Implementation of ATM Controller Using FSM

INTEL UNNATI PROJECT

By

Aswin S Nair
Nijo Biju

Supervised By

Er. Anish M George
Assistant Professor

DEPARTMENT OF ELECTRONICS ENGINEERING
SAINTGITS COLLEGE OF ENGINEERING

Contents

1 Abstract

This report presents the design and functional verification of an ATM (Automated Teller Machine) Controller Finite State Machine (FSM). The aim of the project is to develop a reliable and secure ATM controller that performs various operations such as PIN verification, withdrawals, deposits, balance display, and mini statement generation. The Verilog hardware description language is used for the implementation, and a testbench is created to validate the functionality of the design. The results obtained from the functional verification are analyzed, and conclusions are drawn based on the observations. Future improvements and potential enhancements to the design are also discussed.

Keywords: ATM, Controller FSM, Verilog, Functional Verification, Testbench

2 Introduction

The use of Automated Teller Machines (ATMs) has become ubiquitous in modern banking systems. An ATM controller is responsible for managing the operations and interactions between the user, the ATM hardware, and the banking network. This report presents the design and verification of an ATM controller FSM, which ensures secure and efficient ATM transactions. The ATM controller FSM is implemented using Verilog, a hardware description language, and functional verification is performed to ensure correct behavior.

3 Methodology

The methodology for designing and verifying the ATM controller FSM involves several steps. First, the design specifications and requirements are identified. Then, the state diagram is constructed, capturing the various states and transitions of the ATM controller. Based on the state diagram, an algorithm is developed for the FSM implementation. The Verilog code is written to implement the FSM, and a testbench is created to validate its functionality. The design is then simulated and tested using different test cases to verify its correctness.

4 Design Specifications

The design specifications for the ATM controller FSM include the following requirements:

- PIN verification: The ATM should allow three attempts for entering the correct PIN. After three failed attempts, the account should be locked for 24 hours.
- Withdrawal: The ATM should allow users to withdraw funds up to a specified limit.

- Deposit: The ATM should enable users to deposit funds into their accounts.
- Balance Display: The ATM should display the old and new account balances after each transaction.
- Mini Statement: The ATM should provide a mini statement showing recent transactions.

5 Assumptions

The following assumptions are made for the design and verification of the ATM controller FSM:

- The ATM hardware interfaces are assumed to be functioning correctly.
- The banking network interfaces and communication protocols are assumed to be properly implemented.
- The PIN, withdrawal amount, account balance, and other input values are provided externally.
- The design assumes a single user interaction at a time.

