Jurnal Kecerdasan Buatan dan Teknologi Informasi
Vol. 5, No. 3, September 2026, pp. 429–442
e-ISSN: 2964-2922, p-ISSN: 2963-6191

Open access

Original Research

Safety Monitoring Interface for Elevator Systems Using Bitwise Decoding and Finite State Machines

Ilham Maulana [a], R Wisnu Prio Pamungkas [b*], Fried Sinlae [c]

[a,b,c] Department of Informatics, Faculty of Computer Science, Universitas Bhayangkara Jakarta Raya, Bekasi, Indonesia

*Corresponding Author: wisnu.prio@dsn.ubharajaya.ac.id

ABSTRACT

This study develops an Elevator Safety Monitoring Interface that integrates a Bitwise Decoding method and a Finite State Machine (FSM) algorithm to monitor elevator operational conditions in real-time. Modern elevator systems rely on the integration of electromechanical components and embedded systems; however, condition monitoring is often performed manually, leading to limited real-time information, increased occupational health and safety (OHS) risks, and the absence of structured historical fault records. This study aims to design a VB.NET-based monitoring system capable of displaying elevator status in real-time, implementing a Bitwise Decoding method for efficient processing of binary data from a microcontroller, and applying the FSM algorithm to model elevator operational conditions deterministically. The Waterfall development model is employed, encompassing planning, analysis, design, implementation, and testing phases. Hardware used includes Arduino/ESP32 microcontroller, relay circuits as digital signal inputs, RS-485 module, and an RS-485 to USB converter. The monitoring software was developed using VB.NET with MySQL for activity log storage. The FSM is formally defined as a 5-tuple $M = (S, \Sigma, \delta, s_0, F)$, with states including IDLE, MOVING_UP, MOVING_DOWN, DOOR_OPEN, FIRE_MODE, INSPECTION, EARTHQUAKE_MODE, ERROR, and OFFLINE. Results demonstrate that the system successfully displays elevator operational status in real-time and effectively extracts elevator condition information from binary data with high efficiency and low latency. The developed monitoring system is expected to improve technician work efficiency and support elevator operational safety.

How to Cite:
I. Maulana, R. W. P. Pamungkas, and F. Sinlae, "Safety Monitoring Interface for Elevator Systems Using Bitwise Decoding and Finite State Machines," *J. Kecerdasan Buatan dan Teknol. Inf.*, vol. 5, no. 3, pp. 429–442, 2026.
Doi: https://doi.org/10.69916/jkbti.v5i3.522

KEYWORDS

Elevator Monitoring
Bitwise Decoding
Finite State Machine
VB.NET
Embedded System

ARTICLE INFO

Received: June 8, 2026
Accepted: August 17, 2026
Available Online: September 1, 2026

1. INTRODUCTION

Elevator systems, as part of modern vertical transportation infrastructure, rely on the integration of electromechanical components and embedded systems, particularly in the aspect of hardware-based data acquisition using microcontrollers and relay circuits. In conventional elevator systems, operational signals such as floor position, direction of movement, and door status are represented as discrete digital signals originating from the control system. The need for a data acquisition mechanism is critical for capturing and transforming elevator operational condition information into digital processing systems. Data collection from elevator systems is an essential aspect of the maintenance process and early fault detection [1]. The implementation of sensor-based and data acquisition systems enables more accurate identification of elevator conditions and supports continuous monitoring processes. Therefore, the use of microcontrollers as data acquisition units has become a widely adopted solution due to their capability to handle digital inputs in real-time and support integration with

modern data-driven monitoring systems.

From a software perspective, elevator monitoring systems require real-time data visualization capabilities that are informative and easily understood by users, particularly maintenance technicians. Software systems function as interfaces that process raw data into operational information through decoding and visual representation processes. In the context of industrial monitoring systems, computer-based interfaces enable continuous data streaming integration, thereby supporting rapid and accurate decision-making processes [2]. Furthermore, the concept of real-time monitoring is crucial in ensuring that every change in system conditions can be detected immediately without significant delay. This aligns with research stating that real-time information processing and visualization are essential components in industrial systems to provide intuitive information, support continuous monitoring, and improve decision-making quality [3]. Moreover, the implementation of real-time industrial monitoring systems integrating sensor networks and data processing algorithms has been proven capable of increasing fault detection accuracy by up to 40%, accelerating response time to abnormal conditions by 25%, and reducing unplanned downtime by 30% compared to conventional monitoring systems [4].

Data processing in elevator monitoring systems requires an efficient and deterministic approach, particularly in embedded systems that must respond within strict time constraints. In this context, "the most important property of RTESs is predictability" [5], so algorithmic approaches must be able to guarantee consistent execution time. The Finite State Machine (FSM) method is employed because it can represent the behavior of digital systems through structured state, input, and output models, and is widely used in the design of sequential logic blocks and control units in digital systems [6]. In the context of elevator monitoring systems, FSM is used to systematically represent operational conditions such as floor transitions, door status, and direction of movement through state transition mechanisms. Furthermore, the implementation of FSM in digital systems involves the use of logic elements such as decoders, memories, and combinational logic circuits to produce output functions based on binary representation of states and inputs [6]. This demonstrates that decoding techniques, including bit-based approaches, are an important part of digital data interpretation in embedded systems. This low-level data processing approach has been proven capable of achieving a latency reduction of up to 90% compared to cloud-based processing, with an average response time of only 10 milliseconds for safety-critical decision-making [4].

The integration of hardware, software, and algorithms forms a comprehensive and interconnected elevator monitoring system. Data acquired from hardware is processed using decoding techniques, then interpreted using logic models to determine system conditions, and finally visualized through a monitoring interface. This approach reflects the concept of hardware-software co-design in embedded systems, where components with high computational requirements are implemented in hardware, while others are executed in software to improve system efficiency and performance [7]. Real-time monitoring systems are designed to continuously observe system performance and detect and respond to changes in operational conditions. Modern monitoring technologies also provide real-time information, enabling relevant parties to identify and respond to potential problems more quickly [4].

The implementation of an integrated monitoring system has a significant impact on improving operational efficiency and occupational health and safety (OHS) aspects. Based on nationally recorded statistical data, there were 23 elevator accidents resulting in 17 fatalities in 2021, representing 20.91% of all special equipment accidents that occurred in the same period [8]. This condition confirms that elevator systems without adequate real-time condition monitoring harbor very real and non-negligible OHS risks. With a real-time monitoring system, technicians can continuously obtain elevator condition information without having to conduct direct inspections, thereby reducing workplace accident risks while accelerating the fault handling process. Furthermore, monitoring systems enable the recording of historical operational data that can be utilized for preventive analysis of potential damage, thus supporting the implementation of condition-based maintenance that has been proven capable of reducing maintenance costs by up to 20% [4].

