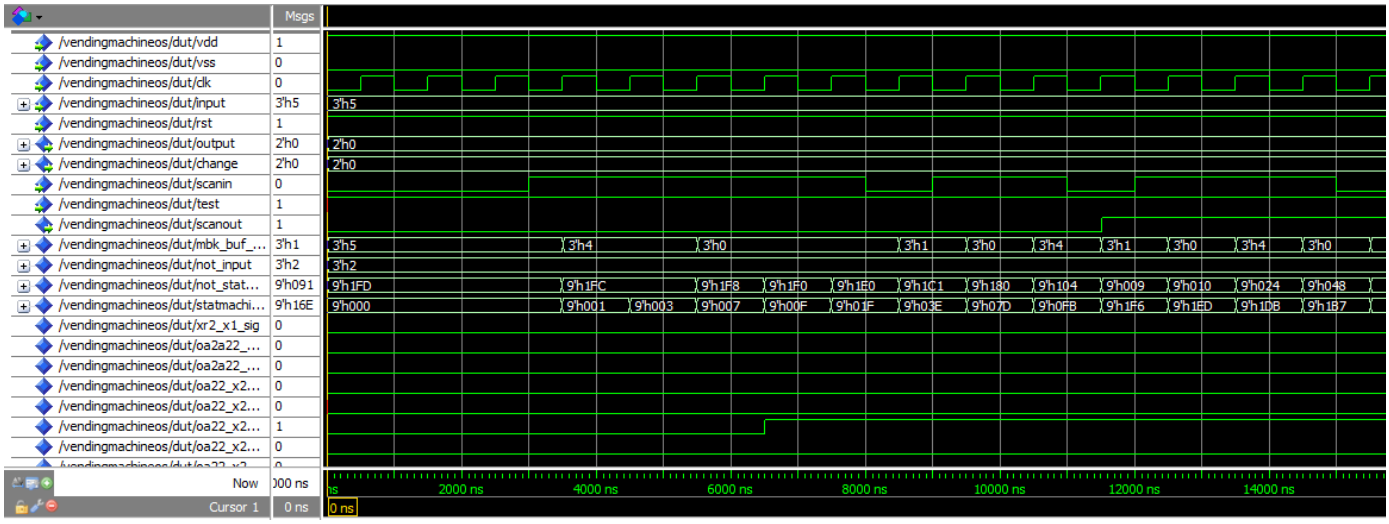


Fig (5): Simulation wave output of the -o encoded (loon) vhdl code

We can see in the last snapshot of the wave, that the output is delayed by 4 ns



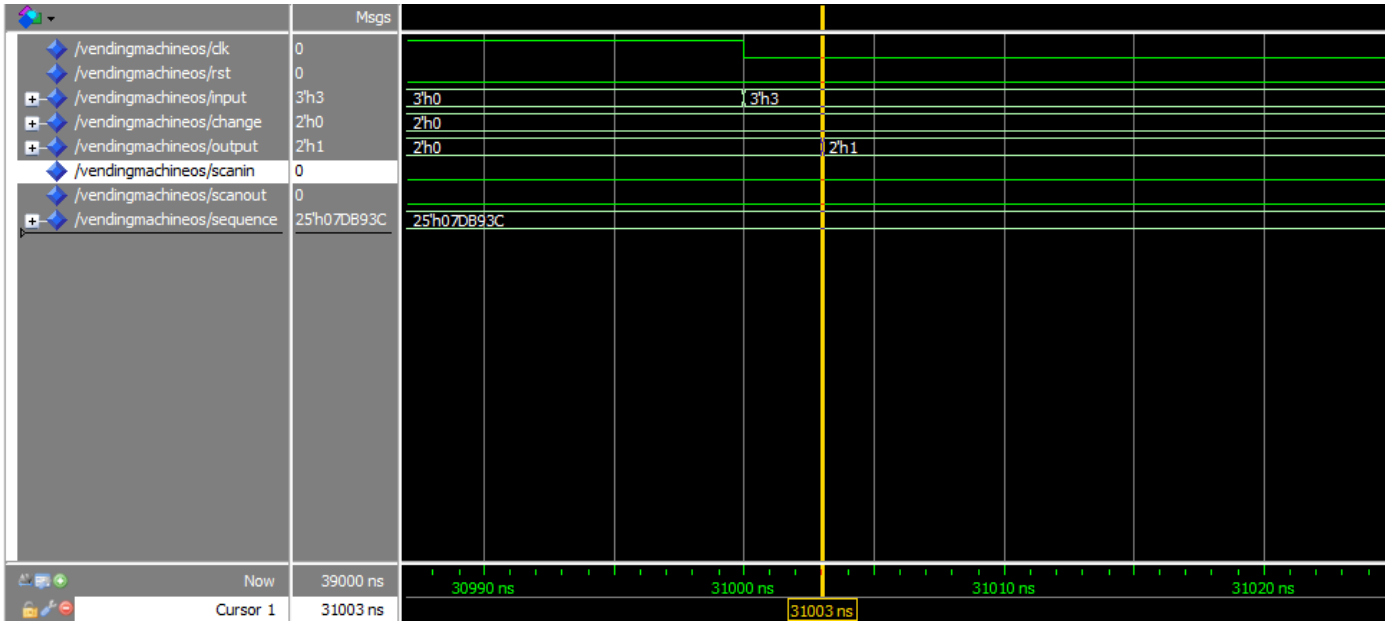


Fig (6): Simulation wave output of the -o encoded (scapin) tool output

We can see in the last snapshot of the wave, that the output is delayed by 3 ns

GRAAL Tool Output

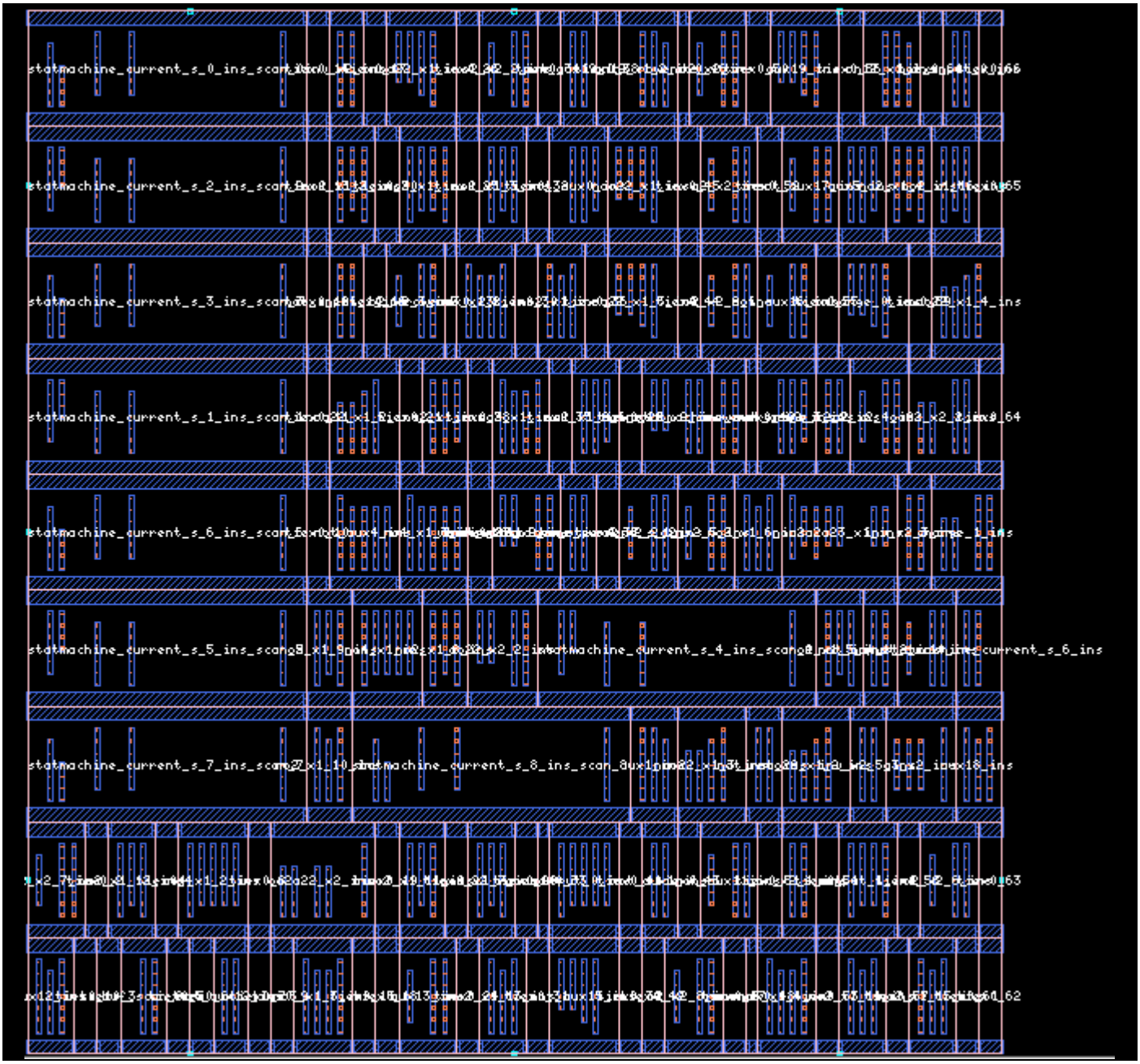


Fig (7) Graal tool on ocp file output

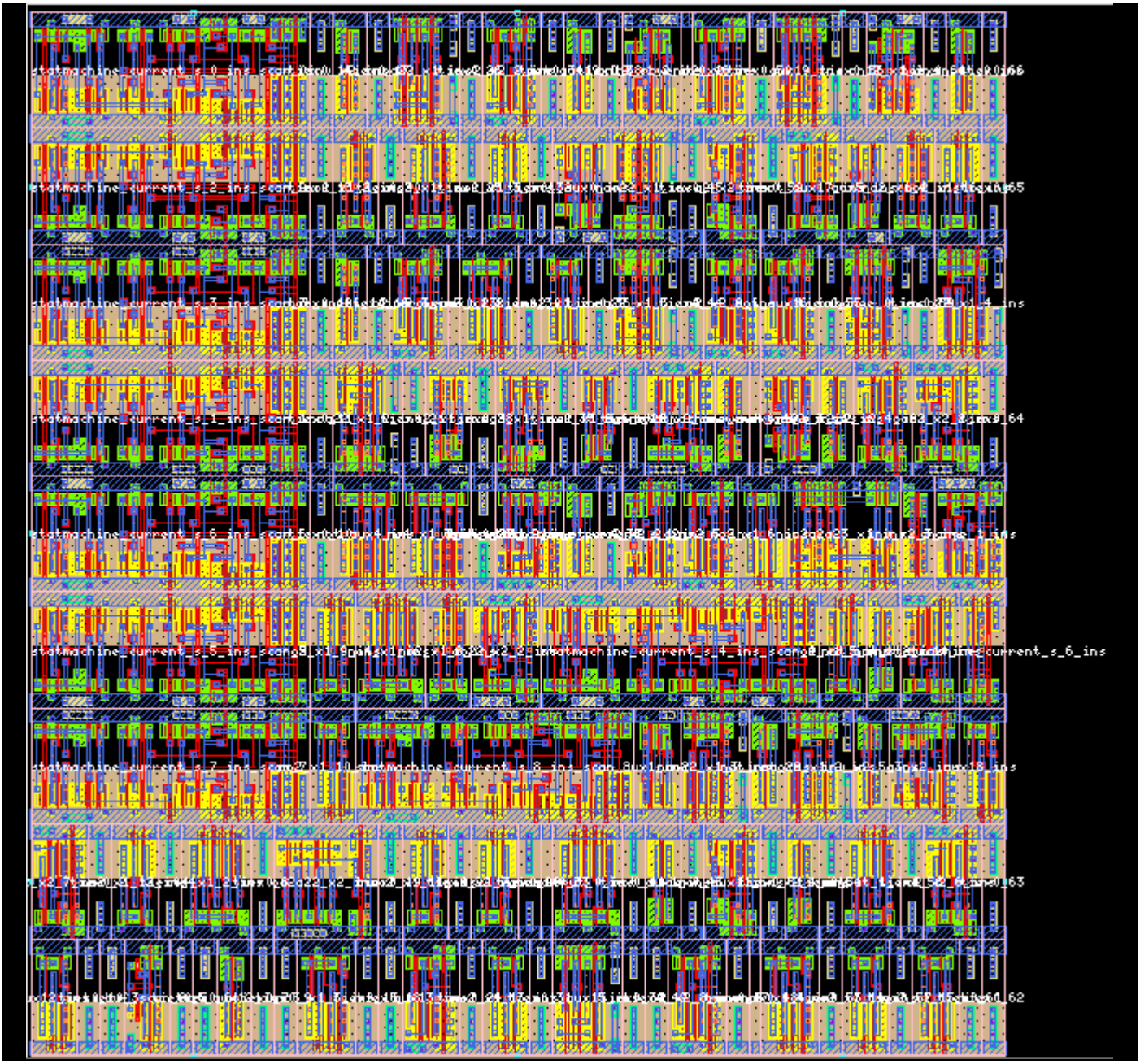


Fig (8) Graal tool on ocp file flatten output

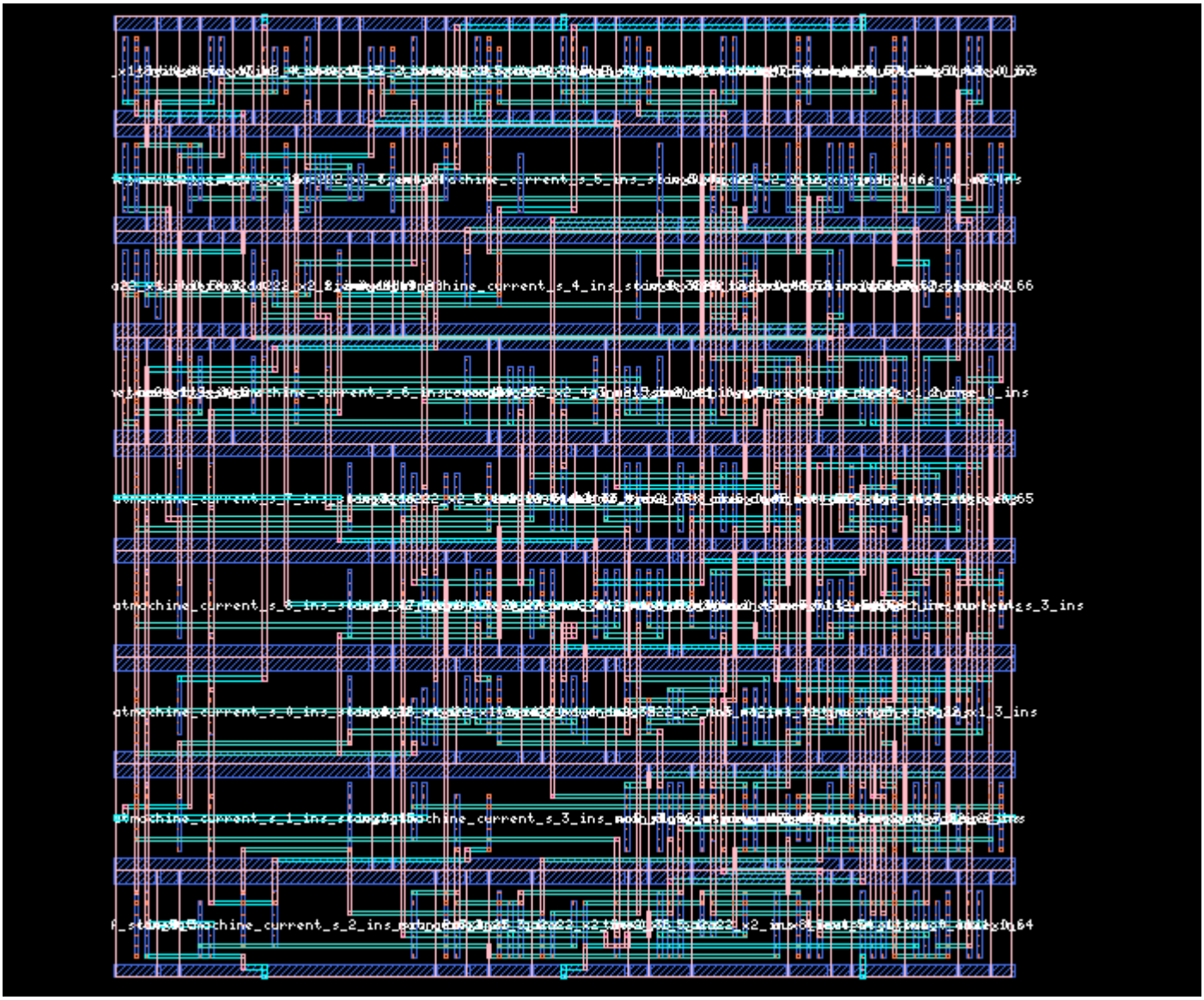


Fig (9) Graal tool on nero file output

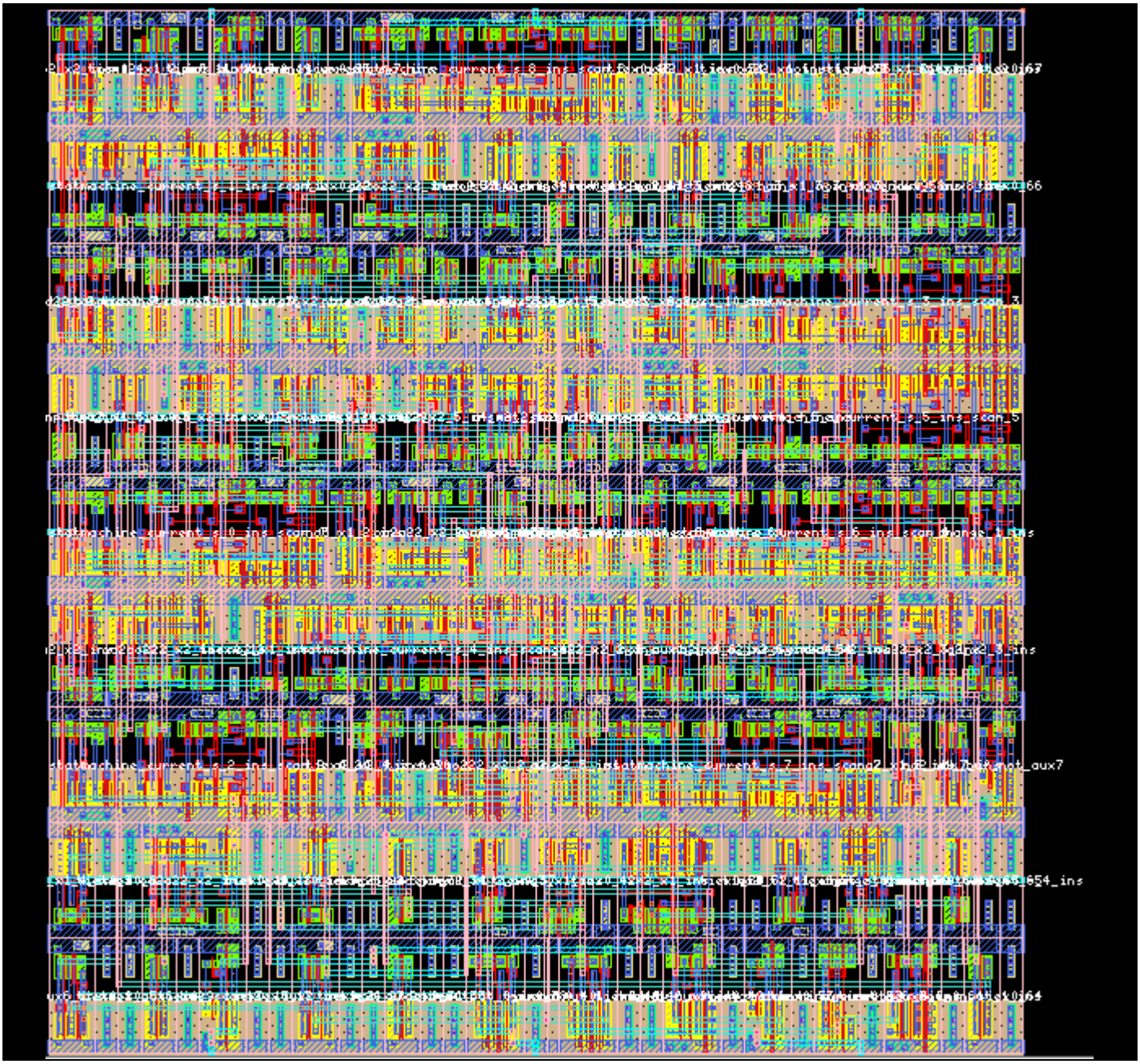


Fig (10) Graal tool on nero file flatten output

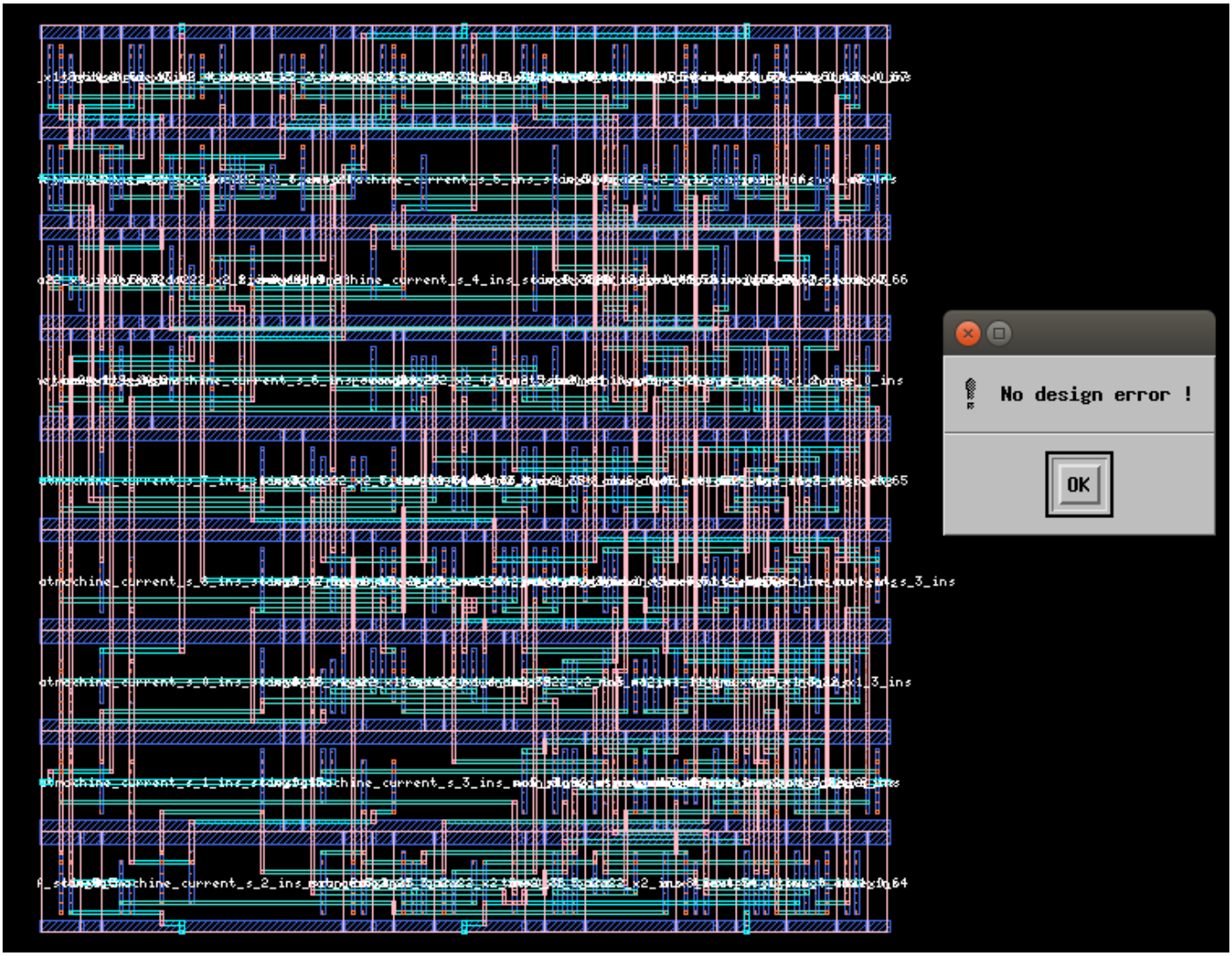


Fig (11) Graal tool on nero file druc output

DREAL Tool Output

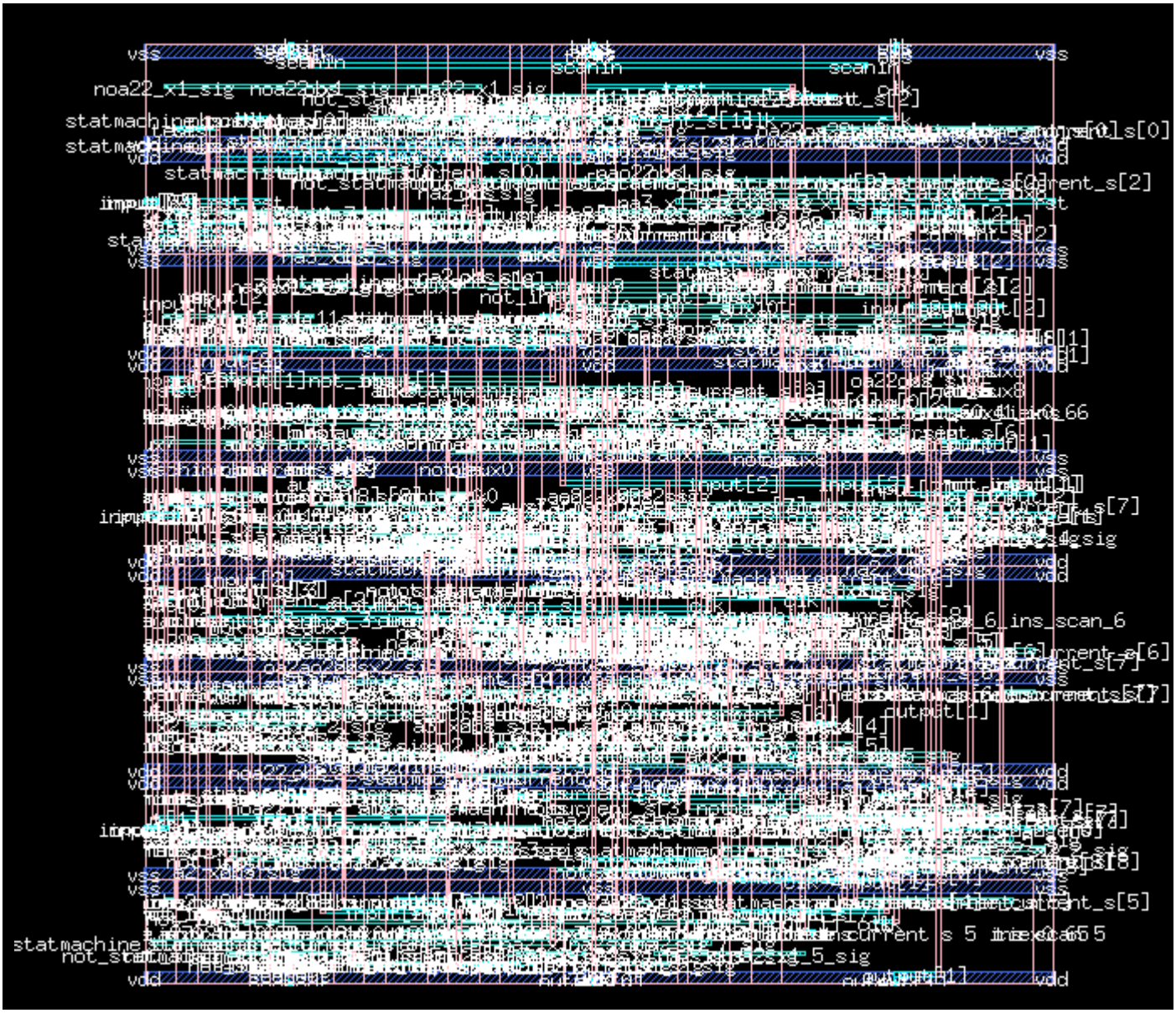


Fig (12) DREAL tool on s2r file output

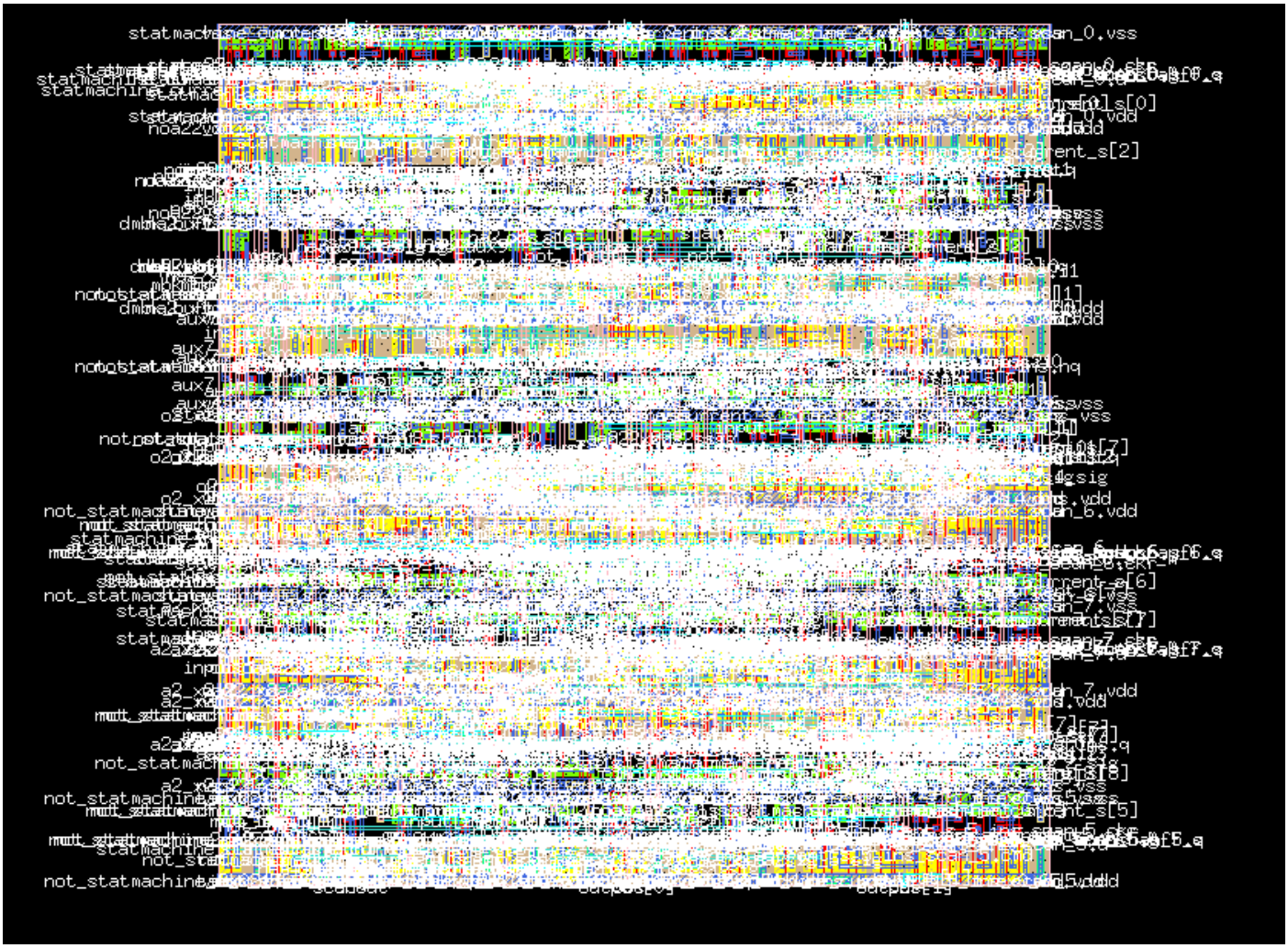


Fig (13) DREAL tool on s2r file flatten output

Appendix

vendingmachine.vhd file

```
entity vendingMachine is
	port(
		vdd: in bit;
		vss: in bit;
		clk: in bit;
		input: in bit_vector(2 downto 0);
		rst: in bit;
		output: out bit_vector(1 downto 0);
		change: out bit_vector(1 downto 0)
	);
end vendingMachine;

architecture beh of vendingMachine is
	type state_type is (s0, s1, s2, s3, s4, s5, s6, s7, s8);
	signal current_s, next_s: state_type;

	--Synthesis directives:
	--pragma current_state current_s
	--pragma next_state next_s
	--pragma clock clk

	begin
	process(clk, rst)
		begin
			if (rising_edge(clk) and rst = '1') then
```

```
                current_s <= s0;
            elsif(rising_edge(clk)) then
                current_s <= next_s;
            end if;
    end process;

    process (current_s, input)
        begin
            case current_s is
                when s0 =>
                    output <= "00";
                    change <= "00";
                    if(input = "000") then
                        next_s <= s1;
                    elsif(input = "001") then
                        next_s <= s2;
                    elsif(input = "010") then
                        next_s <= s4;
                    end if;

                when s1 =>
                    output <= "00";
                    change <= "00";
                    if(input = "000") then
                        next_s <= s2;
                    elsif(input = "001") then
                        next_s <= s3;
                    elsif(input = "010") then
                        next_s <= s5;
                    end if;

                when s2 =>
                    output <= "00";
                    if(input = "000") then
                        next_s <= s3;
                        change <= "00";
                    elsif(input = "001") then
                        next_s <= s4;
                        change <= "00";
                    elsif(input = "010") then
                        next_s <= s6;
                        change <= "01";
                    end if;

                when s3 =>
                    output <= "00";
                    if(input = "000") then
                        next_s <= s4;
                        change <= "00";
                    elsif(input = "001") then
                        next_s <= s5;
                        change <= "00";
                    elsif(input = "010") then
                        next_s <= s7;
                        change <= "10";
                    end if;

                when s4 =>
                    output <= "00";
                    if(input = "000") then
                        next_s <= s5;
                        change <= "00";
                    elsif(input = "001") then
                        next_s <= s6;
```

```
                              change <= "01";
                        elsif(input = "010") then
