DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

PDPM-IIITDM JABALPUR,

MADHYA PARDESH -482005

Project Report

Vending Machine Using Verilog

IT 3E01: - (Verilog),

2022-23

Submitted to:

DR. Koushik Dutta

Submitted by:

1. Ritik Sharma

(Roll no 20BME047)

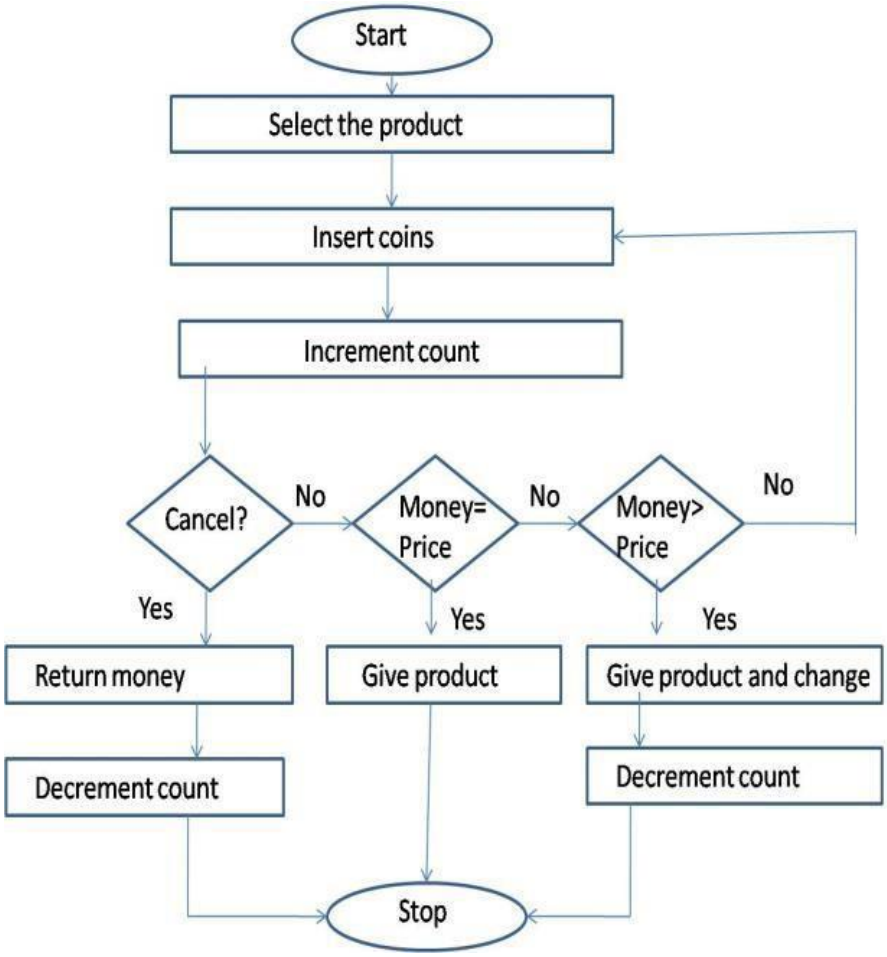
Vending Machine

Introduction: The Vending Machine is an automatic machine that sells food such as canned soups and packaged sandwiches, snacks or other items such as newspapers or tickets. The vending machine market is a big business with a huge annual revenue for leading nations like the USA, North America, Japan, China and some other Asian countries including India. The machines usually work when a product is selected, and some money (usually coins or paper money) is put in a slot. Then, a button needs to be pushed, or a lever pulled. If there is enough money, the selected item will be dropped to a tray, where it can be taken out by the person making the purchase. Most of the vending machines which are currently in use are based on CMOS, Microcontroller technology. The latest FPGA based machine is programmable and reconfigurable.

