ABSTRACT

The design is based on a finite state machine (mealy machine) that responds to user input and dispenses the product once the correct amount of money has been inserted and returns change if any.

Deepak Raj (SC21B088)
Neelanshu Panigrahi (SC21B131)
Kanishk Sunda (SC21B133)

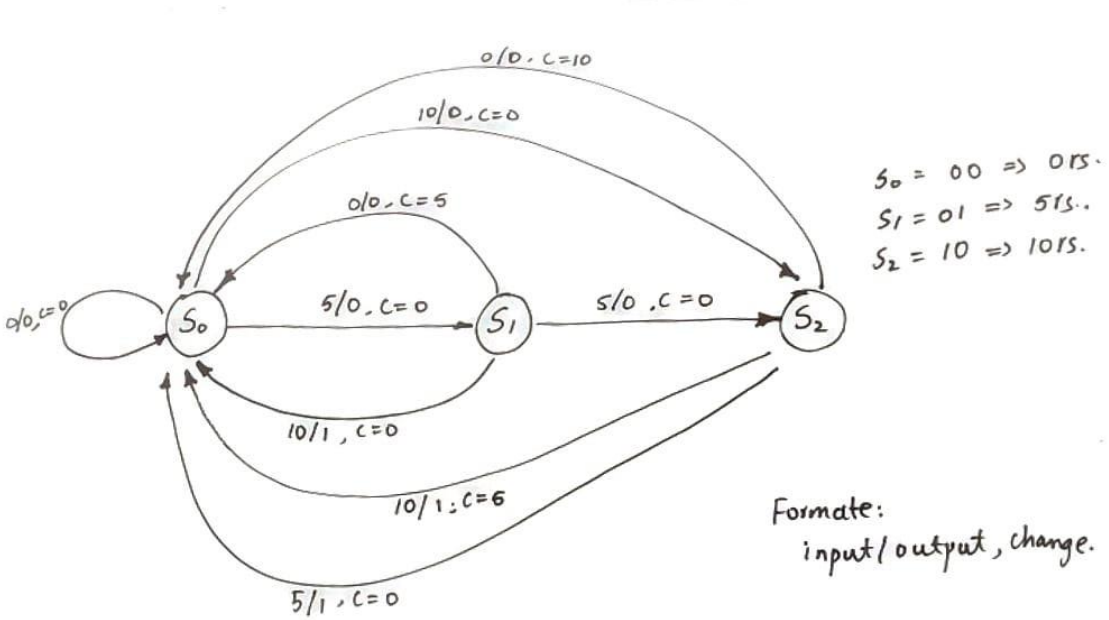
VLSI PROJECT:
FPGA-BASED VENDING MACHINE

A vending machine is a type of automated machine that dispenses products such as snacks, drinks, etc. to customers after money or a credit card is inserted into the machine. In this project, we use Verilog HDL to design a vending machine controller on an FPGA board.

The design is based on a finite state machine that responds to user input and dispenses the product once the correct amount of money has been inserted and returns the change if any. The Verilog code includes modules for the state machine, the coin slot (push buttons), and the display(led). The design has been tested on an FPGA board and can be further optimized for real-world implementation.

