ISSN 2277-2685
IJESR/April-June. 2026/ Vol-16/Issue-2/1059-1065
Salveru Pragathi *et. al.*, / International Journal of Engineering & Science Research

Full Length Article

Design and Simulation of Two-Floor Elevator Controller FSM Using VHDL

Ms B Jyothsna[1], Salveru Pragathi[2], Rajaram Varshini[3], Ravvala Ramya[4]
[1]Associate Professor; Department of Electronics and Communications Engineering Bhoj Reddy Engineering College for Women Hyderabad India.
[2,3,4]B.Tech Student's; Department of Electronics and Communications Engineering Bhoj Reddy Engineering College for Women Hyderabad India
Mail Id's; pragathisalveru@gmail.com[2],Varshinirajaram3@gmail.com[3],ravvalaramya22@gmail.com[4]

Accepted 02-06-2026

Abstract

An elevator controller is a crucial digital control system used to ensure safe and efficient movement of elevators between floors. This project presents the design and implementation of an elevator controller using a Finite State Machine (FSM) approach. In this method, the elevator operation is divided into well-defined states such as idle, moving up, moving down, door open, and door close. State transitions occur based on floor requests and sensor inputs. The primary objective of this project is to develop a reliable control system that accurately responds to user floor selections, manages motor direction, and ensures proper door operation. The FSM approach simplifies the overall design by clearly defining system states and transitions, making the controller easy to design, analyze, and implement. The system can be implemented using hardware description languages such as Verilog or VHDL and validated through simulation tools. This project enhances understanding of sequential logic design and digital control systems. It is highly applicable in embedded systems and automation environments where structured state-based decision-making is required. The proposed design provides efficient performance, reduced complexity, and scalability for multi-floor elevator systems.

Keywords: *Elevator controller, Finite State Machine (FSM), digital control system, sequential logic, Verilog, VHDL, embedded systems, automation, state transition, motor control.*

Introduction

Elevators are essential vertical transportation systems widely used in modern buildings to ensure safe and efficient movement of people between floors. The control of elevator operations requires a well-structured system capable of handling multiple requests, determining movement direction, and managing door operations in a synchronized manner. Since elevator behavior follows a clearly defined sequence of operations, it can be effectively modeled using a Finite State Machine (FSM).

In this project, an elevator controller is designed based on FSM principles, where the entire operation is divided into a set of discrete states. These include the idle state, upward movement state, downward movement state, door open state, and door close state. Each state represents a specific function of the elevator system, and transitions between states occur based on external inputs such as floor requests and sensor feedback. The controller operates by continuously monitoring user inputs from inside and outside the elevator. When a floor request is detected, the system evaluates the current position of the elevator and decides the direction of movement accordingly. Once the requested floor is reached, the system transitions to the door open state, allows passengers to enter or exit, and then proceeds to close the door after a defined delay. After completing the cycle, the system either returns to idle or processes the next pending request.

The FSM-based approach simplifies the overall design by clearly separating system behavior into well-defined states and transitions. This improves design clarity and makes implementation more systematic. The controller is implemented using Verilog Hardware Description Language, where states and transitions are described using sequential logic structures such as always blocks and case statements.

For verification, simulation tools such as ModelSim or Xilinx Vivado are used. A testbench is developed to evaluate different operating scenarios, including

multiple floor requests, direction changes, and door timing conditions. Waveform analysis is used to confirm correct system behavior. The design can also be implemented on FPGA hardware to validate real-time performance.

This project enhances understanding of sequential logic design, FSM modeling, and digital system implementation. It also demonstrates how complex real-world control systems can be efficiently designed using structured digital design techniques.

Literature Survey

Elevator control systems have been an important area of research in digital electronics and automation due to their widespread use in modern infrastructure. Initially, elevator systems were controlled using mechanical switches and relay-based circuits, which provided basic functionality but were not suitable for complex multi-floor operations.

