INTEL UNNATI INDUSTRIAL TRAINING-SUMMER 2023

Design and Implementation of Any Time Electricity Bill Payment (ATP) Machine Controller

Team Spark

Name 1: Surepally Vishnnu Vardhini

Name 2: Aditya Kumar Sharma

Institution: Nitte Meenakshi Institute of Technology, Bangalore.

Institution Mentor:

Dr. Shashidhara K S, Professor, Dept of ECE.

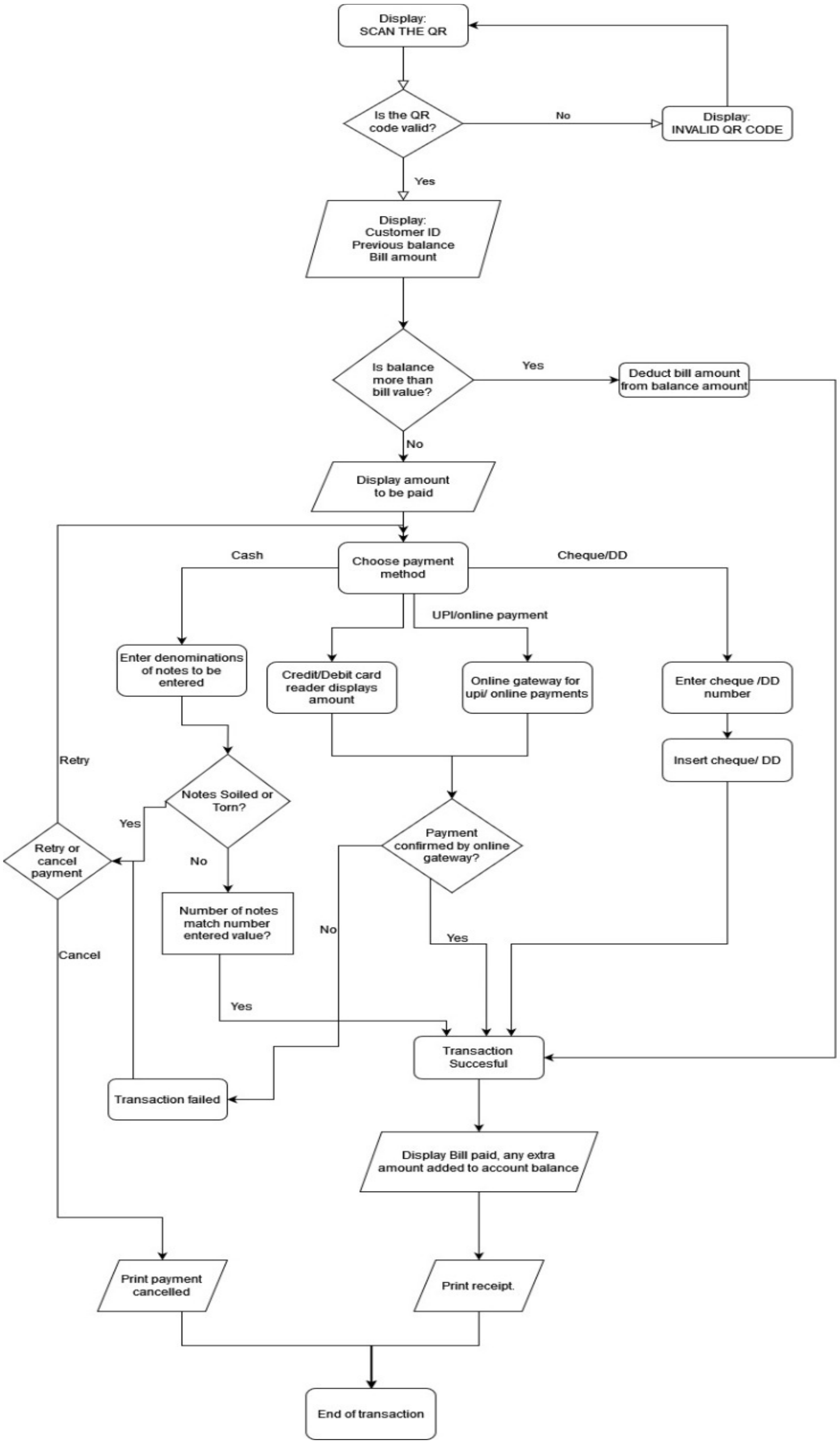
Industry Mentor:

Mr. Abhishek Nandy, Intel Unnati.