Objectives: The problem is defined as the design of a vending machine that accepts coins of denominations five and ten and dispenses three products of different prices. The machine should possess additional features of returning change when a coin of higher denomination is inserted and returning money when request is cancelled. The machine asks the customer to select the product first and then insert coins as per the price of the selected product. The select and coin signals are therefore inputs to the machine. There is also a cancel option which is another input. The machine checks the inserted amount with the price of the selected product and dispenses the product if both are equal. If the customer inserts a coin of higher denomination, the machine gives the product along with the change. Whenever change is not available in the machine, it returns the total amount. Product and change are therefore outputs of the machine. Money is returned when the request is cancelled. Therefore, returning money is another issue. Two registers are needed, one to keep track of the coin count of the present transaction and the other to keep track of the total coin count in the machine. The registers are taken to be ten-bit width each. When a higher denomination coin is inserted, change can be given only if the total coin count is higher than the coin count of the present transaction. In this

way, the machine checks whether the change is available or not. To meet the above specifications, a finite state machine approach is adopted.

Theory and Method: The diagram below is the block diagram of the vending machine. The inputs are the coins and the output is the Product.



Initially, the product needs to be selected followed by the insertion of coins. The count is incremented as the coins are inserted. If a cancel signal is given, the machine returns inserted money and decrements count. Otherwise, the machine checks the inserted money with the price of the selected product. If both are equal, the machine dispenses the appropriate product. If inserted money is greater than price, the machine gives the appropriate product along with the change. If the inserted money is less than the price of the selected product, the machine waits for the customer to insert more coins.

Design Methodology :

- **Finite state Method:**

Any Sequential digital circuit can be converted into a state machine using a state diagram. In a state machine the circuit's output is defined in a different set of states i.e.. Each output is a state. There is a State Register to hold the state of the machine and a next state logic to decode the next state. There is also an output register that defines the output of the machine. The next state logic is

the sequential part of the machine, and the Output and Current State are the Register part of the logic.

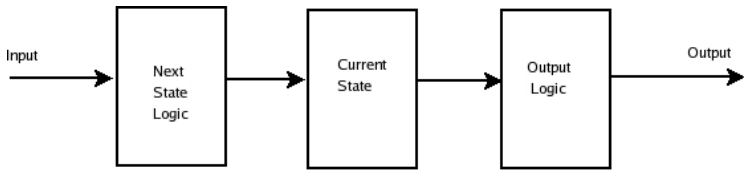
There are two types of state machines:

1. MOORE

2. MEALY

MOORE

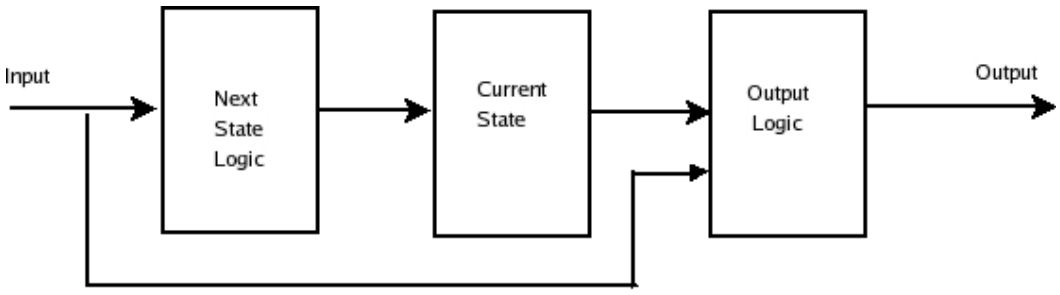
In a Moore machine the output state is totally dependent on the present state



MOORE machine

MEALY

In a mealy machine the output depends on the input as well as the present state.