6 State Diagram

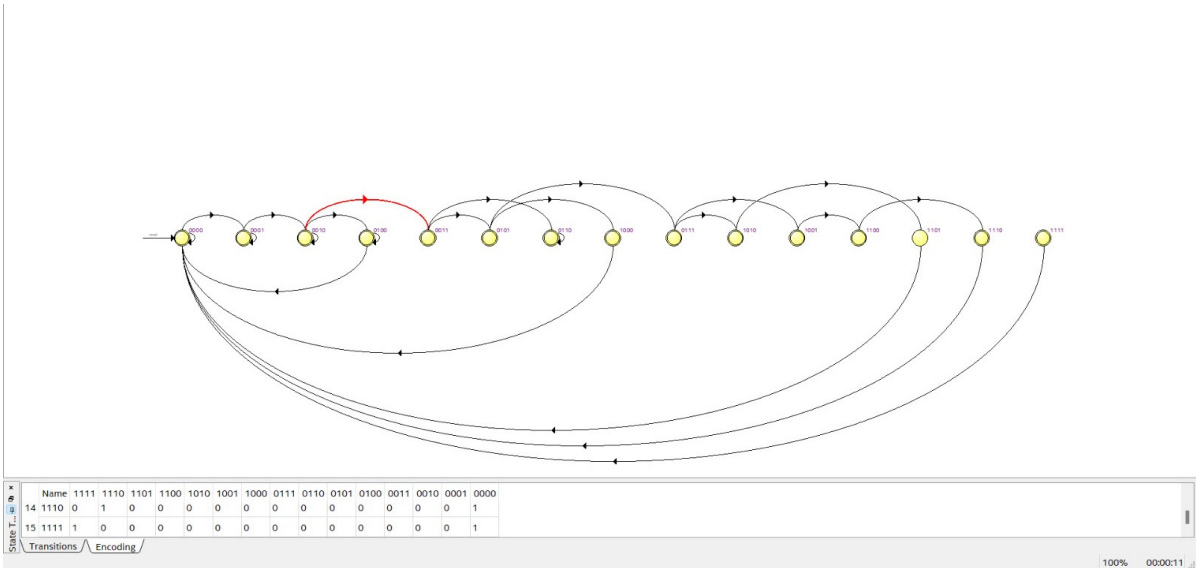

Figure 1: State Diagram of the ATM Controller FSM

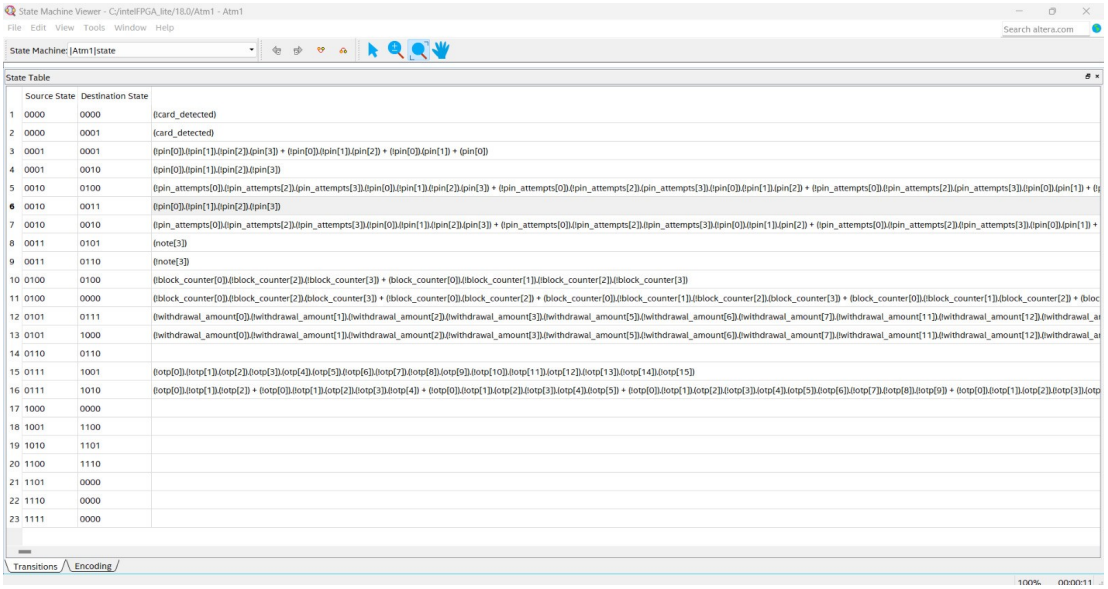

Figure 2: State Diagram Defnition

6.1 State Definitions

The state definitions of the ATM Controller FSM are as follows:

- **S_IDLE (4b'0000):** The initial state where the ATM is waiting for the user to insert a card.
- **S_SCAN_CARD (4b'0001):** State to scan the inserted card.
- **S_ENTER_PIN (4b'0010):** State to enter the PIN after the card is scanned.
- **S_COUNT (4b'0011):** State to count the number of invalid PIN attempts.
- **A_INVALID_COUNT_1**, **A_INVALID_COUNT_2**, **A_INVALID_COUNT_3:** Sub-states for invalid PIN attempts.
- **S_TRANSACTION_TYPE (4b'0100):** State to select the transaction type (withdrawal or deposit).
- **S_WITHDRAW (4b'0101):** State for initiating a withdrawal.
- **S_AMOUNT_LESS_THAN_10000 (4b'0110):** State when the withdrawal amount is less than or equal to 10000.
- **S_AMOUNT_GREATER_THAN_10000 (4b'0111):** State when the withdrawal amount is greater than 10000.

