# Finite State Machine Design for the ATM

Harshil Soni
*Department of Electronics Engineering*
*Birla Vishvakarma Mahavidyalaya*
*Engineering College*, Anand, Gujarat,
India
soniharshil15@gmail.com

***Abstract*— This paper describes the working of a simple Finite State Machine for an ATM. It has user data like account numbers and pin numbers. Where the FSM will identify the user. It has multiple features like checking the balance, withdrawing some amount, transferring an amount to other accounts, etc. This FSM can verify that the correct user is operating the system. An error message will be displayed if the amount withdrawn or transferred exceeds the account balance.**

***Keywords*—Finite State Machine, ATM, Withdraw, Transfer, Balance, Error**

## I. DESCRIPTION

A finite state machine is a mathematical abstraction used to design algorithms. In simpler terms, a state machine will read a series of inputs. When it reads an input, it will switch to a different state. Each state specifies which state to switch to for a given input. A FSM can be implemented in two ways: mealy, where the output depends on current input as well as current state, and moore where the output only depends on current state. This is a simple FSM for an ATM machine. In the code, the 10 different account numbers and their respective pin numbers are predefined in the authentication module. If a valid account number and a pin number are not entered, then it will not proceed further. The features of this FSM provide users with the ability to check balances, withdraw funds, and make transactions to other accounts. The errors will occur when the amount sent is more than the balance.

## II. STATE DIAGRAM

The state diagram of the ATM is shown in Figure 1. The design is mealy FSM. There are a total of five states in which this FSM operates. The initial state is "waiting". In that, the user has to enter the account number as well as the pin number. If the entered details match the data, it will go to state "menu". In this state, users have four options. According to the selected option, the state will be changed. The states are: (1)balance, (2) withdrawal without balance, (3) withdrawal with balance, and (4)transaction. In states 2 and 3, it requires an amount as input. In state 4, it requires an amount and an account number as input. If an invalid operation is performed, then it will show error messages and it will go to the state of "menu". After successfully completing the task again, it will go to the state of "menu". If the current user has finished his operation, then it will exit the menu state and go to the state "waiting".

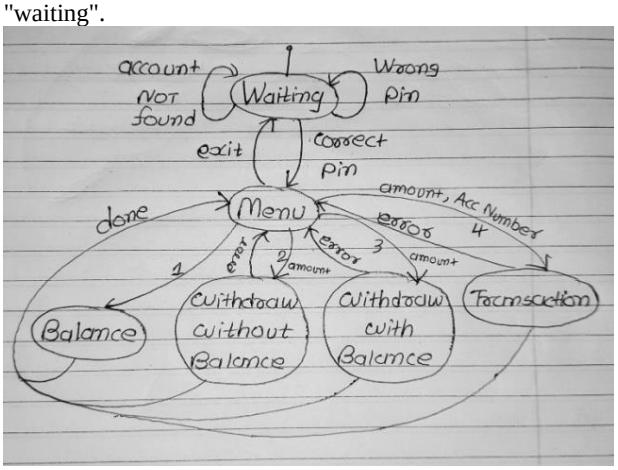


Fig. 1. State diagram for ATM

## III. SIMULATION RESULTS

Figure 2 shows the simulated results after performing certain operations mentioned in the testbench file. The first line is for a welcoming message. Then the wrong login credentials were used to login. So the user is unable to login because of a mismatch in data. Then, after entering the right credentials, the user is logged in. The next user will withdraw some money and then show the balance. After that, the user checks the balance. The next user is withdrawing too much money, resulting in an error. After that, transfer some money to the destination account with the number 2816. After that, the user transfers too much money to the destination account with the number 2816, which exceeds 2047 and causes an error. The next user exits the system and logs in using the account with the number 2816. The balance is now more than the default value because a previous user transferred some money to this account a while ago.

```
Welcome to the ATM
Account number or password was incorrect
Logged In.
Account 2178 has balance   400 after withdrawing  100
Account 2178 has balance   400
Error!, action causes an invalid operation.
Account 2178 has balance   400
Destination account 2816 after transaction has a total balance of   550
Error!, action causes an invalid operation.
Logged In.
Account 2816 has balance   550
Done
```

Fig. 2. Simulated results

## REFERENCES

[1] Verilog HDL code and Finite State Machine (FSM) for a simple ATM (https://github.com/arashsm79/Verilog-HDL-FSM-ATM.git