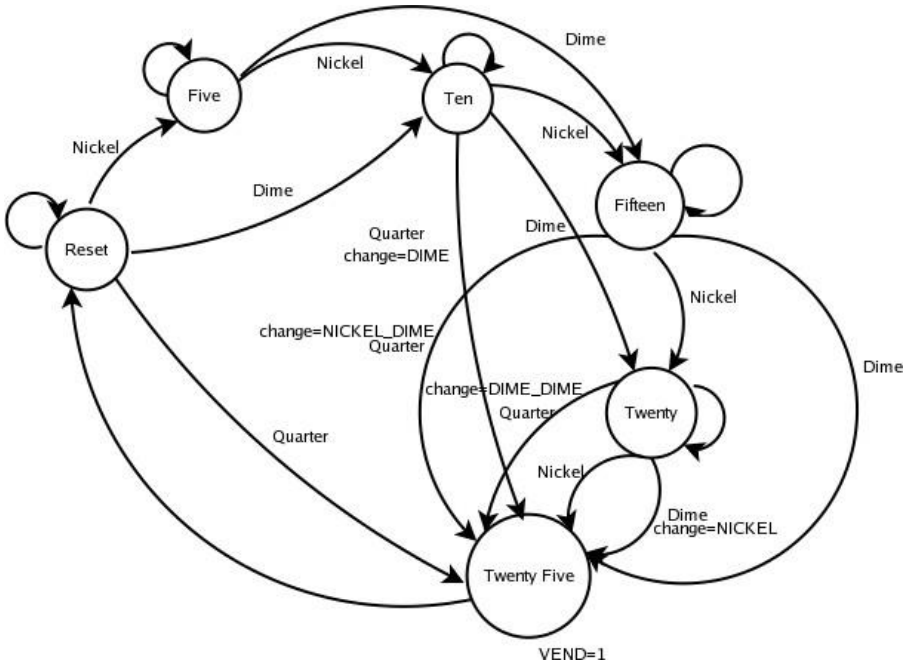


MEALY machine

STATE DIAGRAM OF VENDING MACHINE :

Sel = 00 – Product A (Rs. 5/-) Coin = 01 - Five Sel = 01 – Product B (Rs. 10/-) Coin = 10 -
Ten Sel = 10 – Product C (Rs. 20/-)

Initially, the machine is in initial state where no product is given. The customer selects the product and inserts coins. If a five coin is inserted, the machine goes to ‘five’ state. If cancel button is pressed, the machine returns the money and goes to initial state. Otherwise, it checks for selection and gives product if select signal is of product A. Similarly, when ten coin is inserted, the machine goes to ‘ten’ state. If cancel button is pressed, the machine returns the money and goes to initial state. Otherwise, it checks for selection. If selected product has the same price as money inserted, it gives the product and change. If the entered money is of higher denomination than the price of the product selected, the machine gives product B along with change of five. Similarly, the machine enters other states such as fifteen and ten as coins are inserted and checks for selection after reaching each state. The product is dispensed accordingly. The state diagram indicates the flow of various signals and states during the working of vending machine.



What happens is whenever we get a coin we jump to the next state. So

for example, we get a coin from the reset state say a NICKEL, then we jump to the next state FIVE. Otherwise, we stay in the same state. When we get Extra amount we come back to the reset state and the difference is given back to the user.

