Université Saint-Joseph de Beyrouth

Faculté d'ingénierie

Institut national des télécommunications et de l'informatique

Firmware Final Project

Submitted By:

Melody El Hage – melody.hage@net.usj.edu.lb

Supervisor(s):

Dr. Majdi Richa

Date:

10/01/2025

1. Abstract:

This project focuses on creating a 4-floor elevator control system that demonstrates teamwork, advanced system design, and efficient embedded firmware management. The system handles key functions like floor selection, controlling elevator movement, and displaying the current floor, all built on reliable hardware and optimized firmware. Inside the elevator cabin, the user experience is enhanced with real-time updates, including temperature, time, and date, made possible through an LM35 temperature sensor and an I2C-based Real-Time Clock (RTC) module. Safety and efficiency are prioritized, with built-in error-handling mechanisms ensuring smooth and secure operation.

This report dives into the conceptual design, technical structure, and step-by-step implementation of the elevator system. The aim is to deliver a dependable, user-friendly, and innovative solution, showcasing practical applications of embedded system concepts while leaving room for future improvements.

2. Introduction:

Elevators are a vital part of modern buildings, offering reliable and efficient vertical transportation. Developing an elevator control system involves seamlessly integrating hardware and software components to ensure smooth operation, safety, and user convenience. This project centers on building a 4-floor elevator controller that serves the Ground Floor (GF), Floor 1 (F1), Floor 2 (F2), and Floor 3 (F3), providing a real-world application of embedded system concepts.

The system is designed with features that prioritize user-friendliness and efficiency, ensuring a smooth and intuitive experience for users.

- **Floor Indication:** Each floor, along with the elevator cabin, is equipped with 7-segment displays that clearly show the current floor in real time, ensuring users are always informed of the elevator's location.

- **User Input Buttons:** Each floor is equipped with UP and DOWN buttons, allowing users to call the elevator in their desired direction. Inside the cabin, individual buttons let passengers select specific floors with ease.
- **Visual Indicators:** LED arrows on each floor provide clear visual cues, showing whether the elevator is moving UP or DOWN, enhancing usability and feedback for passengers.
- **Sensor Feedback:** The system integrates limit switches to deliver precise feedback on the cabin's position and speed adjustments, ensuring accurate control and a seamless riding experience.
- **Real-Time Information:** Inside the cabin, additional displays provide live updates on temperature, time, and date, powered by an LM35 temperature sensor and an I2C-based Real-Time Clock (RTC). These features enrich the user experience by adding convenience and functionality.
- **Efficient Motor Control:** The motor supports dual-speed and dual-direction operation, enabling smooth transitions between floors while optimizing energy use for an efficient and reliable system.

The schematic highlights the seamless integration of all system features, with a centralized microcontroller (PIC18F67K22) at its core, efficiently managing input, processing, and output tasks. The integration of an I2C-based RTC module and an LM35 temperature sensor ensures precise real-time data collection, offering users valuable environmental insights. Inside the cabin, internal displays work in tandem with floor indicators to provide clear, intuitive visual feedback, significantly enhancing the user experience.

By combining advanced hardware components with meticulously optimized firmware, this project aims to deliver a robust, reliable, and secure elevator control system. It exemplifies essential principles of system design, real-time data processing, and user-focused interface development, creating a versatile and future-ready solution for modern infrastructure needs.

3. Firmware Architectures:

The firmware serves as the backbone of the elevator system, orchestrating seamless communication between all hardware components. Designed with a modular approach, each module is dedicated to a specific function, ensuring the system operates efficiently and reliably. Together, these modules work in harmony to deliver a smooth, intuitive, and user-friendly experience. The microcontroller, communication protocols, and task management are central to enabling this robust functionality.

- **Microcontroller Management:**
  - The PIC18F67K22 microcontroller serves as the brain of the system, efficiently managing all inputs, processing, and outputs.
  - It supports multitasking, ensuring operations like motor control, data processing, and user interactions occur simultaneously without delays.
  - Dedicated interrupt handling prioritizes time-sensitive events, such as button presses and sensor feedback, ensuring smooth and responsive operation.
- **Input Module:**
  - Captures input from UP and DOWN buttons on each floor and floor selection buttons inside the cabin.
  - Monitors signals from limit switches to accurately detect the cabin's position and adjust speed during movement.
  - Employs debouncing techniques to eliminate false triggers and ensure reliable input processing.
- **Output Module:**
  - Updates 7-segment displays on each floor and inside the cabin to show real-time floor information.
  - Manages LED indicators to visually communicate the elevator's movement direction (UP or DOWN).
  - Enhances user experience by displaying real-time data, such as time, date, and temperature, inside the cabin.
- **Motor Control Module:**

    - Controls the motor's direction (UP or DOWN) and adjusts between high and low speeds for smooth and accurate cabin movement.
    - Uses pulse-width modulation (PWM) to enable precise motor control and optimize energy efficiency.
    - Implements controlled acceleration and deceleration for enhanced passenger safety and comfort.
- **Real-Time Data Module:**
    - Retrieves accurate time and date data using an external I2C RTC module for cabin display.
    - Processes analog temperature data from the LM35 sensor, converting it into a user-friendly format for users.
    - Utilizes I2C communication for reliable interaction with the RTC module, ensuring synchronized and accurate data updates.
- **Task Management:**
    - Employs a scheduler to handle multiple concurrent tasks, such as motor control, data updates, and safety checks.
    - Prioritizes tasks based on their importance, ensuring smooth and uninterrupted system performance.
- **Safety Module:**
    - Detects and addresses system errors, such as invalid button inputs, sensor failures, or motor malfunctions.
    - Implements fallback mechanisms to maintain safe and reliable operation in all scenarios.
    - Provides alerts and safety measures to prevent accidents and ensure system security.