                              next_s <= s8;
                              change <= "11";
                        end if;

                  when s5 =>
                        change <= "00";
                        if(input = "011") then
                              next_s <= s0;
                              output <= "01";
                        elsif(input = "100") then
                              next_s <= s0;
                              output <= "10";
                        end if;

                  when s6 =>
                        change <= "00";
                        if(input = "011") then
                              next_s <= s0;
                              output <= "01";
                        elsif(input = "100") then
                              next_s <= s0;
                              output <= "10";
                        end if;

                  when s7 =>
                        change <= "00";
                        if(input = "011") then
                              next_s <= s0;
                              output <= "01";
                        elsif(input = "100") then
                              next_s <= s0;
                              output <= "10";
                        end if;

                  when s8 =>
                        change <= "00";
                        if(input = "011") then
                              next_s <= s0;
                              output <= "01";
                        elsif(input = "100") then
                              next_s <= s0;
                              output <= "10";
                        end if;
            end case;
      end process;
end beh;
```

testbench.vhd file

```
-- Entity declaration for your testbench. Don't declare any ports here
ENTITY vendingMachine_testbench IS
END ENTITY vendingMachine_testbench;

ARCHITECTURE testbench OF vendingMachine_testbench IS

-- Component Declaration for the Device Under Test (DUT)
COMPONENT vendingMachine IS
port(
            vdd: in bit;
            vss: in bit;
            clk: in bit;
            input: in bit_vector(2 downto 0);
```

```
            rst: in bit;
            output: out bit_vector(1 downto 0);
            change: out bit_vector(1 downto 0)
      );
END COMPONENT vendingMachine;

FOR dut: vendingMachine USE ENTITY WORK.vendingMachine (beh);

-- Inputs
SIGNAL clk       : bit := '0';
SIGNAL rst  : bit := '1';
SIGNAL vdd       : bit := '1';
SIGNAL vss       : bit := '0';
SIGNAL input     : bit_vector(2 Downto 0) := "101";

-- Outputs
SIGNAL change    : bit_vector(1 Downto 0);
SIGNAL output    : bit_vector(1 Downto 0);

-- Constants and Clock period definitions
constant clk_period : time := 1000 ns;
BEGIN
      dut: vendingMachine PORT MAP (vdd, vss, clk, input, rst, output, change);
      clk_process :process
   begin
        clk <= '0';
        wait for clk_period/2;
        clk <= '1';
        wait for clk_period/2;
   end process;
   stim_proc: PROCESS IS
BEGIN
      WAIT FOR clk_period; --For the output to be stable
      ASSERT change = "00" and output = "00"
      REPORT "Reset error"
      SEVERITY error;

      rst <= '0';

      input <= "000";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "000";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "000";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "000";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "000";
```

```
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "011";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "01"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "001";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "010";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "100";
      WAIT FOR clk_period;
      ASSERT change = "01" and output = "10"
      REPORT "Outputs error"
      SEVERITY error;
end process;
end;
```

