INDIAN INSTITUTE OF INFORMATION TECHNOLOGY GUWAHATI

VLSI Design Lab

Mini Project

SUBMITTED BY :--

NAME → ANISH KUMAR CHAUDHARY

ROLL NO →1901028

GROUP →EC31

Date of Submission → 2nd Mar 2022

SUBMITTED TO :--

Dr. Rakesh Biswas

T/A

Birabrata Seal ,

Jitumani Sarma

ATM USING MOORE STATE MACHINE

Objective :-

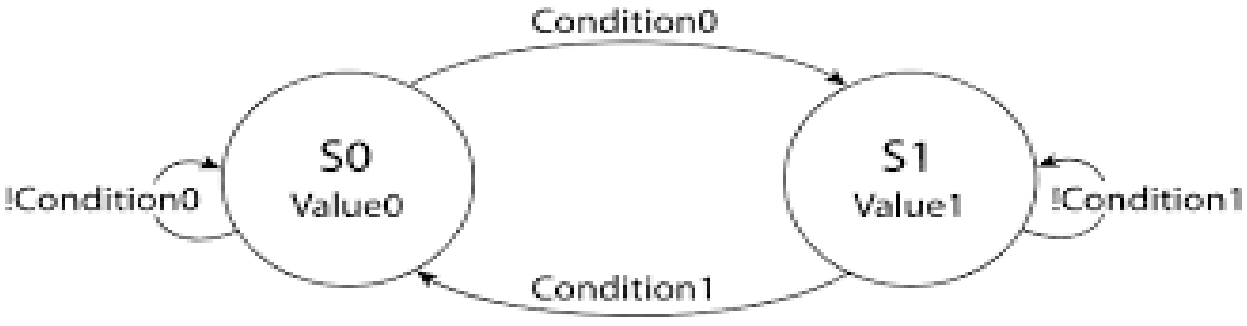
- Design ATM controller for secure financial transaction using Moore State Machine.

Platform :-

- Vivado 2017.4

Theory :-

Moore State Machine – A finite state machine whose current output values are determined only by its current state. Like other finite state machine, in Moore machine the input typically influences the next state by indirectly influencing subsequent output, but not the current or immediate output.



ATM (automated teller machine) — ATM is a very essential tool required in order to facilitate the need of safe transaction of money. In the growing technological world, people want their work to be very simple and safe in order to save time. As we know some new technology is becoming very popular in banking sector which is referred as ATMs. A finite state machine is used to model a simple ATM in this project. The developed design is modelled using Verilog HDL language which hardware description language. The simulation is carried out using Xilinx tool.

Algorithm :-

1. First of all, Different states of ATM controller need to be defined for different operations and unique binary code is assigned to them. In this project overall 12 states where defined and hence for that 4 bit binary code would be sufficient for unique assigning. Different states and there code is given below –

STATE CODE	STATE DISCRIPTION	CODE IN BINARY
S0	Welcome	0000
S1	Scan Card	0001
S2	Enter PIN	0010
S3	Option For Transaction	0011
S4	Invalid	0100
S5	Withdraw	0101
S6	Balance Check	0110
S7	Deposit	0111
S8	Withdraw amount	1000
S9	Balance Show	1001
S10	Deposit amount	1010
S11	Next	1011

2. We also need to define 1 bit or 2 bit signal to decide which state to go next. These signals are defined below which decides the next state for the whole transaction.

SIGNAL CODE	DISCRIPTION
IC	Insert Card
S	Scanning account
M	Matching Pin
OP	Option for Transaction
WC	Withdrawal Condition
DC	Deposit Condition
N	Next Operation

3. Now block diagram of ATM controller were made using different states and input signals defined earlier. This block diagram is attached later in this report.

4. Functioning of our ATM controller –

   Initially the system is in ideal state which is our welcome screen and is ready to operate, hence S0 state is selected. When card is detected in the machine then scan state goes high

which represent the next state S1. In S2, pin is needed for the further transaction. If invalid pin is given then system will goes to invalid state i.e. S4 state goes high and then it again goes to ideal state i.e. S0 state. When the valid pin is given control signal then goes to S3 state in which user can select the option for transaction.

There are Four types of transaction option that is used in this project.

1. Withdraw -> OP = 01
2. Balance Check -> OP = 10
3. Deposit -> OP = 11
4. Exit -> OP = 00

Withdraw state is represented by S5, in this system checks whether the withdrawal amount is less then the available balance of that particular account number, if condition is satisfied than cash will be withdrawn and S8 goes high otherwise it goes to invalid state and S4 state goes high. Balance check is represented by S6 state. In balance check no condition is required it just goes to S6 state where balance is displayed on the screen. Deposit state is represented by S7 state. If deposit amount is less than the daily limit of deposition then amount is deposited and S10 goes high otherwise it goes to invalid state i.e. S4 state. To exit transaction, just OP=00 needed which makes system ideal.

For further transaction, there is next state i.e. S11 state. Here, either option for transaction i.e. S3 state (N=0) is selected or abort the transaction by again going to ideal state (N=1).

5. Now Moore Machine for the overall ATM controller is designed which is an effective way to implement the control functions. The Moore State machine is attached later in this report.

6. For the Verilog implementation inputs and outputs were decided

   Inputs –

   clk, rst, account, pin, bal, ammount, IC, S, OP and N

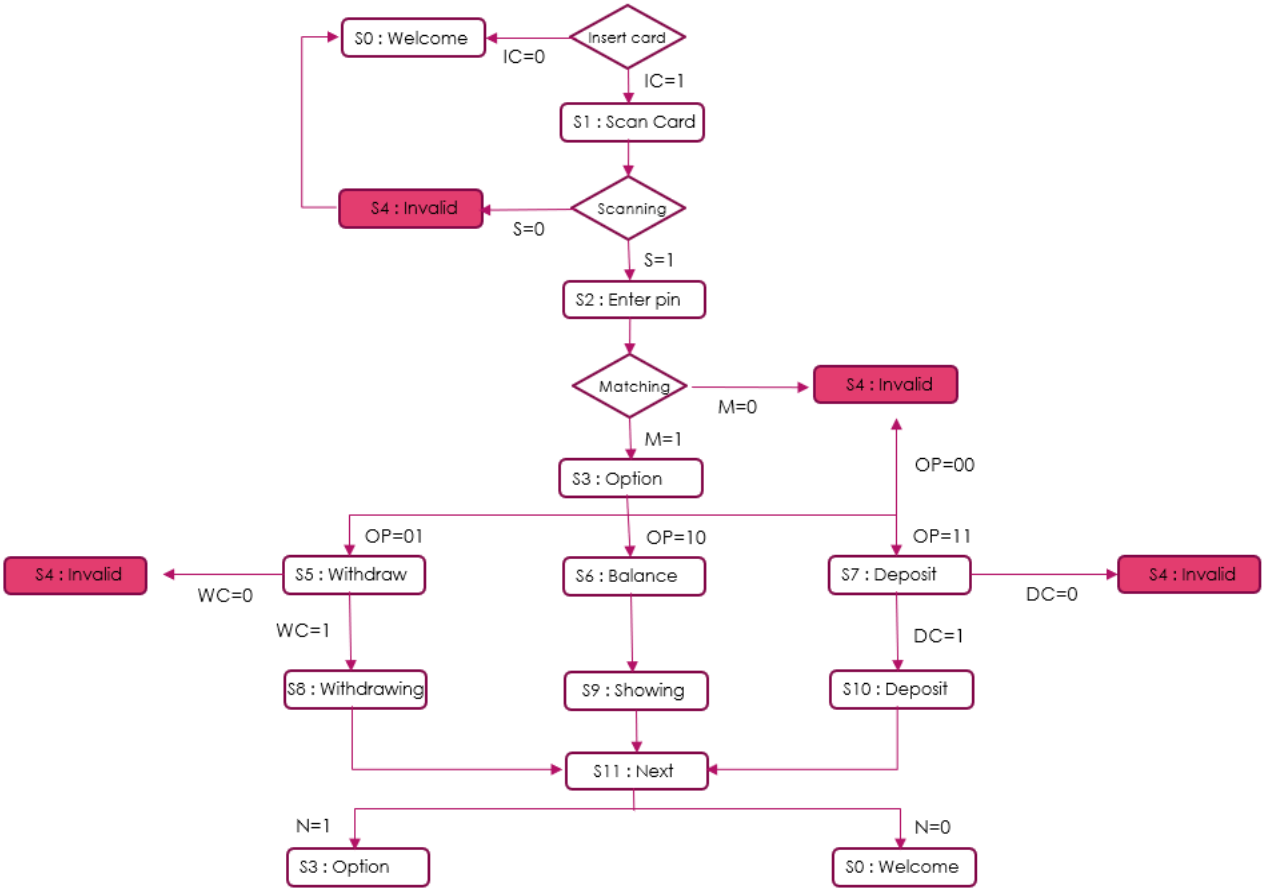
   Outputs –

   Balance – This shows the updated balance after transaction
   Y – This is 11 : 0 bus which shows the outputs of all the states during state transition.