Code of Machine:

```
module candy(d,n,q, reset, clk, y);
output reg y;
input d,n,q;    //n=5,d=10,q=25;
input clk;
input reset;
reg [2:0] cst, nst;
parameter S0 = 3'b000,
        S1 = 3'b001,
        S2 = 3'b010,
        S3 = 3'b100,
           S4 = 3'b101,
           S5 = 3'b110,
           S6 = 3'b111;
always @(cst or d or n or q)
 begin
 case (cst)
  S0: if (n== 1'b1 && d==1'b0 && q==1'b0)
         begin
       nst = S1;
          y=1'b0;
         end
    else if(n== 1'b0 && d==1'b1 && q==1'b0)
         begin
         nst=S2;
         y=1'b0;
         end
         else if(n== 1'b0 && d==1'b0 && q==1'b1)
         begin
         nst=S5;
         y=1'b0;
         end
         else
         begin
         nst = cst;
         y=1'b0;
         end
  S1: if (n== 1'b1 && d==1'b0 && q==1'b0)
         begin
```

```
      nst = S2;
         y=1'b0;
        end
   else if(n== 1'b0 && d==1'b1 && q==1'b0)
        begin
        nst=S3;
        y=1'b0;
        end
        else if(n== 1'b0 && d==1'b0 && q==1'b1)
        begin
        nst=S6;
        y=1'b0;
        end
        else
        begin
        nst = cst;
        y=1'b0;
        end
S2:if (n== 1'b1 && d==1'b0 && q==1'b0)
        begin
      nst = S3;
         y=1'b0;
        end
   else if(n== 1'b0 && d==1'b1 && q==1'b0)
        begin
        nst=S4;
        y=1'b0;
        end
        else if(n== 1'b0 && d==1'b0 && q==1'b1)
        begin
        nst=S0;
        y=1'b1;
        end
        else
        begin
        nst = cst;
        y=1'b0;
        end
S3: if (n== 1'b1 && d==1'b0 && q==1'b0)
        begin
      nst = S4;
         y=1'b0;
        end
   else if(n== 1'b0 && d==1'b1 && q==1'b0)
        begin
        nst=S5;
        y=1'b0;
```

```
        end
        else if(n== 1'b0 && d==1'b0 && q==1'b1)
        begin
        nst=S0;
        y=1'b1;
        end
        else
        begin
        nst = cst;
        y=1'b0;
        end
S4: if (n== 1'b1 && d==1'b0 && q==1'b0)
        begin
      nst = S5;
        y=1'b0;
        end
    else if(n== 1'b0 && d==1'b1 && q==1'b0)
        begin
        nst=S6;
        y=1'b0;
        end
        else if(n== 1'b0 && d==1'b0 && q==1'b1)
        begin
        nst=S0;
        y=1'b1;
        end
        else
        begin
        nst = cst;
        y=1'b0;
        end
S5: if (n== 1'b1 && d==1'b0 && q==1'b0)
        begin
      nst = S6;
        y=1'b0;
        end
    else if(n== 1'b0 && d==1'b1 && q==1'b0)
        begin
        nst=S0;
        y=1'b1;
        end
        else if(n== 1'b0 && d==1'b0 && q==1'b1)
        begin
        nst=S0;
        y=1'b1;
        end
         else
```