As technology progressed, digital control methods replaced traditional systems, enabling more efficient and reliable operation. Among these methods, the Finite State Machine approach has gained significant importance because elevator operation naturally follows a sequence of defined stages.

FSM-based designs typically include states such as idle, moving up, moving down, door opening, and door closing. Each state performs a specific function, and transitions occur based on input conditions such as floor requests and sensor signals. This structured approach simplifies system design and improves reliability.

Research also shows that FSM-based controllers are easier to implement using hardware description languages like Verilog and VHDL. These tools allow designers to simulate and verify system behavior before hardware implementation, reducing errors and improving efficiency.

Modern studies also explore FPGA-based implementations, where FSM logic is deployed on programmable hardware for real-time operation. Such systems offer higher speed, flexibility, and scalability compared to traditional designs.

Existing System and Proposed System

Existing Methods

From existing studies and literature, it is evident that elevator control systems have evolved significantly, with FSM emerging as a widely accepted design approach. Earlier systems were implemented using mechanical relays and basic electronic circuits, which provided limited automation and lacked flexibility. These designs were suitable for simple applications but struggled to handle complex multi-floor environments.

FSM-based modeling introduced a structured way of designing elevator controllers by breaking the system into distinct states. Each state represents a specific function such as idle condition, movement in upward or downward direction, and door operations. Transitions between these states are triggered by inputs like floor selection signals, sensor feedback, and timing constraints. This structured representation improves clarity, reduces design ambiguity, and simplifies implementation.

Research indicates that FSM-based elevator systems offer improved reliability and safety since the system always operates in a defined state and transitions only occur under valid conditions. This reduces the risk of unexpected behavior such as incorrect movement or unsafe door operation. Additionally, FSM simplifies system design by dividing complex logic into smaller, manageable components, making debugging and modification easier.

The use of hardware description languages such as Verilog, along with FPGA platforms, has further enhanced system development. These tools enable simulation, verification, and hardware realization of elevator controllers before deployment. This reduces design errors and improves system reliability. Moreover, FPGA-based implementations provide flexibility and scalability for future enhancements.

Modern research also explores advanced techniques such as intelligent scheduling and energy-efficient control strategies. However, despite these advancements, FSM continues to serve as the core structure due to its simplicity and dependability.

Overall, existing methods confirm that FSM-based design remains one of the most effective approaches for implementing elevator control systems in both academic and practical applications.

Proposed System

The proposed elevator controller is developed using a Finite State Machine approach to achieve organized, efficient, and reliable system behavior. The entire operation is divided into a set of predefined states, including idle, moving upward, moving downward, door opening, and door closing. Each state performs a specific function, and transitions between states are determined by user inputs and sensor feedback.

When a floor request is generated, the controller compares the requested floor with the current

position of the elevator. If the requested floor is higher, the system transitions to the upward movement state; if it is lower, it moves to the downward state. Once the elevator reaches the desired floor, the motor is stopped, and the system enters the door opening state. After a defined delay, the door closes automatically, and the system returns either to idle or processes additional requests.

The FSM-based design reduces system complexity by clearly defining operational behavior through states and transitions. It improves reliability by ensuring that each operation occurs in a controlled sequence. The design is implemented using Verilog HDL, allowing simulation and verification before hardware deployment. The proposed method also supports scalability, enabling easy extension to multi-floor systems and integration of advanced features in future developments.

Proposed Method

The proposed method implements the elevator controller using a Finite State Machine designed in Verilog HDL. The system continuously monitors inputs such as floor requests and sensor signals to determine its operational state. Based on these inputs, the controller transitions between states in a structured manner to ensure smooth operation.