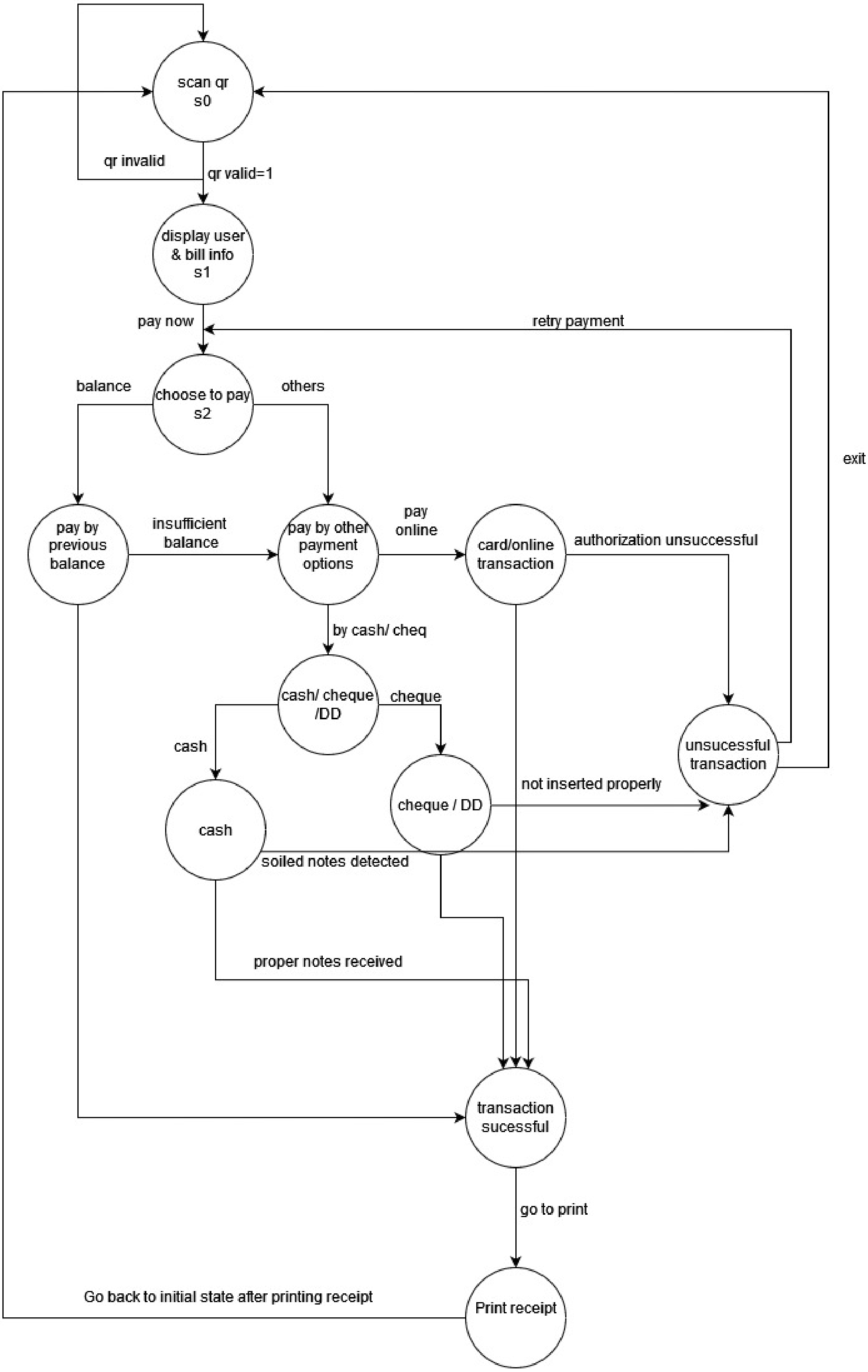
Contents

In this project, we have tried to design and implement an Any Time Electricity Bill Payment (ATP) Machine Controller using FPGA.