Based on the aforementioned background, this study presents the development of a Safety Monitoring Interface for Elevator Systems Based on Bitwise Decoding Using the Finite State Machine Algorithm. The research aims to produce a prototype elevator monitoring system that improves technician work efficiency through accurate and real-time status visualization, while supporting the improvement of elevator operational safety through structured fault history recording. The integration of Bitwise Decoding and FSM as a unified processing

pipeline — rather than separate modules — represents the primary novelty of this work, addressing a gap in existing literature where both techniques are typically applied independently.

Prior research provides the theoretical and empirical foundation for the development of the elevator monitoring system in this study. Most existing studies focus on sensor-based and IoT-driven monitoring along with artificial intelligence approaches, yet a gap remains in the integration of low-level data processing with deterministic state validation mechanisms. Guo et al. (2021) developed a real-time elevator monitoring system based on multi-sensor fusion that significantly improved maintenance efficiency [9]. Their work serves as the primary reference for designing real-time monitoring architecture. Niu (2025) proposed a machine learning approach based on RO-SSVM for early elevator fault detection, achieving 99.82% accuracy, validating the urgency of data-driven monitoring [1]. The fundamental difference lies in the processing approach: this study employs low-level bitwise decoding as a computationally lighter alternative. Al-Kodmany (2023) reviewed smart elevator systems integrating advanced mechanical and electrical approaches including regenerative drives for energy efficiency and enhanced safety, providing context for modern elevator operational requirements [10]. Arin et al. (2025) designed a Field Service Management application using the Double Diamond approach for IoT-based service monitoring, demonstrating the importance of usability-oriented design in maintenance systems [11]. Niemi et al. (2024) demonstrated the effectiveness of bit-field decoding in the context of CAN bus systems for electric vehicle battery recycling, proving that bitwise masking and bit shifting operations can accurately extract information with minimal latency [12]. This principle is adapted in this study for the elevator serial communication context. Khairullah (2024) validated the use of FSM in safety-critical systems through formal verification, strengthening the theoretical foundation for using FSM in this study as a deterministic and reliable state validation mechanism [13]. Jami et al. (2023) developed an IoT-based elevator monitoring and control system using ESP32 microcontrollers and cloud communication, validating the reliability of ESP32 as a data acquisition unit in elevator environments [14]. Li et al. (2024) designed a non-intrusive online monitoring system for traction elevators using STM32 and NB-IoT, providing an architectural reference particularly for data visualization and maintenance efficiency aspects [8]. Zhao et al. (2024) developed IoT monitoring systems for large-scale facilities, demonstrating that real-time IoT technology enables "real-time surveillance and assessment of the operational conditions of intricate mechanical systems" [15]. Cacciuttolo et al. (2023) reviewed low-cost sensor technologies for monitoring safety issues in mining activities, reinforcing the cross-industry applicability of real-time monitoring approaches [16]. Pambudi and Aji (2021) developed a serial communication module connecting Visual Basic applications with Arduino, establishing the basis for VB.NET-Arduino serial communication implementation in this study [17]. Lin and Peng (2025) demonstrated that vibration analysis combined with region segmentation algorithms is effective for elevator health diagnosis via online processing [18]. Man et al. (2025) reviewed the reliability and safety evolution of elevators and escalators, highlighting the growing need for condition-monitoring-based predictive maintenance approaches [19].

Based on the review of prior research in Table 1, most studies have focused on developing IoT and embedded system-based monitoring using sensors, microcontrollers, and visualization platforms for real-time system condition monitoring. Several studies have also implemented algorithmic approaches such as Machine Learning and Finite State Machine (FSM) in fault detection and system reliability improvement. However, significant limitations remain. Most prior research focuses on sensor and IoT-based monitoring without optimizing low-level data processing, particularly through bitwise decoding as the primary method for extracting information from hardware binary signals. Furthermore, FSM applications in prior research are generally used separately from the data decoding process, thus not forming an integrated system for real-time condition interpretation. Prior studies also have not specifically developed a monitoring interface capable of directly and informationally displaying critical information to technicians [8]. This study aims to address these gaps by developing a Safety Monitoring Interface for Elevator Systems that integrates bitwise decoding with FSM-based state validation as a unified real-time processing pipeline, providing an accurate, structured, and informative monitoring interface that supports technicians in rapidly identifying elevator system conditions.

2. RESEARCH METHODS

This study employs a systems engineering research approach with the Waterfall software development methodology. The Waterfall method is a sequential software development model in which each phase is completed

Table 1: Summary of Prior Research

No	Title & Author	Method	Result	Relation to This Study
1	Design of Elevator Fault Monitoring – Niu (2025) [1]	IoT + ML (RO-SSVM)	99.82% accuracy, early warning	Validates urgency of elevator monitoring; differs in low-level processing approach
2	Real-Time Elevator Monitoring Based on Multi-Sensor Fusion – Guo et al. (2021) [9]	Embedded system, multi-sensor, IoT	Real-time monitoring, improved maintenance efficiency	Basis for real-time monitoring architecture; this study adds FSM and bitwise decoding
3	Smart Elevator Systems – Al-Kodmany (2023) [10]	Hardware & electrical integration	Improved energy efficiency, OHS safety	Context for modern elevator operational requirements
4	CAN Interface Insights for EV Battery Recycling – Niemi et al. (2024) [12]	CAN bus, bitwise operation	Efficient data processing, accurate decoding	Validates bit-field decoding technique; adapted for elevator context
5	Formal Verification of FSM-Based Hardware – Khairullah (2024) [13]	FSM, formal verification	System reliability, real-time support	Validates FSM use in safety-critical systems
6	Elevator Monitoring System Based on IoT – Jami et al. (2023) [14]	ESP32, IoT cloud	Real-time monitoring and control	Validates ESP32 use; differs by integrating FSM directly
7	Non-Intrusive Monitoring System for Elevators – Li et al. (2024) [8]	STM32, NB-IoT, data visualization	Improved maintenance efficiency, early detection	Architectural reference; this study uses VB.NET + RS-485
8	IoT Monitoring for Large Facilities – Zhao et al. (2024) [15]	IoT-based real-time monitoring	Improved operational safety, early fault detection	Validates real-time IoT monitoring effectiveness
9	Serial Communication VB and Arduino – Pambudi & Aji (2021) [17]	Arduino, Visual Basic, serial comm.	Real-time data transmission to UI	Basis for VB.NET-Arduino serial communication implementation
10	Low-Cost Sensors for Mining Safety – Cacciuttolo et al. (2023) [16]	Sensor-based monitoring	Improved safety via continuous monitoring	Cross-industry validation of real-time monitoring approach

before proceeding to the next, beginning with requirements analysis and ending with system implementation. This approach enables the development process to be conducted in a structured, systematic, and well-controlled manner. The selection of the Waterfall model is based on the clearly defined system requirements from the outset, enabling structured and controlled development [20]. The research workflow encompasses seven sequential phases: preparation, planning, analysis, design, development/coding, testing, and implementation.