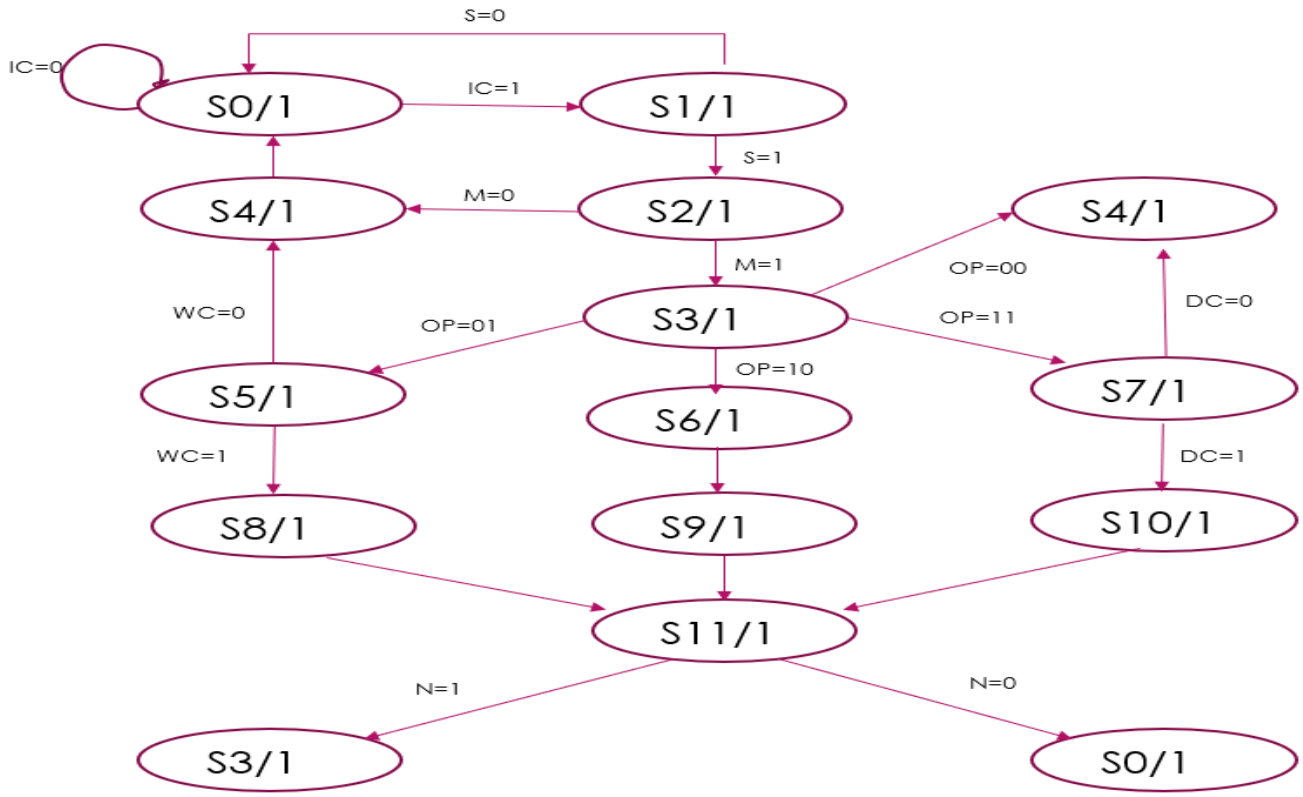
7. Finally state machine was implemented in Verilog along with all the conditions and input signals. Test bench for this project was created where four cases were taken –
   1. Ideal State Setting
   2. Wrong PIN
   3. Showing Balance
   4. Cash Withdrawal

8. After Simulation RTL schematics, Simulation results and observation will be attached in the report.

Block Diagram for ATM Controller :-



Moore State Machine for ATM Controller :-



Code and Test Bench for ATM controller :-

Code for ATM Controller

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 13.02.2022 15:10:48
// Design Name:
// Module Name: ATM_Controller
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////

module ATM_Controller(clk,rst,account,pin,bal,ammount,IC,S,OP,N,Y,balance);

input wire clk,rst,IC,S,N;                          //IC=insert_card,S=card_scan,N=more_transaction
input wire [3:0] pin;
input wire [4:0] account;
input wire [2:0] bal;
input wire [2:0] ammount;
input wire [1:0] OP;                                 //OP= option that is selected by the user
output reg [11:0] Y;
output reg [2:0] balance;

//Definining all the state of ATM controller
parameter [3:0]  Welcome=4'b0000,                    //s0
                 Scan_Card=4'b0001,                  //s1
                 Enter_Pin=4'b0010,                  //s2
                 Option_For_Txn=4'b0011,             //s3
                 Invalid=4'b0100,                    //s4
                 Withdraw=4'b0101,                   //s5
```

```
            Balance_Check=4'b0110,          //s6
            Deposit=4'b0111,                //s7
            Withdrawn_Ammount=4'b1000,      //s8
            Balance_Show=4'b1001,           //s9
            Deposited_Ammount=4'b1010,      //s10
            Next=4'b1011;                   //s11


reg [3:0] current_state, next_state;

always @(*)
begin
    Y[0]  <= 0;
    Y[1]  <= 0;
    Y[2]  <= 0;
    Y[3]  <= 0;
    Y[4]  <= 0;
    Y[5]  <= 0;
    Y[6]  <= 0;
    Y[7]  <= 0;
    Y[8]  <= 0;
    Y[9]  <= 0;
    Y[10] <= 0;
    Y[11] <= 0;

    next_state <= 0;
    balance <= bal;

case(current_state)
     Welcome:begin
                if(IC)
                    begin
                    next_state <= Scan_Card;
                    Y[1] <= 1;
                    end
                else
                    begin
                    next_state <= Welcome;
                    Y[0] <= 1;
                    end
            end
```

```
 Scan_Card: begin
            if(S)
               begin
               next_state <= Enter_Pin;
               Y[2] <= 1;
               end
            else
               begin
               next_state <= Welcome;
               Y[0] <= 1;
               end
         end

Enter_Pin: begin
            if(pin == 4'b1111)
               begin
               next_state <= Option_For_Txn;
               Y[3] <= 1;
               end
            else
               begin
               next_state <= Invalid;
               Y[4] <= 1;
            end
         end

Invalid: begin
            next_state <= Welcome;
            Y[0] <= 1;
       end

Option_For_Txn: begin
             if(OP == 2'b01)
                begin
                next_state <= Withdraw;
                Y[5] <= 1;
                end
             else if(OP == 2'b10)
                begin
                next_state <= Balance_Check;
                Y[6] <= 1;
                end
```

```
            else if(OP == 2'b11)
                begin
                next_state <= Deposit;
                Y[7] <= 1;
                end
            else
                begin
                next_state <= Invalid;
                Y[4] <= 1;
                end
        end

Withdraw: begin
            if(ammount <= bal)
                begin
                balance<=bal-ammount;
                next_state <= Withdrawn_Ammount;
                Y[8] <= 1;
                end
            else
                begin
                next_state <= Invalid;
                Y[4] <= 1;
                end
        end

Balance_Check: begin
            next_state <= Balance_Show;
            Y[9] <= 1;
          end

Deposit: begin
          if(ammount <= 11)
                begin
                balance<=bal+ammount;
                next_state <= Deposited_Ammount;
                Y[10] <= 1;
                end
          else
                begin
                next_state <= Invalid;
                Y[4] <= 1;
                end
          end
```

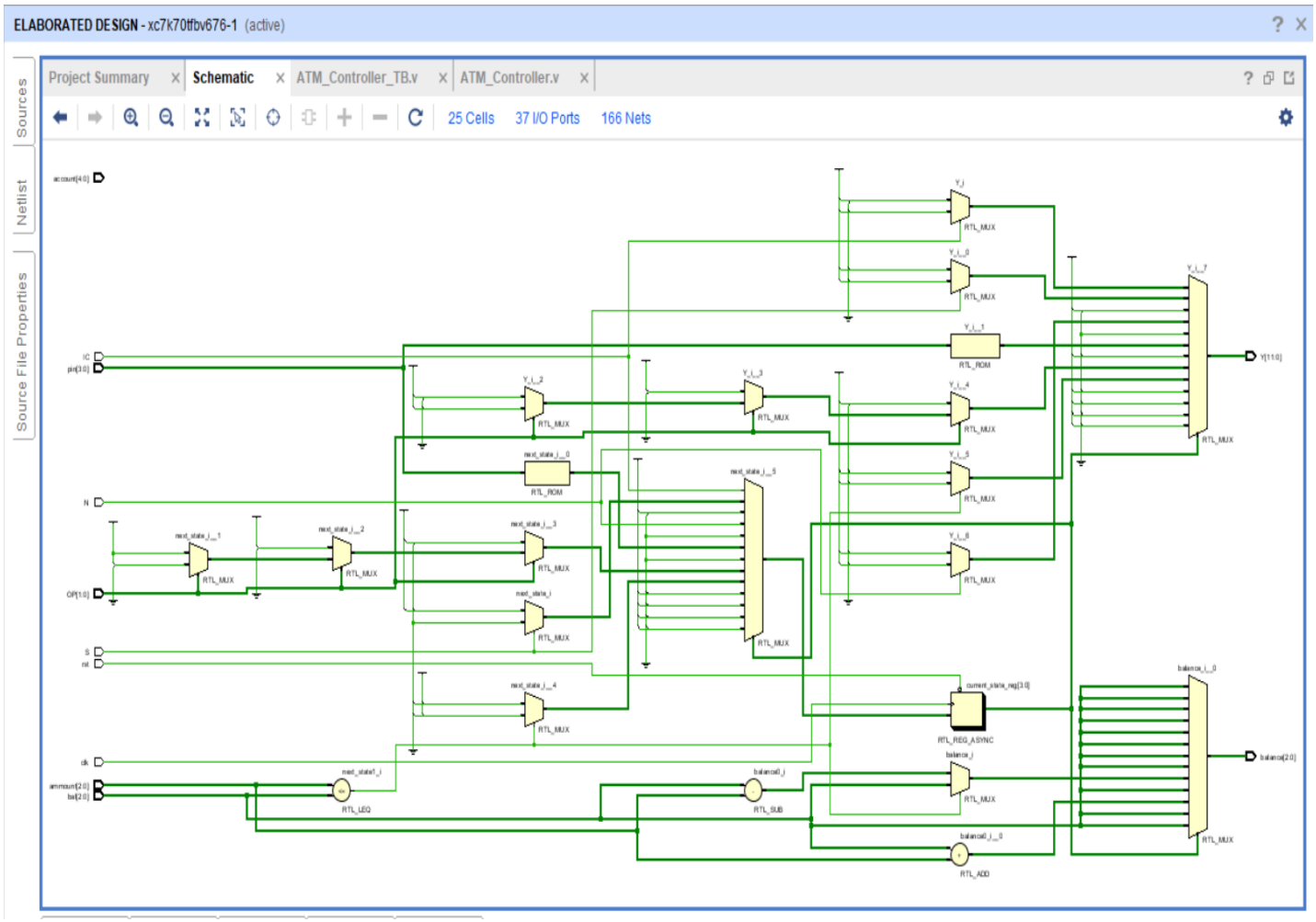
```
    Withdrawn_Ammount: begin
                    next_state <= Next;
                    Y[11] <= 1;
                  end

    Balance_Show: begin
                    next_state <= Next;
                    Y[11] <= 1;
                end

    Deposited_Ammount: begin
                    next_state <= Next;
                    Y[11] <= 1;
                  end