I. BLOCK DIAGRAM


Display:
SCAN THE QR
Is the QR code valid?
No
Display: INVALID QR CODE
Yes
Display: Customer ID Previous balance Bill amount
Is balance more than bill value?
Yes
Deduct bill amount from balance amount
No
Display amount to be paid
Choose payment method
Cash
Cheque/DD
UPI/online payment
Enter denominations of notes to be entered
Credit/Debit card reader displays amount
Online gateway for upi/ online payments
Enter cheque /DD number
Insert cheque/ DD
Retry
Notes Soiled or Torn?
Yes
Retry or cancel payment
No
Payment confirmed by online gateway?
Number of notes match number entered value?
No
Yes
Cancel
Yes
Transaction Succesful
Transaction failed
Display Bill paid, any extra amount added to account balance
Print payment cancelled
Print receipt.
End of transaction

II. FSM


scan qr
s0
qr invalid
qr valid=1
display user
& bill info
s1
pay now
retry payment
choose to pay
s2
balance
others
exit
pay by
previous
balance
insufficient
balance
pay by other
payment
options
pay
online
card/online
transaction
authorization unsuccessful
by cash/ cheq
cash/ cheque
/DD
cheque
cash
unsucessful
transaction
cheque / DD
not inserted properly
cash
soiled notes detected
proper notes received
transaction
sucessful
go to print
Go back to initial state after printing receipt
Print receipt

III. APPROACH

1.In order to solve this problem, we drew a flowchart first which roughly summarizes our agenda and gives us a brief on methodology and how to work on it.

2. Users can pay through existing Balance, Cash, Card, Online Transactions, DD and Cheque.

3.The machine controller uses QR Scanner in order to initiate the process.

4.After the QR gets scanned, the machine controller displays Customer ID, Existing Balance, Amount to be paid and Pay option.

5.If Balance is greater than the Bill amount then the amount gets deducted automatically else the machine controller moves further and displays other payment options.

6.The machine controller displays QR for UPI payments, redirects to Online Gateway for Online payments, displays Insert DD or Cheque option for DD or Cheque and displays Insert Cash option for Cash mode of payment.

7.The machine controller does not accept soiled or torn notes.

8.If the transaction fails then the machine controller redirects to Retry Payment option.

9.If the transaction is successful then the machine controller displays Bill paid successfully and Extra amount to be added option.

10. If the user wants to add balance, then the machine controller redirects the user to choose payment mode option else the receipt is printed and transaction ends.

IV. METHODOLOGY/FLOW

1.Intel Quartus Prime Lite Software has been used to obtain results.

2.A Verilog HDL code and the test bench has been compiled successfully without errors to obtain the State Diagram of the given problem statement using block diagram as reference as shown in figure below.

Verilog code:

```
module atp (output reg [3:0] data_out,input in0,in1,in2,in3,in4,in5,in6,in7,in8,in9,in10,in11,
input clk,rst);

 parameter s0= 4'b0000; //scanqr

 parameter s1= 4'b0001; //data display

 parameter s2= 4'b0010; //balance or oth
 parameter s3= 4'b0011; // pay by bal
 parameter s4= 4'b0100; // pay by oths
 parameter s5= 4'b0101; //pay caash or dd or cheq
 parameter s6= 4'b0110; // card or upi
 parameter s7= 4'b0111; //cash
 parameter s8= 4'b1000; //dd/chq
 parameter s9= 4'b1001; //FAIL
 parameter s10=4'b1010; // success
 parameter s11=4'b1011; // print receipt

 reg [1:0] c_state,n_state;

 always @(posedge clk)
  begin
 if (rst==1)
 begin
   c_state=0;
            n_state=0;
            end
else
    c_state=n_state;

case(c_state)

 s0: begin
 if (in0==0) n_state = s0;
 else if ( in0==1) n_state = s1;
 end

 s1: begin
 if (in1==0) n_state = s0;
 else if ( in1==1) n_state = s2;
 end

 s2: begin
 if ( in2==0) n_state = s3;

 else if ( in2==1) n_state = s4;

end

s3: begin
if (in3==0) n_state = s4;
else if ( in3==1) n_state = s10;
end

s4: begin
if (in4==0) n_state = s5;

else if (in4==1)n_state = s6;

end

s5: begin
if (in5==0) n_state = s7;
else if (in5==1)n_state = s8;
end
s6: begin
if (in6==0) n_state = s9;
else if (in6==1)n_state = s10;
```

```
end
s7: begin
if (in7==0) n_state = s9;
else if (in7==1)n_state = s10;
end
s8: begin
if (in8==0) n_state = s9;
else if (in8==1)n_state = s10;
end
s9: begin
if (in9==0) n_state = s4;
else if (in9==1)n_state = s0;
end
s10: begin
if (in10==0) n_state = s11;
else if (in10==1)n_state = s11;
end
s11: begin
if (in11==0) n_state = s0;
else if (in11==1)n_state = s0;
end
endcase
end
endmodule
```