By integrating these modules into a unified architecture, the firmware ensures that the elevator system functions reliably, efficiently, and safely. The combination of a powerful microcontroller, I2C communication, and efficient task scheduling optimizes the system's performance, making it scalable and adaptable to future enhancements and evolving requirements.

4. Block Diagrams:

The block diagrams below showcase the structure and workflow of the 4-floor elevator controller system, emphasizing the interaction between the hardware components and the microcontroller. These diagrams visually represent the modular design of the system and the communication pathways that ensure seamless operation.

a) System Overview Block Diagram:

The system revolves around the **PIC18F67K22 microcontroller**, which functions as the central control unit, managing inputs, processing data, and controlling outputs. The microcontroller facilitates seamless communication among various components, ensuring the smooth operation of the elevator system. Below are the primary components involved:

Input Devices:

- UP and DOWN Buttons: Located on each floor, these allow users to call the elevator in the desired direction.
- Cabin Floor Selection Buttons: Enable passengers inside the cabin to select their destination floors.
- Limit switches : Provide real-time feedback on the elevator cabin's position and control speed adjustments during movement.

i. Output Devices:

- 7-segment displays : Show the current floor number on each floor and inside the cabin, providing real-time information to users.
- LED indicators : Use visual cues to indicate the elevator's movement direction (UP or DOWN) for enhanced user feedback.

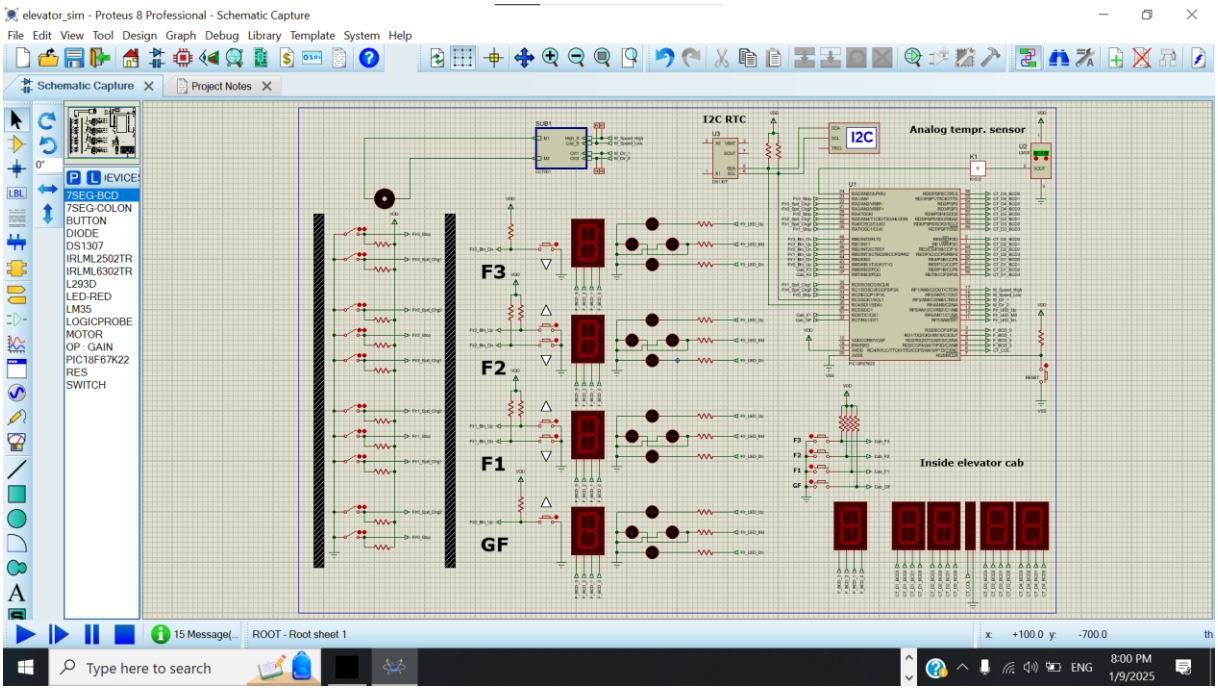
ii. Sensors:

- LM35 temperature sensor : Measures and displays the cabin temperature, improving user comfort.
- I2C-based RTC module : Ensures accurate real-time data for time and date display, enhancing the system's functionality.

iii. Motor Control:

iv. Dual-speed, dual-direction motor : Utilizes Pulse Width Modulation (PWM) signals to manage the elevator's movement with precision.

- Dual-Speed: Allows smooth acceleration and deceleration, ensuring passenger safety and comfort during transitions between floors.
- Dual-Direction: Controls upward and downward motion, enabling efficient and reliable cabin movement.



System Overview: High-level schematic showing all components and their connections to the microcontroller.

b) Input System Diagram:

The input subsystem acts as the bridge between users and the elevator system, capturing commands and providing real-time feedback about the cabin's position. This subsystem is critical for ensuring smooth operation, accurate control, and user safety. The main components include :

i. Buttons:

- UP and DOWN buttons Located on each floor, these allow users to call the elevator in their desired direction (up or down).
- Cabin floor selection buttons : Let passengers inside the cabin choose their destination floor.