vendingmachineo_l.vhd file

```
LIBRARY sxlib_ModelSim;
entity vendingmachineo_l is
   port (
      vdd    : in      bit;
      vss    : in      bit;
      clk    : in      bit;
      input  : in      bit_vector(2 downto 0);
      rst    : in      bit;
      output : out     bit_vector(1 downto 0);
      change : out     bit_vector(1 downto 0)
 );
end vendingmachineo_l;

architecture structural of vendingmachineo_l is
Component a4_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component inv_x4
   port (
      i   : in      bit;
      nq  : out     bit;
      vdd : in      bit;
```

```
      vss : in      bit
 );
end component;

Component ao2o22_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component noa22_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component oa2a2a23_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      i4  : in      bit;
      i5  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component oa2ao222_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      i4  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component no4_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
```

```
end component;

Component oa22_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component oa2a22_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component na2_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component a3_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component na3_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component sff1_x4
   port (
      ck  : in      bit;
      i   : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
```

```
 );
end component;

Component o3_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component o2_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component a2_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component nao22_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component no2_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component ao22_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
```

```
end component;

Component inv_x2
   port (
      i   : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component no3_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component buf_x2
   port (
      i   : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

signal mbk_buf_statmachine_current_s : bit_vector( 4 downto 4);
signal not_input                     : bit_vector( 2 downto 0);
signal not_statmachine_current_s     : bit_vector( 8 downto 0);
signal statmachine_current_s         : bit_vector( 8 downto 0);
signal oa2ao222_x2_sig               : bit;
signal oa2a2a23_x2_sig               : bit;
signal oa2a22_x2_sig                 : bit;
signal oa22_x2_sig                   : bit;
signal oa22_x2_9_sig                 : bit;
signal oa22_x2_8_sig                 : bit;
signal oa22_x2_7_sig                 : bit;
signal oa22_x2_6_sig                 : bit;
signal oa22_x2_5_sig                 : bit;
signal oa22_x2_4_sig                 : bit;
signal oa22_x2_3_sig                 : bit;
signal oa22_x2_2_sig                 : bit;
signal oa22_x2_10_sig                : bit;
signal o3_x2_sig                     : bit;
signal o2_x2_sig                     : bit;
signal o2_x2_2_sig                   : bit;
signal not_aux9                      : bit;
signal not_aux8                      : bit;
signal not_aux7                      : bit;
signal not_aux6                      : bit;
signal not_aux4                      : bit;
signal not_aux15                     : bit;
signal not_aux14                     : bit;
signal not_aux13                     : bit;
signal not_aux12                     : bit;
signal not_aux11                     : bit;
signal not_aux10                     : bit;
signal not_aux1                      : bit;
signal not_aux0                      : bit;
```

```
signal noa22_x1_sig                  : bit;
signal no4_x1_sig                    : bit;
signal no4_x1_2_sig                  : bit;
signal no3_x1_sig                    : bit;
signal no3_x1_4_sig                  : bit;
signal no3_x1_3_sig                  : bit;
signal no3_x1_2_sig                  : bit;
signal no2_x1_sig                    : bit;
signal no2_x1_5_sig                  : bit;
signal no2_x1_4_sig                  : bit;
signal no2_x1_3_sig                  : bit;
signal no2_x1_2_sig                  : bit;
signal nao22_x1_sig                  : bit;
signal nao22_x1_2_sig                : bit;
signal na3_x1_sig                    : bit;
signal na3_x1_2_sig                  : bit;
signal na2_x1_sig                    : bit;
signal na2_x1_4_sig                  : bit;
signal na2_x1_3_sig                  : bit;
signal na2_x1_2_sig                  : bit;
signal mbk_buf_not_aux7              : bit;
signal inv_x2_sig                    : bit;
signal inv_x2_3_sig                  : bit;
signal inv_x2_2_sig                  : bit;
signal aux5                          : bit;
signal aux16                         : bit;
signal aux14                         : bit;
signal aux13                         : bit;
signal aux1                          : bit;
signal ao22_x2_sig                   : bit;
signal ao22_x2_4_sig                 : bit;
signal ao22_x2_3_sig                 : bit;
signal ao22_x2_2_sig                 : bit;
signal a3_x2_sig                     : bit;
signal a3_x2_3_sig                   : bit;
signal a3_x2_2_sig                   : bit;
signal a2_x2_sig                     : bit;
signal a2_x2_8_sig                   : bit;
signal a2_x2_7_sig                   : bit;
signal a2_x2_6_sig                   : bit;
signal a2_x2_5_sig                   : bit;
signal a2_x2_4_sig                   : bit;
signal a2_x2_3_sig                   : bit;
signal a2_x2_2_sig                   : bit;

begin

not_aux4_ins : a4_x2
   port map (
      i0  => not_statmachine_current_s(1),
      i1  => not_statmachine_current_s(3),
      i2  => not_statmachine_current_s(2),
      i3  => not_statmachine_current_s(0),
      q   => not_aux4,
      vdd => vdd,
      vss => vss
   );

not_aux15_ins : o2_x2
   port map (
      i0  => not_aux14,
      i1  => not_input(1),
      q   => not_aux15,
      vdd => vdd,
```

```
      vss => vss
   );

not_aux12_ins : a2_x2
   port map (
      i0  => not_input(2),
      i1  => not_statmachine_current_s(8),
      q   => not_aux12,
      vdd => vdd,
      vss => vss
   );

not_aux11_ins : a2_x2
   port map (
      i0  => input(2),
      i1  => not_statmachine_current_s(6),
      q   => not_aux11,
      vdd => vdd,
      vss => vss
   );

not_aux10_ins : a2_x2
   port map (
      i0  => not_input(2),
      i1  => not_statmachine_current_s(7),
      q   => not_aux10,
      vdd => vdd,
      vss => vss
   );

not_aux9_ins : a2_x2
   port map (
      i0  => input(2),
      i1  => not_statmachine_current_s(5),
      q   => not_aux9,
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_8_ins : inv_x2
   port map (
      i   => statmachine_current_s(8),
      nq  => not_statmachine_current_s(8),
      vdd => vdd,
      vss => vss
   );

not_aux8_ins : a2_x2
   port map (
      i0  => not_input(2),
      i1  => not_statmachine_current_s(6),
      q   => not_aux8,
      vdd => vdd,
      vss => vss
   );

not_aux7_ins : a2_x2
   port map (
      i0  => input(2),
      i1  => not_statmachine_current_s(4),
      q   => not_aux7,
      vdd => vdd,
      vss => vss
   );
```

```
not_statmachine_current_s_3_ins : inv_x2
   port map (
      i   => statmachine_current_s(3),
      nq  => not_statmachine_current_s(3),
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_7_ins : inv_x2
   port map (
      i   => statmachine_current_s(7),
      nq  => not_statmachine_current_s(7),
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_6_ins : inv_x2
   port map (
      i   => statmachine_current_s(6),
      nq  => not_statmachine_current_s(6),
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_2_ins : inv_x2
   port map (
      i   => statmachine_current_s(2),
      nq  => not_statmachine_current_s(2),
      vdd => vdd,
      vss => vss
   );

not_aux6_ins : o2_x2
   port map (
      i0  => input(2),
      i1  => not_statmachine_current_s(5),
      q   => not_aux6,
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_5_ins : inv_x2
   port map (
      i   => statmachine_current_s(5),
      nq  => not_statmachine_current_s(5),
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_1_ins : inv_x2
   port map (
      i   => statmachine_current_s(1),
      nq  => not_statmachine_current_s(1),
      vdd => vdd,
      vss => vss
   );

not_aux14_ins : inv_x2
   port map (
      i   => aux14,
      nq  => not_aux14,
      vdd => vdd,
      vss => vss
```

```
   );

not_aux0_ins : o2_x2
   port map (
      i0  => input(2),
      i1  => not_statmachine_current_s(4),
      q   => not_aux0,
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_4_ins : inv_x4
   port map (
      i   => statmachine_current_s(4),
      nq  => not_statmachine_current_s(4),
      vdd => vdd,
      vss => vss
   );

not_aux13_ins : inv_x2
   port map (
      i   => aux13,
      nq  => not_aux13,
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_0_ins : inv_x2
   port map (
      i   => statmachine_current_s(0),
      nq  => not_statmachine_current_s(0),
      vdd => vdd,
      vss => vss
   );

not_aux1_ins : inv_x2
   port map (
      i   => aux1,
      nq  => not_aux1,
      vdd => vdd,
      vss => vss
   );

not_input_2_ins : inv_x2
   port map (
      i   => input(2),
      nq  => not_input(2),
      vdd => vdd,
      vss => vss
   );

not_input_1_ins : inv_x2
   port map (
      i   => input(1),
      nq  => not_input(1),
      vdd => vdd,
      vss => vss
   );

not_input_0_ins : inv_x2
   port map (
      i   => input(0),
      nq  => not_input(0),
      vdd => vdd,
```

```
      vss => vss
   );

aux16_ins : ao2o22_x2
   port map (
      i0  => not_aux13,
      i1  => not_aux1,
      i2  => not_aux14,
      i3  => aux5,
      q   => aux16,
      vdd => vdd,
      vss => vss
   );

aux14_ins : no2_x1
   port map (
      i0  => input(0),
      i1  => rst,
      nq  => aux14,
      vdd => vdd,
      vss => vss
   );

aux13_ins : no2_x1
   port map (
      i0  => rst,
      i1  => not_input(0),
      nq  => aux13,
      vdd => vdd,
      vss => vss
   );

aux5_ins : no2_x1
   port map (
      i0  => input(1),
      i1  => not_input(2),
      nq  => aux5,
      vdd => vdd,
      vss => vss
   );

aux1_ins : na2_x1
   port map (
      i0  => input(1),
      i1  => not_input(2),
      nq  => aux1,
      vdd => vdd,
      vss => vss
   );

oa22_x2_2_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(0),
      i1  => not_input(2),
      i2  => input(1),
      q   => oa22_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_ins : na2_x1
   port map (
      i0  => not_aux0,
      i1  => not_statmachine_current_s(0),
```

```
      nq  => na2_x1_sig,
      vdd => vdd,
      vss => vss
   );

a3_x2_ins : a3_x2
   port map (
      i0  => aux14,
      i1  => na2_x1_sig,
      i2  => oa22_x2_2_sig,
      q   => a3_x2_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_ins : no2_x1
   port map (
      i0  => not_aux13,
      i1  => not_aux1,
      nq  => no2_x1_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(0),
      i1  => no2_x1_sig,
      i2  => a3_x2_sig,
      q   => oa22_x2_sig,
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_0_ins : sff1_x4
   port map (
      ck  => clk,
      i   => oa22_x2_sig,
      q   => statmachine_current_s(0),
      vdd => vdd,
      vss => vss
   );

oa22_x2_4_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(1),
      i1  => not_input(2),
      i2  => input(1),
      q   => oa22_x2_4_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_2_ins : na2_x1
   port map (
      i0  => not_aux6,
      i1  => not_statmachine_current_s(1),
      nq  => na2_x1_2_sig,
      vdd => vdd,
      vss => vss
   );

a3_x2_2_ins : a3_x2
   port map (
```

```
      i0  => aux14,
      i1  => na2_x1_2_sig,
      i2  => oa22_x2_4_sig,
      q   => a3_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_2_ins : no2_x1
   port map (
      i0  => not_aux13,
      i1  => not_aux1,
      nq  => no2_x1_2_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_3_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(1),
      i1  => no2_x1_2_sig,
      i2  => a3_x2_2_sig,
      q   => oa22_x2_3_sig,
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_1_ins : sff1_x4
   port map (
      ck  => clk,
      i   => oa22_x2_3_sig,
      q   => statmachine_current_s(1),
      vdd => vdd,
      vss => vss
   );

oa22_x2_5_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(2),
      i1  => not_input(2),
      i2  => input(1),
      q   => oa22_x2_5_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_6_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(6),
      i1  => not_input(2),
      i2  => statmachine_current_s(2),
      q   => oa22_x2_6_sig,
      vdd => vdd,
      vss => vss
   );

na3_x1_ins : na3_x1
   port map (
      i0  => oa22_x2_6_sig,
      i1  => aux14,
      i2  => oa22_x2_5_sig,
      nq  => na3_x1_sig,
      vdd => vdd,
      vss => vss
```

```
   );

o2_x2_ins : o2_x2
   port map (
      i0  => not_aux13,
      i1  => not_aux1,
      q   => o2_x2_sig,
      vdd => vdd,
      vss => vss
   );

a2_x2_ins : a2_x2
   port map (
      i0  => not_aux0,
      i1  => not_statmachine_current_s(2),
      q   => a2_x2_sig,
      vdd => vdd,
      vss => vss
   );

nao22_x1_ins : nao22_x1
   port map (
      i0  => a2_x2_sig,
      i1  => o2_x2_sig,
      i2  => na3_x1_sig,
      nq  => nao22_x1_sig,
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_2_ins : sff1_x4
   port map (
      ck  => clk,
      i   => nao22_x1_sig,
      q   => statmachine_current_s(2),
      vdd => vdd,
      vss => vss
   );

ao22_x2_ins : ao22_x2
   port map (
      i0  => not_aux6,
      i1  => not_input(0),
      i2  => not_statmachine_current_s(3),
      q   => ao22_x2_sig,
      vdd => vdd,
      vss => vss
   );

a2_x2_2_ins : a2_x2
   port map (
      i0  => not_input(1),
      i1  => mbk_buf_statmachine_current_s(4),
      q   => a2_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_3_ins : no2_x1
   port map (
      i0  => input(2),
      i1  => not_input(1),
      nq  => no2_x1_3_sig,
      vdd => vdd,
```

```
      vss => vss
   );

a2_x2_3_ins : a2_x2
   port map (
      i0  => no2_x1_3_sig,
      i1  => statmachine_current_s(7),
      q   => a2_x2_3_sig,
      vdd => vdd,
      vss => vss
   );

nao22_x1_2_ins : nao22_x1
   port map (
      i0  => a2_x2_3_sig,
      i1  => a2_x2_2_sig,
      i2  => not_input(0),
      nq  => nao22_x1_2_sig,
      vdd => vdd,
      vss => vss
   );

noa22_x1_ins : noa22_x1
   port map (
      i0  => nao22_x1_2_sig,
      i1  => ao22_x2_sig,
      i2  => aux16,
      nq  => noa22_x1_sig,
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_3_ins : sff1_x4
   port map (
      ck  => clk,
      i   => noa22_x1_sig,
      q   => statmachine_current_s(3),
      vdd => vdd,
      vss => vss
   );

no3_x1_ins : no3_x1
   port map (
      i0  => not_aux8,
      i1  => mbk_buf_not_aux7,
      i2  => input(1),
      nq  => no3_x1_sig,
      vdd => vdd,
      vss => vss
   );

a2_x2_4_ins : a2_x2
   port map (
      i0  => input(1),
      i1  => mbk_buf_statmachine_current_s(4),
      q   => a2_x2_4_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_4_ins : no2_x1
   port map (
      i1  => not_aux7,
      i0  => not_input(1),
```

```
      nq  => no2_x1_4_sig,
      vdd => vdd,
      vss => vss
   );

inv_x2_ins : inv_x2
   port map (
      i   => not_aux6,
      nq  => inv_x2_sig,
      vdd => vdd,
      vss => vss
   );

oa2a2a23_x2_ins : oa2a2a23_x2
   port map (
      i0  => inv_x2_sig,
      i1  => not_input(1),
      i2  => no2_x1_4_sig,
      i3  => statmachine_current_s(8),
      i4  => mbk_buf_statmachine_current_s(4),
      i5  => input(2),
      q   => oa2a2a23_x2_sig,
      vdd => vdd,
      vss => vss
   );

oa2ao222_x2_ins : oa2ao222_x2
   port map (
      i0  => oa2a2a23_x2_sig,
      i1  => aux14,
      i2  => a2_x2_4_sig,
      i3  => no3_x1_sig,
      i4  => aux13,
      q   => oa2ao222_x2_sig,
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_4_ins : sff1_x4
   port map (
      ck  => clk,
      i   => oa2ao222_x2_sig,
      q   => statmachine_current_s(4),
      vdd => vdd,
      vss => vss
   );

no4_x1_ins : no4_x1
   port map (
      i0  => not_aux9,
      i1  => not_aux1,
      i2  => not_aux8,
      i3  => not_aux14,
      nq  => no4_x1_sig,
      vdd => vdd,
      vss => vss
   );

no3_x1_2_ins : no3_x1
   port map (
      i0  => not_aux9,
      i1  => not_aux10,
      i2  => input(1),
      nq  => no3_x1_2_sig,
```

```
      vdd => vdd,
      vss => vss
   );

oa22_x2_8_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(5),
      i1  => input(1),
      i2  => no3_x1_2_sig,
      q   => oa22_x2_8_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_7_ins : oa22_x2
   port map (
      i0  => oa22_x2_8_sig,
      i1  => aux13,
      i2  => no4_x1_sig,
      q   => oa22_x2_7_sig,
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_5_ins : sff1_x4
   port map (
      ck  => clk,
      i   => oa22_x2_7_sig,
      q   => statmachine_current_s(5),
      vdd => vdd,
      vss => vss
   );

no4_x1_2_ins : no4_x1
   port map (
      i0  => not_aux10,
      i1  => not_aux1,
      i2  => not_aux11,
      i3  => not_aux14,
      nq  => no4_x1_2_sig,
      vdd => vdd,
      vss => vss
   );

no3_x1_3_ins : no3_x1
   port map (
      i0  => not_aux12,
      i1  => not_aux11,
      i2  => input(1),
      nq  => no3_x1_3_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_10_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(6),
      i1  => input(1),
      i2  => no3_x1_3_sig,
      q   => oa22_x2_10_sig,
      vdd => vdd,
      vss => vss
   );
```

```
oa22_x2_9_ins : oa22_x2
   port map (
      i0  => oa22_x2_10_sig,
      i1  => aux13,
      i2  => no4_x1_2_sig,
      q   => oa22_x2_9_sig,
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_6_ins : sff1_x4
   port map (
      ck  => clk,
      i   => oa22_x2_9_sig,
      q   => statmachine_current_s(6),
      vdd => vdd,
      vss => vss
   );

no3_x1_4_ins : no3_x1
   port map (
      i0  => not_aux1,
      i1  => not_aux12,
      i2  => not_aux14,
      nq  => no3_x1_4_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_3_ins : na2_x1
   port map (
      i0  => input(2),
      i1  => not_statmachine_current_s(7),
      nq  => na2_x1_3_sig,
      vdd => vdd,
      vss => vss
   );

ao22_x2_2_ins : ao22_x2
   port map (
      i0  => input(2),
      i1  => input(1),
      i2  => aux13,
      q   => ao22_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

oa2a22_x2_ins : oa2a22_x2
   port map (
      i0  => ao22_x2_2_sig,
      i1  => statmachine_current_s(7),
      i2  => na2_x1_3_sig,
      i3  => no3_x1_4_sig,
      q   => oa2a22_x2_sig,
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_7_ins : sff1_x4
   port map (
      ck  => clk,
      i   => oa2a22_x2_sig,
      q   => statmachine_current_s(7),
```

```
      vdd => vdd,
      vss => vss
   );

inv_x2_2_ins : inv_x2
   port map (
      i   => rst,
      nq  => inv_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

ao22_x2_4_ins : ao22_x2
   port map (
      i0  => not_input(1),
      i1  => input(0),
      i2  => inv_x2_2_sig,
      q   => ao22_x2_4_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_4_ins : na2_x1
   port map (
      i0  => input(0),
      i1  => not_input(1),
      nq  => na2_x1_4_sig,
      vdd => vdd,
      vss => vss
   );

a3_x2_3_ins : a3_x2
   port map (
      i0  => na2_x1_4_sig,
      i1  => not_statmachine_current_s(3),
      i2  => ao22_x2_4_sig,
      q   => a3_x2_3_sig,
      vdd => vdd,
      vss => vss
   );

a2_x2_5_ins : a2_x2
   port map (
      i0  => not_statmachine_current_s(1),
      i1  => not_statmachine_current_s(0),
      q   => a2_x2_5_sig,
      vdd => vdd,
      vss => vss
   );

a2_x2_6_ins : a2_x2
   port map (
      i0  => not_statmachine_current_s(8),
      i1  => not_statmachine_current_s(2),
      q   => a2_x2_6_sig,
      vdd => vdd,
      vss => vss
   );

na3_x1_2_ins : na3_x1
   port map (
      i0  => a2_x2_6_sig,
      i1  => a2_x2_5_sig,
      i2  => a3_x2_3_sig,
```

```
      nq  => na3_x1_2_sig,
      vdd => vdd,
      vss => vss
   );

a2_x2_7_ins : a2_x2
   port map (
      i0  => input(2),
      i1  => statmachine_current_s(8),
      q   => a2_x2_7_sig,
      vdd => vdd,
      vss => vss
   );

ao22_x2_3_ins : ao22_x2
   port map (
      i0  => a2_x2_7_sig,
      i1  => aux16,
      i2  => na3_x1_2_sig,
      q   => ao22_x2_3_sig,
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_8_ins : sff1_x4
   port map (
      ck  => clk,
      i   => ao22_x2_3_sig,
      q   => statmachine_current_s(8),
      vdd => vdd,
      vss => vss
   );

o3_x2_ins : o3_x2
   port map (
      i0  => not_aux13,
      i1  => input(1),
      i2  => not_aux0,
      q   => o3_x2_sig,
      vdd => vdd,
      vss => vss
   );

o2_x2_2_ins : o2_x2
   port map (
      i0  => input(2),
      i1  => not_aux15,
      q   => o2_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

a2_x2_8_ins : a2_x2
   port map (
      i0  => not_statmachine_current_s(6),
      i1  => not_statmachine_current_s(4),
      q   => a2_x2_8_sig,
      vdd => vdd,
      vss => vss
   );

change_0_ins : nao22_x1
   port map (
      i0  => a2_x2_8_sig,
```

```
      i1  => o2_x2_2_sig,
      i2  => o3_x2_sig,
      nq  => change(0),
      vdd => vdd,
      vss => vss
   );

no2_x1_5_ins : no2_x1
   port map (
      i0  => input(2),
      i1  => not_aux15,
      nq  => no2_x1_5_sig,
      vdd => vdd,
      vss => vss
   );

change_1_ins : ao22_x2
   port map (
      i0  => mbk_buf_statmachine_current_s(4),
      i1  => statmachine_current_s(5),
      i2  => no2_x1_5_sig,
      q   => change(1),
      vdd => vdd,
      vss => vss
   );

output_0_ins : no3_x1
   port map (
      i0  => aux1,
      i1  => not_aux4,
      i2  => not_aux13,
      nq  => output(0),
      vdd => vdd,
      vss => vss
   );

inv_x2_3_ins : inv_x2
   port map (
      i   => aux5,
      nq  => inv_x2_3_sig,
      vdd => vdd,
      vss => vss
   );

output_1_ins : no3_x1
   port map (
      i0  => inv_x2_3_sig,
      i1  => not_aux14,
      i2  => not_aux4,
      nq  => output(1),
      vdd => vdd,
      vss => vss
   );

mbk_buf_statmachine_current_s_4 : buf_x2
   port map (
      i   => statmachine_current_s(4),
      q   => mbk_buf_statmachine_current_s(4),
      vdd => vdd,
      vss => vss
   );

mbk_buf_not_auxx7 : buf_x2
   port map (
```

```
      i   => not_aux7,
      q   => mbk_buf_not_aux7,
      vdd => vdd,
      vss => vss
   );


end structural;
```

