AY 2022-23
PROJECT REPORT
ON

Vending Machine Module

By
ANUJ KUMAR

Abstract

This report outlines the design, implementation, and functionality of a Verilog-based Vending Machine Module. The module simulates a vending machine that processes transactions for items priced at 5 or 10 units. Details regarding its functionality, design methodology, simulation results, and potential applications are presented.

TABLE OF CONTENTS

Project Overview

1.1 Objective

The objective of this project was to create a hardware module in Verilog that emulates the behavior of a vending machine, allowing transactions for items with 5 or 10 unit prices.

1.2 Design Specifications

Inputs:

clk: Clock signal controlling module operations.

rst: Reset signal for initialization.

in: Input determining transaction value (01 for 5-unit item, 10 for 10-unit item).

Outputs:

out: Signal indicating transaction success (1) or failure (0).

change: Signal representing given change (2'b00 for no change, 2'b01 for 5-unit change, 2'b10 for 10-unit change).

Methodology

2.1 Module Design

The module was designed using Verilog and structured as a finite state machine (FSM) with states S0, S1, and S2. State transitions were governed by the input in, triggering appropriate changes in out and change.

2.2 Implementation

Parameters S0, S1, and S2 were assigned to represent distinct states.

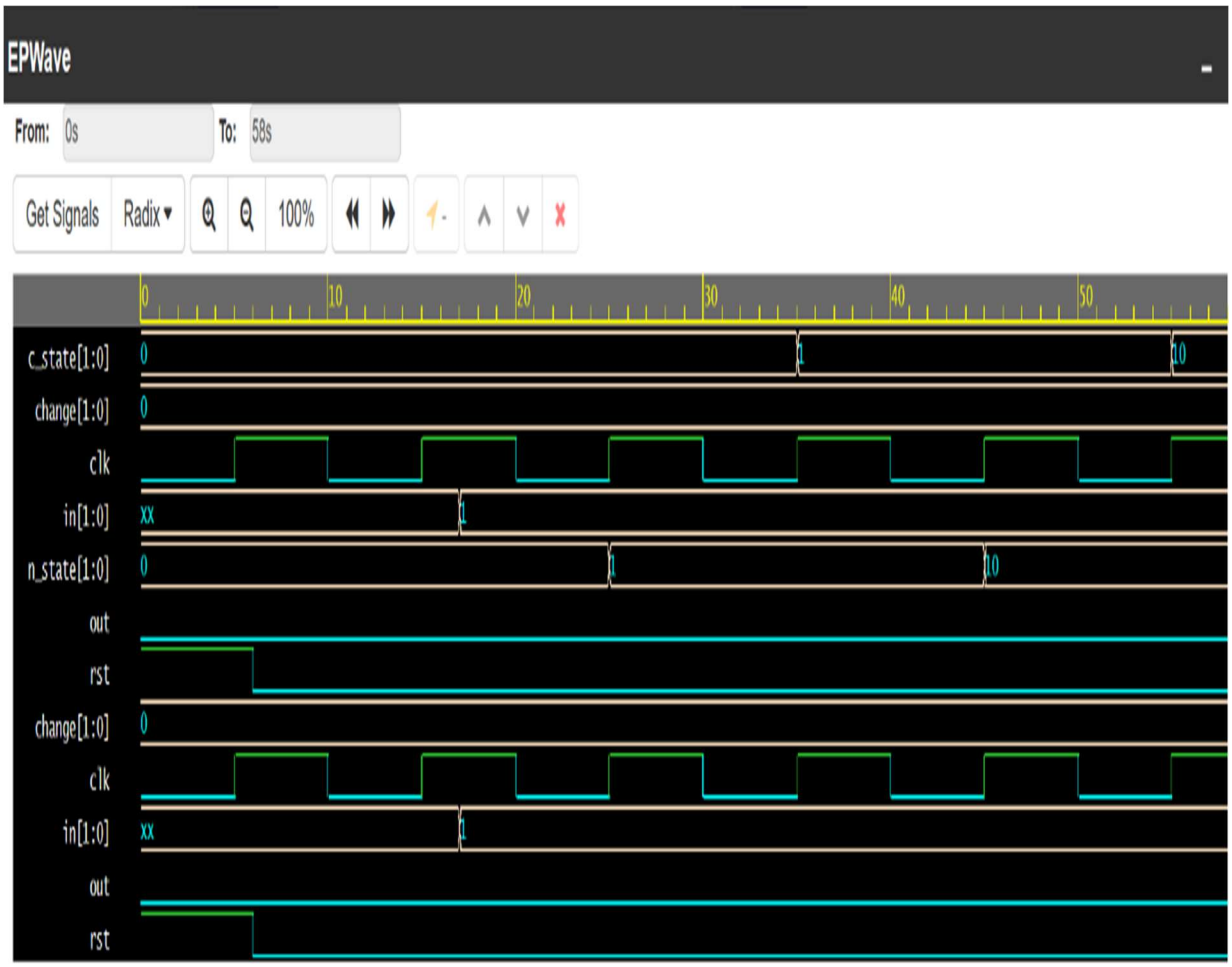
Registers c_state and n_state managed current and next states, respectively.