V

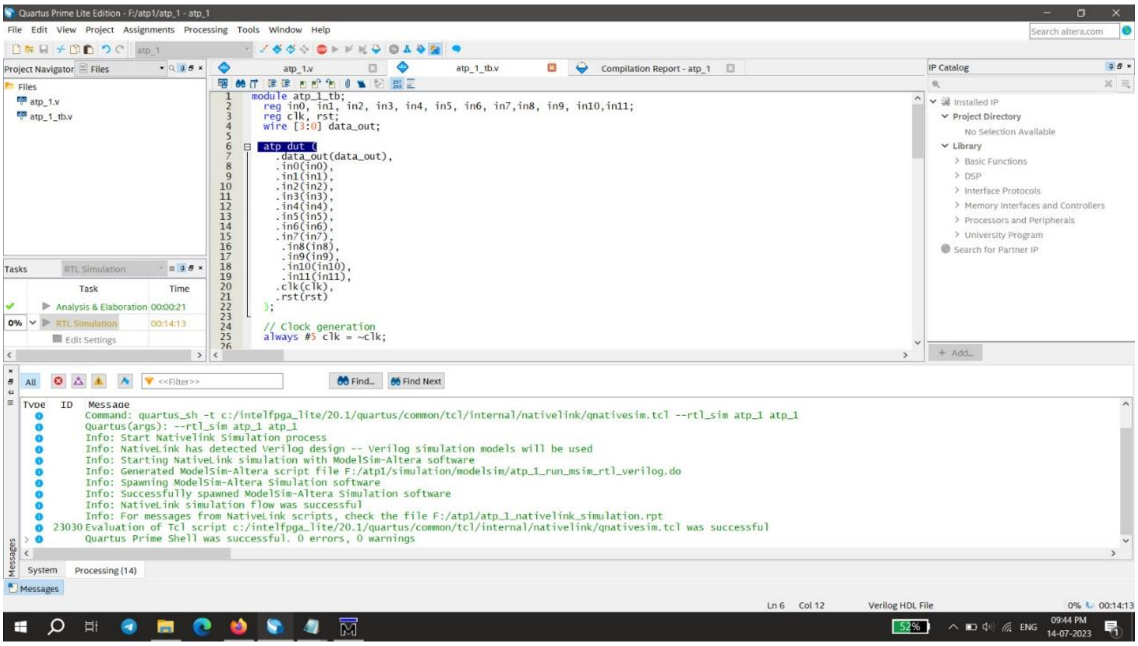
Verilog testbench:

```
module atp_tb;
  reg in0, in1, in2, in3, in4, in5, in6, in7,in8, in9, in10,in11;
  reg clk, rst;
  wire [3:0] data_out;

  atp (
    .data_out(data_out),
    .in0(in0),
    .in1(in1),
    .in2(in2),
    .in3(in3),
    .in4(in4),
    .in5(in5),
    .in6(in6),
    .in7(in7),
            .in8(in8),
            .in9(in9),
            .in10(in10),
            .in11(in11),
    .clk(clk),
    .rst(rst)
  );

  // Clock generation
  always #5 clk = ~clk;

  // Test stimulus
  initial begin
    // Initialize inputs
```

```
in0 = 0;
in1 = 0;
in2 = 0;
in3 = 0;
in4 = 0;
in5 = 0;
in6 = 0;
in7 = 0;
rst = 1;
clk = 0;

// Reset sequence
#10 rst = 0;
#20 rst = 1;

// Test case 1: Transition from S0 to S1
#10 in0 = 1;
#10 in1 = 1;
#10 in0 = 0;
#10 in1 = 0;

// Test case 2: Transition from S1 to S2
#10 in1 = 1;
#10 in2 = 1;
#10 in1 = 0;
#10 in2 = 0;
//3
#10 in2 = 1;
#10 in3 = 1;
#10 in2 = 0;
```

```
   #10 in3 = 0;
            //4
#10 in3 = 1;
  #10 in4 = 1;
  #10 in3 = 0;
  #10 in4 = 0;
            //5
#10 in4 = 1;
  #10 in5 = 1;
   #10 in4 = 0;
   #10 in5 = 0;
//6
#10 in5 = 1;
   #10 in6 = 1;
   #10 in5 = 0;
   #10 in6 = 0;
//7
#10 in6 = 1;
  #10 in7 = 1;
   #10 in6 = 0;
  #10 in7 = 0;
//8
#10 in7 = 1;
  #10 in8 = 1;
  #10 in7 = 0;
   #10 in8 = 0;
//9
#10 in8 = 1;
  #10 in9 = 1;
  #10 in8 = 0;
  #10 in9 = 0;
//10
#10 in9 = 1;
  #10 in10 = 1;
  #10 in9= 0;
  #10 in10 = 0;
//11
#10 in10= 1;
 #10 in11= 1;
 #10 in10 = 0;
 #10 in11= 0;

  // End simulation
  #10 $finish;
end
endmodule
```

