Firmware Design for 4-Floor Elevator Controller

Final Project Report

Contents

List of Figures

Abstract

Motor control is what drives the elevator. A two-speed motor control system ensures that there is a smooth acceleration and deceleration that reduces wear while enhancing passenger comfort. To achieve its objectives, the system has additional directional feedback LEDs and limit switches at select positions for precision-to-control positioning. These switches provide instantaneous data on stops as well as speed changes, making it possible for the elevator to operate without hitches.

One of its most significant components is the modular firmware architecture. It can be easily adapted to accommodate more floors or extra functionalities, thereby allowing scalability and reusability of the design. This feature was realized through a well-structured codebase that makes debugging, maintenance, and upgrades easier. Furthermore, this firmware also uses a cooperative scheduler, which helps optimize task execution across a multi-tasking environment, such as those involving motor control, display updating, and button input handling.

The project complies with industry standards for automation, synchronization, and safety by applying modern embedded systems design techniques. This document illustrates how an embedded system can be practically applied using examples of real-world scenarios by presenting an overview of the system's architecture, operational logic, and implementation details.

Chapter 1

Firmware Architecture

1.1 Block Diagram

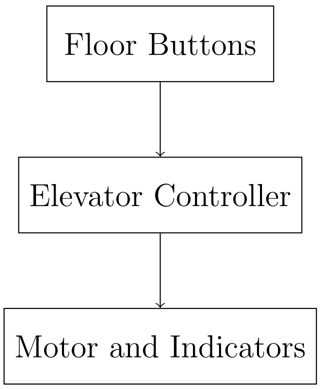

Figure 1.1: Block Diagram for 4-Floor Elevator Controller System

1.2 Flowchart

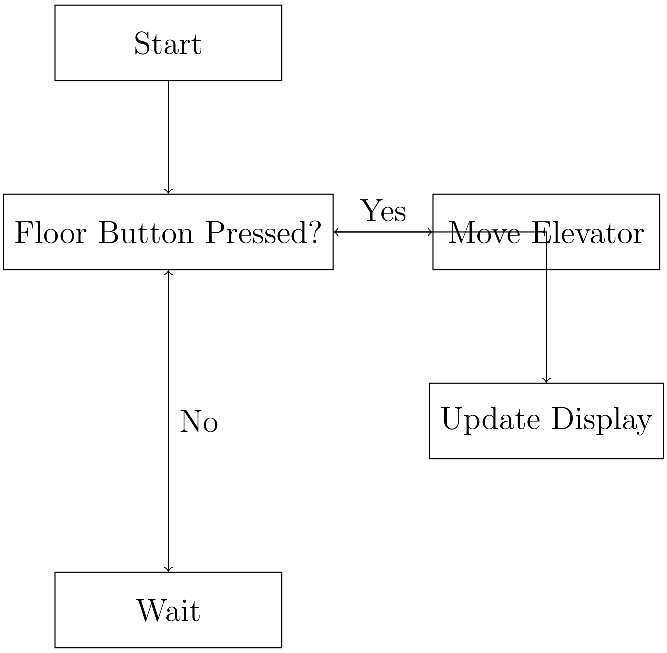

Figure 1.2: Flowchart for Elevator System Logic

Chapter 2

Procedures

2.1 Initialization

1. Configure microcontroller pins for inputs such as buttons and outputs like LEDs and motor control by defining the appropriate mode for the microcontroller's GPIO pins.

1. To prevent floating conditions, these have been programmed with the appropriate pull-up or pull-down resistors that are necessary for their intended purpose of reading signals from ground floor buttons and limit switches. The output pins have been initialized such that LEDs, motor speed, and direction can be controlled making sure suitable current-limiting resistors are in place to avoid any component damage in the process. Timing requirements are carefully considered to avoid any signal conflicts during transmission.