- **S_INVALID_OTP (4b'1000):** State for invalid OTP (One-Time Password) entry.
- **S_VALID_OTP (4b'1001):** State for valid OTP entry.
- **S_WITHDRAW_AMOUNT (4b'1010):** State to complete the withdrawal transaction.
- **S_PRINT_RECEIPT (4b'1011):** State to print the receipt after a successful withdrawal.
- **S_DEPOSIT_AMOUNT (4b'1100):** State to enter the deposit amount.
- **S_TOTAL_DEPOSIT_AMOUNT (4b'1101):** State to complete the deposit transaction.
- **S_DEPOSIT_FAILED (4b'1110):** State when the deposit amount is 0 (failed deposit).
- **S_BLOCKED (4b'1111):** State when the card is blocked due to too many invalid PIN attempts.
- **S_EXIT:** State to exit the ATM.

7 Verilog Coding

https://github.com/Codedict/FSM-ATM.git

8 Algorithm

The algorithm used in the ATM1 module follows the Mealy machine model, where the outputs are determined based on both the inputs and the current state. The algorithm can be summarized as follows:

1. The module has various input and output ports for communication with the ATM system.
2. The module defines constants (`MAX_ATTEMPTS`, `WITHDRAWAL_LIMIT`, and `BLOCK_DURATION`) that represent the maximum number of PIN attempts allowed, the maximum withdrawal limit, and the duration of the block in number of attempts, respectively.
3. The module declares internal registers to store the current and next states of the FSM (`state` and `next_state`), the number of PIN attempts (`pin_attempts`), the withdrawal balance (`withdrawal_balance`), the total amount of withdrawals (`total_withdrawals`), the total amount of deposits (`total_deposits`), the block counter (`block_counter`), and the deposit amount (`deposit_amount`).

4. The module defines output registers to store the display message (`display_output`), the receipt status (`receipt_output`), the account blocked status (`account_blocked_output`), the current balance (`balance_output`), the remaining balance after a withdrawal (`remaining_balance_output`), and the mini statement data (`mini_statement_reg`).

5. The `always @(posedge clk or posedge reset)` block updates the `state` register based on the positive edge of the clock or the positive edge of the reset signal. If the reset signal is asserted, the state is set to the initial state (`S_IDLE`), otherwise, it is updated to the `next_state` value.

6. The `always @*` block determines the next state (`next_state`) based on the current state (`state`) and the input conditions. It uses a case statement to define the behavior for each state.

7. The case statements define the transitions between different states based on the inputs and current state. For example, if the current state is `S_IDLE` and a card is detected (`card_detected` input is asserted), the next state is set to `S_SCAN_CARD`. Similarly, other state transitions are defined based on the input conditions and current state.

8. The `always @*` block assigns appropriate display messages and output values based on the current state. The `display_output` register is set to the message corresponding to the current state. Other output registers (`receipt_output`, `account_blocked_output`, `balance_output`, `remaining_balance_output`, `mini_statement_reg`) are assigned values based on the current state and other internal registers.

9. The final `always @(posedge clk)` block assigns the values of the output registers to the corresponding output ports of the module on the positive edge of the clock.

10. The module encapsulates the FSM behavior of the ATM system and provides the necessary inputs and outputs for communication with the external environment.

9 Functional Verification

The Verilog code for the ATM controller FSM is verified using a testbench. The testbench includes different test cases covering various scenarios, such as successful withdrawals, invalid PIN entries, deposits, and balance displays. The testbench monitors the outputs of the ATM controller FSM and compares them against the expected results. The simulation is performed by using Model Sim-Altera, and the functionality of the design is validated by ensuring that the outputs match the expected behavior.