Clocked process managed state transitions and output assignments.

Results

3.1 Simulation

The module was simulated using test vectors for various inputs (in). Outputs (out and change) were validated based on expected behavior defined by the module's states and input combinations.



{Fig. 1:- EPwave at 0 sec}

{Fig. 1:- EPwave at 6 sec}

3.2 Observations

Correct functionality observed for different transaction scenarios.

Outputs aligned with expected results for specific inputs, validating the module's behavior.

4. Conclusion

The Vending Machine Module successfully emulates the behavior of a vending machine, handling transactions for 5 and 10 unit-priced items. Its FSM-based design facilitates accurate state transitions and output generation, demonstrating reliable functionality.

References

[1]David Harris and Sarah Harris, "Digital Design and Computer Architecture"

[2]Pong P. Chu, "FPGA Prototyping by Verilog Examples: Xilinx Spartan-3 Version"

[3]Samir Palnitkar, "Verilog HDL: A Guide to Digital Design and Synthesis"

[4]Mohamed El-Sharkawi, "Introduction to Digital Systems: Modeling, Synthesis, and Simulation Using VHDL"

Code

Verilog design:

```
module vending_machine_18105070(
  input clk,
  input rst,
  input [1:0] in, // 01 = 5 rs, 10 = 10 rs
  output reg out,
  output reg [1:0] change
);

parameter S0 = 2'b00;
parameter S1 = 2'b01;
parameter S2 = 2'b10;

reg [1:0] c_state, n_state;

always @(posedge clk or posedge rst)
  begin
    if (rst)
      begin
```

```
    c_state <= S0;
    n_state <= S0;
    change <= 2'b00;
    out <= 0;
  end
else
  c_state <= n_state;

case(c_state)
  S0: // State 0 : 0 rs
    begin
      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
    end
```

```
S1: // State 1 : 5 rs
  begin
    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 <= 0;
        change <= 2'b00;
      end
  end

S2: // State 2 : 10 rs
  begin
    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
    end
  endcase
```

testbench:

```
module vending_machine_tb;

  // Inputs
  reg clk;
  reg [1:0] in;
  reg rst;

  // Outputs
  wire out;
  wire [1:0] change;

  // Instantiate vending machine module
  vending_machine_18105070 uut (
```

```
  .clk(clk),
  .rst(rst),
  .in(in),
  .out(out),
  .change(change)
);

initial begin
  // Initialize inputs
  $dumpfile("vending_machine_18105070.vcd");
  $dumpvars(0, vending_machine_tb);
  rst = 1;
  clk = 0;

  #6 rst = 0;
  #11 in = 2'b01;
  #16 in = 2'b01;
  #25 $finish;
end

always #5 clk = ~clk;
```