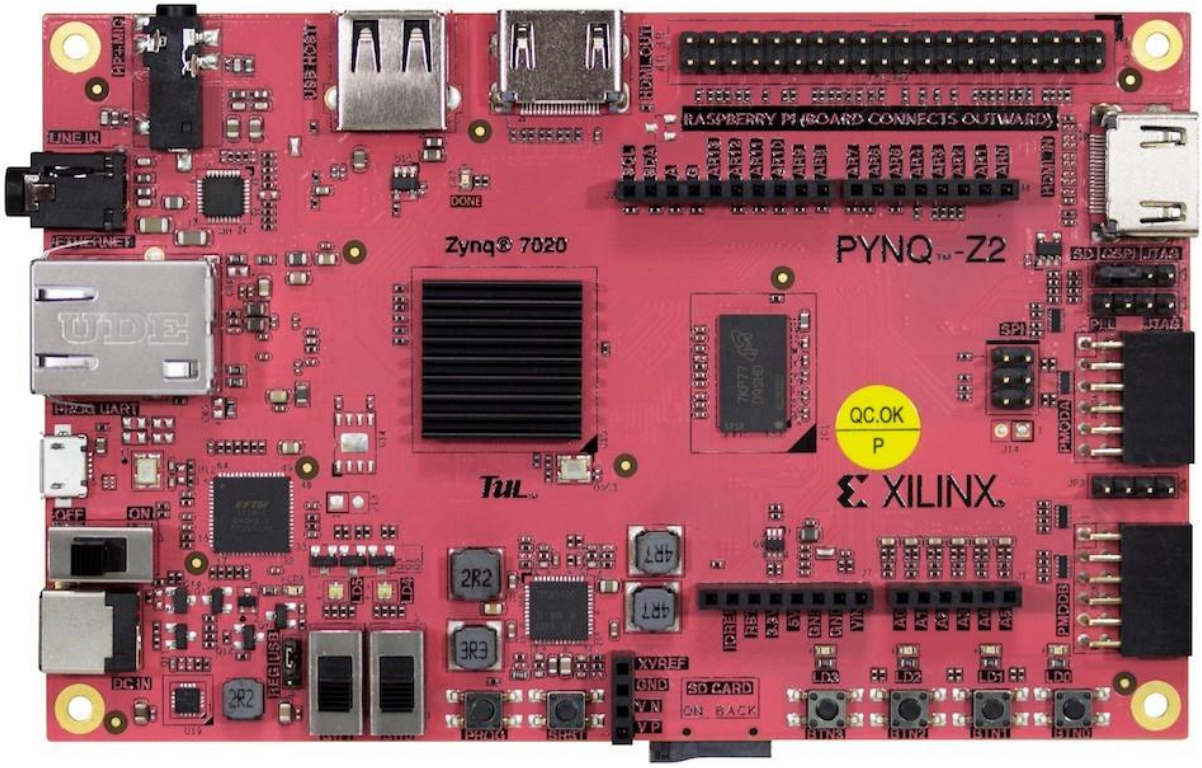
Theory:

State Diagram:



Note: the numbers used for input and change are in decimal and output in binary. And the product cost is of rupees 15.

How to use it?



From the above picture. We allotted the 2 switches in bottom left as the coin slot right switch being the MSB i.e., we enter 01 for 5rs and 10 for 10rs and 00 for nothing.

The push buttons at the bottom right are used for clock and reset i.e., BTN3 is clock and BTN2 is reset. Remaining push-buttons are not in use.

For the output we use the LEDs that are above the push-buttons, LD3 and LD2 are LSB and MSB for change respectively. The led at the right end is the product i.e., 1 for product delivered and 0 for not delivered.

Verilog Code:

Vending_Machine.v | Vending_Machine_TB.v | vendingX

C:/Users/acer/Desktop/vendingmachine_2/vendingmachine_2.srcs/s

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////
3  // Company:
4  // Engineer:
5  //
6  // Create Date: 05/04/2023 03:51:50 PM
7  // Design Name:
8  // Module Name: Vending_Machine
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////////////////////////////////////
21
22
23 module Vending_Machine(
24     input clk,
25     input rst,
26     input [1:0]in,
27     output reg out,
28     output reg[1:0]change
29 );
30
31 parameter S0 = 2'b00;
32 parameter S1 = 2'b01;
33 parameter S2 = 2'b10;
34     reg[1:0] c_state,n_state;
35 always@(posedge clk)
36 begin
37     if(rst==1)
38         begin
39             c_state = 2'b00;
40             n_state = 2'b00;
41             change = 2'b00;
42         end
43     else
44         c_state = n_state;
```