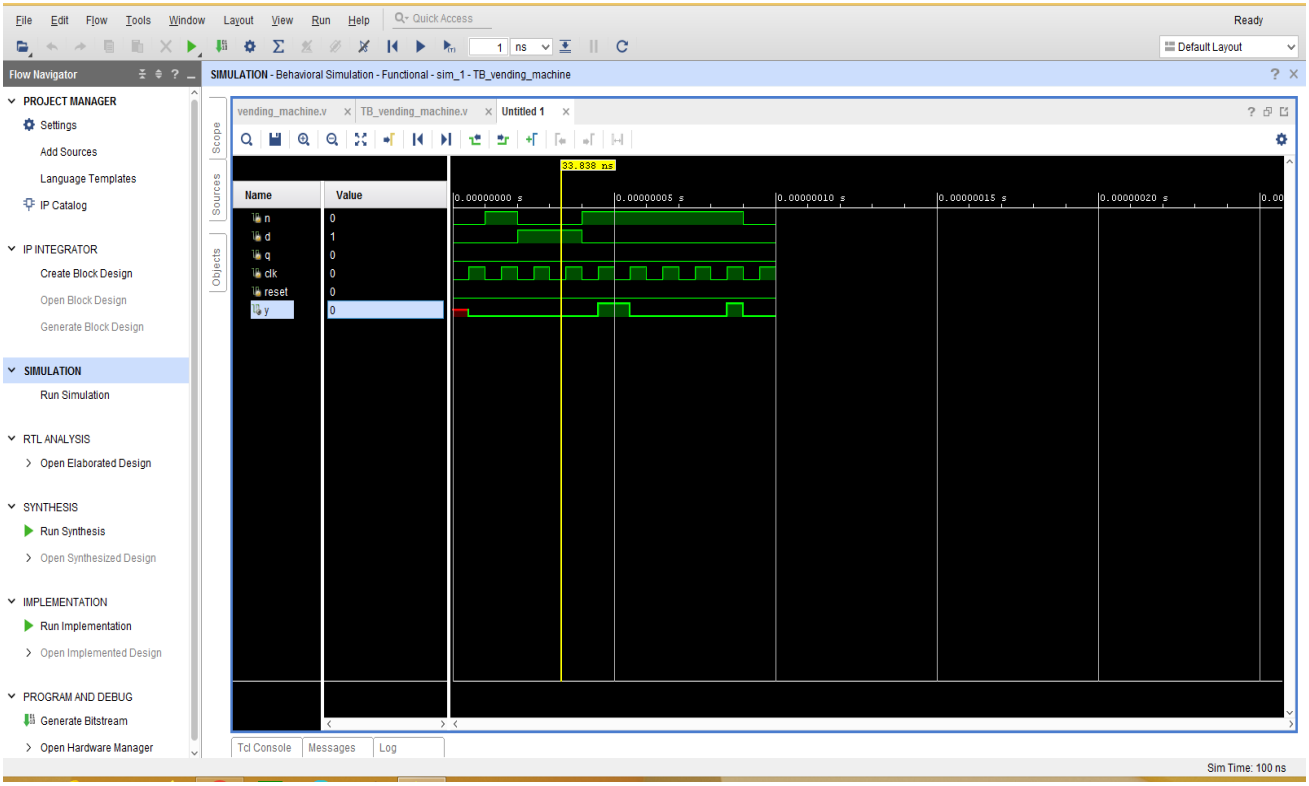
```
          begin
          nst = cst;
          y=1'b0;
          end
S6: if (n== 1'b1 && d==1'b0 && q==1'b0)
          begin
        nst =S0;
           y=1'b1;
          end
     else if(n== 1'b0 && d==1'b1 && q==1'b0)
          begin
          nst=S0;
          y=1'b1;
          end
          else if(n== 1'b0 && d==1'b0 && q==1'b1)
          begin
          nst=S0;
          y=1'b1;
          end
          else
          begin
          nst = cst;
          y=1'b0;
          end

   default: nst = S0;
  endcase
end
always@(posedge clk) //or posedge reset)
begin
 if (reset)
 cst <= S0;
 else
   cst <= nst;
end
endmodule
```

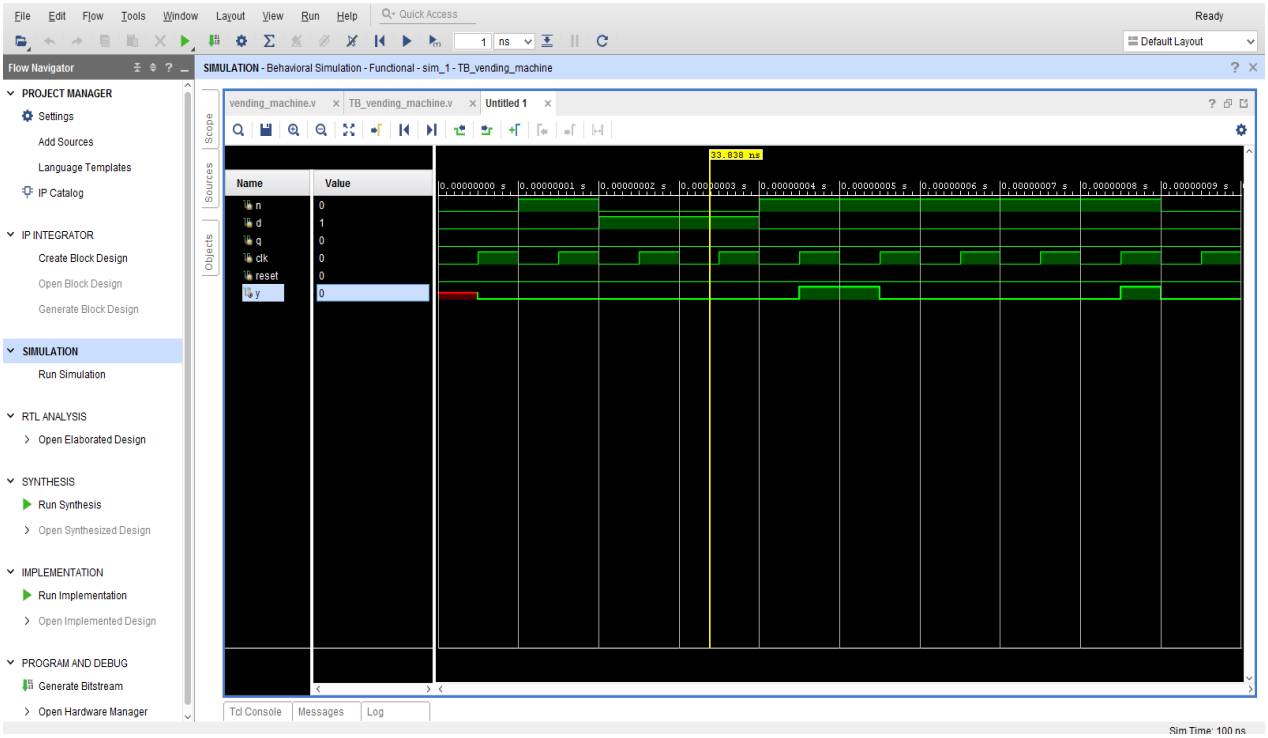
Test Bench Code:

```
module  candy_tb;
reg n,d,q,clk,reset;
wire y;
```