- All buttons are connected to the microcontroller, which processes their signals to determine the appropriate elevator movement.t.

ii. Limit Switches:

- Provide precise feedback on the elevator cabin's position and speed during operation.
- For example:

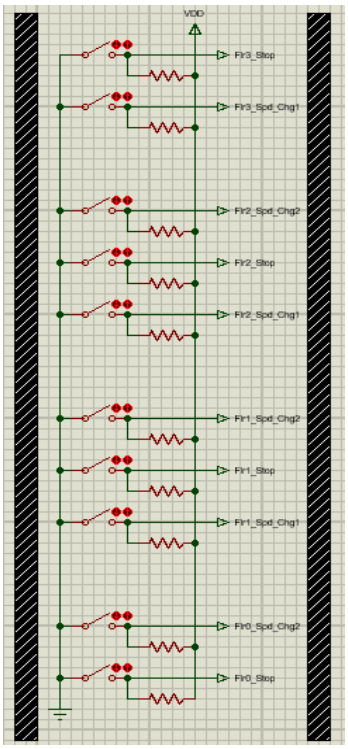
  When the cabin reaches a specific floor, the limit switch sends a signal to the microcontroller, instructing it to stop the motor and align the cabin accurately.

  This ensures smooth stops and prevents overshooting or misalignment.

iii. Connection and Processing :

- Both the buttons and limit switches are directly wired to the GPIO pins of the PIC18F67K22 microcontroller**.**
- The microcontroller processes these input signals in real time, triggering corresponding actions such as activating the motor, updating displays, or controlling LED indicators.

The buttons and limit switches are directly interfaced with the GPIO pins of the PIC18F67K22 microcontroller**.** The microcontroller processes these input signals in real time and triggers the necessary actions, such as initiating elevator movement, stopping the motor, or updating the floor displays and LED indicators.
This direct connection ensures fast and accurate responses to user commands and cabin position feedback, enhancing the system's overall reliability and efficiency.



Input Subsystem: buttons for capturing user commands (e.g., UP, DOWN, and floor selection) and limit switches for providing real-time feedback on the elevator cabin's position. Both are directly connected to the microcontroller, enabling precise control and responsive operation.

c) Output Subsystem Diagram:

The output subsystem ensures users receive clear and real-time visual feedback about the elevator's status and direction of movement. This enhances the user experience and provides vital operational information. The key components of this subsystem include:

i. 7-Segment Displays:

- Installed on each floor and inside the cabin to display the current floor number in real time.
- Updates dynamically as the elevator moves, keeping users informed of the cabin's position.

ii. LED Indicators:

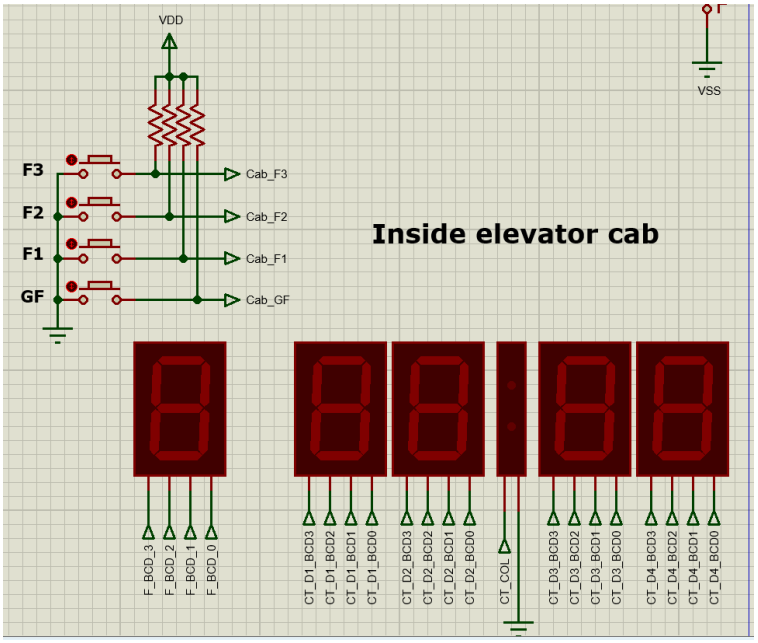
- Located on each floor, these LEDs visually indicate the elevator's movement direction (UP or DOWN).
- Provides intuitive and immediate feedback to users waiting for the elevator.

iii. Cabin Real-Time Displays:

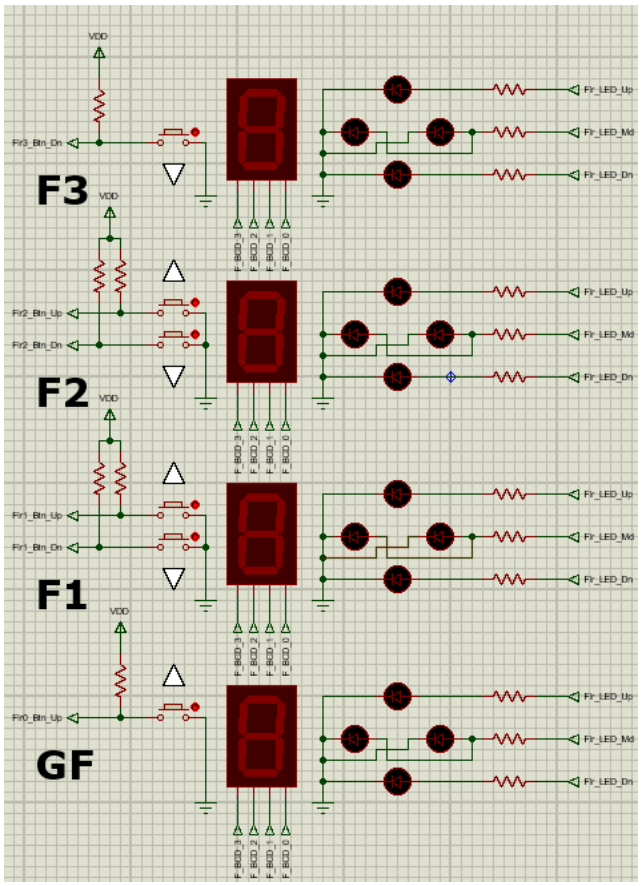
- Displays additional data inside the elevator cabin, such as :

  Temperature: Measured by the LM35 sensor.