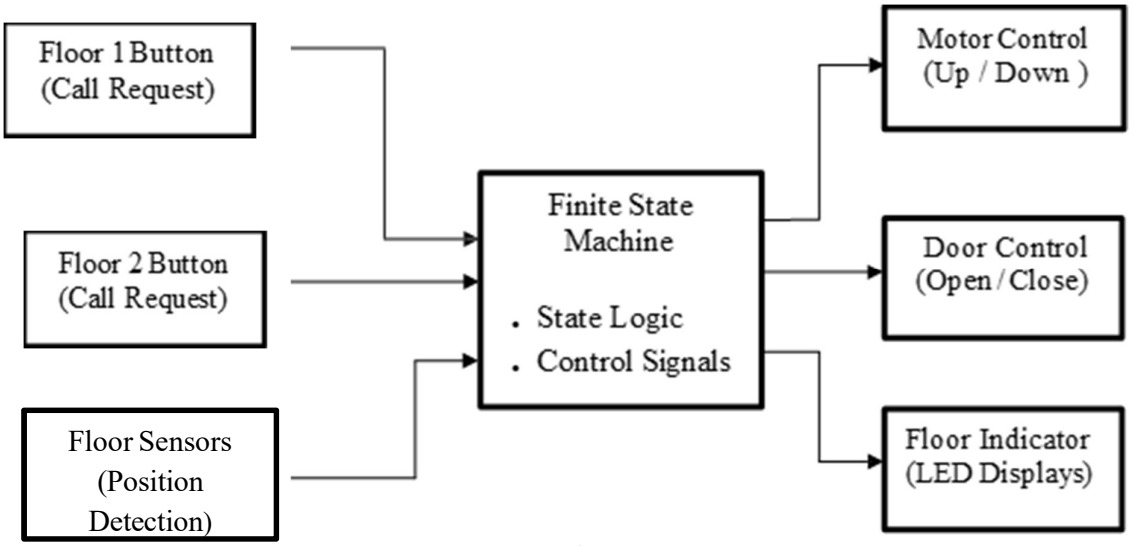
Initially, the system remains in the idle state, waiting for user input. Once a request is received, the controller evaluates the target floor in relation to the current floor. Depending on this comparison, the system transitions to either upward or downward movement. During motion, floor sensors provide continuous feedback to update the current position.

When the destination floor is reached, the system enters the door open state, allowing passengers to enter or exit. After a fixed time delay, the controller moves to the door close state, ensuring safe closure before further operation. If additional requests exist, they are processed sequentially; otherwise, the system returns to idle.

The FSM is implemented using Verilog by defining states as parameters and controlling transitions using a clock-driven sequential logic structure. A case-based structure is used to define state behavior and transition conditions. A testbench is developed to simulate various scenarios such as multiple requests, directional changes, and timing conditions, ensuring correctness of operation.

This method provides a structured, scalable, and efficient solution for elevator control systems and supports future enhancements such as priority scheduling and multi-floor optimization.

Block Diagram



Block Diagram

The block diagram represents the overall architecture of a two-floor elevator control system designed using FSM principles. It consists of three major components: input unit, control unit, and output unit.

The input unit includes floor selection buttons and floor position sensors. Floor buttons allow users to request specific floors, while sensors provide real-time feedback about the elevator's current position.

These inputs are sent to the FSM controller, which acts as the central decision-making unit.

The FSM controller processes input signals and determines the appropriate state of operation. It includes states such as idle, moving upward, moving downward, door opening, and door closing. Based on current conditions, the controller generates output signals that manage motor direction, door operations, and floor indication.

The output unit includes motor control, door control, and floor display indicators. The motor control module handles upward and downward movement, while the door control module manages opening and closing operations. Floor indicators display the current position of the elevator to users.

The system operates synchronously using a clock signal, ensuring proper timing of state transitions. A reset signal is also included to return the system to its initial state when required. Overall, the block diagram provides a clear representation of how inputs are processed through FSM logic to generate controlled elevator movement.

Software Implementation

Software Description

The elevator controller is developed using Verilog HDL, which is widely used for digital system modeling and simulation. The design is structured into modules representing inputs, outputs, and control logic. The FSM-based architecture defines each elevator operation as a separate state.

