ECE 310
Lab 5 Report: Moore FSM Vending Machine

Ohm Patel

October 26, 2025

Abstract

This lab involved designing and implementing a Moore finite state machine (FSM) to model a 25-cent vending machine that accepts dimes (10¢) and quarters (25¢). The machine outputs a vend signal when 25¢ or more is entered and raises an additional change output if the total exceeds 25¢. Inputs `D` and `Q` are binary coin indicators, and the design includes an active-low synchronous reset. Both the FSM module and testbench were implemented in Verilog and verified through simulation in Vivado. Waveforms confirm correct behavior for all possible coin sequences, including invalid and idle cases.

1 Introduction, Background, and Theory

Finite state machines (FSMs) are fundamental control elements that transition between discrete states based on inputs and timing. A **Moore machine** defines outputs as functions of the current state only, producing deterministic timing and clean signal transitions.

In this lab, the Moore FSM models a vending machine that:

- Accepts only one coin at a time: dime (`D`=1, `Q`=0) or quarter (`D`=0, `Q`=1).
- Ignores simultaneous or no inputs (`DQ`=11 or 00).
- Vends a product (`P`=1) when 25¢ or more has been inserted.
- Returns change (`C`=1) whenever the total exceeds 25¢.
- Waits in the vend state until the user provides new input or resets by idle/invalid input.

The FSM transitions are synchronous with the system clock, and all state logic follows Moore conventions.

2 Design and State Description

2.1 State Encoding and Meaning

The vending machine FSM uses five states, encoded with the minimum number of bits ($n = 3$). Each state represents the total amount of money accumulated so far, with additional states for vending (with or without change). Unused binary encodings are assigned transitions to `S0` by default to ensure safe recovery.

State	Binary Code	Description
S0	000	Idle / 0¢ inserted
S1	001	10¢ total (one dime)
S2	010	20¢ total (two dimes)
S3	011	Vend, exact 25¢ (product dispensed)
S4	100	Vend with change (>25¢ total)

2.2 State Transition Table

Table 1 lists the next-state and output behavior for all relevant input combinations. The inputs are represented as `DQ`, where `D`=1 indicates a dime and `Q`=1 indicates a quarter. Inputs "00" and "11" are treated as idle or invalid and therefore cause the machine to hold or reset, depending on state.

Current State	Input (DQ)	Next State	P	C
S0	10 (dime)	S1	0	0
S0	01 (quarter)	S3	0	0
S0	00 or 11	S0	0	0
S1	10 (dime)	S2	0	0
S1	01 (quarter)	S4	0	0
S1	00 or 11	S1	0	0
S2	10 or 01	S4	0	0
S2	00 or 11	S2	0	0
S3	10 (dime)	S1	1	0
S3	01 (quarter)	S3	1	0
S3	00 or 11	S0	1	0
S4	10 (dime)	S1	1	1
S4	01 (quarter)	S3	1	1
S4	00 or 11	S0	1	1

Table 1: State transition table for the vending machine Moore FSM. Outputs `P` (product) and `C` (change) depend only on the current state, consistent with Moore-type behavior.

2.3 Transition Summary

The FSM transitions are summarized as follows:

- **S0 → S1:** Dime inserted (`DQ`=10). **S0 → S3:** Quarter inserted (`DQ`=01).
- **S1 → S2:** Additional dime; **S1 → S4:** Dime + quarter combination (total exceeds 25¢).
- **S2 → S4:** Any valid coin input adds beyond 25¢.
- **S3/S4:** Vending states—`P`=1; remain until `DQ`=00 or 11, then reset to S0.

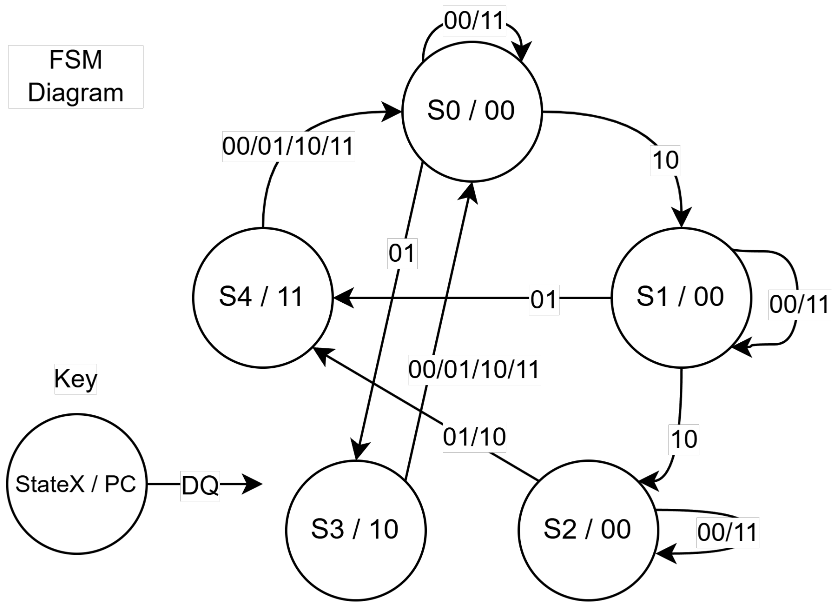

Figure 1: Moore FSM state transition diagram for the vending machine.