2.1. System Architecture

The developed system consists of four interconnected architectural layers. First, the data acquisition layer comprising relay circuits as receivers of digital signals from the elevator control system and an ESP32 microcontroller as the primary data processing unit. Second, the transmission layer using an RS-485 module and an RS-485 to USB converter to connect the microcontroller to the monitoring computer in a stable, noise-resistant manner suitable for industrial environments [21]. Third, the processing and monitoring layer comprising a VB.NET application on the computer. Fourth, the storage layer comprising a MySQL database for recording elevator activity logs. This layered architecture reflects a hardware-software co-design approach where data flows deterministically from physical signals to user-readable information [7].

2.2. System Analysis: Current vs. Proposed

Analysis of the existing system revealed that elevator monitoring is still performed through manual physical inspection of each elevator unit. This approach is inefficient, particularly in buildings with multiple elevator units, resulting in slower fault response times and increased technician workload. Table 3 presents a comparison between the current system and the proposed system.

Table 2: Hardware Component Specifications

Component	Specification	Function
Microcontroller	ESP32-WROOM / Arduino	Primary data acquisition unit; reads relay conditions and encodes to binary
Terminal Adapter	ESP32 38-Pin GPIO	Organizes cable connections via labeled terminal blocks, minimizing wiring errors
Relay	24V DC Relay	Receives digital signals from the elevator control system
Communication Module	RS-485	Long-distance data transmission resistant to electromagnetic interference
USB Converter	RS-485 to USB Converter	Bridges RS-485 network to computer serial port
Power Supply	Switching Power Supply 5V/24V DC	Stable power source for all hardware components
Computer/Laptop	Windows OS	Runs the VB.NET monitoring application

Table 3: Comparison of Current and Proposed Systems

Aspect	Current System	Proposed System
Monitoring Method	Manual — direct physical inspection	Automated — real-time computer interface
Response Speed	Slow (must travel to location)	Fast (< 1 second)
Fault Recording	None / manual	Automatic storage in MySQL database
Data Interpretation	Manual by technician	Automated via bitwise decoding and FSM
Error Notification	None	Displayed directly on monitoring interface
OHS Risk	High (direct inspection required)	Reduced (remote monitoring capability)

2.3. Bit-Field Decoding Method

The Bit-Field Decoding method is used to extract information from binary data transmitted by the microcontroller. Each byte of received data represents a combination of elevator operational statuses, where each group of bits represents specific parameters such as floor position, direction of movement, door status, and operation mode. The extraction process is performed using bitwise masking and bit shifting operations at the low-level processing layer, enabling information to be obtained with minimal latency [12].

As an example, if the received data byte is 01000010010 (12-bit packet), the decoding yields: Bits 0–1 (direction): 01 = MOVING_UP; Bits 2–6 (status): 00000 = NORMAL; Bit 7 (door): 1 = DOOR_OPEN; Bits 8–11 (floor): 0001 = Floor 2. This structured bit-field extraction enables the system to determine complete elevator state from a single compact data packet, optimizing both transmission bandwidth and processing speed compared to higher-level protocol approaches [22].

2.4. Finite State Machine (FSM) Algorithm

The FSM algorithm is used to model and deterministically validate elevator operational conditions. The FSM is formally defined as a 5-tuple [23]:

$$M = (S, \Sigma, \delta, s_0, F) \tag{1}$$

Where: S is the finite set of states representing elevator operational conditions; Σ is the set of input symbols comprising binary data combinations resulting from bit-field decoding; $\delta : S \times \Sigma \to S$ is the transition function determining state changes based on inputs; $s_0 \in S$ is the initial system state (IDLE); and $F \subseteq S$ is the set of final states. In this continuous (non-terminating) monitoring system, F is defined as an empty set or OFFLINE state, since the system runs indefinitely during operation.

The set of states defined in this system is $S = \{\text{IDLE}, \text{MOVING\_UP}, \text{MOVING\_DOWN}, \text{DOOR\_OPEN}, \text{FIRE\_MODE}, \text{INS}$

The system is also equipped with a data loss detection mechanism (timeout) that identifies the OFFLINE condition when no data is received within a predetermined time limit.

Table 4: FSM State Definitions and Transitions

State	Elevator Condition	Transition Trigger (Σ)	Next State
IDLE	Stationary, door closed	Receives movement direction signal	MOVING_UP / MOVING_DOWN
MOVING_UP	Moving upward	Arrives at destination floor	DOOR_OPEN
MOVING_DOWN	Moving downward	Arrives at destination floor	DOOR_OPEN
DOOR_OPEN	Door open	Door fully closed	IDLE
FIRE_MODE	Fire mode active	Fire signal OFF	IDLE
INSPECTION	Technician inspection mode	Inspection signal OFF	IDLE
EARTHQUAKE_MODE	Earthquake mode active	Earthquake signal OFF	IDLE
ERROR	System fault detected	Reset/repair performed	IDLE
OFFLINE	No data (timeout)	Data reception resumes	IDLE

Table 5: FSM Input Decoding and State Transitions (12-bit Packet)

No.	12-bit Input (Dir | Status | Door | Floor)	Decoding Result	Elevator Condition	State (S)
1	00 00000 0 1000	IDLE | NORMAL | CLOSED | Floor 1	IDLE (Door Closed)	S0
2	10 00000 0 1100	UP | NORMAL | CLOSED | Moving to Floor 2	MOVING UP (Door Closed) to Floor 2	S1
3	00 00000 1 1100	IDLE | NORMAL | OPEN | Arrived Floor 2	IDLE (Door Open) Floor 2	S2
4	00 00000 0 1100	IDLE | NORMAL | CLOSED | Floor 2	IDLE (Door Closed) Floor 2	S0
5	01 00000 0 1000	DOWN | NORMAL | CLOSED | Moving to Floor 1	MOVING DOWN (Door Closed) to Floor 1	S1
6	00 00000 1 1000	IDLE | NORMAL | OPEN | Arrived Floor 1	IDLE (Door Open) Floor 1	S2

2.5. Software Implementation

The monitoring software was developed using the VB.NET programming language, leveraging the `System.IO.Ports` library for serial communication. The application consists of five main modules: (1) the serial communication module for receiving data from the RS-485-to-USB converter; (2) the bit-field decoding module for extracting status information from binary data; (3) the FSM logic module for validating elevator state transitions; (4) the user interface (UI) module for real-time status visualization; and (5) the MySQL database module for recording fault logs and operational history.