    Next: begin
              if(N)
                 begin
                 next_state <= Option_For_Txn;
                 Y[3] <= 1;
                 end
              else
                 begin
                 next_state <= Welcome;
                 Y[0] <= 1;
                 end
          end
 endcase

end

always @(posedge clk or negedge rst)
       begin
          if(!rst)
               begin
               current_state<=Welcome;
               end
          else
               current_state<=next_state;
               end
endmodule
```

Test Bench for ATM Controller

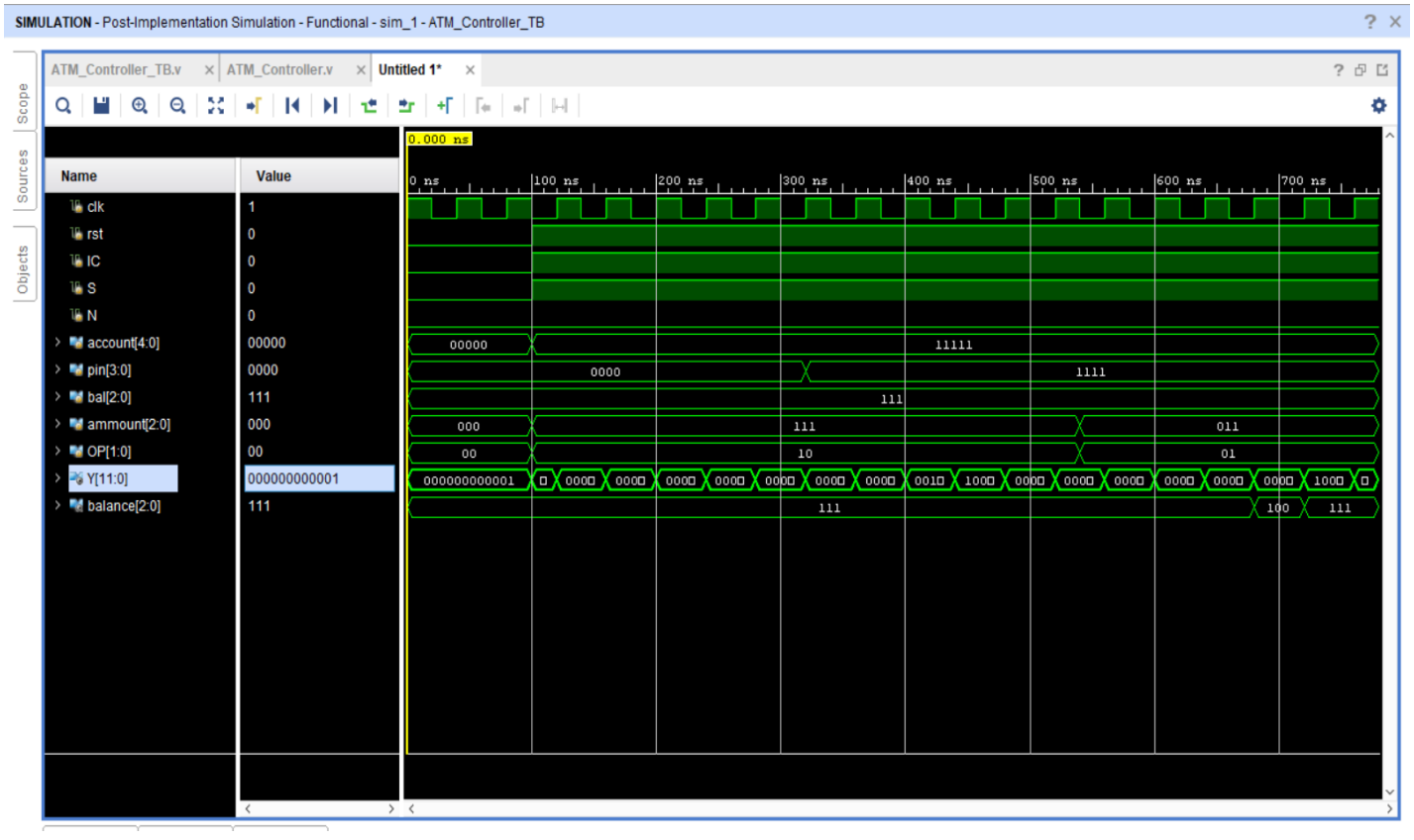
```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 13.02.2022 16:36:48
// Design Name:
// Module Name: ATM_Controller_TB
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module ATM_Controller_TB();
reg clk,rst,IC,S,N;
reg [4:0] account;
reg [3:0] pin;
reg [2:0] bal;
reg [2:0] ammount;
reg [1:0] OP;
wire [11:0] Y;
wire [2:0] balance;


ATM_Controller
uut(.clk(clk),.rst(rst),.IC(IC),.S(S),.N(N),.account(account),.pin(pin),.bal(bal),.ammount(ammount),
.OP(OP),.Y(Y),.balance(balance));