3 Implementation (Verilog)

3.1 FSM Module

Listing 1: Condensed Verilog Moore FSM implementation

```
module vending_25c_moore(input wire clock, reset_n, D, Q,
                          output reg P, C);
    reg [2:0] state, next_state;
    parameter S0=3'b000, S1=3'b001, S2=3'b010,
              S3=3'b011, S4=3'b100;

    always @(*) begin
        next_state = state; P = 0; C = 0;
        case (state)
            S0: case({D,Q})
                    2'b10: next_state = S1;
                    2'b01: next_state = S3;
                endcase
            S1: case({D,Q})
                    2'b10: next_state = S2;
                    2'b01: next_state = S4;
                endcase
            S3: begin P=1; case({D,Q})
                    2'b00,2'b11: next_state=S0;
                    2'b10: next_state=S1;
                endcase end
            S4: begin P=1; C=1; case({D,Q})
                    2'b00,2'b11: next_state=S0;
                endcase end
        endcase
    end

    always @(posedge clock)
        if(!reset_n) state <= S0;
        else state <= next_state;
endmodule
```

3.2 Testbench

Listing 2: Condensed Verilog testbench for the vending machine FSM

```
module vending_25c_moore_tb;
    reg clock, reset_n, D, Q;    wire P, C;
    vending_25c_moore dut (...);    // Instantiate DUT
    initial begin clock = 0; forever #5 clock = ~clock; end // Clock generation
    initial begin
        // Reset
        reset_n = 0; D = 0; Q = 0; #10; reset_n = 1; #10;

        // Test 1
        D = 0; Q = 1; #10; D = 0; Q = 0; #20;

        // Test 2
        reset_n = 0; #10; reset_n = 1; #10;
        repeat(3) begin D = 1; Q = 0; #10; D = 0; Q = 0; #10; end

        // Test 3
        reset_n = 0; #10; reset_n = 1; #10;
        D = 1; Q = 1; #10; D = 0; Q = 0; #10;
        D = 0; Q = 1; #10; D = 0; Q = 0; #20;

        // Test 4
        reset_n = 0; #10; reset_n = 1; #10;
        D = 1; Q = 0; #10; D = 0; Q = 0; #10;
        D = 0; Q = 1; #10; D = 0; Q = 0; #20;
        $finish; end
endmodule
```

4 Test Case Explanation and Justification

The FSM was validated with four representative scenarios that meet and exceed the lab's minimum of two required input vectors.

TC	Input Sequence (D,Q)	Description	Expected Output	Purpose / Coverage
1	(01) → 00	Insert quarter	P=1, C=0	Best-case vend, exact 25¢.
2	(10)(10)(10)	Three dimes	P=1, C=1	Overpay (30¢) → change.
3	(10)(11)(00)(01)	Invalid input	P=1, C=1	Tests robustness to 11, idle reset.
4	(10)(10)(01)	20¢ + quarter	P=1, C=1	Multi-coin path, change case.

Each case demonstrates a distinct state path and validates correct Moore output timing relative to the clock.

5 Results and Discussion

All simulations were run in Vivado. Figure 2 shows representative sections of the waveform for both best-case and non-best-case inputs. The FSM correctly held its vend output high until an idle or invalid input occurred, fulfilling the "wait-in-state" specification. Outputs P and C changed only on clock edges and matched the expected state-based behavior.

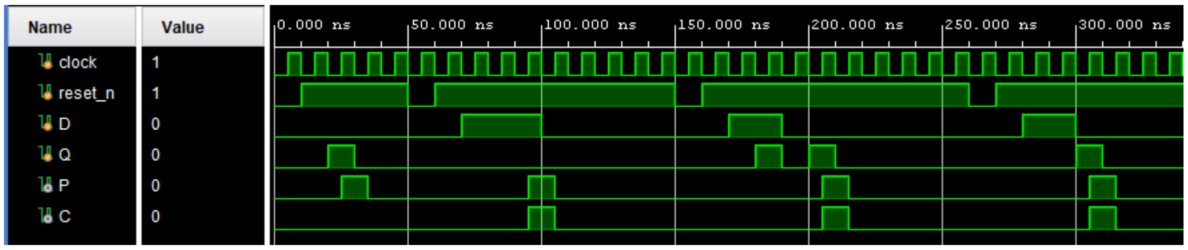

Figure 2: Simulation waveform showing vending machine FSM behavior.

6 Conclusion

The 25¢ vending machine Moore FSM was successfully designed, implemented, and verified. The design used five states to represent coin accumulation and vend conditions, meeting all specifications in the lab description. Outputs depend only on state, confirming Moore-style operation. Testbench simulations verified correct vending, change handling, invalid input behavior, and state retention until user input reset. All design goals and deliverables were achieved with full compliance to ECE 310 Lab 5 requirements.

This report was typeset in LaTeX.