  Time and Date: Provided by the I2C-based RTC module.
- Enhances user comfort by offering real-time environmental information



Internal Cabin Displays: Floor selection buttons and real-time data displays inside the elevator cabin connected to the microcontroller.



Output Subsystem: Displays and LED indicators for floor number and movement direction connected to the microcontroller.

d) Motor Control Subsystem :

The motor control subsystem is integral to ensuring the elevator cabin moves smoothly and efficiently between floors. It combines precise direction control, speed adjustments, and energy-efficient operation. The components include:

i. Motor Driver Circuit:

- Controls the motor's direction (UP or DOWN) and speed (high or low).

- Receives input from the microcontroller to manage smooth acceleration and deceleration, providing a seamless ride experience.

ii. Dual-Speed Functionality:

- Supports two operating speeds for optimal performance:
  High Speed : Used for regular movement between floors.
  Low Speed: Ensures precise alignment when the cabin stops at a floor, enhancing accuracy and safety.

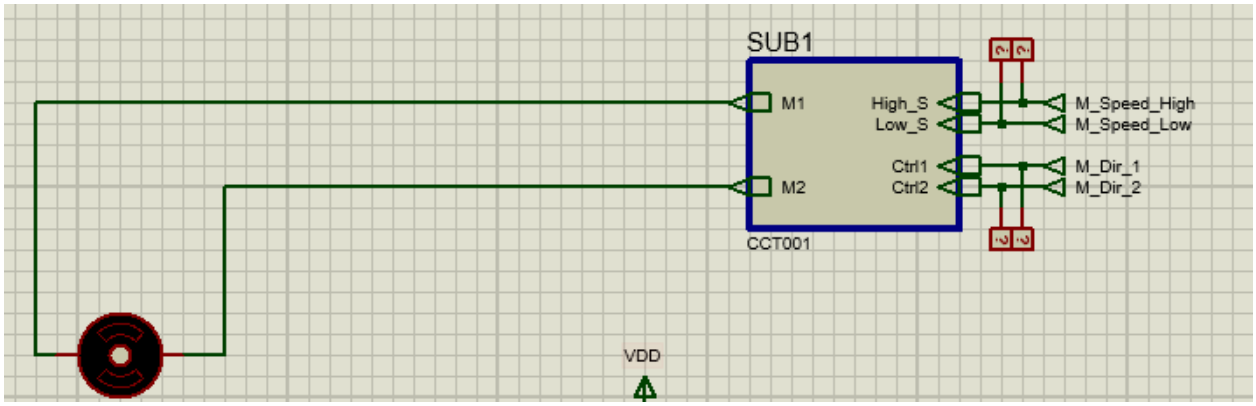
iii. PWM Signals:

- The microcontroller generates Pulse-Width Modulation (PWM) signals to regulate motor speed.
- PWM ensures energy efficiency and reduces wear on the motor, contributing to the system's longevity.

i. Direction Control:

- Specific GPIO pins on the microcontroller determine the motor's direction:
  UP: The motor rotates in one direction to move the cabin upward.
  DOWN: The motor rotates in the opposite direction to move the cabin downward.

This subsystem is crucial for delivering smooth, safe, and efficient elevator operation, ensuring precise cabin movement while prioritizing user comfort and system reliability.



Output Subsystem: Includes 7-segment displays for showing the floor number and LED indicators for visually signaling the elevator's movement direction (UP or DOWN). All components are directly connected to the microcontroller, which updates them in real time to provide clear and accurate feedback to users.

e) Real-Time Data Subsystem :

The real-time data subsystem enriches the elevator system by delivering accurate and valuable environmental information to users. It integrates sensors to measure and retrieve key data, ensuring this information is displayed promptly and accurately. The main components include :

i. LM35 Temperature Sensor:

- Measures the cabin's temperature and provides analog signals to the microcontroller.
- The microcontroller processes these signals and converts them into a user-friendly format for display inside the cabin.

