EE 421/621, Fall 2024

# Project 3:
# Moore FSM Vending Machine Design

November 26th, 2024

**Michael Pittenger**

# Background

Finite State Machines (FSMs) are foundational tools in digital electronics design and are used for sequential decision making logic. FSMs consist of defined (finite) states in which the next state is determined by the current state and input. The output of a Moore FSM is determined solely by the state the machine is in and the output of a Mealy FSM is determined by both the current state and input of the machine.

# Design

The task for this project is to design a Moore FSM controlled vending machine. The vending machine sells newspapers that cost 35 cents, and the logic in the FSM will dictate how much is deposited and what response the user should get. The machine should take an input from the user in the form of coins (nickels, dimes, and quarters), and output a combination of signals representing refunds, releases, unlatching, and coin return. The outputs of the FSM are the signals stated previously, and the outputs are determined based on the user input and the current state of the machine (how much and what money was deposited by the user). The project will be implemented with the Quartus Prime software and on the Intel DE1-Soc FPGA board.

## Additional information:

If exact changed is entered, a mechanical latch is release (Unlatch) so the customer can get the paper. If the amount of money deposited exceeds 35 cents, change is given if possible. Otherwise, deposited coins are refunded to the customer and the mechanical latch is not released.
If sufficient change is available, the FSM pulses a nickel release (NR) or a dime release (DR) signal to release one coin of change at a time.
If insufficient change is available, the coins just deposited are refunded by the FSM by asserting a refund (REF) signal. Otherwise, the deposited coins join the repository as the FSM asserts a release (REL) signal.

To start, a state transition diagram was drafted to give a clear image of what states will be available and how states will transition between each other, shown in **Figure 1**. One method would be to have a different state for every single possible balance and combination of coins deposited for said balance. However, that method requires a very large number of states and the transition diagram would become very difficult to keep track of. Instead, the current balance in the machine will be used as states, and each state will simply update a counter to keep track of if the user inserted a nickel or a dime.

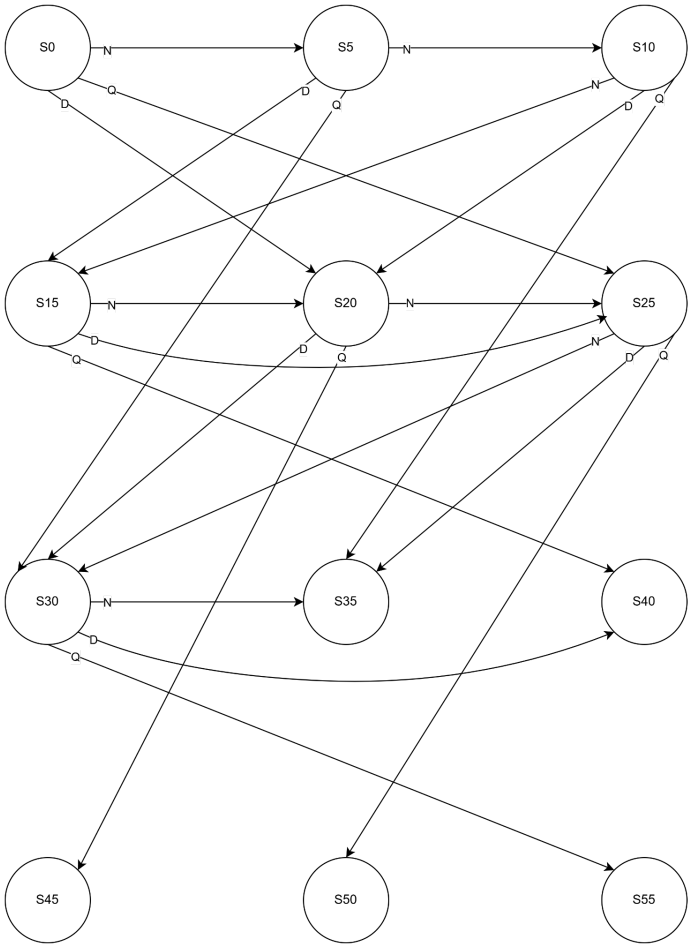


Figure 1: State Transition Diagram

**Table 1** shows the state transition table, which is based off of the state transition diagram mentioned previously. This table might make it easier to see what exactly is going on at each state and with a given input. Coin 00, represented by SW(1 downto 0) on the FPGA, will represent nickels, 01 will represent dimes, 10 will represent quarters, and 11 will represent an attempt to buy.