Input signals such as floor requests and sensor feedback are continuously monitored by the system. Based on these inputs, the controller transitions between states including idle, upward movement, downward movement, door opening, and door closing. Each state generates specific output signals for motor operation, door control, and floor indication.

The design consists of sequential logic, which updates the current state based on the clock signal, and combinational logic, which determines the next state and outputs. A reset function is included to initialize the system when required.

Simulation is performed to validate system behavior before hardware implementation. This ensures correct operation under multiple scenarios, including different floor requests and movement directions. The software-based approach improves reliability and allows easy modification for future enhancements.

Features of Vivado

Vivado Design Suite provides an integrated environment for FPGA-based digital system design. It supports Verilog and VHDL coding, simulation, synthesis, and implementation in a unified platform. It also offers waveform analysis tools for observing system behavior and debugging design issues. Additionally, it enables performance optimization and hardware mapping for FPGA devices, making it suitable for FSM-based controller design.

Algorithm

The elevator control algorithm begins with system initialization in the idle state. The controller continuously waits for user input through floor selection buttons. When a request is received, the system compares the requested floor with the current position.

If the requested floor is higher, the elevator moves upward; if it is lower, it moves downward. During movement, sensors continuously track the elevator position. Once the destination floor is reached, the motor stops and the door opens for a fixed duration. After the delay, the door closes automatically and the system returns to idle or processes additional requests. This sequence ensures smooth, safe, and efficient elevator operation.

Finite State Machine Design

Basic Components of FSM

A Finite State Machine consists of several fundamental components that collectively define its behavior. The first component is the **state**, which represents a specific condition of the system at a given time. At any instant, the system exists in only one state, and its behavior is determined by that state. In an elevator system, typical states include idle, moving up, moving down, and door open, each defining a unique operational mode.

The second component is **inputs**, which are external or internal signals that influence state transitions. In this project, inputs include floor request buttons, clock signals, and reset signals. These inputs determine how and when the system changes its state.

The third component is **outputs**, which represent system actions based on the current state. In an elevator controller, outputs include motor activation signals and door control signals, which directly control physical operations of the system.

The **state register** acts as memory, storing the current state of the FSM. It is typically implemented using flip-flops and is updated at each clock cycle.

Without this memory element, sequential behavior cannot be achieved.

The **next state logic** determines the future state of the system based on current inputs and present state. This logic is implemented using combinational circuits and is responsible for decision-making within the FSM.

State transitions define the movement of the system from one state to another based on input conditions. These transitions describe how the system evolves over time and are essential for modeling dynamic behavior.

The **clock signal** synchronizes all state changes in the system. It ensures that transitions occur at fixed time intervals, maintaining system stability and avoiding timing conflicts.

The **reset signal** initializes the FSM to a known starting state, usually idle. It is essential for ensuring predictable system behavior during startup or fault recovery.

Finally, **output logic** generates system outputs based on the current state (Moore model) or both state and inputs (Mealy model). It ensures that the FSM produces correct and meaningful external actions.

Types of FSM

A **Moore machine** is a type of FSM where outputs depend only on the current state. This makes the system more stable because outputs change only during state transitions. In an elevator system, actions such as motor control or door operation remain consistent within a state, improving safety and predictability.

In contrast, a **Mealy machine** produces outputs based on both the current state and input signals. This allows faster response since outputs can change immediately with input variation. However, this can introduce instability due to rapid input changes.

When comparing both models, Moore machines offer better stability and simplicity, while Mealy machines provide faster response with increased complexity. For safety-critical systems like elevators, stability is more important than speed.

In this project, the **Moore machine model is selected** because it ensures that outputs change only during state transitions, reducing the risk of glitches and improving system reliability.

Elevator System States

The elevator system operates through a sequence of well-defined states to ensure organized and safe functioning. Initially, the system remains in the idle state where it waits for a floor request. When a request is received, the system evaluates the requested floor relative to the current position.