testbenchl.vhd file

```
-- Entity declaration for your testbench. Don't declare any ports here
ENTITY vendingmachineox IS
END ENTITY vendingmachineox;

ARCHITECTURE testbench OF vendingmachineox IS

-- Component Declaration for the Device Under Test (DUT)
COMPONENT vendingmachineo_l IS
port (
      vdd     : in      bit;
      vss     : in      bit;
      clk     : in      bit;
      input   : in      bit_vector(2 downto 0);
      rst     : in      bit;
      output  : out     bit_vector(1 downto 0);
      change  : out     bit_vector(1 downto 0);
 );
END COMPONENT vendingmachineo_l;

FOR dut: vendingmachineo_l USE ENTITY WORK.vendingmachineo_l (structural);

-- Inputs
SIGNAL clk       : bit := '0';
SIGNAL rst  : bit := '1';
SIGNAL vdd       : bit := '1';
SIGNAL vss       : bit := '0';
SIGNAL input     : bit_vector(2 Downto 0) := "101";

-- Outputs
SIGNAL change    : bit_vector(1 Downto 0);
SIGNAL output    : bit_vector(1 Downto 0);

-- Constants and Clock period definitions
constant clk_period : time := 1000 ns;
BEGIN
      dut: vendingmachineo_l PORT MAP (vdd, vss, clk, input, rst, output,
change);
      clk_process :process
   begin
        clk <= '0';
        wait for clk_period/2;
        clk <= '1';
        wait for clk_period/2;
   end process;
   stim_proc: PROCESS IS
BEGIN
      WAIT FOR clk_period; --For the output to be stable
      ASSERT change = "00" and output = "00"
      REPORT "Reset error"
      SEVERITY error;
```

```
        rst <= '0';

        input <= "000";
        WAIT FOR clk_period;
        ASSERT change = "00" and output = "00"
        REPORT "Outputs error"
        SEVERITY error;

        input <= "000";
        WAIT FOR clk_period;
        ASSERT change = "00" and output = "00"
        REPORT "Outputs error"
        SEVERITY error;

        input <= "000";
        WAIT FOR clk_period;
        ASSERT change = "00" and output = "00"
        REPORT "Outputs error"
        SEVERITY error;

        input <= "000";
        WAIT FOR clk_period;
        ASSERT change = "00" and output = "00"
        REPORT "Outputs error"
        SEVERITY error;

        input <= "000";
        WAIT FOR clk_period;
        ASSERT change = "00" and output = "00"
        REPORT "Outputs error"
        SEVERITY error;

        input <= "011";
        WAIT FOR clk_period;
        ASSERT change = "00" and output = "01"
        REPORT "Outputs error"
        SEVERITY error;

        input <= "001";
        WAIT FOR clk_period;
        ASSERT change = "00" and output = "00"
        REPORT "Outputs error"
        SEVERITY error;

        input <= "010";
        WAIT FOR clk_period;
        ASSERT change = "00" and output = "00"
        REPORT "Outputs error"
        SEVERITY error;

        input <= "100";
        WAIT FOR clk_period;
        ASSERT change = "01" and output = "10"
        REPORT "Outputs error"
        SEVERITY error;
end process;
end;
```