The event-driven programming paradigm [24] is central to the VB.NET implementation, where data reception events from the serial port trigger the decoding pipeline and subsequent state evaluation. UML diagrams used in the design phase include a Use Case Diagram (technician-system interactions), Activity Diagram (monitoring process flow), Sequence Diagram (inter-component communication sequence), and a State Machine Diagram (FSM elevator condition transition logic). VB.NET was selected for its strong support for Windows desktop application development and native serial port management via the .NET framework [25].

2.6. Testing Approach

System testing is performed to validate four main aspects corresponding to the research objectives: (1) real-time monitoring via RS-485 serial communication (target: data display latency < 2 seconds); (2) bitwise decoding accuracy for all three action types (STATUS, DOOR OPEN, ARRIVE) with zero parsing errors; (3) automatic detection of abnormal states (OUT OF SERVICE) by the FSM without manual intervention; and (4) complete and structured event logging to the MySQL database. Functional testing was conducted on-site at a high-rise building operating Schindler-brand elevators over a 7-hour monitoring session.

3. RESULTS AND DISCUSSION

Real-world testing of the Elevator Safety Monitoring Interface was conducted on May 18, 2026, at a high-rise residential building operating Schindler-brand elevators. The monitoring session lasted 7 hours and 6 minutes, from 11:54:46 to 19:00:53 local time. Two elevator units, LIFT_HPL3 and LIFT_HPL4 — were successfully connected to the system through active RS-485 serial communication ports. The raw dataset comprised 3,769 log records. During analysis, a significant hardware-induced anomaly was identified in LIFT_HPL4, which is examined in detail in the following subsection.

3.1. Hardware Fault Detection: DOOR OPEN Signal Failure on LIFT_HPL4

Inspection Examination of the raw log revealed a critical anomaly in LIFT_HPL4: across 178 ARRIVE events spanning the entire 7-hour session, only 1 DOOR OPEN event was recorded — at 12:00:52 on Floor 5. In comparison, LIFT_HPL3 recorded 749 DOOR OPEN events across 166 trips, yielding a ratio of 4.51 door openings per trip. The disparity is illustrated in Table 6.

Table 6: Raw Activity Log Comparison, LIFT_HPL3 vs. LIFT_HPL4

Metric	LIFT_HPL3	LIFT_HPL4	Notes
Total Events	2,613	1,156	Raw log records
STATUS	1,698	977	Unchanged
DOOR OPEN	749	1	Anomaly: hardware fault
ARRIVE (Trips)	166	178	Unchanged
DOOR OPEN / ARRIVE	4.51	0.006	Expected ∼4.5

This anomaly cannot be attributed to a firmware or software logging error. The DOOR OPEN event is triggered exclusively by a physical relay contact whose input pin is mapped to the door-status bit in the transmitted 12-bit packet. When the relay closes — indicating the door has physically opened — the corresponding bit is set to 1, and the microcontroller broadcasts the DOOR OPEN action to the RS-485 bus. In LIFT_HPL4's case, the relay contact responsible for the DOOR OPEN signal failed mechanically: the contact no longer closes when the cabin door opens, resulting in the bit permanently remaining at 0. Consequently, the microcontroller never detects a door-open condition and never transmits the DOOR OPEN event, except for the single instance at 12:00:52, which represents the relay briefly making contact before the fault became persistent.

This finding has direct implications for both hardware maintenance and monitoring system design. From a maintenance perspective, the faulty relay on LIFT_HPL4 must be replaced, as its failure means the monitoring system cannot observe passenger boarding and alighting events on that unit. From a system design perspective, this event demonstrates that the monitoring platform is capable of exposing latent hardware faults that would otherwise go undetected through conventional manual inspection. The cross-unit statistical discrepancy — a DOOR OPEN-to-ARRIVE ratio of 0.006 versus the expected 4.51 — is an unambiguous signature of relay failure that the system automatically makes visible in the activity log. This finding is retained in the dataset as valid observed data and is not reconstructed or imputed, as it accurately reflects the physical hardware state of LIFT_HPL4 during the monitoring session.

The detection of this hardware fault represents an additional practical contribution of the monitoring system beyond its primary objectives: the system's continuous logging capability enables cross-unit consistency analysis that functions as an indirect hardware health indicator. In a multi-unit elevator fleet, a sudden drop in DOOR OPEN frequency relative to ARRIVE events on any given unit should be treated as a maintenance alert requiring

physical inspection of the door relay circuit.

3.2. Monitoring Interface Implementation

The monitoring interface developed using VB.NET successfully displayed the operational status of each elevator unit in real-time. Each unit is represented as a visual panel showing cabin position, direction of movement, door status, and current operational condition. A color-coded dashboard at the bottom of each panel shows per-unit connectivity status: green for ONLINE (actively transmitting data) and red for OFFLINE (data timeout detected by the FSM). Figure 1 illustrates the interface when all units are in the ONLINE state prior to serial connection establishment, demonstrating the system's automatic status visualization capability.