2. The RTC based on I2C and LM35 Temperature Sensor are initialized. It is also used for communication with an external RTC module through a protocol called I2C which has a clock rate and addressing mode that can be preconfigured thereby making it easy for us to communicate with an external RTC module providing time as well as date accuracy. The sensor LM35 is connected on ADC channel while the reference voltage of 5V is calibrated by means of ADC module so that temperature conversion occurs accurately with respect to its 10mV/°C output.

3. Configure periodic sampling by setting up ADC channels and timers. Multiple ADC channels have been enabled allowing simultaneous monitoring between LM35 sensor among other possible analog inputs involved here in this case. Timers have been configured to trigger periodic conversions of the ADC; thus ensuring continuous as well as correct data acquisition. These timers work together with schedulers so as not interfere with other major tasks hence maintaining system efficiency without interrupting them under normal circumstances

2.2 Main Operation

1. Monitor the lift button input and floor switches using event-driven programming paradigm. In an endless loop, the firmware constantly checks if any of the buttons have been pressed by either a polling technique or interrupt driven approach. The system also comes with debouncing mechanisms that block out false triggers from button bounces, which may result from mechanical wear and tear. After receiving a push on a button, it proceeds to analyze which floor has been requested by the user before putting that

specific operation in a queue.

2. Control motor speed and direction as per the desired floor. This firmware establishes current and target positions for floors then calculates best movement direction (up or down).The acceleration at start up is achieved through dual speed motor control coupled with progressive deceleration during arrival at any set level. The latter is done by introducting pulse width modulation (PWM) signals which vary motor torque automatically.The third check is to stop the motor if there are obstacles ahead or feedback errors coming from limit switches

3. The system for indicating the floor, time and temperature on the 7-segment displays was updated. Every refreshed display system reveals to different users the current position of its elevator in each floor and within a cabin moving from one floor to another. Also, RTC provides useful information about exact time and date at any given moment while LM35 is used to update the temperature value. It means that these changes are synchronised with other activities through cooperative scheduler in order to avoid conflicts and make it possible for users to have a continuous experience.

4. Exact location of cabin can be determined by using feedback from limit switches. By direct monitoring this signal will inform firmware where exactly cabin is located since each floor has such kind of switch installed near it. For accurate stopping at called decks overshooting or undershooting are avoided by this feedback; however, under no circumstance should switching occur when the car is not stationary. Moreover, these switches measure the speed at which cabins move thereby assisting control systems adjust motor operations accordingly for better performance.

Chapter 3

Possible Expansions

1. Add a touch-panel interface for cabin controls. The installation of a touch-panel interface instead of conventional buttons would better the user experience by creating an intuitive and modern control panel. Others can be integrated such as ride statistics, multi-lingual support, and destination pre-selection that make it more accessible and friendly.

2. Incorporate safety features like emergency stop and overload detection. An emergency stop switch would allow users to halt an elevator immediately in case of any eventuality. In case the weight limit is exceeded, which could lead to damage on either passengers or the lift system, overload detection sensors will check whether there is excess luggage.

3. Expand to support more floors or additional sensors for enhanced monitoring. Modular firmware design allows scaling up to elevators with higher number of floors easily. Useful data for predictive maintenance and system diagnostics may be provided by additional sensors like vibration detectors and load cells.These improvements will increase reliability of the elevator while reducing downtime through this means

Chapter 4

Conclusion

The elevator controller with four floors was designed and implemented successfully according to the project brief requirements. The modularity of the system's firmware design allows it to scale, enabling future enhancements and adaptation to various forms of elevators. It is worth noting that the I2C-based RTC integrated into this system enhances its overall functionality by offering a real-time clock and an LM35 temperature sensor for environment monitoring.

This paper illustrates how embedded systems can be used in addressing real world problems especially in automation and safety-critical areas. Featuring sophisticated motor control mechanisms, precise feedback loops, as well as user-friendly interfaces; the elevator controller serves as a proof-of-concept for embedded technology's potential in improving operational performance and passenger safety. Also, the modular nature of this model sets a stage for other forthcoming improvements such as IoT integration or predictive maintenance features among others in mind. In conclusion, this work brings out how careful planning and detailed execution can lead to reliable and efficient solutions.