vendingmachineo_s.vhd file

```
LIBRARY sxlib_ModelSim;
entity vendingmachineo_s is
```

```
   port (
      vdd     : in      bit;
      vss     : in      bit;
      clk     : in      bit;
      input   : in      bit_vector(2 downto 0);
      rst     : in      bit;
      output  : out     bit_vector(1 downto 0);
      change  : out     bit_vector(1 downto 0);
      scanin  : in      bit;
      test    : in      bit;
      scanout : out     bit
 );
end vendingmachineo_s;

architecture structural of vendingmachineo_s is
Component a2_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component inv_x2
   port (
      i   : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component noa2ao222_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      i4  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component a4_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component noa2a22_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
```

```
      i3  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component ao2o22_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component nao2o22_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component oa22_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component xr2_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component na3_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component no4_x1
```

```
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component a3_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component o2_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component noa22_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component na2_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component oa2a22_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;
```

```
Component no2_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component ao22_x2
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component no3_x1
   port (
      i0  : in      bit;
      i1  : in      bit;
      i2  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component no4_x4
   port (
      i1  : in      bit;
      i0  : in      bit;
      i2  : in      bit;
      i3  : in      bit;
      nq  : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component buf_x4
   port (
      i   : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

Component sff2_x4
   port (
      ck  : in      bit;
      cmd : in      bit;
      i0  : in      bit;
      i1  : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
```

```
end component;

Component buf_x2
   port (
      i   : in      bit;
      q   : out     bit;
      vdd : in      bit;
      vss : in      bit
 );
end component;

signal mbk_buf_not_statmachine_current_s : bit_vector( 2 downto 0);
signal not_input                         : bit_vector( 2 downto 0);
signal not_statmachine_current_s         : bit_vector( 8 downto 0);
signal statmachine_current_s             : bit_vector( 8 downto 0);
signal xr2_x1_sig                        : bit;
signal oa2a22_x2_sig                     : bit;
signal oa2a22_x2_2_sig                   : bit;
signal oa22_x2_sig                       : bit;
signal oa22_x2_8_sig                     : bit;
signal oa22_x2_7_sig                     : bit;
signal oa22_x2_6_sig                     : bit;
signal oa22_x2_5_sig                     : bit;
signal oa22_x2_4_sig                     : bit;
signal oa22_x2_3_sig                     : bit;
signal oa22_x2_2_sig                     : bit;
signal o2_x2_sig                         : bit;
signal o2_x2_7_sig                       : bit;
signal o2_x2_6_sig                       : bit;
signal o2_x2_5_sig                       : bit;
signal o2_x2_4_sig                       : bit;
signal o2_x2_3_sig                       : bit;
signal o2_x2_2_sig                       : bit;
signal not_rst                           : bit;
signal not_aux9                          : bit;
signal not_aux8                          : bit;
signal not_aux7                          : bit;
signal not_aux6                          : bit;
signal not_aux5                          : bit;
signal not_aux4                          : bit;
signal not_aux2                          : bit;
signal not_aux10                         : bit;
signal not_aux1                          : bit;
signal not_aux0                          : bit;
signal noa2ao222_x1_sig                  : bit;
signal noa2a22_x1_sig                    : bit;
signal noa22_x1_sig                      : bit;
signal no4_x1_sig                        : bit;
signal no3_x1_sig                        : bit;
signal no2_x1_sig                        : bit;
signal no2_x1_9_sig                      : bit;
signal no2_x1_8_sig                      : bit;
signal no2_x1_7_sig                      : bit;
signal no2_x1_6_sig                      : bit;
signal no2_x1_5_sig                      : bit;
signal no2_x1_4_sig                      : bit;
signal no2_x1_3_sig                      : bit;
signal no2_x1_2_sig                      : bit;
signal no2_x1_13_sig                     : bit;
signal no2_x1_12_sig                     : bit;
signal no2_x1_11_sig                     : bit;
signal no2_x1_10_sig                     : bit;
signal nao2o22_x1_sig                    : bit;
signal na3_x1_sig                        : bit;
```

```
signal na3_x1_7_sig                     : bit;
signal na3_x1_6_sig                     : bit;
signal na3_x1_5_sig                     : bit;
signal na3_x1_4_sig                     : bit;
signal na3_x1_3_sig                     : bit;
signal na3_x1_2_sig                     : bit;
signal na2_x1_sig                       : bit;
signal na2_x1_9_sig                     : bit;
signal na2_x1_8_sig                     : bit;
signal na2_x1_7_sig                     : bit;
signal na2_x1_6_sig                     : bit;
signal na2_x1_5_sig                     : bit;
signal na2_x1_4_sig                     : bit;
signal na2_x1_3_sig                     : bit;
signal na2_x1_2_sig                     : bit;
signal na2_x1_21_sig                    : bit;
signal na2_x1_20_sig                    : bit;
signal na2_x1_19_sig                    : bit;
signal na2_x1_18_sig                    : bit;
signal na2_x1_17_sig                    : bit;
signal na2_x1_16_sig                    : bit;
signal na2_x1_15_sig                    : bit;
signal na2_x1_14_sig                    : bit;
signal na2_x1_13_sig                    : bit;
signal na2_x1_12_sig                    : bit;
signal na2_x1_11_sig                    : bit;
signal na2_x1_10_sig                    : bit;
signal inv_x2_sig                       : bit;
signal inv_x2_2_sig                     : bit;
signal aux12                            : bit;
signal aux11                            : bit;
signal ao2o22_x2_sig                    : bit;
signal a4_x2_sig                        : bit;
signal a4_x2_3_sig                      : bit;
signal a4_x2_2_sig                      : bit;
signal a3_x2_sig                        : bit;

begin

inv_x2_ins : inv_x2
   port map (
      i   => statmachine_current_s(1),
      nq  => inv_x2_sig,
      vdd => vdd,
      vss => vss
   );

not_aux4_ins : a3_x2
   port map (
      i0  => inv_x2_sig,
      i1  => not_statmachine_current_s(3),
      i2  => not_aux2,
      q   => not_aux4,
      vdd => vdd,
      vss => vss
   );

not_aux2_ins : a2_x2
   port map (
      i0  => not_statmachine_current_s(0),
      i1  => not_statmachine_current_s(2),
      q   => not_aux2,
      vdd => vdd,
      vss => vss
```

```
   );

not_aux7_ins : o2_x2
   port map (
      i0  => rst,
      i1  => not_statmachine_current_s(7),
      q   => not_aux7,
      vdd => vdd,
      vss => vss
   );

not_aux9_ins : o2_x2
   port map (
      i0  => input(0),
      i1  => input(2),
      q   => not_aux9,
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_8_ins : inv_x2
   port map (
      i   => statmachine_current_s(8),
      nq  => not_statmachine_current_s(8),
      vdd => vdd,
      vss => vss
   );

not_aux8_ins : o2_x2
   port map (
      i0  => input(2),
      i1  => not_input(0),
      q   => not_aux8,
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_7_ins : inv_x2
   port map (
      i   => statmachine_current_s(7),
      nq  => not_statmachine_current_s(7),
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_3_ins : inv_x2
   port map (
      i   => statmachine_current_s(3),
      nq  => not_statmachine_current_s(3),
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_6_ins : inv_x2
   port map (
      i   => statmachine_current_s(6),
      nq  => not_statmachine_current_s(6),
      vdd => vdd,
      vss => vss
   );

not_aux1_ins : o2_x2
   port map (
      i0  => input(1),
```

```
      i1  => rst,
      q   => not_aux1,
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_2_ins : inv_x2
   port map (
      i   => statmachine_current_s(2),
      nq  => not_statmachine_current_s(2),
      vdd => vdd,
      vss => vss
   );

not_aux6_ins : o2_x2
   port map (
      i0  => input(1),
      i1  => not_input(2),
      q   => not_aux6,
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_5_ins : inv_x2
   port map (
      i   => statmachine_current_s(5),
      nq  => not_statmachine_current_s(5),
      vdd => vdd,
      vss => vss
   );

not_aux0_ins : o2_x2
   port map (
      i0  => rst,
      i1  => not_input(1),
      q   => not_aux0,
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_4_ins : inv_x2
   port map (
      i   => statmachine_current_s(4),
      nq  => not_statmachine_current_s(4),
      vdd => vdd,
      vss => vss
   );

not_aux10_ins : o2_x2
   port map (
      i0  => rst,
      i1  => not_input(0),
      q   => not_aux10,
      vdd => vdd,
      vss => vss
   );

not_statmachine_current_s_0_ins : inv_x2
   port map (
      i   => statmachine_current_s(0),
      nq  => not_statmachine_current_s(0),
      vdd => vdd,
      vss => vss
   );
```

```
not_aux5_ins : a2_x2
   port map (
      i0  => input(1),
      i1  => not_input(2),
      q   => not_aux5,
      vdd => vdd,
      vss => vss
   );

not_input_2_ins : inv_x2
   port map (
      i   => input(2),
      nq  => not_input(2),
      vdd => vdd,
      vss => vss
   );

not_input_1_ins : inv_x2
   port map (
      i   => input(1),
      nq  => not_input(1),
      vdd => vdd,
      vss => vss
   );

not_input_0_ins : inv_x2
   port map (
      i   => input(0),
      nq  => not_input(0),
      vdd => vdd,
      vss => vss
   );

not_rst_ins : inv_x2
   port map (
      i   => rst,
      nq  => not_rst,
      vdd => vdd,
      vss => vss
   );

aux12_ins : no2_x1
   port map (
      i0  => rst,
      i1  => not_aux8,
      nq  => aux12,
      vdd => vdd,
      vss => vss
   );

aux11_ins : no2_x1
   port map (
      i0  => input(0),
      i1  => rst,
      nq  => aux11,
      vdd => vdd,
      vss => vss
   );

na2_x1_ins : na2_x1
   port map (
      i0  => mbk_buf_not_statmachine_current_s(0),
      i1  => not_statmachine_current_s(4),
```

```
      nq  => na2_x1_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_2_ins : na2_x1
   port map (
      i0  => not_input(1),
      i1  => mbk_buf_not_statmachine_current_s(0),
      nq  => na2_x1_2_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_2_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(0),
      i1  => input(1),
      i2  => not_input(2),
      q   => oa22_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

a4_x2_ins : a4_x2
   port map (
      i0  => oa22_x2_2_sig,
      i1  => aux11,
      i2  => na2_x1_2_sig,
      i3  => na2_x1_sig,
      q   => a4_x2_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_ins : no2_x1
   port map (
      i0  => not_aux10,
      i1  => not_aux5,
      nq  => no2_x1_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(0),
      i1  => no2_x1_sig,
      i2  => a4_x2_sig,
      q   => oa22_x2_sig,
      vdd => vdd,
      vss => vss
   );

inv_x2_2_ins : inv_x2
   port map (
      i   => statmachine_current_s(1),
      nq  => inv_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

o2_x2_ins : o2_x2
   port map (
```

```
      i0  => input(2),
      i1  => not_aux0,
      q   => o2_x2_sig,
      vdd => vdd,
      vss => vss
   );

o2_x2_2_ins : o2_x2
   port map (
      i0  => not_aux10,
      i1  => not_aux5,
      q   => o2_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_3_ins : na2_x1
   port map (
      i0  => not_aux6,
      i1  => aux11,
      nq  => na2_x1_3_sig,
      vdd => vdd,
      vss => vss
   );

noa2ao222_x1_ins : noa2ao222_x1
   port map (
      i0  => na2_x1_3_sig,
      i1  => o2_x2_2_sig,
      i2  => not_statmachine_current_s(5),
      i3  => o2_x2_sig,
      i4  => inv_x2_2_sig,
      nq  => noa2ao222_x1_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_4_ins : na2_x1
   port map (
      i0  => mbk_buf_not_statmachine_current_s(2),
      i1  => not_statmachine_current_s(6),
      nq  => na2_x1_4_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_5_ins : na2_x1
   port map (
      i0  => not_input(1),
      i1  => mbk_buf_not_statmachine_current_s(2),
      nq  => na2_x1_5_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_4_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(2),
      i1  => input(1),
      i2  => not_input(2),
      q   => oa22_x2_4_sig,
      vdd => vdd,
      vss => vss
   );
```

```
a4_x2_2_ins : a4_x2
   port map (
      i0  => oa22_x2_4_sig,
      i1  => aux11,
      i2  => na2_x1_5_sig,
      i3  => na2_x1_4_sig,