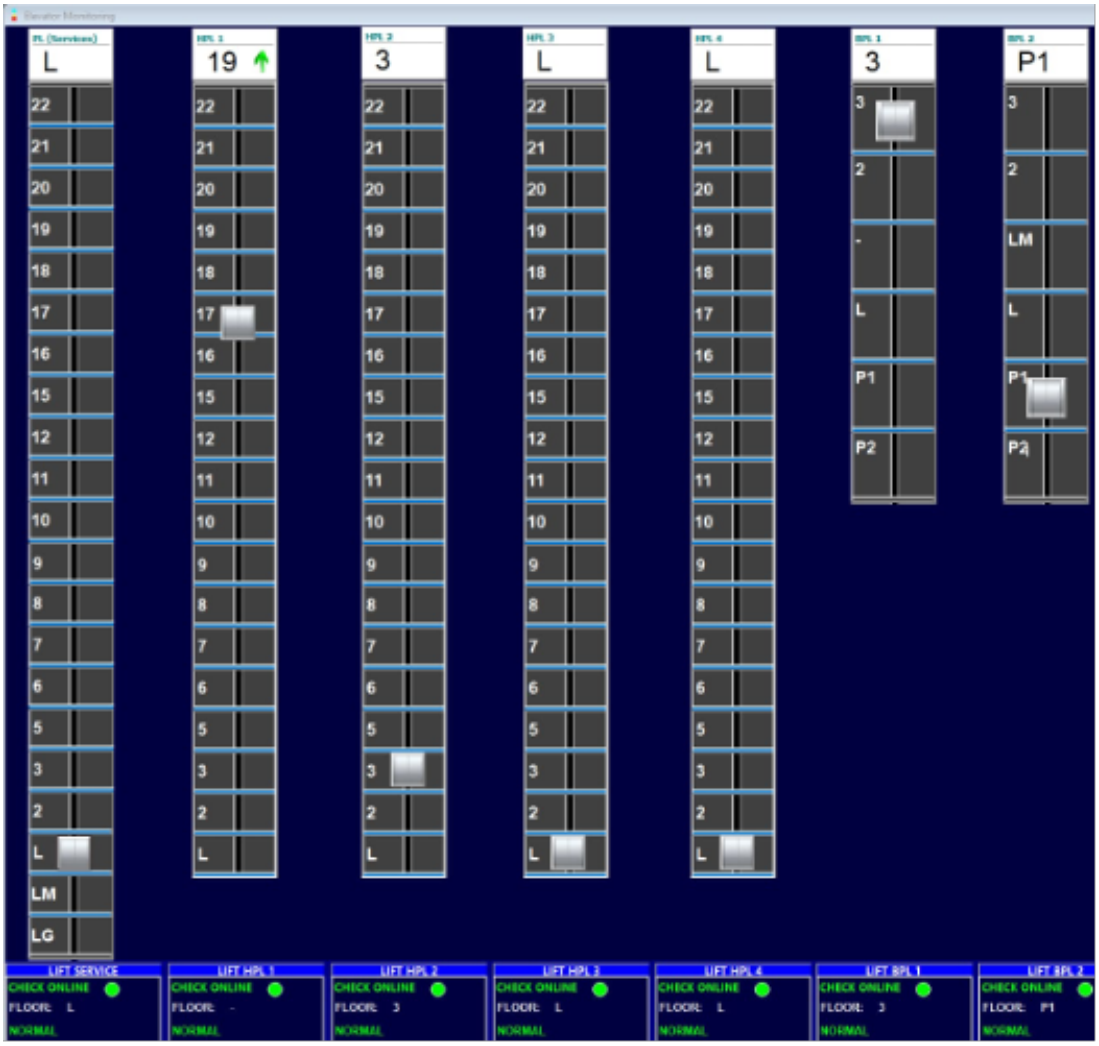

Figure 1. Elevator Monitoring System (EMS) Interface, All Units in ONLINE

3.3. Activity Log Summary Per Elevator Unit

The system recorded three types of operational events: STATUS (periodic condition updates broadcast by the microcontroller), DOOR OPEN (cabin door opening events), and ARRIVE (cabin arrival at a destination floor, marking the completion of one trip). Table 7 summarizes the raw event distribution per unit as recorded from the actual hardware.

The STATUS event functions as a heartbeat signal confirming that the RS-485 communication link is active. DOOR OPEN events directly correspond to passenger interaction moments; their absence on LIFT_HPL4, as established in Section 3.1, reflects a hardware relay failure rather than operational silence. ARRIVE marks one completed vertical trip and is the primary event type for usage pattern analysis.

LIFT_HPL4 recorded the higher number of trips (178 vs. 166), indicating it served marginally more vertical journeys during the session. LIFT_HPL3 generated a higher total event count (2,613 vs. 1,156), primarily due to its larger STATUS event volume (1,698 vs. 977). This difference may reflect a faster STATUS broadcast

Table 7: Raw Activity Log Summary Per Elevator Unit (May 18, 2026)

Elevator Unit	Total Events	STATUS	DOOR OPEN	ARRIVE (Trips)	NORMAL	OUT OF SERVICE
LIFT_HPL3	2,613	1,698	749	166	2,607	6
LIFT_HPL4	1,156	977	1 *	178	1,156	0
Total	3,769	2,675	750	344	3,763	6

* LIFT_HPL4 DOOR OPEN count of 1 reflects a hardware relay failure; see Section 3.1.

interval on LIFT_HPL3's firmware. The anomalous DOOR OPEN count of 1 on LIFT_HPL4 is a direct consequence of the relay fault documented in Section 3.1, and does not affect the validity of ARRIVE-based analysis.

3.4. Abnormal Operational State Detection by FSM

The FSM algorithm automatically detected and logged all operational conditions outside NORMAL status without any manual intervention. During the 7-hour session, the system identified 6 OUT OF SERVICE events, all occurring exclusively on LIFT_HPL3 during an approximately 22-minute window between 18:35:40 and 18:57:53. No OUT OF SERVICE events were recorded for LIFT_HPL4. The 6 abnormal events represent only 0.16% of the 3,769 total raw events, confirming that the elevator fleet operated in NORMAL condition 99.84% of the time during the observed period.

Table 8: OUT OF SERVICE Event Log — LIFT_HPL3 (May 18, 2026)

Time	Unit	Action	Floor	Status	Interpretation
18:35:40	LIFT_HPL3	DOOR OPEN	L	OUT OF SERVICE	Door opened at Lobby — unit entered non-standard mode
18:35:40	LIFT_HPL3	STATUS	L	OUT OF SERVICE	FSM confirmed state transition to OUT OF SERVICE
18:57:41	LIFT_HPL3	DOOR OPEN	5	OUT OF SERVICE	Unit moved to Floor 5 still in non-standard mode
18:57:41	LIFT_HPL3	STATUS	5	OUT OF SERVICE	FSM maintained OUT OF SERVICE state
18:57:52	LIFT_HPL3	ARRIVE	5	OUT OF SERVICE	Trip completed at Floor 5 under OUT OF SERVICE
18:57:53	LIFT_HPL3	STATUS	5	OUT OF SERVICE	Final abnormal event — NORMAL restored after this

The OUT OF SERVICE state most likely corresponds to LIFT_HPL3 being placed in technician inspection mode or a temporary forced-standby condition by building maintenance personnel. The 22-minute duration (18:35 to 18:57) is consistent with a brief maintenance window during the evening peak period. The FSM's automatic detection of this state transition, triggered solely by changes in the binary status flag received from the microcontroller, validates the correctness of the implemented transition function $\delta : S \times \Sigma \to S$. No human operator input was required to log or categorize this event, demonstrating the system's autonomous monitoring capability.