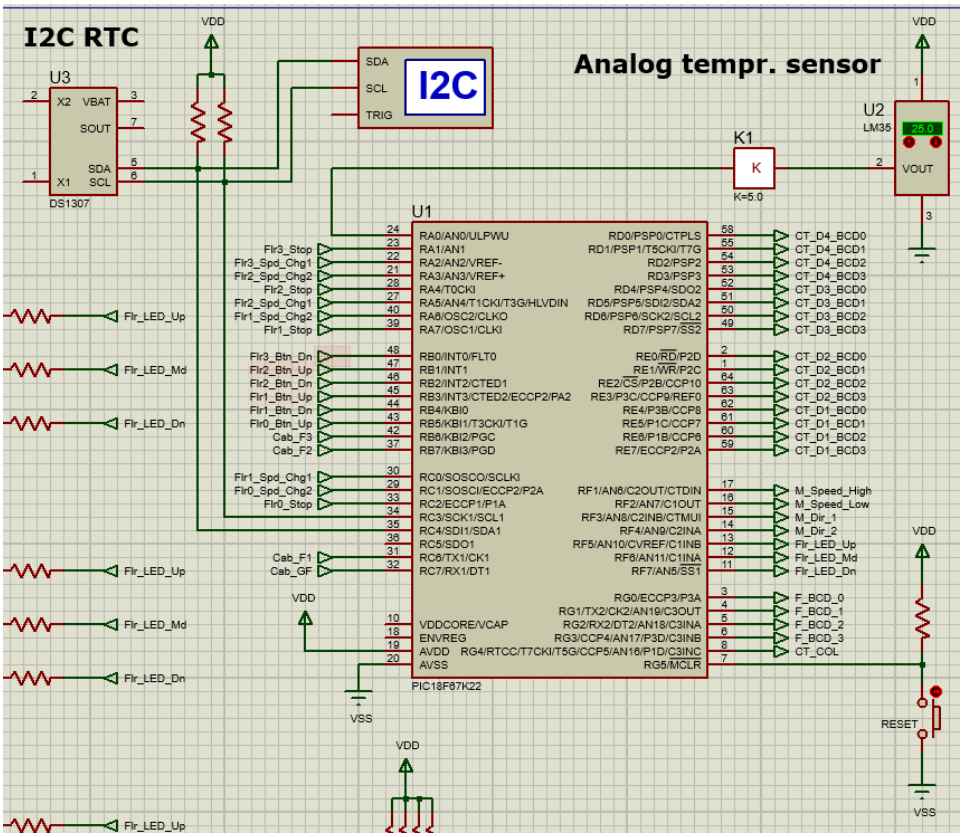
ii. I2C-Based RTC Module :

- Retrieves real-time time and date information.
- Communicates seamlessly with the microcontroller via the I2C protocol to ensure accurate and synchronized data updates.

iii. Microcontroller's Role:

- Real-Time Clock (RTC) Data Retrieval:
  The microcontroller communicates with the RTC module using the I2C bus to continuously fetch accurate time and date information.
- Temperature Data Processing:
  It reads the analog output from the LM35 temperature sensor through its ADC (Analog-to-Digital Converter) pins, converting the signal into a digital format for further processing.
- Display Updates:
  The processed time, date, and temperature values are sent to the internal cabin displays in real time, ensuring users have constant access to up-to-date environmental and temporal information.

This subsystem enhances the user experience by providing timely and accurate environmental information, such as temperature, time, and date. It showcases the seamless integration of real-time data handling within the elevator system, emphasizing both functionality and convenience for users.



Motor Control Subsystem: Motor driver circuit controlled by PWM signals for precise speed and direction management.

5. Flowcharts:

This section provides flowcharts that visually represent the logical processes within the elevator controller system. Each flowchart outlines the steps involved in handling inputs, controlling the motor, managing real-time data, and ensuring system safety.

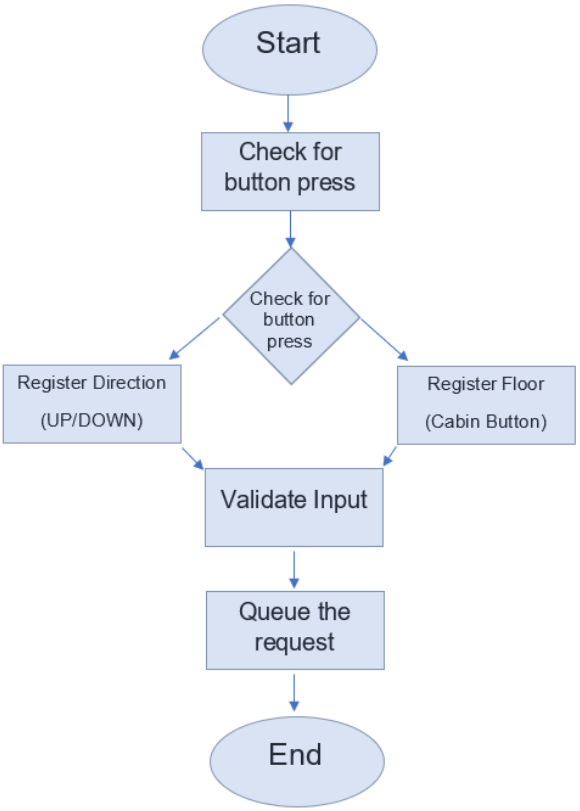
a) Input Handling Flowchart:

The input handling process is designed to detect and process user commands, such as floor requests and direction calls. The flowchart below demonstrates the steps taken to manage inputs and forward them to the appropriate subsystems.

Steps in the Input Handling Process:

1. **Start:** Continuously monitor all input buttons for activity.