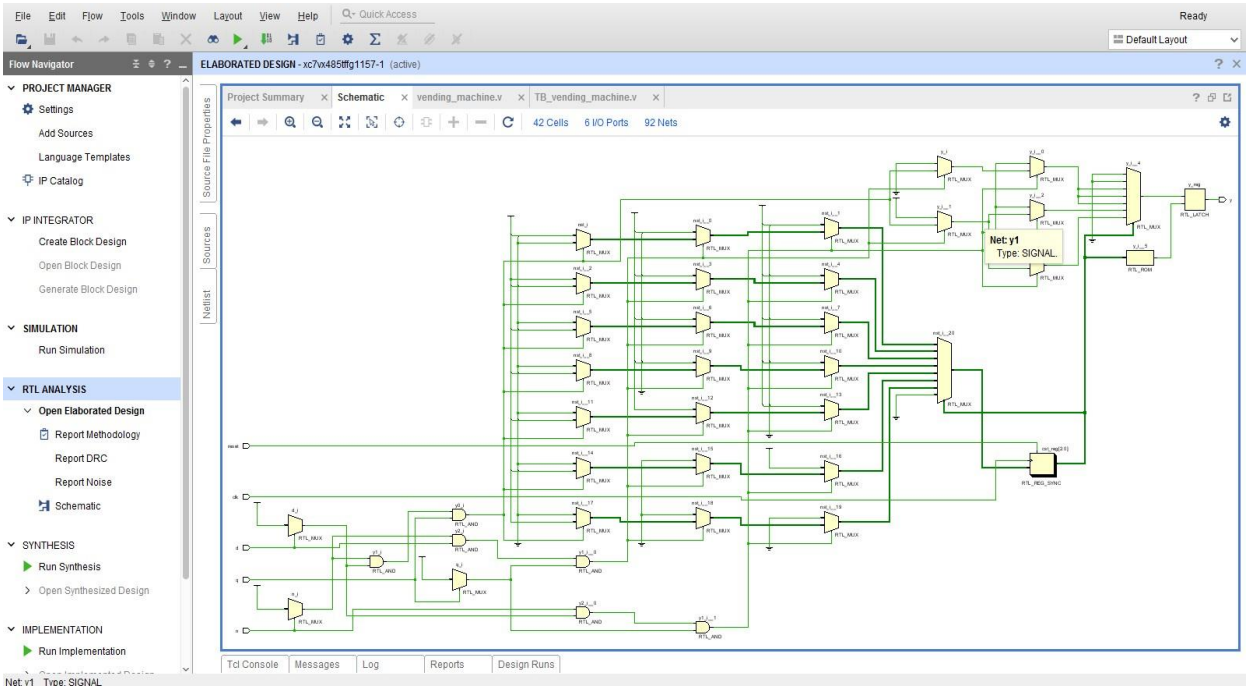
```
candy m1(n,d,q,reset, clk, y);
initial
begin
reset=0         ;clk=0;n=0;q=0;d=0;
$monitor($time, ,
,"c=%b",clk,,"y=%b",y,,"r=%b",reset,,"d=%b",d,,"n=%b",n,,"q=%b",q);
#10 d=0;n=1;q=0;
#10 d=1;n=0;q=0;
#10 d=1;n=0;q=0;
#10 d=0;n=1;q=0;
#10 d=0;n=1;q=0;
#10 d=0;n=1;q=0;
#10 d=0;n=1;q=0;
#10 d=0;n=1;q=0;
#10 d=0;n=0;q=0;
end
always
#5 clk=~clk;
initial
#100 $finish ;

Endmodule
```