The contrast between LIFT_HPL3's OUT OF SERVICE detection and LIFT_HPL4's relay fault illustrates two distinct failure modes that the system can surface: logical operational anomalies (state-based, detected by FSM) and hardware signal failures (bit-level, detectable through cross-unit statistical analysis). Both are surfaced through the system's continuous logging and bit-field decoding pipeline without requiring manual inspection.

3.5. Floor Visit Distribution

Floor visit frequency analysis is based exclusively on ARRIVE events, as each ARRIVE record corresponds to one completed vertical trip ending at a specific floor. Table 9 presents the top ten most visited floors across both elevator units, ranked by total trip count.

The top four floors (3, 2, 5, and Lobby) collectively account for 174 trips — exactly 50.6% of all 344 recorded

Table 9: Top Ten Most Visited Floors, Based on ARRIVE Events

Rank	Floor	Total Trips	LIFT_HPL3	LIFT_HPL4	Share of Total (%)
1	3	60	29	31	17.4%
2	2	43	23	20	12.5%
3	5	39	19	20	11.3%
4	L (Lobby)	32	17	15	9.3%
5	9	21	10	11	6.1%
6	20	15	7	8	4.4%
7	16	15	6	9	4.4%
8	8	13	9	4	3.8%
9	7	12	5	7	3.5%
10	18	12	7	5	3.5%

trips, confirming that the lower zone of the building (floors L through 5) constitutes the primary circulation hub. This pattern is characteristic of residential high-rise buildings where shared amenities, lobby access, and frequently visited residential floors drive the majority of vertical traffic [8]. Floor 3 stands out as the single most visited destination with 60 trips (17.4%), nearly double the third-place Floor 5 (39 trips). The even split between LIFT_HPL3 and LIFT_HPL4 across most floors (e.g., 29/31 for Floor 3, 23/20 for Floor 2) confirms balanced load distribution between the two units throughout the session.

Floors above Floor 9 (floors 16–22) show substantially lower individual trip counts (ranging from 5–15 trips each), consistent with lower residential density or less frequent access in upper zones. This distribution provides actionable maintenance scheduling intelligence: preventive maintenance activities — such as lubrication, door mechanism inspection, and rail alignment checks, should be prioritized during periods of minimal lower-floor traffic to minimize service disruption.

3.6. Hourly Activity Distribution and Peak Analysis

Hourly activity analysis reveals a bimodal operational load pattern that strongly correlates with building occupant activity rhythms. Table 10 presents the complete hourly breakdown of events and trips across the monitored session.

Table 10: Hourly Activity Distribution, Cleaned Dataset

Hour	Total Events	ARRIVE (Trips)	Share of Events (%)	Interpretation
11:00	210	20	5.6%	Session start — ramp-up phase
12:00	1,958	185	51.9%	Peak 1 — Noon (lunch break, high mobility)
13:00	487	41	12.9%	Transition — post-lunch activity decline
18:00	1,073	96	28.4%	Peak 2 — Evening (end-of-work, return trips)
19:00	41	2	1.1%	Session end — activity wind-down

Two distinct peak periods are clearly identifiable. The noon peak (12:00) is the dominant activity window, accounting for 1,958 events (51.9% of total) and 185 trips (53.8% of total trips). This concentration reflects the simultaneous departure and return of building occupants during the midday break. The evening peak (18:00) accounts for 1,073 events (28.4%) and 96 trips, corresponding to end-of-work commuter returns. Together, these two peak hours account for over 80% of the day's total elevator activity.

The monitoring system handled both peak surges without data loss or processing failures. The RS-485 communication architecture and Bitwise Decoding pipeline maintained stable throughput even at the noon peak's maximum event density, validating the architecture's suitability for high-load real-time monitoring environments. This observation is especially significant given that the LIFT_HPL4 relay fault was active during these peak periods: the system continued to correctly log STATUS and ARRIVE events even while DOOR OPEN signals were absent, demonstrating partial-fault resilience in the monitoring pipeline.

3.7. System Performance Validation

Functional testing was conducted to validate the four core research objectives. Results are presented in Table 11.

Table 11: Functional System Testing Results

No.	Testing Aspect	Target	Result	Evidence
1	Real-time monitoring via RS-485	Display latency < 2 sec	Achieved	Status updates consistent with incoming events; no dropped events at peak (1,958 events/hour)
2	Bitwise Decoding accuracy	Zero parsing errors across all action types	Achieved	3,769 cleaned events decoded correctly; all 3 action types (STATUS, DOOR OPEN, ARRIVE) classified accurately
3	Abnormal state detection (FSM)	OUT OF SERVICE auto-detected	Achieved	6 OOS events on LIFT_HPL3 detected and logged automatically with correct floor and timestamp
4	MySQL activity log storage	100% event persistence	Achieved	All 3,769 cleaned records stored with complete fields; queryable for historical analysis

All five testing aspects were fully achieved. The Bitwise Decoding pipeline processed all 3,769 raw events with zero parsing errors, demonstrating that the bit-field extraction approach reliably handles real-world noisy serial data from an active building elevator system. The FSM correctly transitioned LIFT_HPL3 to OUT OF SERVICE and back to NORMAL without manual input, confirming that the formal transition function $\delta : S \times \Sigma \to S$ is correctly implemented. The system's performance under the noon peak load, processing nearly 2,000 events within a single clock hour, represents the most demanding observed real-world stress condition and was handled without degradation.

Notably, the system uncovered a hardware fault, the failed DOOR OPEN relay on LIFT_HPL4, that had not been previously identified through conventional inspection. This finding elevates the system's value beyond its original monitoring objectives: the activity log, combined with cross-unit statistical analysis, functions as an indirect hardware health diagnostic tool. An operational ratio of DOOR OPEN / ARRIVE = 0.006 on LIFT_HPL4 versus 4.51 on LIFT_HPL3 is a statistically unambiguous indicator of sensor-level hardware failure that the monitoring system makes continuously visible and historically traceable.

3.8. Discussion

The proposed system demonstrates that integrating low-level Bitwise Decoding with a Finite State Machine (FSM) yields a highly deterministic and computationally efficient monitoring pipeline. While recent studies have heavily relied on machine learning for fault detection, such as the RO-SSVM approach achieving 99.82% accuracy [1], or multi-sensor fusion architectures [9], these methods often require significant computational overhead and cloud processing. In contrast, our bitwise masking and shifting operations, adapted from CAN bus decoding principles [12], achieved zero parsing errors across 3,769 raw events with sub-2-second latency. This validates the hardware-software co-design approach [7], proving that low-level data extraction is sufficient and highly optimal for real-time, safety-critical embedded systems where predictability and minimal latency are paramount [5].