Table 1: State Transition Table

| Present State | Next State at “BUY” | | |
|---|---|---|---|
| | Coin = 00<br>N++ | Coin = 01<br>D++ | Coin = 10<br>Q |
| S0<br>Output | S5<br>REF=1 | S10<br>REF=1 | S25<br>REF=1 |
| S5<br>Output | S10<br>REF=1 | S15<br>REF=1 | S30<br>REF=1 |

| | | | |
|---|---|---|---|
| S10<br>Output | S15<br>REF=1 | S20<br>REF=1 | S35<br>REL=1, UNLATCH=1 |
| S15<br>Output | S20<br>REF=1 | S25<br>REF=1 | S40<br>CHECK Nx & Dx |
| S20<br>Output | S25<br>REF=1 | S30<br>REF=1 | S45<br>CHECK Nx & Dx |
| S25<br>Output | S30<br>REF=1 | S35<br>REL=1, UNLATCH=1 | S50<br>CHECK Nx & Dx |
| S30<br>Output | S35<br>REL=1, UNLATCH=1 | S40<br>CHECK Nx & Dx | S55<br>CHECK Nx & Dx |
| S35<br>Output | X | X | X |
| S40<br>Output | X | X | X |
| S45<br>Output | X | X | X |
| S50<br>Output | X | X | X |
| S55<br>Output | X | X | X |

When the balance is below 35 and no buy input is selected, then the FSM waits until another coin is deposited. If buy is pushed at any time, then a REF signal will be sent because there is not enough balance to purchase a newspaper.

When the balance is at 35, it is the perfect amount of a newspaper, so the REL signal is sent to store the coins and the UNLATCH signal is sent to give the user their newspaper.

When the balance is over 35, the FSM will not transition to any more states or accept any new coins from the user. If there are enough nickels and/or dimes in the repository to refund just the excess amount that was deposited, then the machine will dispense those coins and also a newspaper (DR, NR, REL, and UNLATCH signals). The machine will attempt to distribute dimes first. If there is not an amount of dimes and/or nickels sufficient for a refund of the excess, then a REF signal will be sent to give all change back to the user.

On the DE1-SoC FPGA board, the inputs will be handled by the KEY buttons and SW switches, and the outputs will be displayed using the LEDR LEDs and the HEX seven segment displays. KEY(0) will be the START/reset button, KEY(3) will be the clock, and SW(1 downto 0) will represent the coins inserted as well as the BUY signal. The NR, DR, REF, REL, and UNLATCH signals will be shown on LEDR(9 downto 5), and the number of nickels, dimes, and balance will be converted to binary coded decimal and displayed on the seven segment displays.

**Figure 2** shows the netlist generated by the Quartus Prime software for the completed FSM machine, with the yellow box representing the state transition diagram shown in **Figure 3**.