always #20 clk=!clk;
```

```
initial begin
        clk=1;
        rst=0;
        IC=1'b0;
        S=1'b0;
        N=1'b0;
        account=5'b00000;
        pin=4'b0000;
        bal= 3'b111;
        ammount=3'b000;
        OP=2'b00;
    #100
        rst=1;
        IC=1'b1;
        S=1'b1;
        N=1'b0;
        account=5'b11111;
        pin=4'b0000;
        bal= 3'b111;
        ammount=3'b111;
        OP=2'b10;
    #220
        rst=1;
        IC=1'b1;
        S=1'b1;
        N=1'b0;
        account=5'b11111;
        pin=4'b1111;
        bal= 3'b111;
        ammount=3'b111;
        OP=2'b10;
    #220
        rst=1;
        IC=1'b1;
        S=1'b1;
        N=1'b0;
        account=5'b11111;
        pin=4'b1111;
        bal= 3'b111;
        ammount=3'b011;
        OP=2'b01;
    #240 $stop;
    end
endmodule
```

Schematic :-

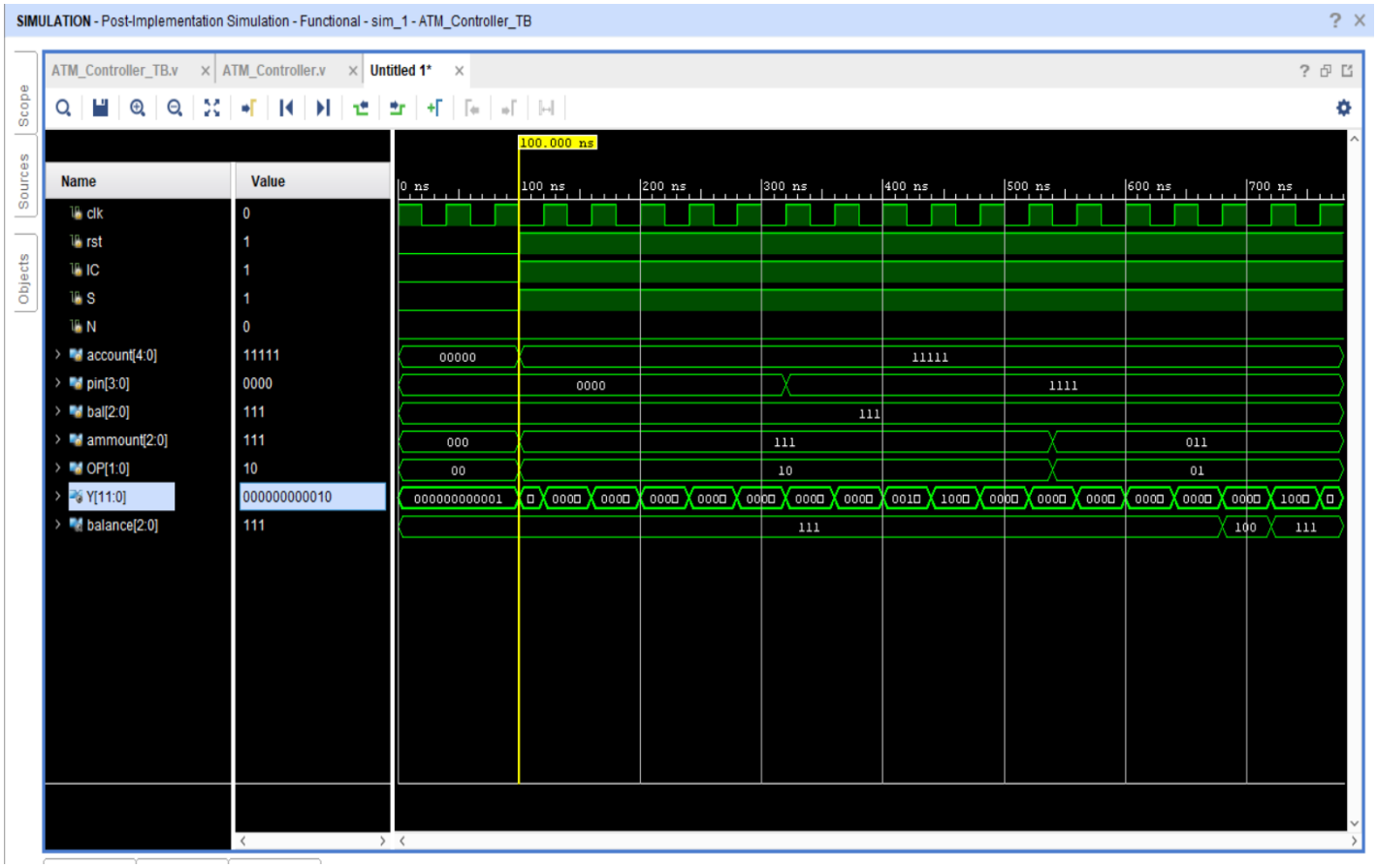


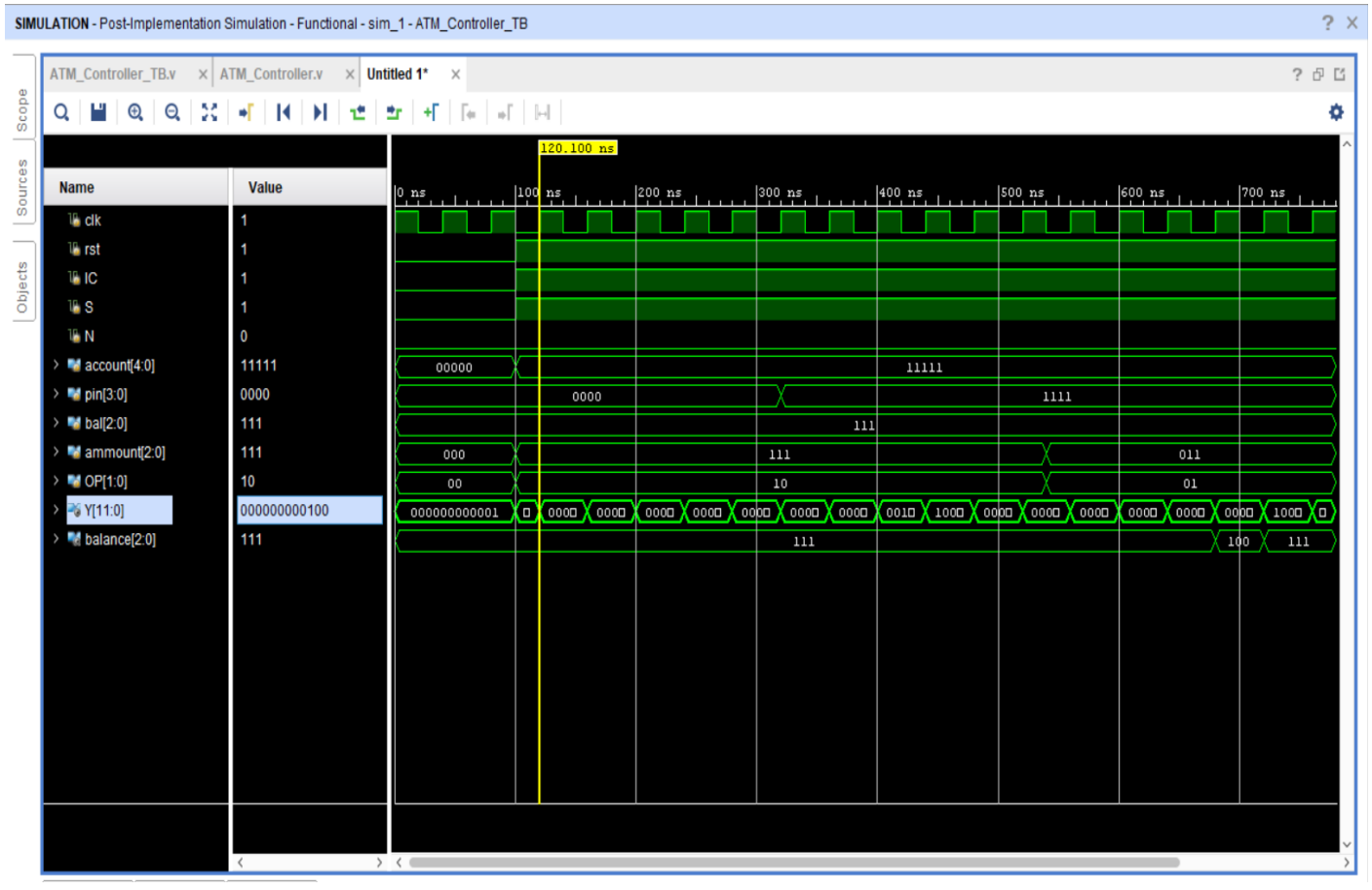
Simulation Result :-

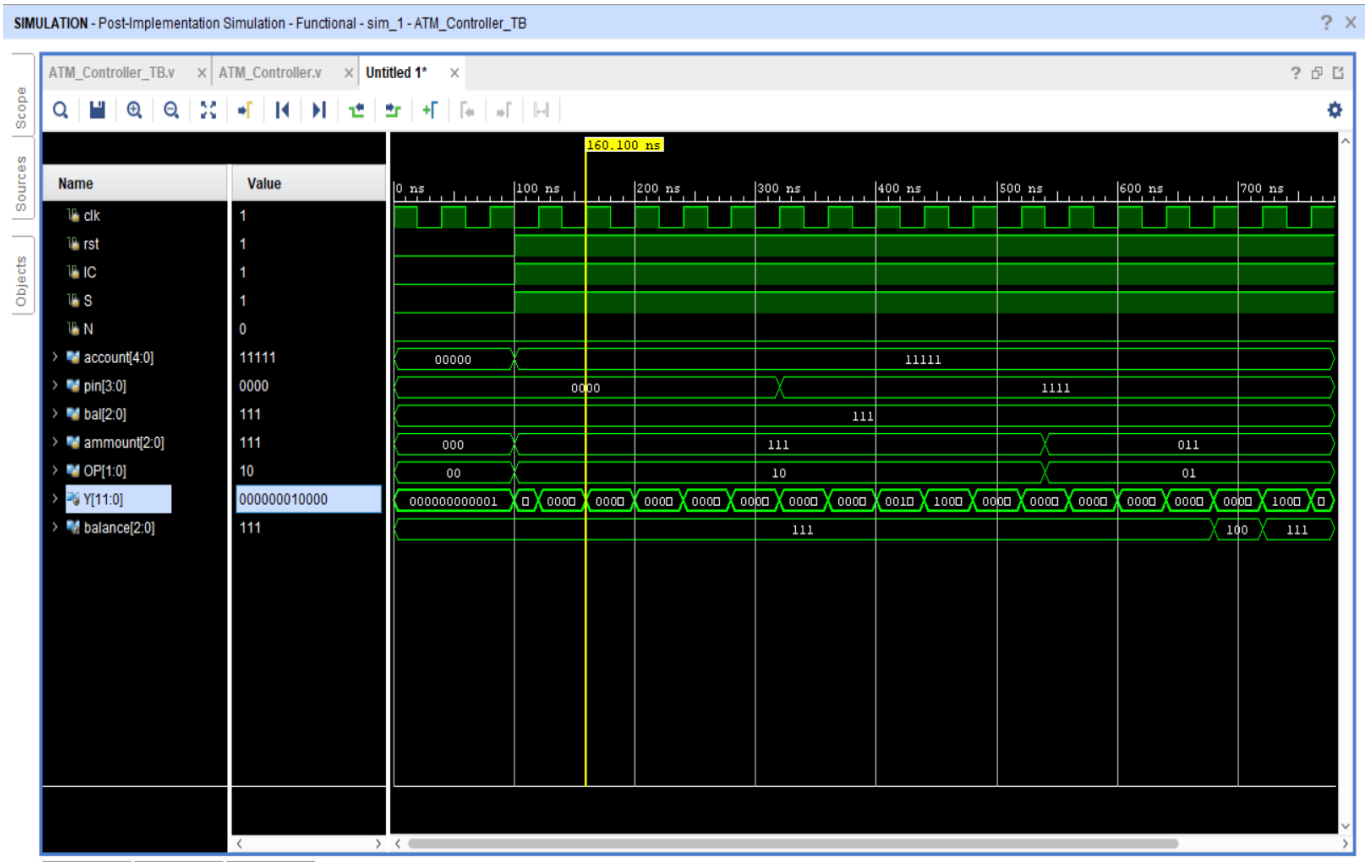


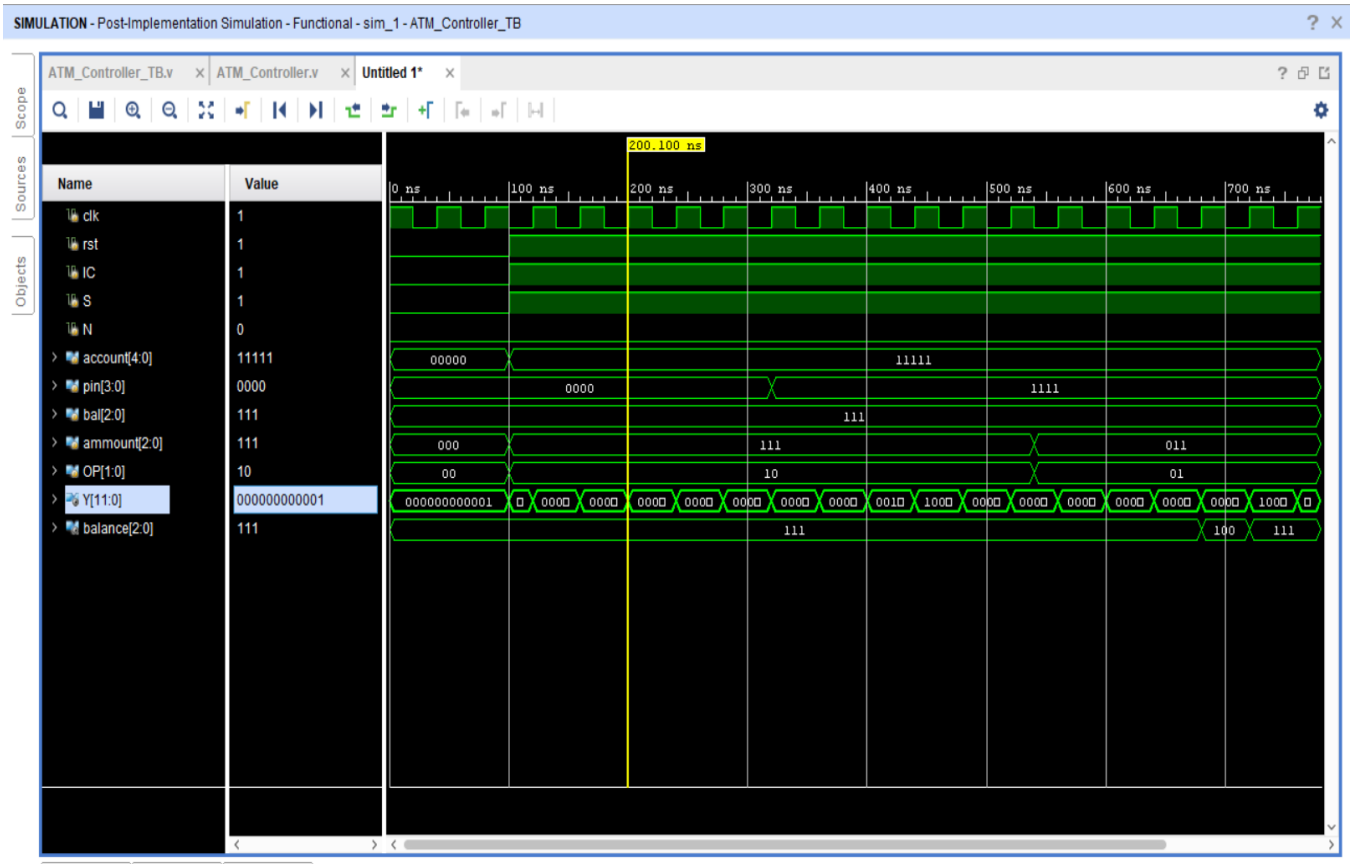
First test bench is written which sets ATM in ideal state i.e. Welcome page. This was done by assigning all the input value to 0 and reset 0. It can be seen above that initially ATM is in ideal State till 100 ns (000000000001 -> S0 is 1 that is Welcome state).

Case I – Wrong Pin code



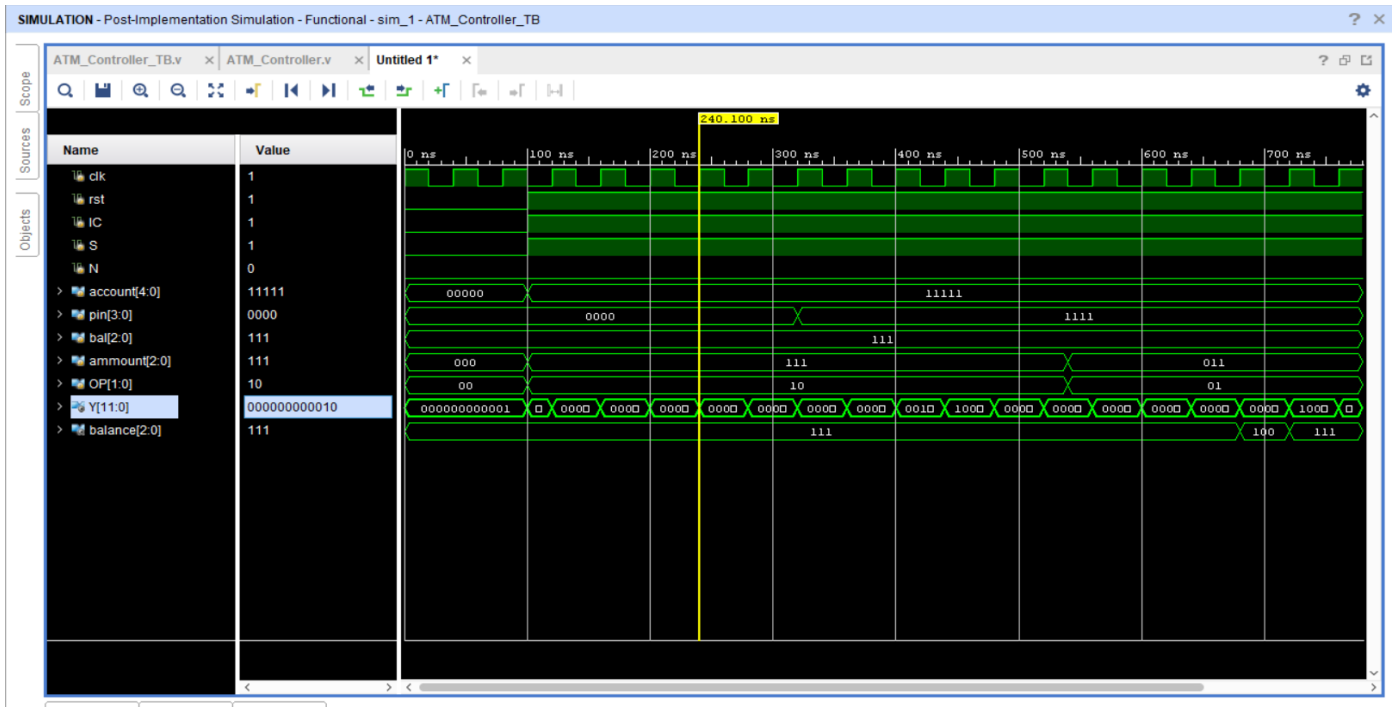


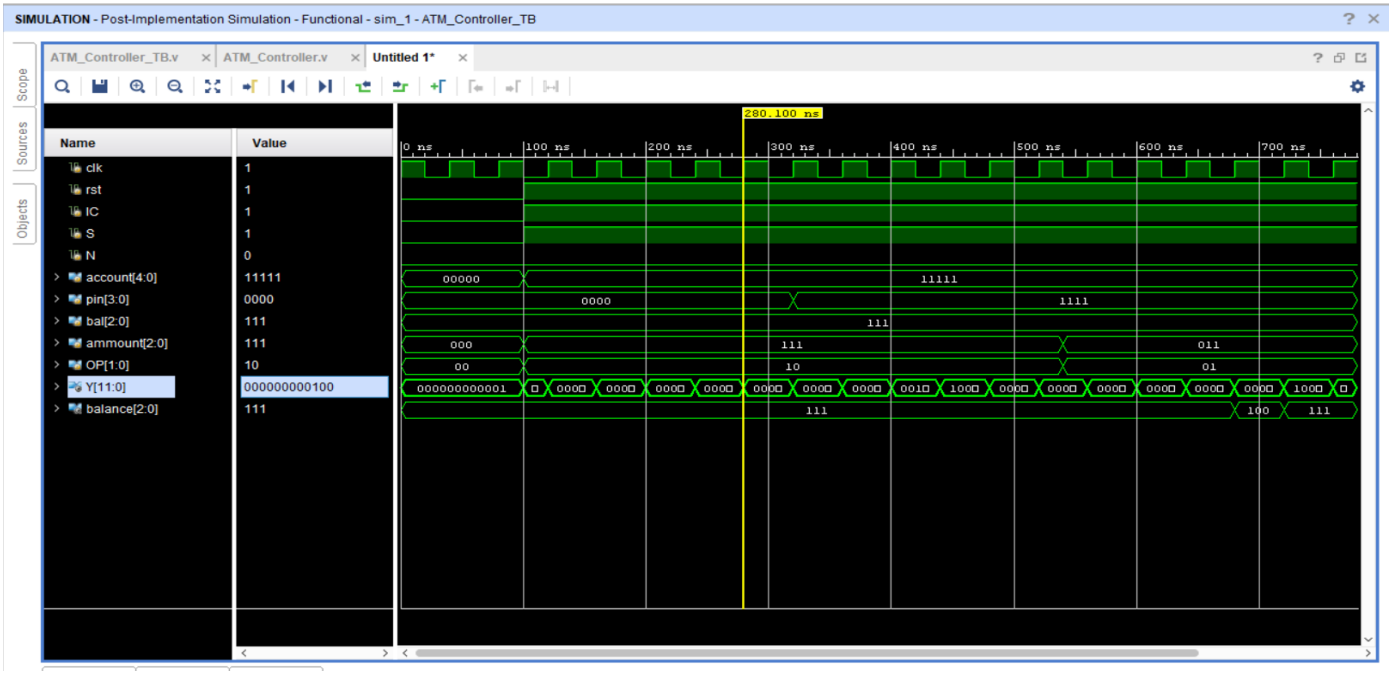


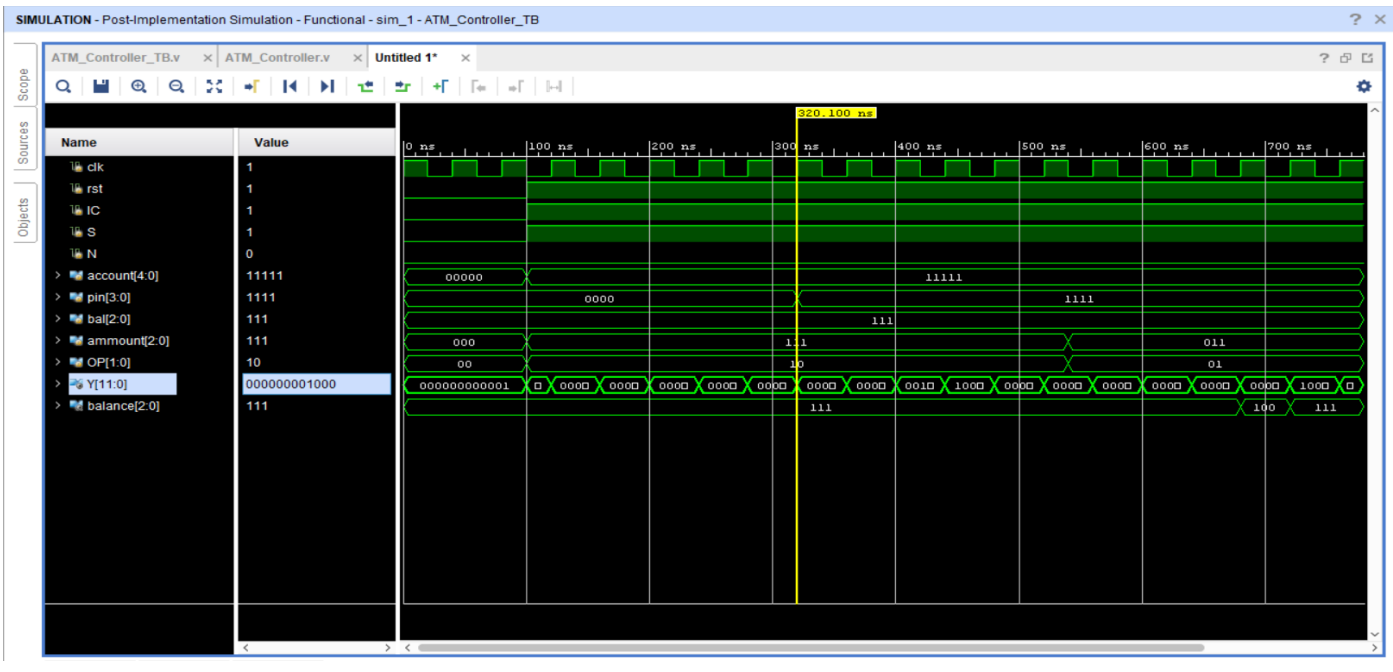


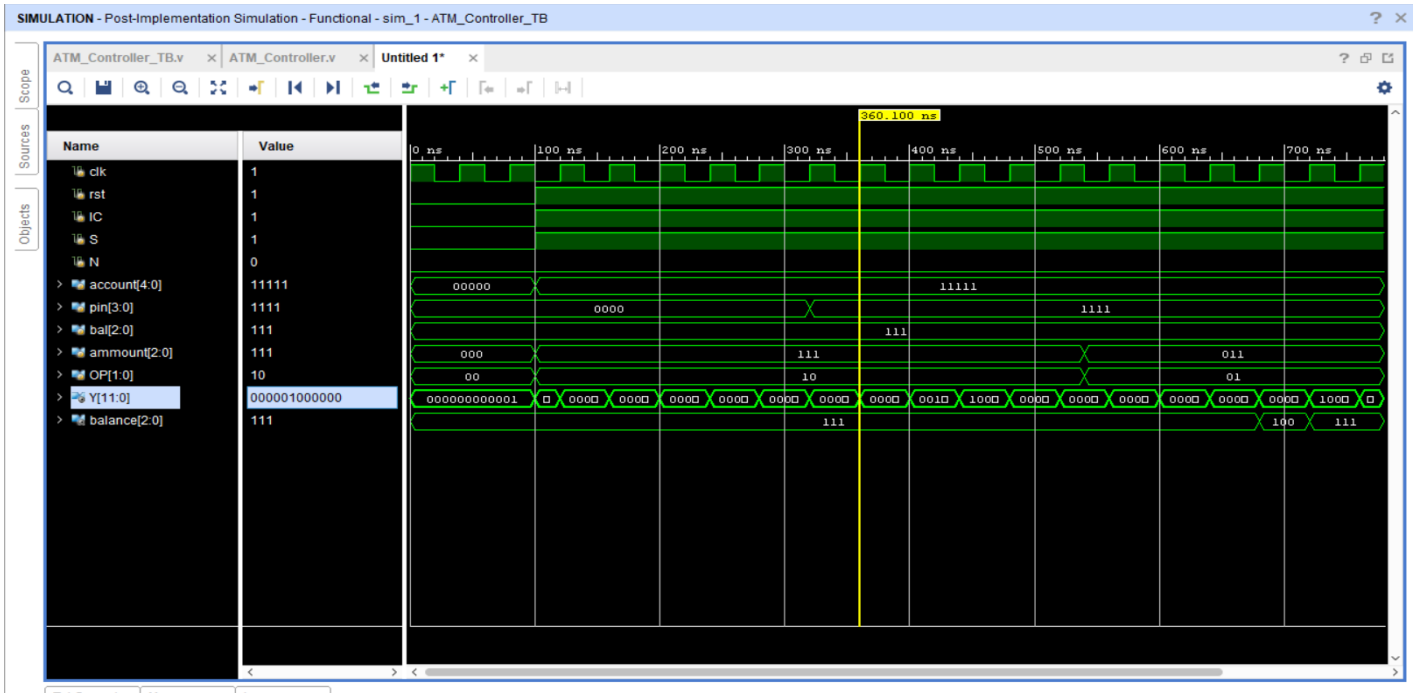