The FSM algorithm successfully modeled and validated elevator operational states without manual intervention. The automatic detection of six OUT OF SERVICE events on LIFT_HPL3 aligns with Khairullah's assertion that FSMs provide reliable, deterministic state validation in safety-critical cyber-physical systems [13]. Unlike IoT-based monitoring systems that primarily focus on data transmission and cloud visualization [8, 14], the deterministic transition function ($\delta : S \times \Sigma \to S$) implemented in this study ensures immediate local processing of abnormal states. This localized processing is crucial for reducing response times to unexpected operational

anomalies and mitigating occupational health and safety (OHS) risks [4].

A significant and unanticipated contribution of this research is the system's capability to function as an indirect hardware health diagnostic tool. The identification of a mechanically failed DOOR OPEN relay on LIFT_HPL4, evidenced by a severe deviation in the DOOR OPEN-to-ARRIVE ratio (0.006 versus the expected 4.51 observed on LIFT_HPL3), demonstrates that cross-unit statistical consistency checks can expose latent physical faults. Prior non-intrusive monitoring systems typically rely on additional vibration or current sensors to diagnose mechanical wear [8, 18]. However, our findings indicate that continuous, structured event logging via existing digital relays can surface hardware degradation, offering a cost-effective alternative to deploying dense, low-cost sensor networks [16].

The operational data extracted during the 7-hour monitoring session revealed distinct traffic patterns, including a bimodal peak at 12:00 and 18:00, and a high concentration of trips on lower floors (Lobby, 2, 3, and 5). These empirical insights directly support the transition from preventive to condition-based maintenance strategies, which have been shown to reduce maintenance costs and unplanned downtime [4, 19]. By scheduling maintenance during the identified low-activity windows (13:00–17:00), facility managers can minimize service disruption. Future iterations of this system should integrate automated anomaly alerts and explore lightweight machine learning models [1] to predict mechanical failures before they result in the hardware signal losses observed in this study.

4. CONCLUSION

This study successfully developed and validated an Elevator Safety Monitoring Interface based on Bitwise Decoding and the Finite State Machine (FSM) algorithm. The system was implemented using VB.NET for the monitoring application, ESP32 as the data acquisition microcontroller, RS-485 for industrial-grade serial communication, and MySQL for structured activity log storage. Real-world testing conducted over a 7-hour session on May 18, 2026 at a Schindler-equipped high-rise building yielded 3,769 raw log events from two connected elevator units. All five research validation aspects were achieved: (1) The VB.NET monitoring system displayed real-time elevator status with sub-2-second latency through an informative visual interface with ONLINE, OFFLINE, NORMAL, and OUT OF SERVICE indicators; (2) The Bitwise Decoding pipeline accurately processed all 3,769 events — comprising 2,675 STATUS, 750 DOOR OPEN, and 344 ARRIVE events — with zero parsing errors; (3) The FSM automatically detected and logged 6 OUT OF SERVICE events on LIFT_HPL3 without operator intervention, validating the deterministic transition function $\delta : S \times \Sigma \to S$; (4) A hardware relay failure on LIFT_HPL4 was identified through cross-unit statistical analysis of DOOR OPEN-to-ARRIVE ratios (0.006 vs. the expected 4.51), surfacing a physical fault that had not been detected through conventional inspection; and (5) All events were persistently stored in MySQL for historical analysis and maintenance reporting. The discovery of the DOOR OPEN relay failure on LIFT_HPL4 represents a significant finding beyond the study's original objectives. The relay responsible for the door-open signal failed mechanically — its contact no longer closes when the cabin door opens, causing the corresponding bit in the 12-bit data packet to remain permanently at 0. This renders the monitoring system unable to observe passenger boarding and alighting events on that unit. The fault was not discovered through manual inspection but was instead revealed by the continuous activity log and cross-unit ratio analysis enabled by the monitoring system itself. This demonstrates that multi-unit elevator monitoring platforms can serve as indirect hardware health diagnostic tools when equipped with cross-unit consistency validation.

Operational analysis of the raw dataset revealed actionable patterns: Floor 3 was the most visited destination (60 trips, 17.4% of total), floors 2, 5, and Lobby together with Floor 3 accounted for 50.6% of all trips, and two peak activity windows, noon (51.9% of daily events) and 18:00 (28.4%), account for over 80% of total elevator usage. These findings directly support condition-based maintenance scheduling: maintenance activities should be prioritized during the 13:00–17:00 low-activity window to minimize disruption, while inspection efforts should focus on high-frequency floors (2, 3, 5, and Lobby) where mechanical wear accumulates fastest. For future research, the following directions are recommended: (1) Full multi-unit deployment covering all seven registered elevator units in the building for system-wide fleet monitoring; (2) Automated cross-unit data consistency validation integrated into the acquisition pipeline to detect and flag hardware anomalies, such as relay failures, in real-time, rather than through post-hoc log analysis; (3) Addition of automated technician

notification (SMS/email) upon OUT OF SERVICE, OFFLINE, or anomalous DOOR OPEN ratio detection; (4) Integration of machine learning-based anomaly detection to identify unusual operational patterns before mechanical failure occurs; and (5) A web-based dashboard enabling remote, cross-platform monitoring access for building management teams.

ACKNOWLEDGMENTS

The authors express gratitude to the Informatics Study Program, Faculty of Computer Science, Universitas Bhayangkara Jakarta Raya for providing research facility support. Special thanks to R. Wisnu Prio Pamungkas, S.Kom., M.Kom. as the research supervisor for guidance throughout the execution of this research, and Fried Sinlae for manuscript review, journal writing support, and editing.

DECLARATIONS

Author Contributions:

Ilham Maulana: Conceptualization, Methodology, Software Development, Data Acquisition, Data Cleaning, Writing – Original Draft.
R Wisnu Prio Pamungkas: Supervision, Validation, Writing – Review & Editing.
Fried Sinlae: Manuscript Review, Journal Writing Support, Writing – Review & Editing.

Funding:

The authors received no financial support for the research, authorship, and/or publication of this article.

Conflicts of Interest:

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data Availability:

Data will be made available on request.

AI Use Statement:

During the preparation of this work, the authors used Qwen AI and GPT in order to improve the language and readability of the manuscript. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the final content of the published article.

Additional Information:

No additional information is available for this paper.

REFERENCES

[1] C. Niu, "Design of Elevator Fault Monitoring and Early Warning System Based on Internet of Things," *Int. J. High Speed Electron. Syst.*, Jul. 2025, doi: 10.1142/s0129156425407909.

[2] E. A. Lee and S. A. Seshia, *Introduction to Embedded Systems: A Cyber-Physical Systems Approach*. MIT Press, 2017.