2. **Check for Button Press:** If a button is pressed, proceed to the next step. If no button is pressed, loop back to continue monitoring.
3. **Identify Button Type:** UP/DOWN Button: Register the direction call (UP or DOWN). Cabin Button: Register the requested floor.
4. **Validate Input:** Check that the button press corresponds to a valid floor or direction. Ignore any invalid or duplicate requests.
5. **Queue the Request:** Add the validated request to the command queue for further processing by the system.
6. **End:** Return to the start to handle the next input.



b) Motor Control Logic Flowchart:

The motor control logic ensures the elevator cabin moves smoothly and efficiently between floors by dynamically adjusting the motor's direction and speed based on floor requests and feedback from limit switches. The process includes the following steps:

i. **Monitor Current Position:** The system continuously monitors the elevator's current position using limit switches or internal counters.

ii. **Compare Requested Floor:** The microcontroller compares the current floor to the requested floor to determine the required direction:

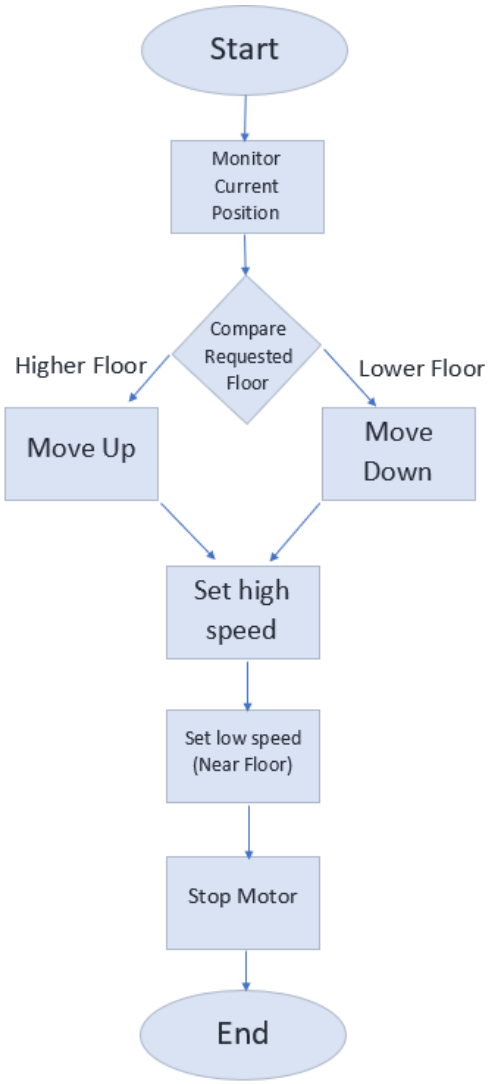
- If the requested floor is above the current floor, the motor moves UP.
- If the requested floor is below the current floor, the motor moves DOWN.

iii. **Set High-Speed:** During most of the travel, the motor operates at high speed for efficient movement.

iv. **Set Low Speed (Near Floor):** As the elevator approaches the requested floor, the motor reduces to low speed for smooth stopping.

v. **Stop Motor:** Once the elevator reaches the requested floor, the motor stops, ensuring precise alignment with the floor.

vi. **Repeat:** The process repeats for the next floor request in the queue.



This flowchart visually represents the step-by-step logic involved in controlling the elevator motor.

c) Real-Time Data Management Flowchart:

The real-time data management system ensures that the elevator cabin display is always updated with accurate and relevant information, such as the current time, date, and temperature, providing a better experience for passengers. Here's how it works:

i. **Initialization:**

When the system starts, the Real-Time Clock (RTC) module and LM35 temperature sensor are initialized to prepare for collecting data.

ii. **Retrieve Time and Date :**

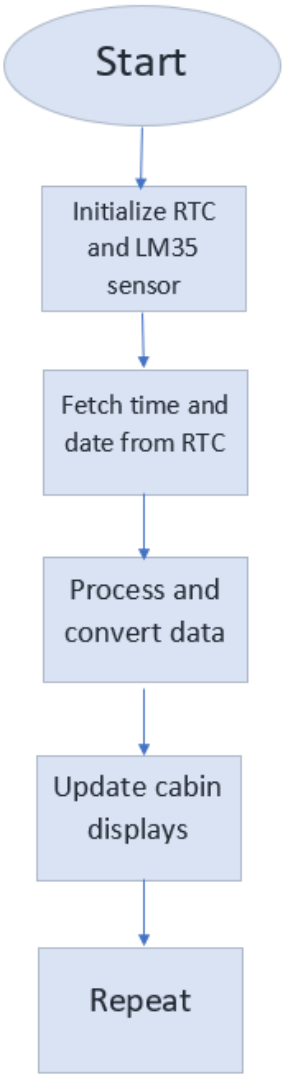
The system communicates with the RTC module via I2C to fetch the current time and date.

iii. **Read Temperature:**
The LM35 sensor measures the cabin's temperature, and the system converts the analog reading into a digital format that's easy to read.

iv. **Process Data:**
The raw data from the RTC and temperature sensor is formatted and processed to ensure it's ready for display.

v. **Update Displays:**
The processed information (time, date, and temperature) is shown on the cabin's screen in real time, keeping passengers informed.

vi. **Continuous Updates:**
This process runs continuously, ensuring the information displayed is always current.