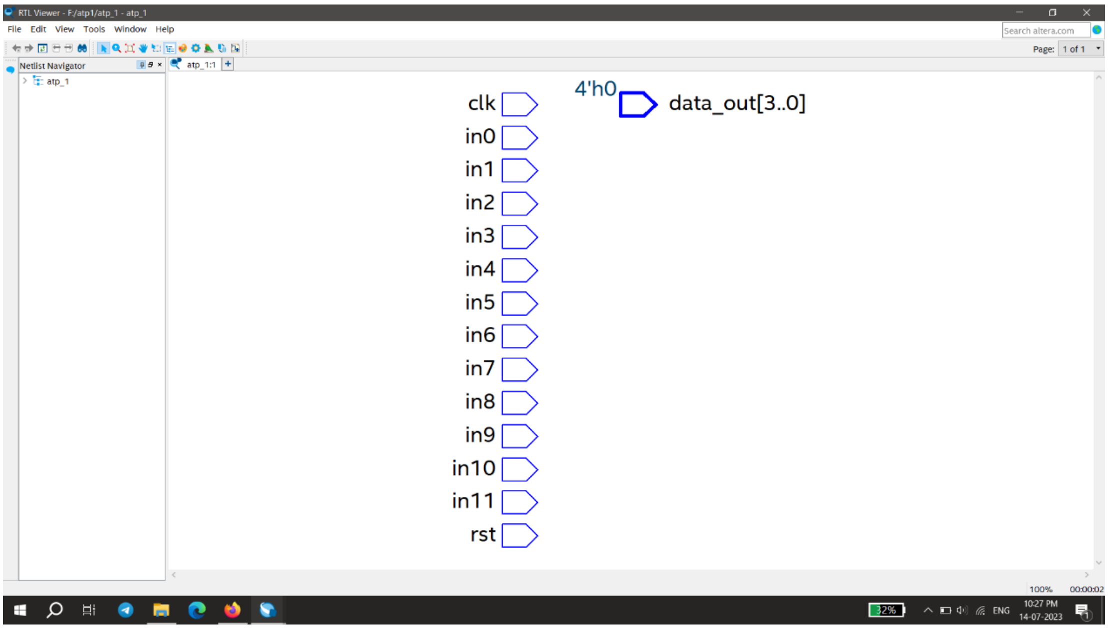
Compilation output



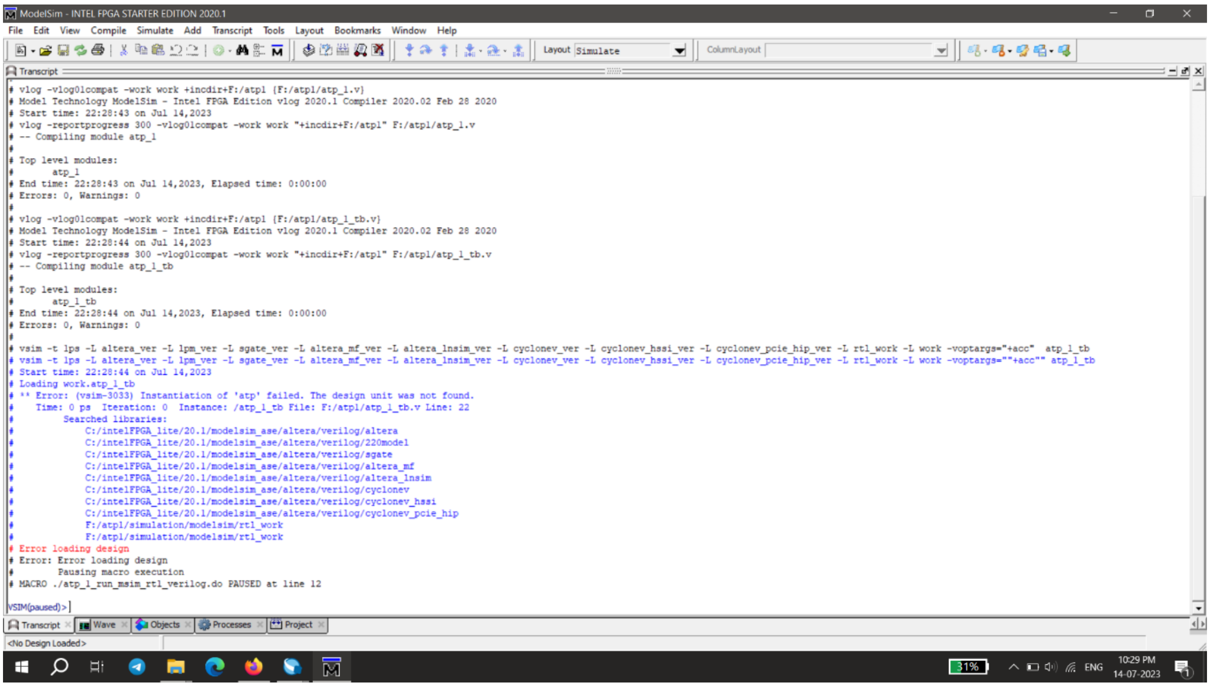
```
module atp_1_tb;
  reg in0, in1, in2, in3, in4, in5, in6, in7,in8, in9, in10,in11;
  reg clk, rst;
  wire [3:0] data_out;

 atp dut (
    .data_out(data_out),
    .in0(in0),
    .in1(in1),
    .in2(in2),
    .in3(in3),
    .in4(in4),
    .in5(in5),
    .in6(in6),
    .in7(in7),
     .in8(in8),
     .in9(in9),
     .in10(in10),
     .in11(in11),
    .clk(clk),
    .rst(rst)
  );

  // Clock generation
  always #5 clk = ~clk;
```

```
Type  ID    Message
            Command: quartus_sh -t c:/intelfpga_lite/20.1/quartus/common/tcl/internal/nativelink/qnativesim.tcl --rtl_sim atp_1 atp_1
            Quartus(args): --rtl_sim atp_1 atp_1
            Info: Start Nativelink Simulation process
            Info: NativeLink has detected Verilog design -- Verilog simulation models will be used
            Info: Starting NativeLink simulation with ModelSim-Altera software
            Info: Generated ModelSim-Altera script file F:/atp1/simulation/modelsim/atp_1_run_msim_rtl_verilog.do
            Info: Spawning ModelSim-Altera Simulation software
            Info: Successfully spawned ModelSim-Altera Simulation software
            Info: NativeLink simulation flow was successful
            Info: For messages from NativeLink scripts, check the file F:/atp1/atp_1_nativelink_simulation.rpt
      23030 Evaluation of Tcl script c:/intelfpga_lite/20.1/quartus/common/tcl/internal/nativelink/qnativesim.tcl was successful
            Quartus Prime Shell was successful. 0 errors, 0 warnings
```