In this case after ideal case ATM card is detected (000000000010 -> S1 is 1 which is Insert card state means card is detected). Then account number is checked or scanned (000000000100 -> S2 is 1 which is scanning state means account number is matched). Then pin is matched, since pin is wrong i.e. 0000 not matched with 1111, it shows error (000000010000 -> S4 is 1 which is invalid state means wrong pin is detected). After showing error it goes to welcome page (000000000001 -> S0 is 1 which is welcome state means ATM is again in ideal state).

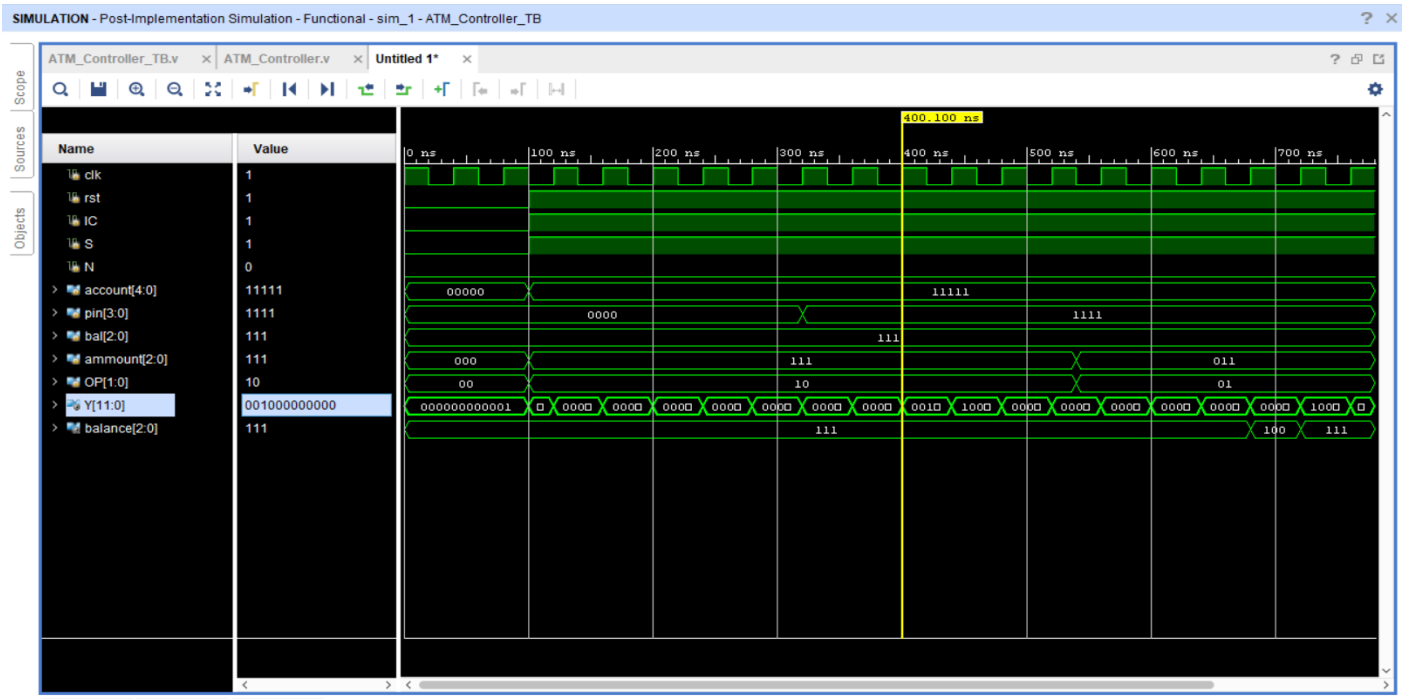
Case II – Balance Checking

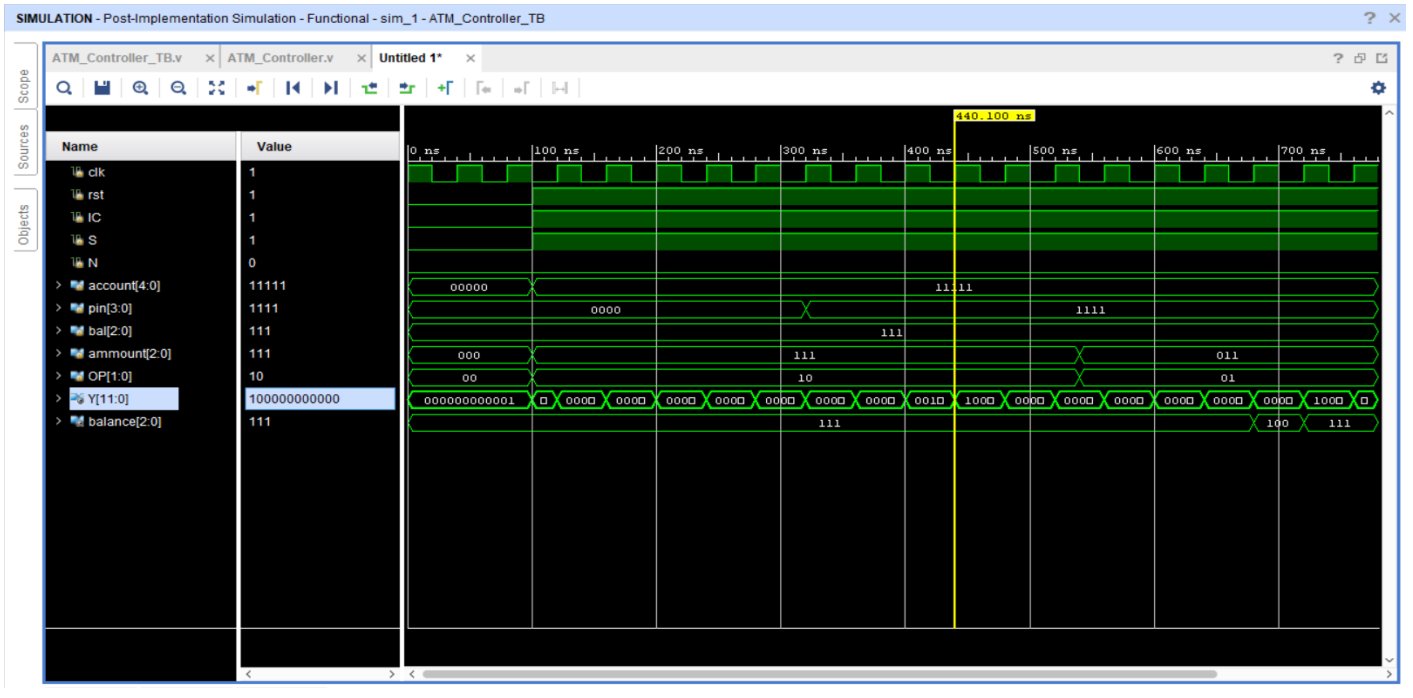


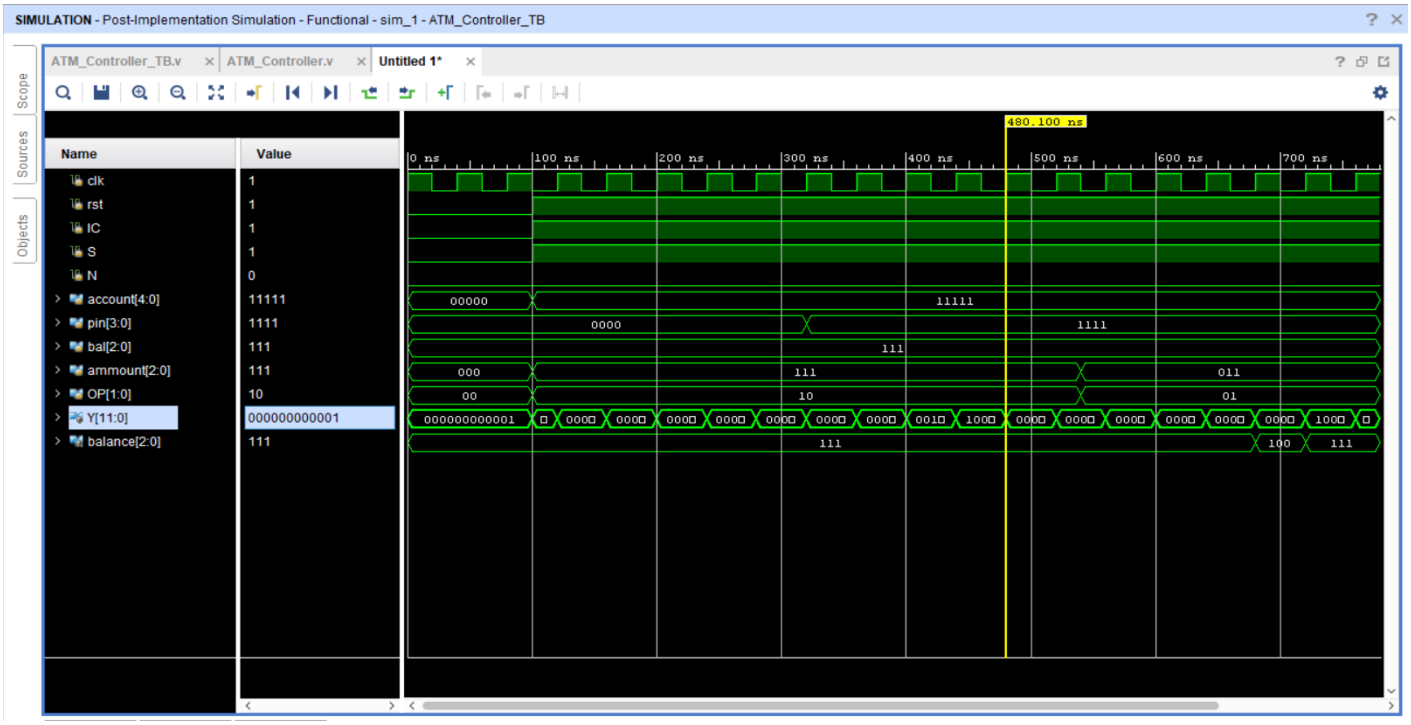





SIMULATION - Post-Implementation Simulation - Functional - sim_1 - ATM_Controller_TB
ATM_Controller_TB.v
ATM_Controller.v
Untitled 1*
Scope
Sources
Objects
Name
Value
clk 1
rst 1
IC 1
S 1
N 0
account[4:0] 11111
pin[3:0] 1111
bal[2:0] 111
ammount[2:0] 111
OP[1:0] 10
Y[11:0] 000001000000
balance[2:0] 111
360.100 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns

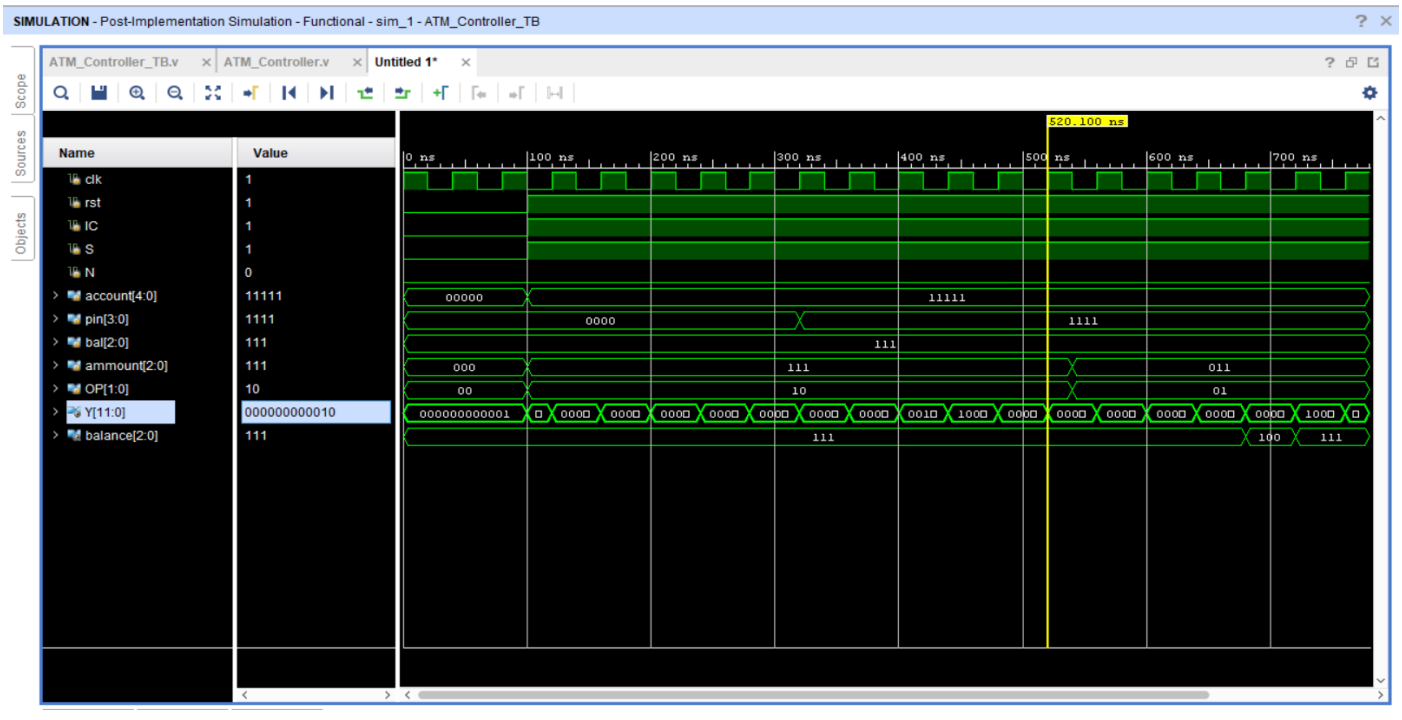

SIMULATION - Post-Implementation Simulation - Functional - sim_1 - ATM_Controller_TB
ATM_Controller_TB.v
ATM_Controller.v
Untitled 1*
Scope
Sources
Objects
Name
Value
clk 1
rst 1
IC 1
S 1
N 0
account[4:0] 11111
pin[3:0] 1111
bal[2:0] 111
ammount[2:0] 111
OP[1:0] 10
Y[11:0] 001000000000
balance[2:0] 111
400.100 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns


SIMULATION - Post-Implementation Simulation - Functional - sim_1 - ATM_Controller_TB
ATM_Controller_TB.v
ATM_Controller.v
Untitled 1*
Scope
Sources
Objects
Name
Value
clk 1
rst 1
IC 1
S 1
N 0
account[4:0] 11111
pin[3:0] 1111
bal[2:0] 111
ammount[2:0] 111
OP[1:0] 10
Y[11:0] 100000000000
balance[2:0] 111
440.100 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns

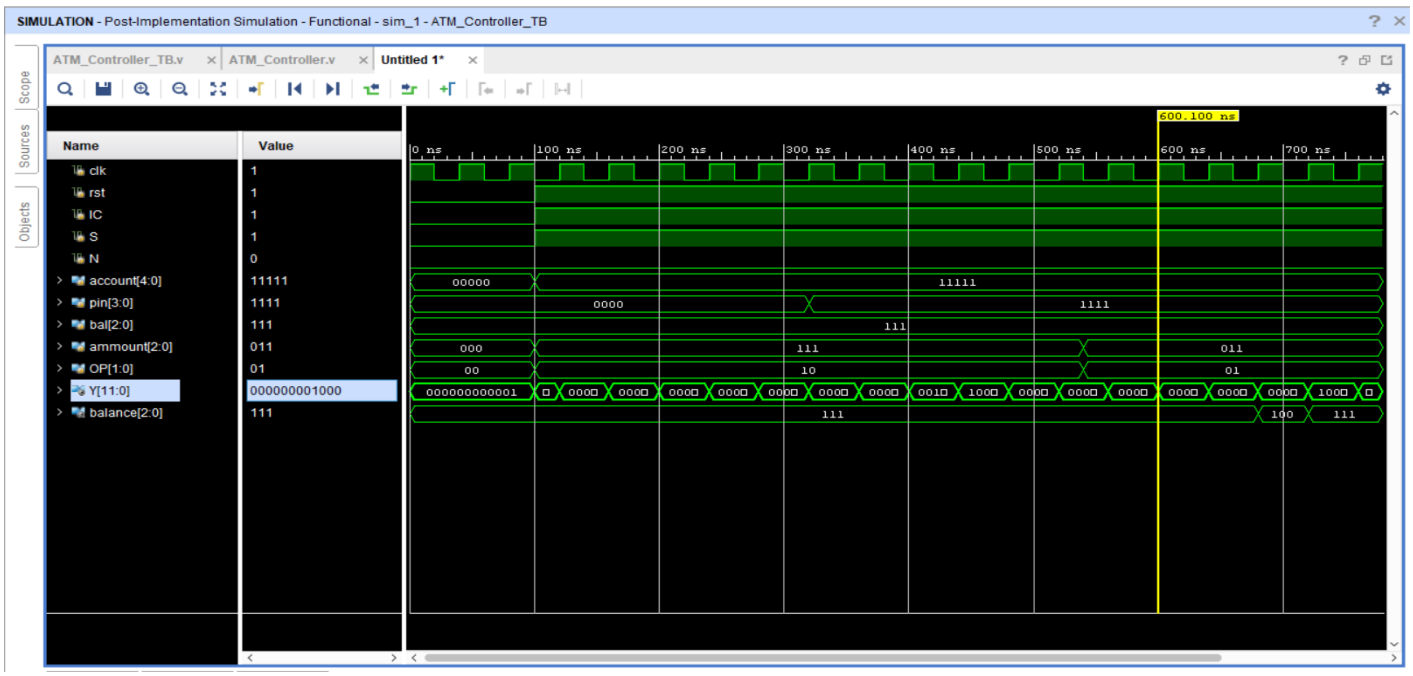