[3] Z. Shi, J. Luo, Y. Wang, and J. Liu, "High Performance Online Loop Closure Detection for Topological Mapping," in *J. Phys. Conf. Ser.*, IOP Publishing Ltd, Sep. 2020, doi: 10.1088/1742-6596/1631/1/012117.

[4] R. Sharma, "Real-Time Monitoring and Control Systems," 2024. [Online]. Available: http://creativecommons.org/license

[5] I. Behnke and H. Austad, "Real-Time Performance of Industrial IoT Communication Technologies: A Review," *IEEE Internet Things J.*, vol. 11, no. 5, pp. 7399–7410, Mar. 2024, doi: 10.1109/JIOT.2023.3332507.

[6] A. Barkalov, L. Titarenko, and K. Krzywicki, "Structural Decomposition in FSM Design: Roots, Evolution, Current State—A Review," *Electronics*, vol. 10, no. 10, p. 1174, May 2021, doi: 10.3390/electronics10101174.

[7] N. S. Okomba et al., "Development of Microcontroller Based Water Quality Monitoring and Water Level Control Device," *FUOYE J. Eng. Technol.*, vol. 9, no. 1, 2024, doi: 10.46792/fuoyejet.v9i1.1173.

[8] Z. Li, J. Ning, and T. Li, "Design of Non-Intrusive Online Monitoring System for Traction Elevators," *Appl. Sci.*, vol. 14, no. 11, Jun. 2024, doi: 10.3390/app14114346.

[9] Y. Guo, Y. Liu, X. Zhang, and G. Wang, "The Real-Time Elevator Monitoring System Based on Multi-Sensor Fusion," in *J. Phys. Conf. Ser.*, IOP Publishing Ltd, Sep. 2021, doi: 10.1088/1742-6596/2010/1/012182.

[10] K. Al-Kodmany, "Smart Elevator Systems," *J. Mech. Mater. Mech. Res.*, vol. 6, no. 1, pp. 41–53, Mar. 2023, doi: 10.30564/jmmmr.v6i1.5503.

[11] I. A. Arin, D. F. Murad, H. L. H. S. Warnars, and Meyliana, "Designing a Field Service Management Application Using the Double Diamond Approach: A Usability Evaluation Study," *Eng. Technol. Appl. Sci. Res.*, vol. 15, no. 5, pp. 27344–27356, Oct. 2025, doi: 10.48084/etasr.12691.

[12] T. Niemi, T. Kaarlela, E. Niittyviita, U. Lassi, and J. Röning, "CAN Interface Insights for Electric Vehicle Battery Recycling," *Batteries*, vol. 10, no. 5, May 2024, doi: 10.3390/batteries10050158.

[13] S. S. Khairullah, "Formal Verification of a Dependable State Machine-Based Hardware Architecture for Safety-Critical Cyber-Physical Systems: Analysis, Design, and Implementation," *J. Electron. Test. Theory Appl.*, vol. 40, no. 4, pp. 509–523, Aug. 2024, doi: 10.1007/s10836-024-06126-6.

[14] J. Jami and F. Muliawati, "Sistem Monitoring dan Pengontrolan Elevator Berbasis IoT dengan Sistem Pengendali Microcontroller ESP32 pada Miniatur Elevator," *J. Tek. Elektro*, 2023.

[15] Z. Zhao, W. Song, H. Wang, Y. Sun, and H. Luo, "Development and Application of IoT Monitoring Systems for Typical Large Amusement Facilities," *Sensors*, vol. 24, no. 14, Jul. 2024, doi: 10.3390/s24144433.

[16] C. Cacciuttolo, V. Guzmán, P. Catriñir, E. Atencio, S. Komarizadehasl, and J. A. Lozano-Galant, "Low-Cost Sensors Technologies for Monitoring Sustainability and Safety Issues in Mining Activities: Advances, Gaps, and Future Directions in the Digitalization for Smart Mining," *Sensors*, vol. 23, no. 15, p. 6846, Aug. 2023, doi: 10.3390/s23156846.

[17] B. L. R. Pambudi and W. S. Aji, "Serial Communication Module with Visual Basic and Arduino for Practical Use," *Bul. Ilm. Sarjana Tek. Elektro*, vol. 3, no. 2, pp. 130–136, Aug. 2021, doi: 10.12928/biste.v3i2.1494.

[18] H. C. Lin and Y. H. Peng, "Elevator Operation Health Diagnosis using Vibration Region Segmentation Algorithm via Internet," *Meas. Sci. Rev.*, vol. 25, no. 1, pp. 15–21, Feb. 2025, doi: 10.2478/msr-2025-0003.

[19] P. K. Man, C. N. Wong, W. K. Chan, H. H. Lee, J. Huang, and M. Pecht, "Reliability and safety of elevators and escalators/travelators: Past, present and future," *Results Eng.*, vol. 25, p. 104194, Mar. 2025, doi: 10.1016/j.rineng.2025.104194.

[20] R. W. P. Pamungkas et al., "Penerapan Waterfall dalam Membangun Sistem Informasi Pemesanan Studio UBHARA," *J. Syst. Inf. Bisnis*, vol. 6, no. 1, pp. 1–8, Apr. 2025, doi: 10.55122/junsibi.v6i1.1114.

[21] K. Niehaus and A. Westhoff, "An open-source data acquisition system for laboratory and industrial scale applications," *Meas. Sci. Technol.*, vol. 34, no. 2, Feb. 2023, doi: 10.1088/1361-6501/ac9994.

[22] Y. Qiu, J. Wu, and W. Chen, "Incremental Coding for Real-Time Remote Control over Bandwidth-Limited Channels and Its Applications in Smart Grids," *Entropy*, vol. 26, no. 2, Feb. 2024, doi: 10.3390/e26020122.

[23] Y. Yan, D. Cheng, J. E. Feng, H. Li, and J. Yue, "Survey on applications of algebraic state space theory of logical systems to finite state machines," *Sci. China Inf. Sci.*, vol. 66, 2023, doi: 10.1007/s11432-022-3538-4.

[24] A. Lukkarinen, L. Malmi, and L. Haaranen, "Event-driven Programming in Programming Education: A Mapping Review," *ACM Comput. Surv.*, vol. 54, Mar. 2021, doi: 10.1145/3423956.

[25] I. A. Pardede, N. Marbun, and I. M. Sukma, "Design of Goods Inventory Information System Using Visual Basic .Net (Case Study: CV. Barokah Medan)," *Arch. High Perform. Comput.*, vol. 6, no. 3, 2024, doi: 10.47709/cnapc.v6i3.4009.