```
    else
        c_state = n_state;
    case(c_state)
        S0: // state_0 : 0rs
        if(in==2'b00)
            begin
                n_state = S0;
                out = 0;
                change = 2'b00;
            end
        else if(in==2'b01)
            begin
                n_state = S1;
                out = 0;
                change = 2'b00;
            end
        else if(in==2'b10)
            begin
                n_state = S2;
                out = 0;
                change = 2'b00;
            end
        S1: //state_1 :5rs
        if(in==2'b00)
            begin
                n_state = S0;
                out = 0;
                change = 2'b01;
            end
        else if(in==2'b01)
            begin
                n_state = S2;
                out = 0;
                change = 2'b00;
            end
         else if(in==2'b10)
            begin
                n_state = S0;
                out = 1;
                change = 2'b00;
            end
         S2: //state_2 :10rs
         if(in==2'b00)
            begin
                n_state = S0;
                out = 0;
                change = 2'b10;
            end
         else if(in==2'b01)
            begin
                n_state = S0;
                out = 1;
                change = 2'b00;
            end
         else if(in==2'b10)
            begin
                n_state = S0;
                out = 1;
                change = 2'b01;
            end
         endcase
     end
endmodule
```

The Testbench:

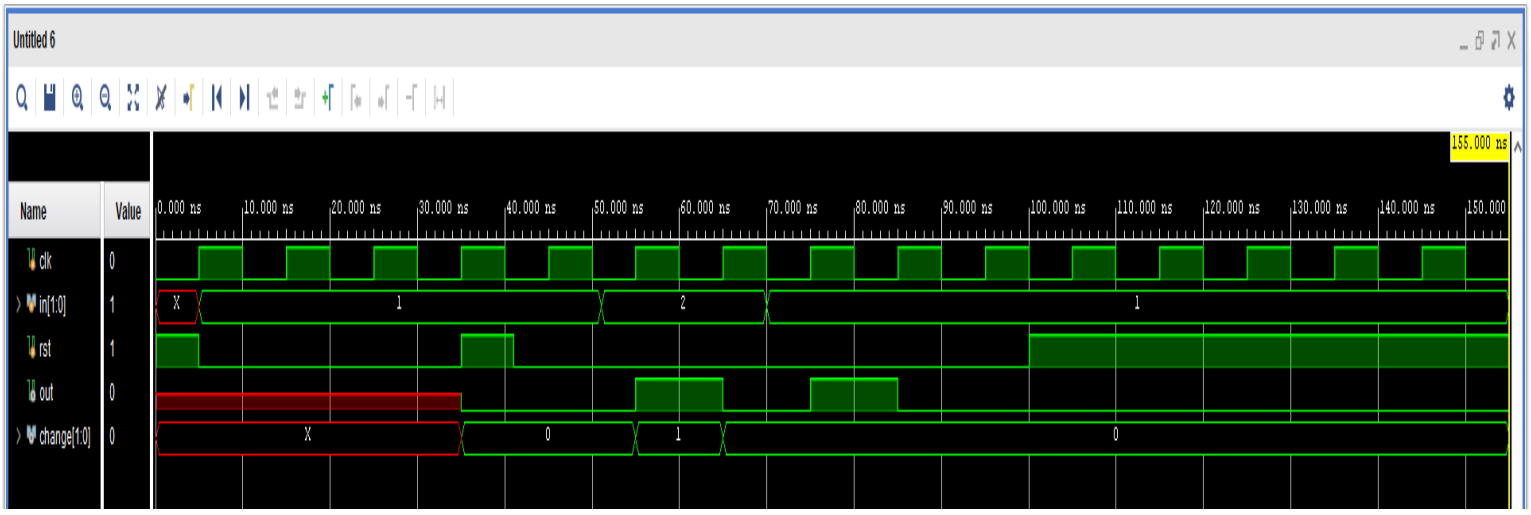
Vending_Machine.v | Vending_Machine_TB.v | vendingXdc.xdc

C:/Users/acer/Desktop/vendingmachine_2/vendingmachine_2.srcs/sim_1/new/Vendir

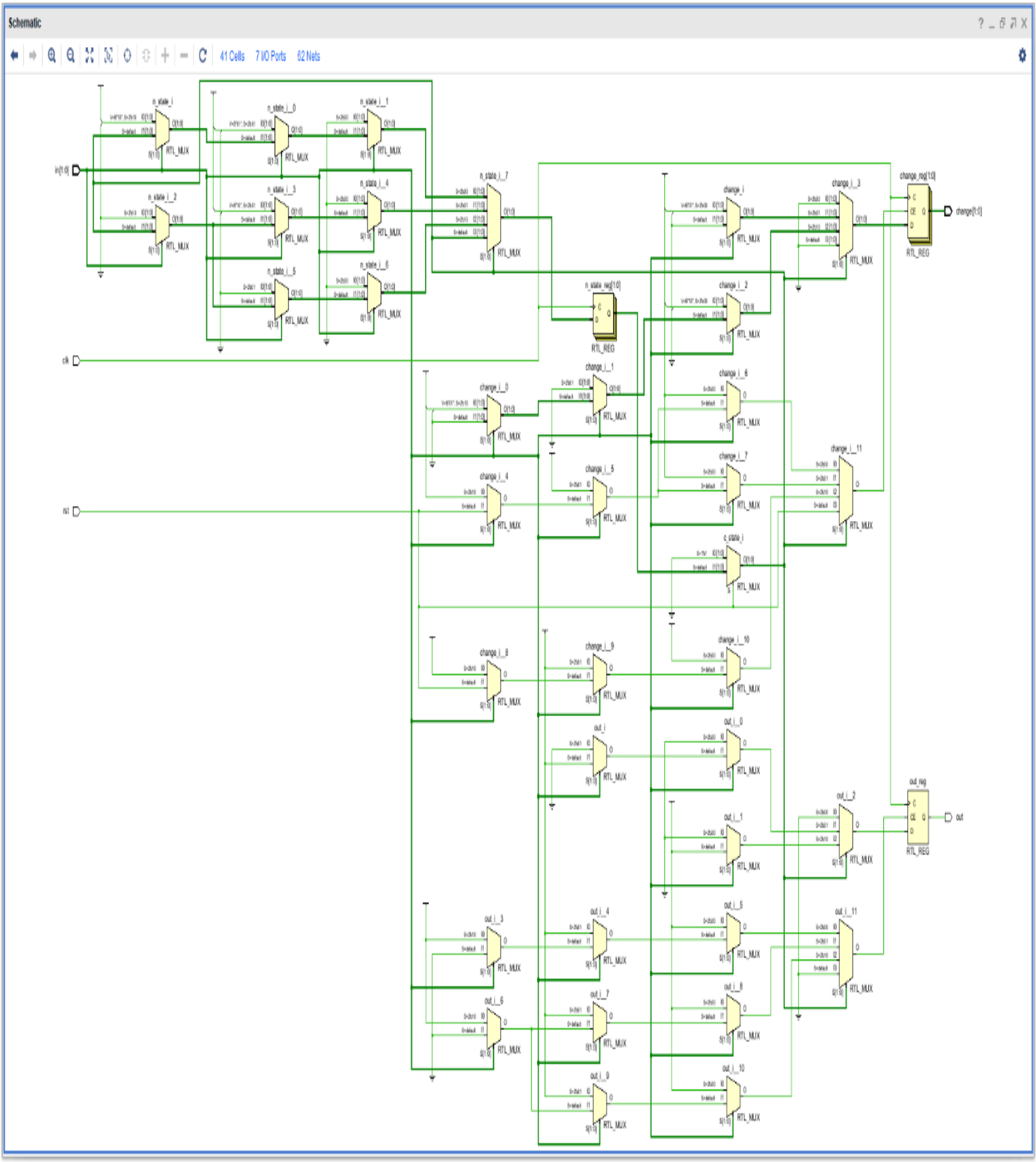
```
14  // Dependencies:
15  //
16  // Revision:
17  // Revision 0.01 - File Created
18  // Additional Comments:
19  //
20  //////////////////////////////////////////////////////////////////////////////////