In this case after ideal case ATM card is detected (000000000010 -> S1 is 1 which is Insert card state means card is detected). Then account number is checked or scanned (000000000100 -> S2 is 1 which is scanning state means account number is matched). Then pin is matched, since pin is correct i.e. 1111 matched with 1111, it goes for transaction option (000000001000 -> S3 is 1 which is option for Txn state means pin is matched hence further transaction can be done). Now option for balance showing is selected i.e. 10 (000001000000 -> S6 is 1 which is balance state means balance is fetched). Then balance is shown on the screen (001000000000 -> S9 is 1 which is balance showing state means balance of that account is shown on the screen). Then it asks whether to go to further transaction or exit (100000000000 -> S11 is 1 which is next state means what to do next, transaction or exit). Since exit option is choose i.e. N=1 hence welcome appears (000000000001 -> S0 is 1 which is welcome state means ATM is again in ideal state).

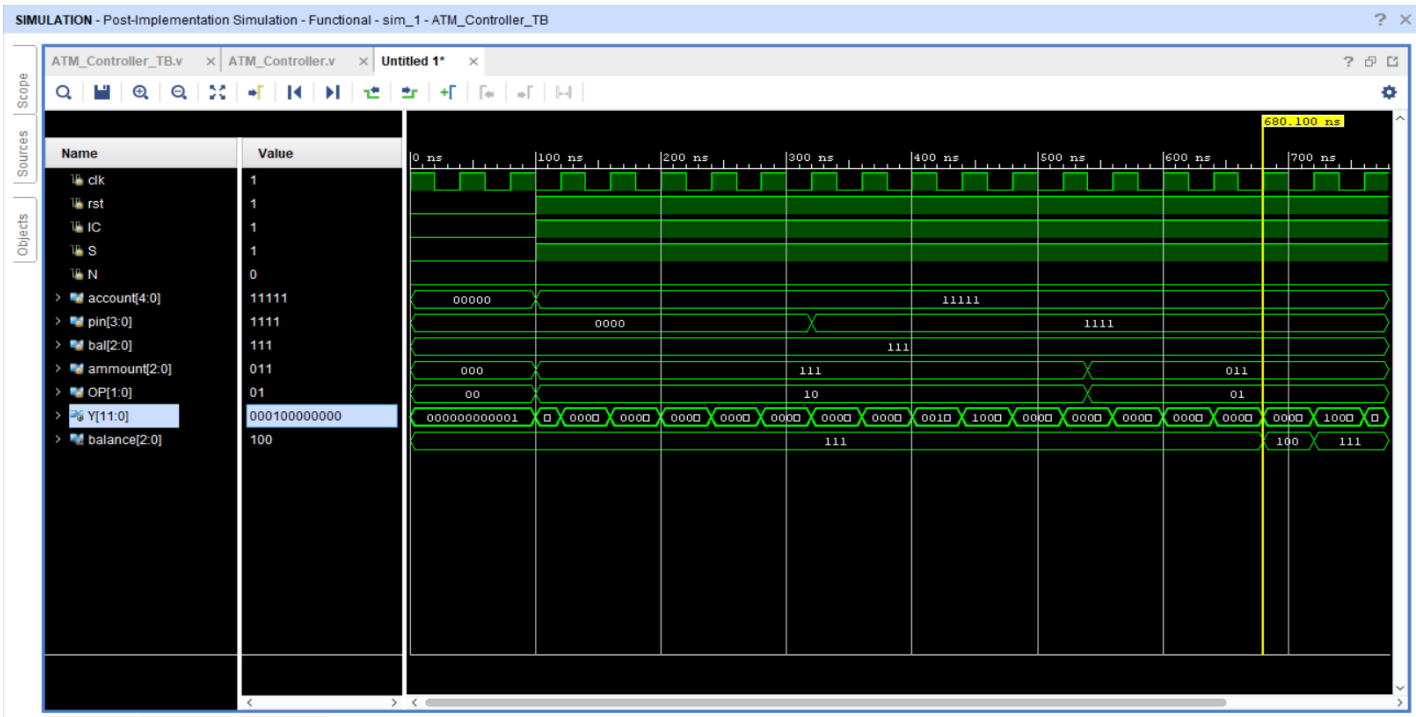
Case III – Amount withdrawing



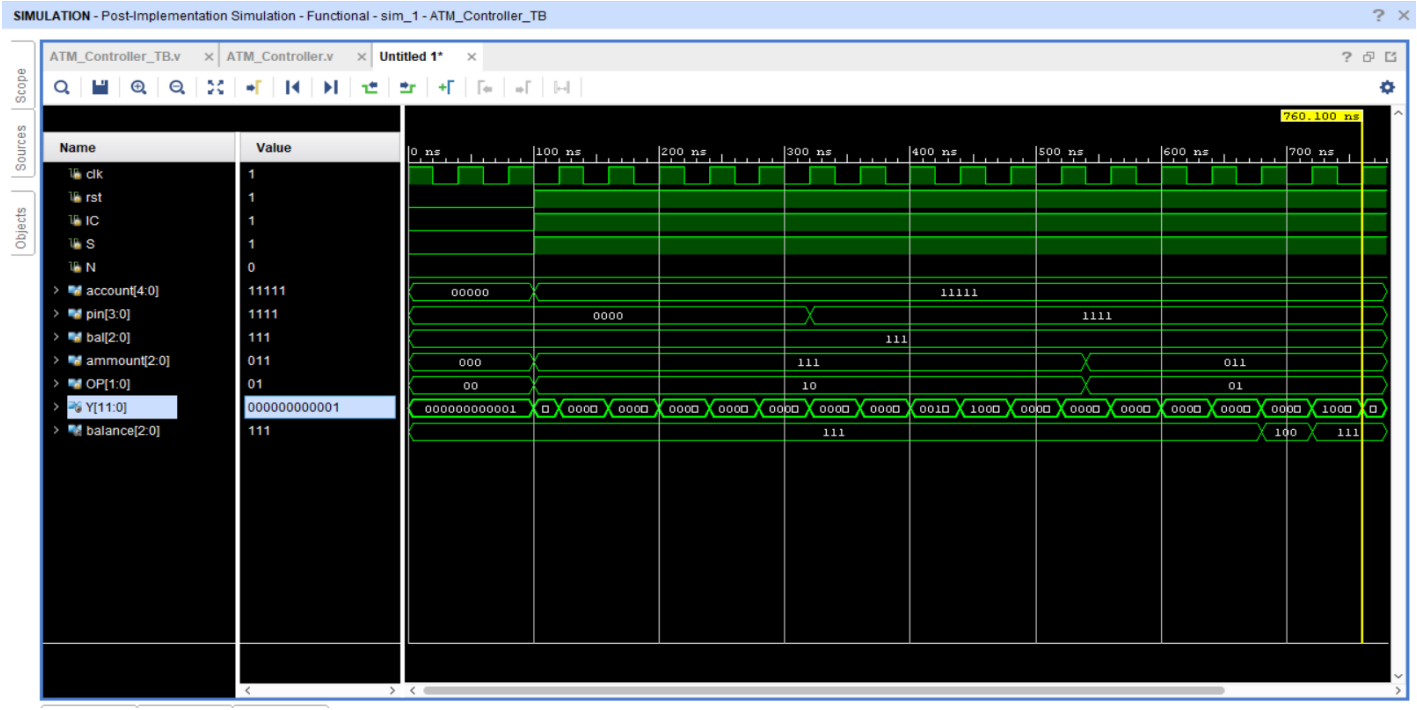

SIMULATION - Post-Implementation Simulation - Functional - sim_1 - ATM_Controller_TB
ATM_Controller_TB.v
ATM_Controller.v
Untitled 1*
Scope
Sources
Objects
Name
Value
clk 1
rst 1
IC 1
S 1
N 0
account[4:0] 11111
pin[3:0] 1111
bal[2:0] 111
ammount[2:0] 011
OP[1:0] 01
Y[11:0] 000000000100
balance[2:0] 111
560.100 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns


SIMULATION - Post-Implementation Simulation - Functional - sim_1 - ATM_Controller_TB
ATM_Controller_TB.v
ATM_Controller.v
Untitled 1*
Scope
Sources
Objects
Name
Value
clk 1
rst 1
IC 1
S 1
N 0
account[4:0] 11111