RTL View:



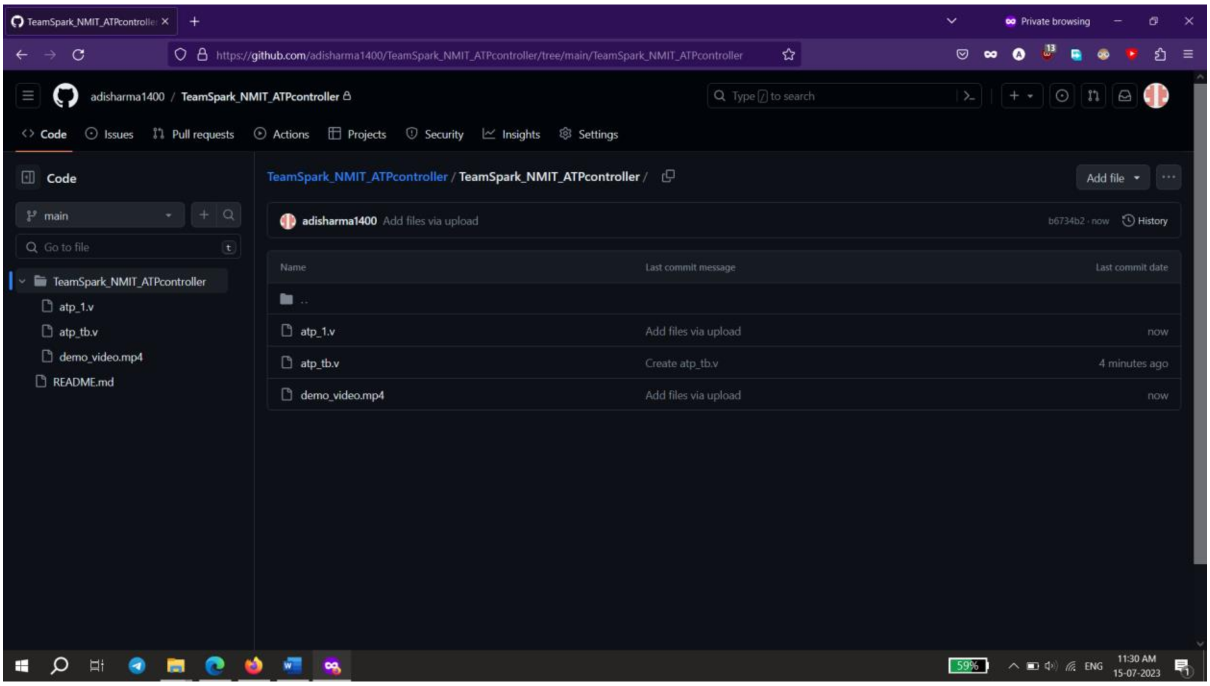
Error in Multisim:


