DESIGN REPORT

PROJECT (1)

CSE312

FALL 2023

Submitted by:

Mohamed Ahmed Esmat 21P0144

Saleh Ahmed Elsayed 21P0324

Youssef Mohamed Al-Sayyed 21P0094

Ahmed Hossam Eldin Ibrahim 21P0271

Abdallah Ahmed Hamdy Ibrahim 21P0333

Zeynat Haytham Farouk 2100588

Shahd Ashraf Mahmoud 2100480

Contents

1. Introduction

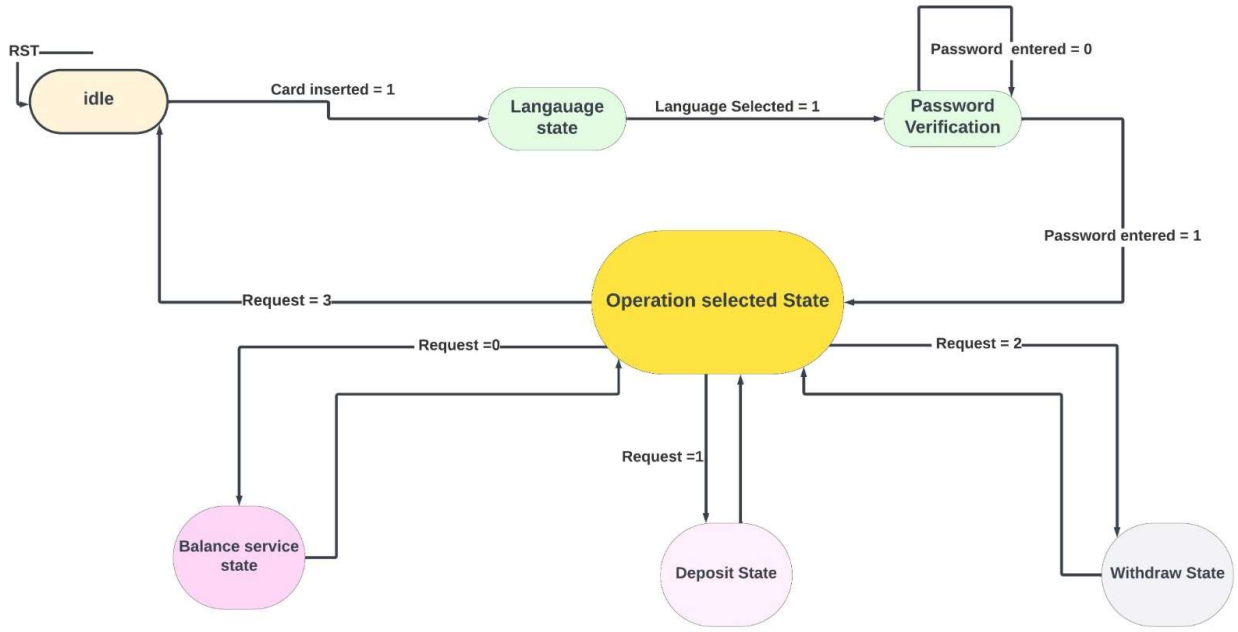
Our aim in this project is to implement the core of the bank ATM design as well as verification environment.

The project assumes all auxiliary devices like card handling, money counting, and timers exist. As well as it assumes that account information like passwords, account numbers and balances exist locally with no need for database connection.

Some of the auxiliaries implemented as:

- Card Handling
- Language Used
- Card Password
- Timers (for 15 minutes)
- Operations (Deposit – Withdraw – Balance service)
- Can input deposit value
- Can input withdraw value

2. FSM Design



Idle:

- This is the initial state where the ATM is waiting for a card to be inserted.
- If the RST is activated, it returns to the Idle state.
- Once a card is inserted (card inserted = 1), the FSM transitions to the Language Selection state.

```
case (state)
    IDLE_STATE: begin
        if (card_inserted && entered == 1'b0) begin
            entered = 1;
            next_state = LANGUAGE_SELECTED_STATE;
        end
        else begin
            next_state = IDLE_STATE;
        end
    end
```

Language Selection:

- The user selects the desired language (language selected = 1).
- The FSM then moves to the Password Verification (Password Entered state).

```
LANGUAGE_SELECTED_STATE:
    if (language_selected === 1'b0 || language_selected === 1'b1) next_state = PASSWORD_ENTERED_STATE;
    else next_state = LANGUAGE_SELECTED_STATE;
```

Password Verification:

- The user enters their password.
- The FSM compares the entered password and card number with the stored values.
- If there's a match (password entered = 1), the FSM transitions to the Operation Selected state.
- If not, the FSM remains in the Password Entered state, allowing further attempts.

```
PASSWORD_ENTERED_STATE: begin
        password_entered = 0;
        index = 3'b000;
        for (i = 0; i < 5; i = i + 1) begin
            if (CardNo == CardNos[i] && Password == Passwords[i]) begin
                password_entered = 1;
                index = i;
            end
        end
    if (password_entered) begin
        next_state = OPERATION_SELECTED_STATE;
    end
    else begin
        next_state = PASSWORD_ENTERED_STATE;
    end
end
```

Operation Selected:

- The user selects a desired operation:
  - Request = 0: Check balance (transitions to Balance Service state).
  - Request = 1: Deposit (transitions to Deposit state).
  - Request = 2: Withdraw (transitions to Withdraw state).
  - Request = 3: Exit request (returns to Idle State).

```
OPERATION_SELECTED_STATE:
    if (request === 2'b00) next_state = BALANCE_SERVICE_STATE;
    else if (request == 2'b01) next_state = DEPOSIT_STATE;
    else if (request == 2'b10) next_state = WITHDRAW_STATE;
    else if (request == 2'b11) next_state = IDLE_STATE;
    else next_state = OPERATION_SELECTED_STATE;
```

Balance Service State:

- The user's account balance for the matching card is displayed.
- The FSM then returns to the Operation Selected state.

Deposit:

- The user enters a deposit amount (deposit value).
- If the amount is less than or equal to the maximum limit the deposit is accepted and added to the balance.
- If the amount exceeds the limit, the deposit is rejected.
- In both cases, the FSM returns to the Operation Selected state.

Withdraw:

- The user enters a withdrawal amount (withdraw value).
- If the amount is less than the current balance, the withdrawal is accepted and deducted from the balance.
- If the amount is greater than or equal to the balance, the withdrawal is rejected.
- Like deposit, the FSM returns to the Operation Selected state after each attempt.