23  module Vending_Machine_TB(

25      );
26      reg clk;
27      reg [1:0] in;
28     reg rst;
29
30     wire out;
31     wire [1:0] change;
32
33     Vending_Machine uut(
34      .clk(clk),
35      .rst(rst),
36      .in(in),
37      .out(out),
38      .change(change)
39
40     );
41
42  initial begin
43  rst = 1;
44  clk = 0;
45  #5 rst = 0;
46  in = 1;
47  #10 in = 1;
48  #14 in = 1;
49  #6 rst =1;
50  #6 rst =0;
51  #10 in = 2'b10;
52  #14 in = 2'b10;
53  #25 $finish;
54  end
55     always #5 clk = ~clk;
56
57  endmodule
```

Scans

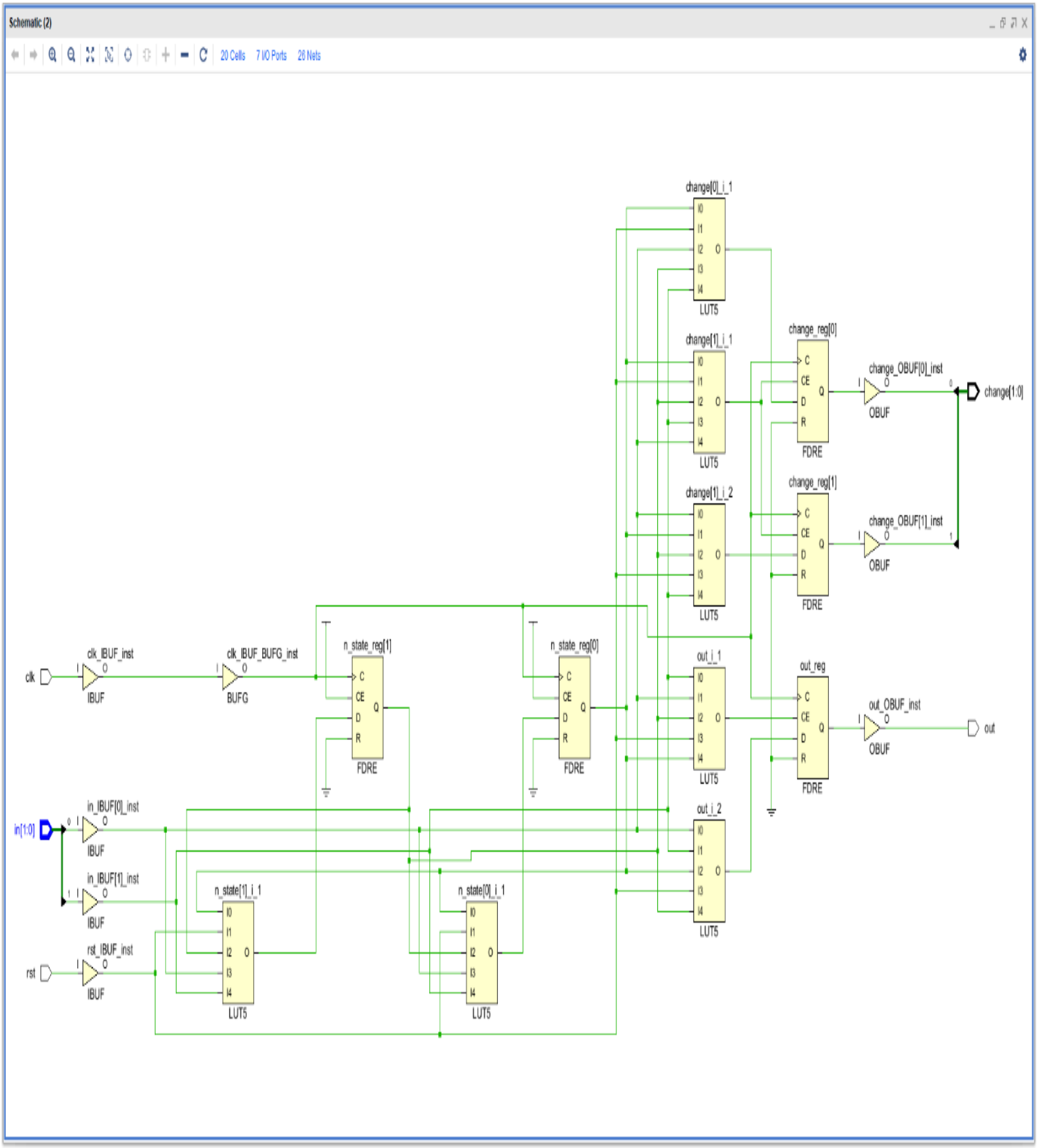
Simulation:



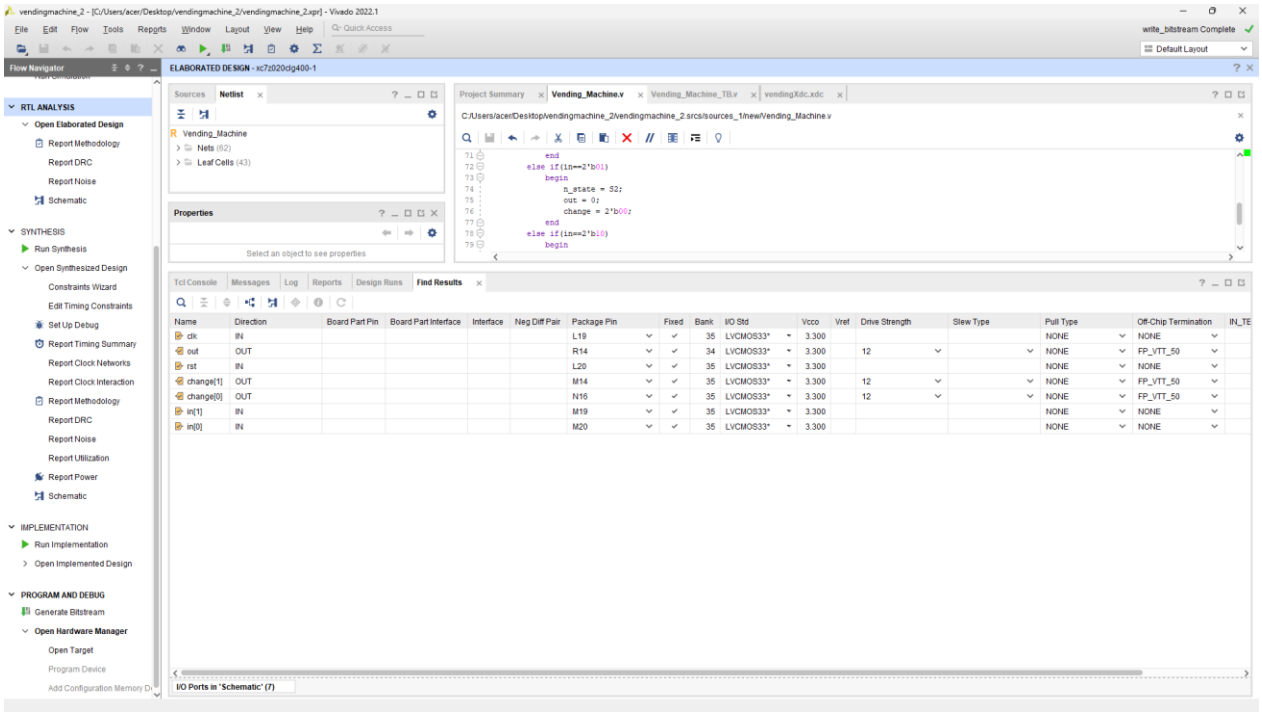
Behavioural schematic:



Synthesized schematic:



Port Definitions:



Implementation:

Vending_Machine.v | Vending_Machine_TB.v | **vendingXdc.xdc**

C:/Users/acer/Desktop/vendingmachine_2/vendingmachine_2.srcs/constrs_1/new/vendingXdc.xdc

```
1  set_property PACKAGE_PIN L19 [get_ports clk]
2  set_property PACKAGE_PIN R14 [get_ports out]
3  set_property PACKAGE_PIN L20 [get_ports rst]
4  set_property PACKAGE_PIN N16 [get_ports {change[1]}]
5  set_property PACKAGE_PIN M14 [get_ports {change[0]}]
6  set_property PACKAGE_PIN M20 [get_ports {in[1]}]
7  set_property PACKAGE_PIN M19 [get_ports {in[0]}]
8  set_property IOSTANDARD LVCMOS33 [get_ports clk]
9  set_property IOSTANDARD LVCMOS33 [get_ports out]
10 set_property IOSTANDARD LVCMOS33 [get_ports rst]
11 set_property IOSTANDARD LVCMOS33 [get_ports {change[1]}]
12 set_property IOSTANDARD LVCMOS33 [get_ports {change[0]}]
13 set_property IOSTANDARD LVCMOS33 [get_ports {in[1]}]
14 set_property IOSTANDARD LVCMOS33 [get_ports {in[0]}]
15 set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets clk_IBUF]
```