      q   => a4_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_2_ins : no2_x1
   port map (
      i0  => rst,
      i1  => not_input(2),
      nq  => no2_x1_2_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_6_ins : na2_x1
   port map (
      i0  => mbk_buf_not_statmachine_current_s(2),
      i1  => not_statmachine_current_s(4),
      nq  => na2_x1_6_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_3_ins : no2_x1
   port map (
      i0  => input(2),
      i1  => not_aux1,
      nq  => no2_x1_3_sig,
      vdd => vdd,
      vss => vss
   );

oa2a22_x2_ins : oa2a22_x2
   port map (
      i0  => no2_x1_3_sig,
      i1  => na2_x1_6_sig,
      i2  => statmachine_current_s(2),
      i3  => no2_x1_2_sig,
      q   => oa2a22_x2_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_3_ins : oa22_x2
   port map (
      i0  => oa2a22_x2_sig,
      i1  => input(0),
      i2  => a4_x2_2_sig,
      q   => oa22_x2_3_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_6_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(3),
      i1  => input(1),
```

```
      i2  => not_input(2),
      q   => oa22_x2_6_sig,
      vdd => vdd,
      vss => vss
   );

na3_x1_ins : na3_x1
   port map (
      i0  => not_input(1),
      i1  => not_statmachine_current_s(3),
      i2  => not_statmachine_current_s(4),
      nq  => na3_x1_sig,
      vdd => vdd,
      vss => vss
   );

o2_x2_3_ins : o2_x2
   port map (
      i0  => statmachine_current_s(7),
      i1  => statmachine_current_s(3),
      q   => o2_x2_3_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_7_ins : oa22_x2
   port map (
      i0  => statmachine_current_s(4),
      i1  => not_input(1),
      i2  => o2_x2_3_sig,
      q   => oa22_x2_7_sig,
      vdd => vdd,
      vss => vss
   );

a4_x2_3_ins : a4_x2
   port map (
      i0  => oa22_x2_7_sig,
      i1  => aux11,
      i2  => na3_x1_sig,
      i3  => oa22_x2_6_sig,
      q   => a4_x2_3_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_4_ins : no2_x1
   port map (
      i0  => rst,
      i1  => not_input(2),
      nq  => no2_x1_4_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_7_ins : na2_x1
   port map (
      i0  => not_statmachine_current_s(5),
      i1  => not_statmachine_current_s(3),
      nq  => na2_x1_7_sig,
      vdd => vdd,
      vss => vss
   );
```

```
no2_x1_5_ins : no2_x1
   port map (
      i0  => input(2),
      i1  => not_aux1,
      nq  => no2_x1_5_sig,
      vdd => vdd,
      vss => vss
   );

oa2a22_x2_2_ins : oa2a22_x2
   port map (
      i0  => no2_x1_5_sig,
      i1  => na2_x1_7_sig,
      i2  => statmachine_current_s(3),
      i3  => no2_x1_4_sig,
      q   => oa2a22_x2_2_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_5_ins : oa22_x2
   port map (
      i0  => oa2a22_x2_2_sig,
      i1  => input(0),
      i2  => a4_x2_3_sig,
      q   => oa22_x2_5_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_8_ins : na2_x1
   port map (
      i0  => input(1),
      i1  => not_statmachine_current_s(4),
      nq  => na2_x1_8_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_9_ins : na2_x1
   port map (
      i0  => not_aux0,
      i1  => not_statmachine_current_s(6),
      nq  => na2_x1_9_sig,
      vdd => vdd,
      vss => vss
   );

na3_x1_3_ins : na3_x1
   port map (
      i0  => aux12,
      i1  => na2_x1_9_sig,
      i2  => na2_x1_8_sig,
      nq  => na3_x1_3_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_10_ins : na2_x1
   port map (
      i0  => not_input(1),
      i1  => not_statmachine_current_s(5),
      nq  => na2_x1_10_sig,
      vdd => vdd,
```

```
      vss => vss
   );

na2_x1_11_ins : na2_x1
   port map (
      i0  => not_aux1,
      i1  => not_statmachine_current_s(8),
      nq  => na2_x1_11_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_6_ins : no2_x1
   port map (
      i0  => rst,
      i1  => not_aux9,
      nq  => no2_x1_6_sig,
      vdd => vdd,
      vss => vss
   );

na3_x1_4_ins : na3_x1
   port map (
      i0  => no2_x1_6_sig,
      i1  => na2_x1_11_sig,
      i2  => na2_x1_10_sig,
      nq  => na3_x1_4_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_7_ins : no2_x1
   port map (
      i0  => rst,
      i1  => not_input(2),
      nq  => no2_x1_7_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_12_ins : na2_x1
   port map (
      i0  => no2_x1_7_sig,
      i1  => statmachine_current_s(4),
      nq  => na2_x1_12_sig,
      vdd => vdd,
      vss => vss
   );

na3_x1_2_ins : na3_x1
   port map (
      i0  => na2_x1_12_sig,
      i1  => na3_x1_4_sig,
      i2  => na3_x1_3_sig,
      nq  => na3_x1_2_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_14_ins : na2_x1
   port map (
      i0  => input(1),
      i1  => not_statmachine_current_s(5),
      nq  => na2_x1_14_sig,
```

```
      vdd => vdd,
      vss => vss
   );

na2_x1_15_ins : na2_x1
   port map (
      i0  => not_aux0,
      i1  => not_statmachine_current_s(7),
      nq  => na2_x1_15_sig,
      vdd => vdd,
      vss => vss
   );

na3_x1_5_ins : na3_x1
   port map (
      i0  => aux12,
      i1  => na2_x1_15_sig,
      i2  => na2_x1_14_sig,
      nq  => na3_x1_5_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_8_ins : no2_x1
   port map (
      i0  => rst,
      i1  => not_input(2),
      nq  => no2_x1_8_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_9_ins : no2_x1
   port map (
      i0  => not_aux9,
      i1  => not_aux1,
      nq  => no2_x1_9_sig,
      vdd => vdd,
      vss => vss
   );

noa2a22_x1_ins : noa2a22_x1
   port map (
      i0  => statmachine_current_s(6),
      i1  => no2_x1_9_sig,
      i2  => statmachine_current_s(5),
      i3  => no2_x1_8_sig,
      nq  => noa2a22_x1_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_13_ins : na2_x1
   port map (
      i0  => noa2a22_x1_sig,
      i1  => na3_x1_5_sig,
      nq  => na2_x1_13_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_17_ins : na2_x1
   port map (
      i0  => input(1),
```

```
      i1  => not_statmachine_current_s(6),
      nq  => na2_x1_17_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_18_ins : na2_x1
   port map (
      i0  => not_aux0,
      i1  => not_statmachine_current_s(8),
      nq  => na2_x1_18_sig,
      vdd => vdd,
      vss => vss
   );

na3_x1_6_ins : na3_x1
   port map (
      i0  => aux12,
      i1  => na2_x1_18_sig,
      i2  => na2_x1_17_sig,
      nq  => na3_x1_6_sig,
      vdd => vdd,
      vss => vss
   );

o2_x2_4_ins : o2_x2
   port map (
      i0  => not_aux9,
      i1  => not_aux1,
      q   => o2_x2_4_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_19_ins : na2_x1
   port map (
      i0  => input(2),
      i1  => not_rst,
      nq  => na2_x1_19_sig,
      vdd => vdd,
      vss => vss
   );

ao2o22_x2_ins : ao2o22_x2
   port map (
      i0  => na2_x1_19_sig,
      i1  => not_statmachine_current_s(6),
      i2  => not_statmachine_current_s(7),
      i3  => o2_x2_4_sig,
      q   => ao2o22_x2_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_16_ins : na2_x1
   port map (
      i0  => ao2o22_x2_sig,
      i1  => na3_x1_6_sig,
      nq  => na2_x1_16_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_10_ins : no2_x1
```

```
   port map (
      i0  => input(1),
      i1  => input(2),
      nq  => no2_x1_10_sig,
      vdd => vdd,
      vss => vss
   );

no3_x1_ins : no3_x1
   port map (
      i0  => no2_x1_10_sig,
      i1  => not_aux7,
      i2  => not_input(0),
      nq  => no3_x1_sig,
      vdd => vdd,
      vss => vss
   );

o2_x2_5_ins : o2_x2
   port map (
      i0  => input(2),
      i1  => not_aux1,
      q   => o2_x2_5_sig,
      vdd => vdd,
      vss => vss
   );

nao2o22_x1_ins : nao2o22_x1
   port map (
      i0  => o2_x2_5_sig,
      i1  => not_statmachine_current_s(8),
      i2  => not_aux7,
      i3  => not_input(2),
      nq  => nao2o22_x1_sig,
      vdd => vdd,
      vss => vss
   );

oa22_x2_8_ins : oa22_x2
   port map (
      i0  => nao2o22_x1_sig,
      i1  => not_input(0),
      i2  => no3_x1_sig,
      q   => oa22_x2_8_sig,
      vdd => vdd,
      vss => vss
   );

xr2_x1_ins : xr2_x1
   port map (
      i0  => input(0),
      i1  => input(2),
      q   => xr2_x1_sig,
      vdd => vdd,
      vss => vss
   );

na3_x1_7_ins : na3_x1
   port map (
      i0  => not_rst,
      i1  => not_aux2,
      i2  => xr2_x1_sig,
      nq  => na3_x1_7_sig,
      vdd => vdd,
```

```
      vss => vss
   );

no4_x1_ins : no4_x1
   port map (
      i0  => statmachine_current_s(8),
      i1  => statmachine_current_s(1),
      i2  => na3_x1_7_sig,
      i3  => statmachine_current_s(3),
      nq  => no4_x1_sig,
      vdd => vdd,
      vss => vss
   );

o2_x2_6_ins : o2_x2
   port map (
      i0  => input(0),
      i1  => not_aux6,
      q   => o2_x2_6_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_20_ins : na2_x1
   port map (
      i0  => input(0),
      i1  => not_aux5,
      nq  => na2_x1_20_sig,
      vdd => vdd,
      vss => vss
   );

a3_x2_ins : a3_x2
   port map (
      i0  => not_rst,
      i1  => na2_x1_20_sig,
      i2  => o2_x2_6_sig,
      q   => a3_x2_sig,
      vdd => vdd,
      vss => vss
   );

o2_x2_7_ins : o2_x2
   port map (
      i0  => not_input(2),
      i1  => not_statmachine_current_s(8),
      q   => o2_x2_7_sig,
      vdd => vdd,
      vss => vss
   );

noa22_x1_ins : noa22_x1
   port map (
      i0  => o2_x2_7_sig,
      i1  => a3_x2_sig,
      i2  => no4_x1_sig,
      nq  => noa22_x1_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_11_ins : no2_x1
   port map (
      i0  => not_aux8,
```

```
      i1  => not_aux1,
      nq  => no2_x1_11_sig,
      vdd => vdd,
      vss => vss
   );

na2_x1_21_ins : na2_x1
   port map (
      i0  => not_statmachine_current_s(6),
      i1  => not_statmachine_current_s(4),
      nq  => na2_x1_21_sig,
      vdd => vdd,
      vss => vss
   );

no2_x1_12_ins : no2_x1
   port map (
      i0  => not_aux9,
      i1  => not_aux0,
      nq  => no2_x1_12_sig,
      vdd => vdd,
      vss => vss
   );

change_0_ins : oa2a22_x2
   port map (
      i0  => no2_x1_12_sig,
      i1  => na2_x1_21_sig,
      i2  => statmachine_current_s(4),
      i3  => no2_x1_11_sig,
      q   => change(0),
      vdd => vdd,
      vss => vss
   );

no2_x1_13_ins : no2_x1
   port map (
      i0  => not_aux9,
      i1  => not_aux0,
      nq  => no2_x1_13_sig,
      vdd => vdd,
      vss => vss
   );

change_1_ins : ao22_x2
   port map (
      i0  => statmachine_current_s(4),
      i1  => statmachine_current_s(5),
      i2  => no2_x1_13_sig,
      q   => change(1),
      vdd => vdd,
      vss => vss
   );

output_0_ins : no3_x1
   port map (
      i0  => not_aux0,
      i1  => not_aux8,
      i2  => not_aux4,
      nq  => output(0),
      vdd => vdd,
      vss => vss
   );
```

```
output_1_ins : no4_x4
   port map (
      i1  => not_aux1,
      i0  => not_input(2),
      i2  => not_aux4,
      i3  => input(0),
      nq  => output(1),
      vdd => vdd,
      vss => vss
   );

dmbk_buf_not_statmachine_current_s_2 : buf_x4
   port map (
      i   => not_statmachine_current_s(2),
      q   => mbk_buf_not_statmachine_current_s(2),
      vdd => vdd,
      vss => vss
   );

mbk_buf_not_statmachine_current_s_0 : buf_x2
   port map (
      i   => not_statmachine_current_s(0),
      q   => mbk_buf_not_statmachine_current_s(0),
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_0_ins_scan_0 : sff2_x4
   port map (
      ck  => clk,
      cmd => test,
      i0  => oa22_x2_sig,
      i1  => scanin,
      q   => statmachine_current_s(0),
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_1_ins_scan_1 : sff2_x4
   port map (
      ck  => clk,
      cmd => test,
      i0  => noa2ao222_x1_sig,
      i1  => statmachine_current_s(0),
      q   => statmachine_current_s(1),
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_2_ins_scan_2 : sff2_x4
   port map (
      ck  => clk,
      cmd => test,
      i0  => oa22_x2_3_sig,
      i1  => statmachine_current_s(1),
      q   => statmachine_current_s(2),
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_3_ins_scan_3 : sff2_x4
   port map (
      ck  => clk,
      cmd => test,
```

```
      i0  => oa22_x2_5_sig,
      i1  => statmachine_current_s(2),
      q   => statmachine_current_s(3),
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_4_ins_scan_4 : sff2_x4
   port map (
      ck  => clk,
      cmd => test,
      i0  => na3_x1_2_sig,
      i1  => statmachine_current_s(3),
      q   => statmachine_current_s(4),
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_5_ins_scan_5 : sff2_x4
   port map (
      ck  => clk,
      cmd => test,
      i0  => na2_x1_13_sig,
      i1  => statmachine_current_s(4),
      q   => statmachine_current_s(5),
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_6_ins_scan_6 : sff2_x4
   port map (
      ck  => clk,
      cmd => test,
      i0  => na2_x1_16_sig,
      i1  => statmachine_current_s(5),
      q   => statmachine_current_s(6),
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_7_ins_scan_7 : sff2_x4
   port map (
      ck  => clk,
      cmd => test,
      i0  => oa22_x2_8_sig,
      i1  => statmachine_current_s(6),
      q   => statmachine_current_s(7),
      vdd => vdd,
      vss => vss
   );

statmachine_current_s_8_ins_scan_8 : sff2_x4
   port map (
      ck  => clk,
      cmd => test,
      i0  => noa22_x1_sig,
      i1  => statmachine_current_s(7),
      q   => statmachine_current_s(8),
      vdd => vdd,
      vss => vss
   );

buf_scan_9 : buf_x2
   port map (
```

```
      i   => statmachine_current_s(8),
      q   => scanout,
      vdd => vdd,
      vss => vss
   );


end structural;
```