Output:





SCHEMATIC DIAGRAM:



Result and discussion:

1. The Verilog code has been formulated for the vending machine using behavioural modelling. The testbench has been written for three products A, B and C and cancellation of two transactions. The simulation has been performed.

2. In the first transaction, Product C – 10 was selected. Cancel signal was given after inserting a five-rupee coin and a ten-rupee coin. The machine returned the five-rupee coin and the ten-rupee coin. In the second transaction, Product A – 00 was selected and a five-rupee coin was inserted. The machine dispensed Product A. In the third transaction, Product C was again selected and cancel signal was given after inserting three five-rupee coins. The machine returned the three five-rupee coins.

3. In the fourth transaction, Product A – 00 was selected and a ten-rupee coin was inserted. The machine dispensed Product A of price five rupees along with a change of five rupees. In the fifth and sixth transactions,

products B and C were selected, and appropriate money was inserted, and the products were dispensed.

4. The first and third transactions indicate cancellation feature. The fourth transaction indicates the return change feature. The fifth and sixth transactions indicate the dispensing of product when money equal to exact price is inserted.

Conclusion and future work:

The vending machine was successful in dispensing three products A, B and C of prices Rs.5/-, Rs.10/- and Rs-20/- respectively, with the additional features of dispensing product along with returning change when higher denomination coin is inserted and returning total money when request is cancelled. The vending machine is successful in meeting the specifications laid out prior to the design. It has been observed in different scenarios, that FPGA based vending machine gives faster response and shows low power consumption as compared to the microcontroller based machines. The results clearly indicate that the computational efficiency is increased by 10% as the algorithm uses less logic utilizations as compared to the existing FPGA based design algorithms

The system has much more enhanced capabilities like withdrawal of money in between the transaction and automatic refill indication. Also we can monitor the FPGA based vending machine with the main frame computer. Its algorithm is very flexible and reliable as the vendor can easily enhance the algorithm for large number of products and money of different denominations at low cost as compared to microprocessor based vending machine. The next step would be to incorporate some of the following features-reverse vending capability, card payment facility and capability to queue & process a set of orders on FIFO basis.

References:

[1] Samir Palnitkar, “Verilog HDL”, Second Edition, Prantice Hall, 2003.

[2] Peter Minns, Ian Elliott, “FSM-based Digital Design using Verilog HDL”, John Wiley & Sons,Ltd 2008.

[3] Phong P. Chu, “FPGA Prototyping Using Verilog HDL- Xilinx Spartan 3 Version”, John Wiley & Sons,Ltd

[4] http://www.xilinx.com/itp/xilinx4/data/docs/sim/vtex9.html

[5] http://www.xilinx.com/itp/3_1i/data/fise/xug/chap07/xug07003.htm

[6] Michael D. Ciletti, “Modeling, synthesis and Rapid prototyping with Verilog HDL”