ModelSim - INTEL FPGA STARTER EDITION 2020.1
File Edit View Compile Simulate Add Transcript Tools Layout Bookmarks Window Help
Layout Simulate
ColumnLayout
Transcript

```
# vlog -vlog01compat -work work +incdir+F:/atpl {F:/atpl/atp_1.v}
# Model Technology ModelSim - Intel FPGA Edition vlog 2020.1 Compiler 2020.02 Feb 28 2020
# Start time: 22:28:43 on Jul 14,2023
# vlog -reportprogress 300 -vlog01compat -work work "+incdir+F:/atpl" F:/atpl/atp_1.v
# -- Compiling module atp_1
#
# Top level modules:
#       atp_1
# End time: 22:28:43 on Jul 14,2023, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
#
# vlog -vlog01compat -work work +incdir+F:/atpl {F:/atpl/atp_1_tb.v}
# Model Technology ModelSim - Intel FPGA Edition vlog 2020.1 Compiler 2020.02 Feb 28 2020
# Start time: 22:28:44 on Jul 14,2023
# vlog -reportprogress 300 -vlog01compat -work work "+incdir+F:/atpl" F:/atpl/atp_1_tb.v
# -- Compiling module atp_1_tb
#
# Top level modules:
#       atp_1_tb
# End time: 22:28:44 on Jul 14,2023, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
#
# vsim -t 1ps -L altera_ver -L lpm_ver -L sgate_ver -L altera_mf_ver -L altera_lnsim_ver -L cyclonev_ver -L cyclonev_hssi_ver -L cyclonev_pcie_hip_ver -L rtl_work -L work -voptargs="+acc"  atp_1_tb
# vsim -t 1ps -L altera_ver -L lpm_ver -L sgate_ver -L altera_mf_ver -L altera_lnsim_ver -L cyclonev_ver -L cyclonev_hssi_ver -L cyclonev_pcie_hip_ver -L rtl_work -L work -voptargs=""+acc"" atp_1_tb
# Start time: 22:28:44 on Jul 14,2023
# Loading work.atp_1_tb
# ** Error: (vsim-3033) Instantiation of 'atp' failed. The design unit was not found.
#    Time: 0 ps  Iteration: 0  Instance: /atp_1_tb File: F:/atpl/atp_1_tb.v Line: 22
#         Searched libraries:
#             C:/intelFPGA_lite/20.1/modelsim_ase/altera/verilog/altera
#             C:/intelFPGA_lite/20.1/modelsim_ase/altera/verilog/220model
#             C:/intelFPGA_lite/20.1/modelsim_ase/altera/verilog/sgate
#             C:/intelFPGA_lite/20.1/modelsim_ase/altera/verilog/altera_mf
#             C:/intelFPGA_lite/20.1/modelsim_ase/altera/verilog/altera_lnsim
#             C:/intelFPGA_lite/20.1/modelsim_ase/altera/verilog/cyclonev
#             C:/intelFPGA_lite/20.1/modelsim_ase/altera/verilog/cyclonev_hssi
#             C:/intelFPGA_lite/20.1/modelsim_ase/altera/verilog/cyclonev_pcie_hip
#             F:/atpl/simulation/modelsim/rtl_work
#             F:/atpl/simulation/modelsim/rtl_work
# Error loading design
# Error: Error loading design
#        Pausing macro execution
# MACRO ./atp_1_run_msim_rtl_verilog.do PAUSED at line 12

VSIM(paused)>
```

V. GITHUB FILE SCREENSHOT

GitHub repo Link:
https://github.com/adisharma1400/TeamSpark_NMIT_ATPcontroller/tree/main/TeamSpark_NMIT_ATPcontroller


TeamSpark_NMIT_ATPcontrolle
Private browsing
https://github.com/adisharma1400/TeamSpark_NMIT_ATPcontroller/tree/main/TeamSpark_NMIT_ATPcontroller
adisharma1400 / TeamSpark_NMIT_ATPcontroller
Type / to search
Code
Issues
Pull requests
Actions
Projects
Security
Insights
Settings
Code
main
Go to file
t
TeamSpark_NMIT_ATPcontroller
atp_1.v
atp_tb.v
demo_video.mp4
README.md
Add file
History
59%
ENG
11:30 AM
15-07-2023

TeamSpark_NMIT_ATPcontroller / TeamSpark_NMIT_ATPcontroller /

adisharma1400 Add files via upload b6734b2 · now

Name	Last commit message	Last commit date
..		
atp_1.v	Add files via upload	now
atp_tb.v	Create atp_tb.v	4 minutes ago
demo_video.mp4	Add files via upload	now

VI. RESULTS AND SUMMARIZATIONS

The main file and the test bench have been compiled without any errors. As observed due to some technical issue the RTL simulations could not be performed. Trying to troubleshoot this issue could not be done and there was also no updated documentation for the shown error.

VII. REFERENCES

1. State Machine Coding Style for Synthesis..............Clifford E. Cummings (1998)
2. System Machine Design Techniques for Verilog HDL and VHDL.......Steve Golson.