testbenchs.vhd file

```
-- Entity declaration for your testbench. Don't declare any ports here
ENTITY vendingmachineos IS
END ENTITY vendingmachineos;

ARCHITECTURE testbench OF vendingmachineos IS

-- Component Declaration for the Device Under Test (DUT)
COMPONENT vendingmachineo_s IS
port (
      vdd     : in      bit;
      vss     : in      bit;
      clk     : in      bit;
      input   : in      bit_vector(2 downto 0);
      rst     : in      bit;
      output  : out     bit_vector(1 downto 0);
      change  : out     bit_vector(1 downto 0);
      scanin  : in      bit;
      test    : in      bit;
      scanout : out     bit
 );
END COMPONENT vendingmachineo_s;

FOR dut: vendingmachineo_s USE ENTITY WORK.vendingmachineo_s (structural);

-- Inputs
SIGNAL clk       : bit := '0';
SIGNAL rst  : bit := '1';
SIGNAL vdd       : bit := '1';
SIGNAL vss       : bit := '0';
SIGNAL input     : bit_vector(2 Downto 0) := "101";
SIGNAL scanin    : bit := '0';
SIGNAL test      : bit := '0';
SIGNAL scanout  : bit := '0';

-- Outputs
SIGNAL change    : bit_vector(1 Downto 0);
SIGNAL output    : bit_vector(1 Downto 0);

-- Constants and Clock period definitions
constant clk_period : time := 1000 ns;
constant sequence: bit_vector := "001111101101111001001111100";

BEGIN
      dut: vendingmachineo_s PORT MAP (vdd, vss, clk, input, rst, output, change,
scanin, test, scanout);
      clk_process :process
   begin
        clk <= '0';
        wait for clk_period/2;
        clk <= '1';
        wait for clk_period/2;
```

```
   end process;
   stim_proc: PROCESS IS
BEGIN
      test <= '1';
      for i In 0 to sequence'length-1 loop
      wait for clk_period; -- Leave time for the output to stabilize
      if i>=4 then -- Assert condition
      Assert scanout=sequence(i-4)
      Report "scanout does not follow scan in"
      Severity error;
      end if;
      scanin <= sequence(i); -- scanin changes on the next wait
      end loop;
      test <= '0';

      WAIT FOR clk_period; --For the output to be stable
      ASSERT change = "00" and output = "00"
      REPORT "Reset error"
      SEVERITY error;

      rst <= '0';

      input <= "000";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "000";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "000";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "000";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "000";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "011";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "01"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "001";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;
```

```
      input <= "010";
      WAIT FOR clk_period;
      ASSERT change = "00" and output = "00"
      REPORT "Outputs error"
      SEVERITY error;

      input <= "100";
      WAIT FOR clk_period;
      ASSERT change = "01" and output = "10"
      REPORT "Outputs error"
      SEVERITY error;
end process;
end;
```