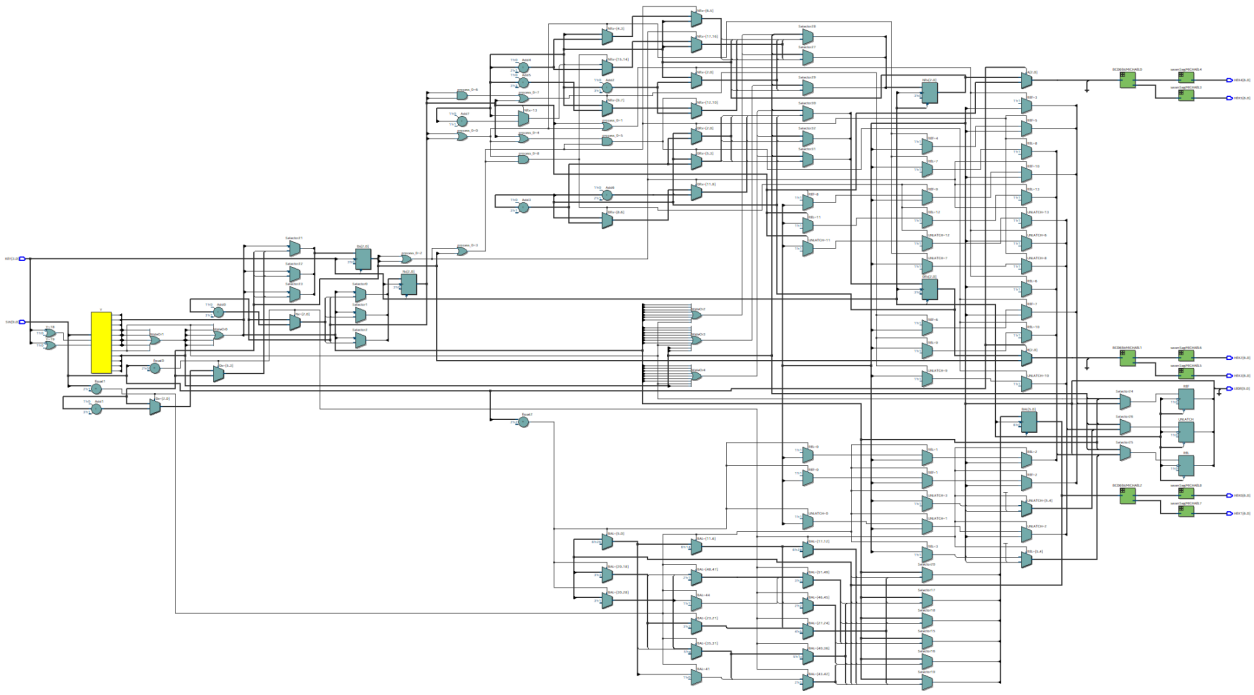

Figure 2: Generated Netlist

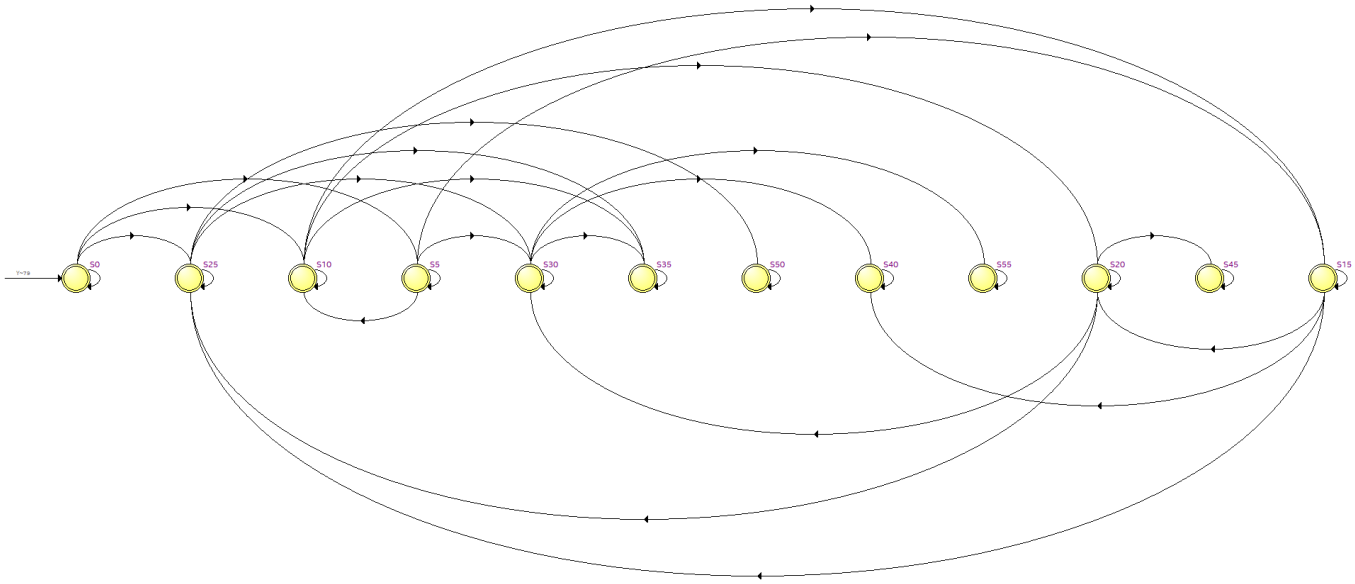


Figure 3: Generated State Transition Diagram

## Conclusion

This project successfully demonstrates the design and implementation of a Moore FSM-controlled vending machine using VHDL. The FSM was structured based on a state transition diagram and table to manage coin deposits (nickels, dimes, and quarters) and handle the various states of the machine. It processes user inputs, tracks the current balance, and controls outputs such as refunds, change dispensing, and the release of the newspaper. The logic was simulated using Quartus Prime Software and tested on an FPGA board, where the system's functionality was verified by ensuring the correct response to different user interactions.

The vending machine design not only provides the correct operation for exact payments but also efficiently handles situations where change is required or insufficient. If the balance exceeds the required amount, the FSM manages the release of change, prioritizing coins like dimes and nickels. When there is insufficient change, the system triggers a refund and prevents further coin acceptance. This project reinforced key principles in sequential logic design and FSM application, showcasing their effectiveness in real-world systems and solidifying my understanding of their practical use in digital systems.

The VHDL code used in the project printed and added to the back of this report.