pin[3:0] 1111
bal[2:0] 111
ammount[2:0] 011
OP[1:0] 01
Y[11:0] 000000001000
balance[2:0] 111
600.100 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns


SIMULATION - Post-Implementation Simulation - Functional - sim_1 - ATM_Controller_TB
ATM_Controller_TB.v
ATM_Controller.v
Untitled 1*
Scope
Sources
Objects
Name
Value
clk 1
rst 1
IC 1
S 1
N 0
account[4:0] 11111
pin[3:0] 1111
bal[2:0] 111
ammount[2:0] 011
OP[1:0] 01
Y[11:0] 000000100000
balance[2:0] 111
640.100 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns


SIMULATION - Post-Implementation Simulation - Functional - sim_1 - ATM_Controller_TB
ATM_Controller_TB.v
ATM_Controller.v
Untitled 1*
Scope
Sources
Objects
Name
Value
clk 1
rst 1
IC 1
S 1
N 0
account[4:0] 11111
pin[3:0] 1111
bal[2:0] 111
ammount[2:0] 011
OP[1:0] 01
Y[11:0] 000100000000
balance[2:0] 100
680.100 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns


SIMULATION - Post-Implementation Simulation - Functional - sim_1 - ATM_Controller_TB
ATM_Controller_TB.v
ATM_Controller.v
Untitled 1*
Scope
Sources
Objects
Name
Value
clk 1
rst 1
IC 1
S 1
N 0
account[4:0] 11111
pin[3:0] 1111
bal[2:0] 111
ammount[2:0] 011
OP[1:0] 01
Y[11:0] 100000000000
balance[2:0] 111
720.100 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns


SIMULATION - Post-Implementation Simulation - Functional - sim_1 - ATM_Controller_TB
ATM_Controller_TB.v
ATM_Controller.v
Untitled 1*
Scope
Sources
Objects
Name
Value
clk 1
rst 1
IC 1
S 1
N 0
account[4:0] 11111
pin[3:0] 1111
bal[2:0] 111
ammount[2:0] 011
OP[1:0] 01
Y[11:0] 000000000001
balance[2:0] 111
760.100 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns

In this case after ideal case ATM card is detected (000000000010 -> S1 is 1 which is Insert card state means card is detected). Then account number is checked or scanned (000000000100 -> S2 is 1 which is scanning state means account number is matched). Then pin is matched, since pin is correct i.e. 1111 matched with 1111, it goes for transaction option (000000001000 -> S3 is 1 which is option for Txn state means pin is matched hence further transaction can be done). Now option for cash withdrawal is selected i.e. 01 (000000100000 -> S5 is 1 which is withdraw state means cash is withdrawing). Then Amount withdrawn is shown on the screen (000100000000 -> S8 is 1 which is amount withdrawn state mean balance is now 100 after withdrawing 011 from 111 which is shown on the screen). Then it asks whether to go to further transaction or exit (100000000000 -> S11 is 1 which is next state means what to do next, transaction or exit). Since exit option is choose i.e. N=1 hence welcome appears (000000000001 -> S0 is 1 which is welcome state means ATM is again in ideal state).

Conclusion :-

We successfully implemented the code for ATM controller using Moore State Machine in vivado 2017.4. ATM controller basically allows user to interact with the memory and also there is feature like pin matching which increases security level. Moreover it also makes the transition in accounts easier and secure.

In this Mini Project, concept of finite State Machine and its architecture designing were understood. Here procedure to develop State Machine specially Moore state Machine diagram of a system through system level analysis is understood. After analysing Moore State Diagram whole ATM controller system is modelled in Verilog and verified all the appropriate scenarios.

THANK YOU