makefile

```
#To run this makefile please type the "make" command#

all:  vendingmachinea.vbe \
      vendingmachinej.vbe \
      vendingmachinem.vbe \
      vendingmachineo.vbe \
      vendingmachiner.vbe \
      syf \
      vendingmachinea_o.vbe \
      vendingmachinej_o.vbe \
      vendingmachinem_o.vbe \
      vendingmachiner_o.vbe \
      vendingmachineo_o.vbe \
      boom \
      vendingmachinea_o.vst \
      vendingmachinej_o.vst \
      vendingmachinem_o.vst \
      vendingmachiner_o.vst \
      vendingmachineo_o.vst \
      boog \
      vendingmachinea_l.xsc \
      vendingmachinej_l.xsc \
      vendingmachinem_l.xsc \
      vendingmachiner_l.xsc \
      vendingmachineo_l.xsc \
      loon \
      vendingmachineo_labs.vbe \
      flatbeh \
      proof \
      scapin \
      ocp \
      nero \
      cougar \
      s2r \
      clean \
      credits
      @echo "-- Done, All Good --"

#----------------- [SYF] -----------------#

vendingmachinea.vbe: vendingmachine.fsm
      @echo "    Encoding Synthesis -> vendingmachinea.vbe"
      syf -CEV -a vendingmachine

vendingmachinej.vbe: vendingmachine.fsm
      @echo "    Encoding Synthesis  -> vendingmachinej.vbe"
      syf -CEV -j vendingmachine
```

```
vendingmachinem.vbe: vendingmachine.fsm
	@echo "    Encoding Synthesis  -> vendingmachinem.vbe"
	syf -CEV -m vendingmachine

vendingmachineo.vbe: vendingmachine.fsm
	@echo "    Encoding Synthesis  -> vendingmachineo.vbe"
	syf -CEV -o vendingmachine

vendingmachiner.vbe: vendingmachine.fsm
	@echo "    Encoding Synthesis  -> vendingmachiner.vbe"
	syf -CEV -r vendingmachine

syf:
	mkdir ./SYF
	cp *.vbe ./SYF
	cp *.enc ./SYF

#----------------- [BOOM] -----------------#

vendingmachinea_o.vbe: vendingmachinea.vbe
	@echo "[BOOM] Boolean optimizing -a -> $@"
	boom -V -d 50 vendingmachinea.vbe vendingmachinea_o.vbe >
vendingmachinea.out

vendingmachinej_o.vbe: vendingmachinej.vbe
	@echo "[BOOM] Boolean optimizing -j -> $@"
	boom -V -d 50 vendingmachinej.vbe vendingmachinej_o.vbe >
vendingmachinej.out

vendingmachinem_o.vbe: vendingmachinem.vbe
	@echo "[BOOM] Boolean optimizing -m -> $@"
	boom -V -d 50 vendingmachinem.vbe vendingmachinem_o.vbe >
vendingmachinem.out

vendingmachineo_o.vbe: vendingmachineo.vbe
	@echo "[BOOM] Boolean optimizing -o -> $@"
	boom -V -d 50 vendingmachineo.vbe vendingmachineo_o.vbe >
vendingmachineo.out

vendingmachiner_o.vbe: vendingmachiner.vbe
	@echo "[BOOM] Boolean optimizing -r -> $@"
	boom -V -d 50 vendingmachiner.vbe vendingmachiner_o.vbe >
vendingmachiner.out

boom:
	mkdir ./BOOM
	cp *_o.vbe ./BOOM
	mv *.out ./BOOM

#----------------- [BOOG] -----------------#

vendingmachinea_o.vst : vendingmachinea_o.vbe
	@echo "[BOOG] Library Mapping -a -> $@ "
	boog -l paramfile vendingmachinea_o

vendingmachinej_o.vst : vendingmachinej_o.vbe
	@echo "[BOOG] Library Mapping -j -> $@ "
	boog -l paramfile vendingmachinej_o

vendingmachinem_o.vst : vendingmachinem_o.vbe
	@echo "[BOOG] Library Mapping -m -> $@ "
	boog -l paramfile vendingmachinem_o

vendingmachineo_o.vst : vendingmachineo_o.vbe
```

```
	@echo "[BOOG] Library Mapping -o -> $@ "
	boog -l paramfile vendingmachineo_o

vendingmachiner_o.vst : vendingmachiner_o.vbe
	@echo "[BOOG] Library Mapping -r -> $@ "
	boog -l paramfile vendingmachiner_o

boog:
	mkdir ./BOOG
	cp *.vst ./BOOG
	cp *.xsc ./BOOG

#----------------- [LOON] -----------------#

vendingmachinea_l.xsc : vendingmachinea_o.vst
	@echo "[LOON] Netlist optimizing -a -> $@ "
	loon vendingmachinea_o vendingmachinea_l paramfile > vendingmachinea_l.out

vendingmachinej_l.xsc : vendingmachinej_o.vst
	@echo "[LOON] Netlist optimizing -j -> $@ "
	loon vendingmachinej_o vendingmachinej_l paramfile > vendingmachinej_l.out

vendingmachinem_l.xsc : vendingmachinem_o.vst
	@echo "[LOON] Netlist optimizing -m -> $@ "
	loon vendingmachinem_o vendingmachinem_l paramfile > vendingmachinem_l.out

vendingmachineo_l.xsc : vendingmachineo_o.vst
	@echo "[LOON] Netlist optimizing -o -> $@ "
	loon vendingmachineo_o vendingmachineo_l paramfile > vendingmachineo_l.out

vendingmachiner_l.xsc : vendingmachiner_o.vst
	@echo "[LOON] Netlist optimizing -r -> $@ "
	loon vendingmachiner_o vendingmachiner_l paramfile > vendingmachiner_l.out

loon:
	mkdir ./LOON
	cp *_l.vst ./LOON
	cp *_l.xsc ./LOON
	mv *.out ./LOON

#----------------- [FLATBEH] -----------------#

vendingmachineo_labs.vbe : ./LOON/vendingmachineo_l.vst
	@echo "[FLATBEH] Netlist checking -o -> $@ "
	flatbeh vendingmachineo_l

flatbeh:
	mkdir ./FLATBEH
	cp *_labs.vbe ./FLATBEH

#----------------- [PROOF] -----------------#

proof :
	@echo "[PROOF] Netlist checking -o -> $@ "
	proof -d ./SYF/vendingmachineo ./FLATBEH/vendingmachineo_labs

#----------------- [SCAPIN] -----------------#

scapin:    ./LOON/vendingmachineo_l.vst
	@echo "[SCAPIN] Scan-path insertion (DFT) $@ "
	scapin -VRB -P sxlib.scapin vendingmachineo_l pathfile vendingmachineo_s
	mkdir ./SCAPIN
	cp *_s.vst ./SCAPIN
```

```
#----------------- [OCP] -----------------#

ocp:  ./SCAPIN/vendingmachineo_s.vst
      @echo "[OCP] Placement $@ "
      alliance-ocp -v -ring -ioc pinorder vendingmachineo_s vendingmachineo_ocp >
vendingmachineo_ocp.out
      mkdir ./OCP
      cp *.ap ./OCP
      mv *.out ./OCP

#----------------- [NERO] -----------------#

nero: ./OCP/vendingmachineo_ocp.ap ./SCAPIN/vendingmachineo_s.vst
      @echo "[NERO] Routing $@ "
      nero -V -p vendingmachineo_ocp vendingmachineo_s vendingmachineo_s >
vendingmachineo_nero.out
      mkdir ./NERO
      cp *_s.ap ./NERO
      mv *_nero.out ./NERO

#----------------- [COUGAR] -----------------#

cougar:     ./NERO/vendingmachineo_s.ap ./SCAPIN/vendingmachineo_s.vst
      @echo "[COUGAR] Netlist extraction $@ "
      MBK_OUT_LO=al; export MBK_OUT_LO; \
      RDS_TECHNO_NAME=./techno/techno-035.rds; \
      export RDS_TECHNO_NAME; \
      cougar -v vendingmachineo_s vendingmachineo_cougar >
vendingmachineo_cougar.out
      @echo "[LVX] Netlist comparison $@ "
      lvx vst al vendingmachineo_s vendingmachineo_cougar -f >
vendingmachineo_lvx.out
      mkdir ./COUGAR
      mv *.al ./COUGAR
      mv *_cougar.out ./COUGAR
      mkdir ./LVX
      mv *_lvx.out ./LVX

#----------------- [S2R] -----------------#
s2r:  ./NERO/vendingmachineo_s.ap
      @echo "[S2R] Symbolic to real conversion $@ "
      s2r -v -r vendingmachineo_s > vendingmachineo_s2r.out
      mkdir ./S2R
      mv *_s2r.out ./S2R
      mv *.cif ./S2R

#----------------- [CREDITS] -----------------#
credits:
      @echo ""
      @echo "[CREDITS]"
      @echo "Mohamed El Ghamry"
      @echo "Faculty of Engineering Student"
      @echo "Ain Shams University"
      @echo "Computer Engineering and Software Systems Department"
      @echo ""

#----------------- [CLEAN] -----------------#

clean :
      rm -f  *.vbe *.enc *.vst *.xsc *.ap *.out *.al *.cif *~
      @echo "Erase all the unnecessary files generated by the makefile"

delete:
      rm -rf SYF
```

```
	rm -rf BOOM
	rm -rf BOOG
	rm -rf LOON
	rm -rf FLATBEH
	rm -rf SCAPIN
	rm -rf OCP
	rm -rf NERO
	rm -rf COUGAR
	rm -rf LVX
	rm -rf S2R
	@echo "Erase all the folders generated by the makefile"
```

paramfile

```
#M{2}
#L{2}
#C{
output(0): 100;
output(1): 100;
change(0): 100;
change(1): 100;
}
```

pathfile

```
BEGIN_PATH_REG
statmachine_current_s_0_ins
statmachine_current_s_1_ins
statmachine_current_s_2_ins
statmachine_current_s_3_ins
statmachine_current_s_4_ins
statmachine_current_s_5_ins
statmachine_current_s_6_ins
statmachine_current_s_7_ins
statmachine_current_s_8_ins
END_PATH_REG

BEGIN_CONNECTOR
SCAN_IN scanin
SCAN_OUT scanout
SCAN_TEST test
END_CONNECTOR
```

pinorder iocfile

```
LEFT (
(IOPIN input(0).0 );
(IOPIN input(1).0 );
(IOPIN input(2).0 );)
TOP (
(IOPIN scanin.0 );
(IOPIN test.0 );
(IOPIN clk.0 );)
RIGHT (
(IOPIN change(0).0 );
(IOPIN change(1).0 );
(IOPIN rst.0 );)
BOTTOM (
(IOPIN scanout.0 );
(IOPIN output(0).0 );
```

```
(IOPIN output(1).0 );)
```

Please note that all source codes and output files are included in the project folder/GitHub repository.