This flowchart visually breaks down the steps, showing the smooth sequence of operations that keep passengers informed with up-to-date information throughout their journey.

d) Error Detection Flowchart:

The error detection subsystem ensures the elevator operates safely and reliably by continuously monitoring for faults or malfunctions and taking immediate action when needed. The process is as follows:

i. **Monitor Components:** Continuously monitor inputs (e.g., buttons, limit switches) and outputs (e.g., displays, motor) to detect irregularities or unusual behavior.

ii. **Detect Errors:** Check for common issues, such as:

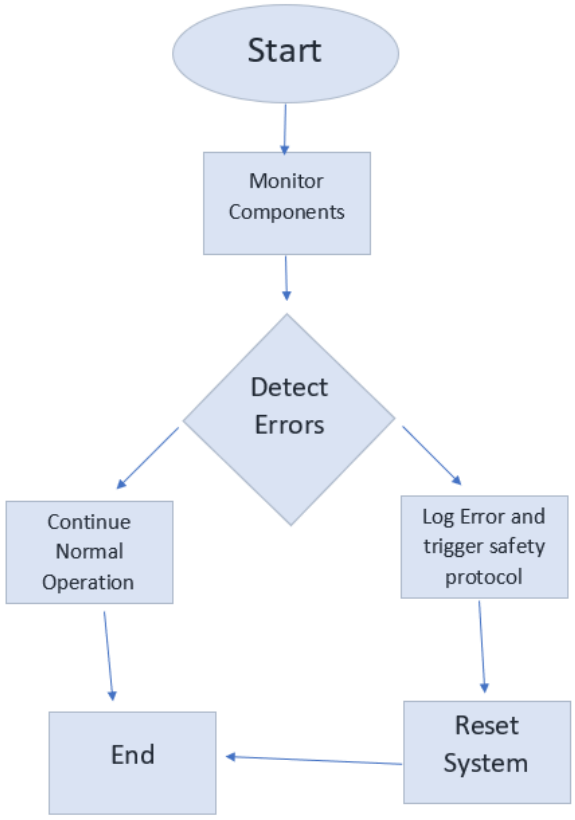
- **Invalid Button Presses:** For example, pressing UP and DOWN simultaneously.
- **Sensor Malfunctions:** Failures in the RTC or LM35 temperature sensor.
- **Motor Faults:** Issues like the motor not responding or overheating.

iii. • **Handle Errors:**

- Log the error for debugging and troubleshooting.
- Trigger safety protocols, such as stopping the motor, resetting the system, or alerting maintenance personnel.

iv. **Reset System:** After resolving the error, reset the affected components to restore normal operation.

v. **Continue Normal Operation:** If no errors are detected, or once errors are handled, the system continues its regular functionality.



This flowchart provides a clear visual representation of the logical steps involved in identifying and addressing system errors, ensuring safety, reliability, and smooth operation.

6. Procedures:

This section outlines the comprehensive approach used to design, implement, and validate the elevator controller system. The methodology emphasizes precise execution, seamless hardware-software integration, user safety, and operational efficiency.

1) **System Design and Setup:**

This phase focused on creating a robust hardware foundation and preparing components for integration.

i. **Hardware Setup:**

- **Component Selection:**
  - **Buttons:** Floor selection and UP/DOWN call buttons for user interaction.
  - **7-Segment Displays :** Show the current floor number dynamically on each floor and in the cabin.
  - **LED Indicators:** Indicate the elevator's movement direction (UP/DOWN).
  - **Motor and Motor Driver:** Supports dual-speed and dual-direction control, ensuring smooth and precise cabin movement.
  - **RTC Module:** Provides accurate time and date via I2C communication.
  - **LM35 Temperature Sensor:** Measures cabin temperature and sends analog data to the microcontroller.
- **Wiring:**
  - Buttons, displays, LEDs, sensors, and the motor driver are connected to the PIC18F67K22 microcontroller.
  - GPIO pins handle digital signals; ADC pins process LM35's analog input; I2C protocol is used for RTC communication.
- **Preliminary Tests:**
  - Verification**:** Verify wiring continuity and proper power levels using a multimeter.
  - Individual Component Testing: Test components individually (e.g., buttons, displays, motor, sensors) to ensure proper functionality.

2) **Firmware Planning:**

The firmware manages hardware interactions using a modular design for ease of debugging and future scalability.

Modules:

- **Input Handling:** Detects and debounces button presses, validates inputs, and queues valid requests.
    - Motor Control: Generates PWM signals to adjust speed, determines direction, and uses limit switches for precise stopping.
    - Real-Time Data Management: Reads and processes time, date, and temperature data from the RTC and LM35 sensors.
    - Output Management: Updates 7-segment displays and LEDs to show floor status and direction.
    - Error Handling: Detects invalid inputs, sensor failures, or motor faults and triggers safety protocols.
- **Task Scheduler:**
    - Manages concurrent tasks like motor control, data processing, and user inputs.
    - Prioritizes high-importance tasks (e.g., stopping at floors) while scheduling periodic updates for lower-priority tasks.

3) **Testing and Debugging:**

Extensive testing ensures the system's reliability and identifies issues early.

- **Subsystem Testing:**
    - Inputs**:** Simulate button presses and validate debouncing logic.
    - Motor**:** Test speed transitions, direction changes, and limit switch accuracy.
    - Outputs**:** Verify 7-segment displays and LEDs reflect the elevator's state accurately.
    - Sensors**:** Compare RTC and LM35 readings with external references for accuracy.
- **Debugging:**
    - Address issues like delayed responses, display flickering, or motor synchronization problems.

4) **Full System Integration:**

Integrating subsystems ensures the elevator operates as a cohesive unit.

- **Integration:**
  Combine input handling, motor control, real-time data, and output updates into a single workflow.
- **Full-System Testing:**
  Conduct real-world simulations, such as simultaneous requests from multiple floors and stress tests with frequent button presses.
- **Stress Testing:**
  Ensure synchronization to avoid conflicts or delays during operation.

5) **Deployment and Validation:**
   The system is deployed in a simulated or real-world environment and evaluated for performance.
   - **Deployment** :
     Install the system on a test rig and allow testers to interact with it.
   - **Validation** :
     Gather feedback to identify areas for improvement, confirm safety protocols work as intended, and ensure the system meets design objectives like reliability, efficiency, and user-friendliness.