If the requested floor is above the current floor, the system transitions to the move-up state. If it is below, it transitions to the move-down state. During these states, the elevator moves in the respective direction until it reaches the target floor.

Once the destination floor is reached, the system enters the door-open state, allowing passengers to enter or exit. After a predefined delay, the system transitions to the door-close state. Finally, the elevator returns to the idle state, ready to process new requests. This structured sequence ensures smooth and reliable operation.

State Diagram

State Diagram

The state diagram represents the behavior of the FSM-based elevator controller and illustrates how the system transitions between different operational states.

The system begins in the **idle state**, where it remains inactive until a request is received. When a request occurs, the controller compares the requested floor with the current floor position.

If the requested floor is higher, the system transitions to the **move-up state**. If it is lower, it moves to the **move-down state**. The elevator continues movement until it reaches the target floor, as confirmed by sensor feedback.

Upon reaching the destination, the system transitions to the **door-open state**, where the door remains open for a fixed duration controlled by a timer. After the timer expires, the system moves to the **door-close state**.

Once the doors are fully closed, the system returns to the idle state, completing one operational cycle and preparing for the next request.

State Transition Table

The state transition table defines system behavior under different input conditions. When the system is in the idle state with no request, it remains idle and all outputs remain inactive. If a request is greater than the current floor, the system transitions to move-up and activates upward motor control. If the request is lower, it transitions to move-down with downward motor activation.

When the elevator reaches the destination floor during movement states, it transitions to the door-open state and disables motor operation. After the timer completes, the system moves to door-close, activating door control signals. Once the door is fully closed, the system returns to idle, completing the cycle.

Results

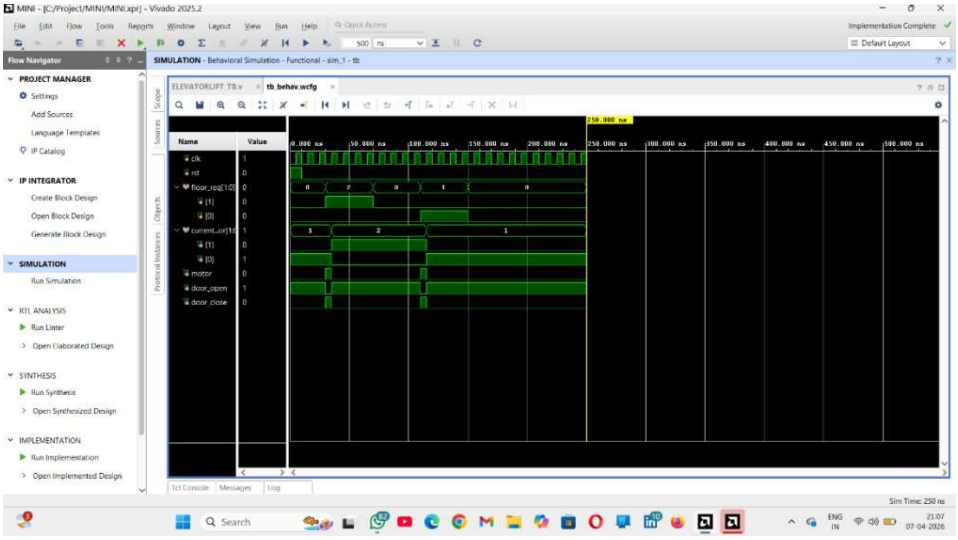


Fig 1 Output Waveform

The simulation waveform generated in Vivado represents the real-time behavior of the elevator controller. The clock signal ensures proper synchronization of all operations, while the reset signal initializes the system to a known starting state.

Different floor requests were applied during simulation to test system response. The controller successfully identified the requested floor and updated the current floor position accordingly. During movement, the motor signal became active, indicating correct upward or downward motion.

Once the elevator reached the target floor, the motor was disabled, and the door_open signal was activated, allowing passenger access. After a fixed delay, the door_close signal was triggered, completing the operation cycle.