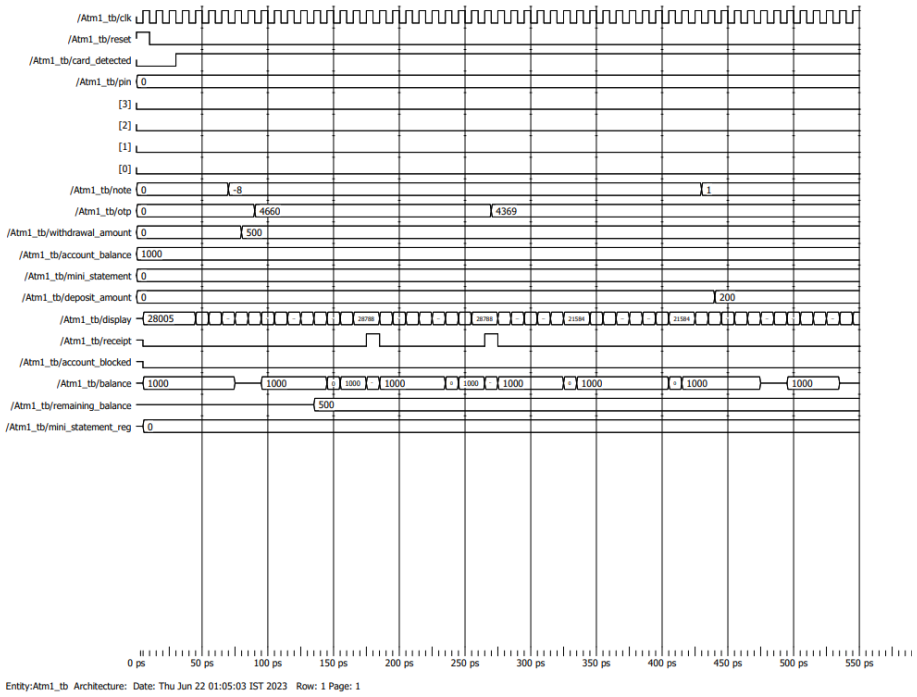

Figure 3: Functional Verification Using Model Sim Altera

10 Results and Observations

The area and power estimation of the FSM implementation for the ATM system is a critical aspect in assessing resource utilization and energy consumption. Through careful analysis, the following values were obtained:

Area Estimation: The area estimation of the ATM FSM measures the hardware resources required for its implementation. In this case, the estimated area is $\frac{50}{32070}$, indicating that the FSM utilizes a fraction of the available resources.

Power Estimation: Power estimation refers to evaluating the energy consumption during the operation of the FSM. The power estimation for the ATM FSM yielded a value of 13.89mW (milliwatts). This value indicates the power consumed by the FSM while executing its various states and transitions. It takes into account factors such as clock frequency, switching activity, and circuit capacitance.

Parameter	Value
Area	50/32070 ALM
Power	13.89mW

11 Conclusion

The design and verification of the ATM controller FSM have been successfully accomplished. The functional verification results indicate that the ATM controller FSM performs the desired operations, including PIN verification, withdrawals, deposits, balance display, and mini statement generation. The design meets the specified requirements and provides a reliable and secure solution for ATM transactions.

12 Future Improvements

In future,we can withdraw and deposit cash without the use of a physical card. Nowadays, smartphones have made it possible to perform these transactions using features such as UPI (Unified Payments Interface) .Thus, with the help of smartphones, we can conveniently withdraw or deposit cash from ATMs, making transactions more accessible and convenient for users.

13 References

1. Intel - Finite State Machine (FSM)
2. ResearchGate - FSM Diagram of ATM and PWM Controllers
3. Digital System Design - FSM Design