This methodology reflects a structured approach to creating a robust, user-friendly, and reliable elevator controller system, ensuring its success in real-world scenarios.

7. Problems Encountered:

During the development of the 4-floor elevator controller system, several challenges arose that required careful analysis and troubleshooting. Addressing these issues was critical to ensuring the system's functionality, safety, and reliability. Below are the key challenges and the solutions implemented:

1) **Synchronization Between Input and Motor Control**

- **Problem:** The elevator motor occasionally failed to respond promptly to button presses, causing operational delays.
- **Solution:** The task scheduler was optimized to prioritize critical tasks like motor control, ensuring a faster and more reliable response to user inputs.

2) **Display Timing Issues**
   - **Problem:** The real-time display for time, date, and temperature did not consistently update every 10 seconds. Delays were inconsistent and exceeded the required interval.
   - **Solution:** The timing mechanism in the firmware was adjusted to maintain consistent updates. After multiple iterations, the issue was resolved, achieving the required 10-second update interval.
3) **Limit Switch Calibration**
   - **Problem:** Limit switches occasionally failed to detect the elevator's exact position, causing inaccuracies in stopping at the requested floor.
   - **Solution:** Recalibration and sensitivity adjustments were made to the limit switches, ensuring reliable position detection and precise alignment with each floor.
4) **Temperature Sensor Readings**
   - **Problem:** The LM35 temperature sensor occasionally produced erratic readings, particularly during extended operation.
   - **Solution:** A data filtering algorithm was implemented to smooth out erratic values, ensuring stable and accurate temperature readings.
5) **Direction Indicators**
   - **Problem:** The LED arrows indicating the elevator's movement direction (UP/DOWN) occasionally flickered or displayed incorrect directions.
   - **Solution:** A wiring issue was identified and resolved, ensuring the LEDs received stable signals from the microcontroller for consistent and accurate direction indication.
6) **Motor Speed Transition**
   - **Problem:** The transitions between high and low motor speeds near floors were jerky, leading to an uncomfortable user experience.

- **Solution:** The PWM signals controlling the motor were fine-tuned to enable smoother acceleration and deceleration, ensuring seamless speed transitions.

By systematically addressing these challenges, the team ensured that the elevator controller system operated efficiently, reliably, and safely. These obstacles provided valuable learning opportunities, ultimately contributing to a more robust final design.

8. Possible Expansions:

1) **Support for More Floors:**
   - **Description:** Expand the system to accommodate high-rise buildings with more than 4 floors.
   - **Implementation:**
     - Modify the microcontroller firmware to support additional buttons and display outputs.
     - Implement multiplexing for 7-segment displays to minimize GPIO pin usage.
     - Upgrade the motor and driver system to handle extended shaft lengths and higher loads.
2) **Voice Announcements:**
   - **Description:** Improve accessibility and user convenience by audibly announcing the current floor, movement direction, or specific messages (e.g., "Doors opening").
   - **Implementation:**
     - Add a pre-recorded sound module and speaker system.
     - Program the microcontroller to trigger announcements based on elevator position and user actions.
3) **Enhanced Emergency Features:**
   - **Description:** Increase safety with advanced emergency functionalities:
     - **Emergency Stop Button:** Allows users to halt the elevator immediately during emergencies.

- **Battery Backup:** Ensures elevator operation during power outages.
- **Alarm System:** Activates audible alarms in case of faults or prolonged inactivity.

➢ **Implementation:**

- Integrate an uninterruptible power supply (UPS) or battery backup.
- Update the microcontroller firmware to prioritize emergency signals and activate alarms.

4) Modernized User Interfaces:

➢ **Description:** Replace traditional buttons with touchscreen interfaces to provide a more modern and interactive user experience.

➢ **Implementation:**

- Install an LCD touchscreen panel for floor selection and dynamic content (e.g., advertisements, building maps).
- Update the firmware to support touch inputs and advanced UI features.

5) Weight Dedication and Load Monitoring:

➢ **Description:** Enhance safety by integrating weight sensors to monitor cabin load and prevent overloading.

➢ **Implementation:**

- Place load cells under the cabin floor to measure weight.
- Program the microcontroller to compare load data with the elevator's weight capacity and issue alerts if the cabin is overloaded.

Implementing these enhancements would elevate the system into a scalable, high-performance solution, suitable for diverse applications. These additions prioritize safety, user experience, and future-proofing, ensuring the elevator controller system remains adaptable to evolving demands.

9. Conclusion:

The development of the 4-floor elevator controller system exemplifies the seamless integration of hardware and firmware to deliver a reliable, efficient, and user-friendly

solution. Through modular design principles and systematic testing, the project addressed key challenges, including precise motor control, real-time data management, and responsive user interfaces.

The system offers essential functionalities such as floor indication, direction signaling, and real-time display of time, date, and temperature, ensuring a smooth and intuitive user experience. Safety and efficiency were prioritized throughout the design, with features like error detection, smooth speed transitions, and precise stopping mechanisms ensuring dependable performance.

This project highlights the practical application of embedded system principles while emphasizing scalability and adaptability. Potential expansions, such as support for more floors, enhanced accessibility features, and integration with building management systems, demonstrate the system's ability to evolve and meet the demands of modern infrastructure.

In conclusion, this project serves as a robust foundational model for elevator control systems, balancing technical precision with user-centric design. It underscores the potential for innovation in automated transportation solutions and contributes meaningfully to advancements in embedded systems.