The waveform confirms that all FSM states operate correctly and transition in a structured manner. The system accurately handles multiple requests and ensures proper sequencing of elevator operations.

Conclusion

The elevator controller developed in this project successfully demonstrates the effectiveness of Finite State Machine (FSM) methodology in designing a structured digital control system. By representing elevator operation through well-defined states such as idle, upward movement, downward movement, door opening, and door closing, the system achieves an organized and predictable flow of operations. Each state performs a specific function, and transitions between states are governed by input conditions such as floor requests and sensor feedback. This state-driven approach ensures safe,

reliable, and deterministic behavior, which is essential for elevator systems.

The implementation using Verilog HDL provided practical exposure to digital system design and enabled accurate modeling and verification of the controller before hardware deployment. Simulation results confirmed that the system correctly handles multiple floor requests, direction changes, and timed door operations. The use of testbenches further validated the correctness and robustness of the design under various operational scenarios. Additionally, FSM-based structuring helped reduce overall design complexity by dividing the system into smaller, manageable functional states, making the design easier to understand, test, and modify.

The deployment of the design on FPGA platforms enhances its real-time applicability and flexibility. FPGA-based implementation allows faster execution, reconfigurability, and scalability, making the system suitable for further development. Overall, the project bridges theoretical concepts of sequential logic with practical hardware implementation, strengthening understanding of digital system design principles.

In conclusion, the proposed elevator controller meets its intended objectives by delivering a reliable, efficient, and well-structured control system. It clearly demonstrates how FSM simplifies complex control operations and ensures safe and systematic elevator functioning. The design also provides a strong foundation for future improvements in embedded systems and automation applications.

Future Scope

The current implementation focuses on a basic two-floor elevator system; however, it can be significantly enhanced to meet real-world requirements. One of the primary extensions is scaling the system to support multiple floors, making it suitable for residential complexes, commercial buildings, and industrial environments.

Advanced request handling strategies can also be introduced, such as priority-based scheduling for emergency cases, which ensures faster response for critical situations. Safety mechanisms like overload detection, fire emergency response, and obstacle detection sensors can be integrated to improve system reliability and passenger protection.

Furthermore, the system can be implemented on real-time FPGA hardware for physical deployment and performance validation. Integration with Internet of Things (IoT) technology can enable remote monitoring, control, and diagnostics of elevator operations. Mobile applications may also be developed to allow users to call elevators remotely and track their status.

Artificial Intelligence (AI)-based optimization techniques can be explored to analyze usage patterns and reduce waiting time by predicting demand and optimizing movement. Energy-efficient algorithms can further enhance system performance by minimizing unnecessary motion and power consumption.

Thus, the proposed system provides extensive opportunities for enhancement, making it adaptable for future smart building technologies and intelligent automation systems.

References

[1] M. Kumar, P. Singh, and S. Singh, “A VLSI Implementation of Parallel Phase Lift Controller Using Verilog HDL,” *International Conference on Materials, Alloys and Experimental Mechanics*, India, 2017.

[2] M. Siddiqui and K. Hasan, “Synthesis & Simulation Model of Parallel Lift Controller Using Verilog,” *International Journal of Engineering Development and Research*, vol. 3, no. 3, pp. 1–5, 2015.

[3] P. Kiran Babu and H. Rao, “A VLSI Implementation of Three-Lift Controller Based on Verilog,” *International Journal of Engineering Trends and Technology*, vol. 6, no. 1, pp. 43–47, 2013.

[4] B. Vaidya and D. Anupama, “A Study of Different Finite State Machine Design and Efficient Coding Techniques,” *Journal of Information, Knowledge and Research in Electronics and Communication Engineering*, vol. 2, no. 2, pp. 493–496, 2008.

[5] I. Chitussin, A. Potapov, and A. Gmur, “Finite State Machine Design and VHDL Coding Techniques,” *10th International Conference on Development and Application Systems*, Suceava, Romania.