Al-Farouk journal of sciences
Volume (2), Issue (3) (JUNE, 2026)
ISSN: 3135-2359

مجلة الفاروق للعلوم
المجلد (2) العدد 3 يونيو 2026
ردمد: 3135-2359

Real-Time Requirements Analysis in Smart Healthcare Monitoring Systems: A Case Study of an ATmega328P-Based Health Monitoring Device

Marwa ALfordag[a], Najwa dirbal[b], Abdusamea I.A Omer[c]
[a]University of Azzawi, Faculty Engineering, Dep. of Computer Engineering and Computer Since
[b]Faculty of Education AL-Ajilat
[c]Sabratha University, Faculty of Engineering, Dep. Of Computer Engineering and information Technology
najway.dirbai2023aa@gmail.com

تاريخ الاستلام: 2026/04/20 تاريخ المراجعة 2026 /05/20 تاريخ القبول:2026/06/07- تاريخ النشر: 2026 /06/21

ABSTRACT

Smart healthcare monitoring systems have become increasingly important for continuous patient observation and timely medical intervention. However, many low-cost embedded healthcare devices are developed with limited consideration of real-time requirements, which may lead to delayed responses during emergency situations. This paper presents a real-time requirements analysis of an ATmega328P-based healthcare monitoring device capable of measuring body temperature, heart rate, blood oxygen saturation (SpO_2), and blood pressure while providing remote alert functionality through GSM communication.

The study begins with an analysis of the implemented prototype and its software architecture. The original system employs a sequential super-loop execution model containing multiple blocking delays that negatively affect responsiveness and emergency handling performance. To address these limitations, a hybrid real-time architecture is proposed that combines event-driven processing, interrupt-assisted emergency detection, and priority-based task scheduling.

Worst-Case Execution Time (WCET) estimation, response-time analysis, Rate Monotonic Scheduling (RMS), and Earliest Deadline First (EDF) scheduling techniques were applied to evaluate the timing behavior of the system. A Python-based simulation framework was developed to assess processor utilization, schedulability, response times, deadline satisfaction, and emergency alert latency.

The results show that the proposed task set achieves a processor utilization of 14.5% and remains schedulable under both RMS and EDF policies. Response-time analysis confirmed that all tasks meet their timing constraints. Furthermore, the proposed hybrid architecture reduced emergency alert latency from approximately 20.5 seconds in the original implementation to 160 milliseconds, representing a latency reduction of 99.22%.

The findings demonstrate that integrating real-time scheduling mechanisms and interrupt-assisted emergency management can significantly improve the responsiveness, predictability, and reliability of resource-constrained healthcare monitoring systems while maintaining compatibility with low-cost embedded platforms.

KEYWORDS Smart Healthcare Monitoring, Real-Time Systems, ATmega328P, WCET Analysis, Rate Monotonic Scheduling, Earliest Deadline First, Response-Time Analysis, Embedded Systems, Healthcare IoT.

1 Introduction

1.1 Background

The rapid growth of embedded systems and Internet of Things (IoT) technologies has significantly transformed modern healthcare services. Smart healthcare monitoring systems enable continuous observation of physiological parameters and support timely medical intervention without requiring constant physical supervision by healthcare professionals [7], [8]. These systems are increasingly deployed in home healthcare, elderly care, chronic disease management, and remote patient monitoring applications.

Recent advances in sensor technologies have facilitated the development of compact and affordable healthcare monitoring devices capable of measuring vital physiological parameters such as body temperature, heart rate, blood oxygen saturation (SpO_2), and blood pressure [7], [11], [12]. The integration of these sensors with low-cost microcontrollers has enabled the development of portable monitoring platforms suitable for both clinical and non-clinical environments.

Despite these advancements, many embedded healthcare systems are primarily designed to achieve functional requirements while paying limited attention to timing constraints and real-time responsiveness. In healthcare applications, delayed processing or notification of critical physiological conditions may reduce the effectiveness of emergency response procedures and negatively impact patient safety [2], [3] According to Kopetz [17] and Burns and Wellings [18], timing correctness is as important as logical correctness in safety-critical embedded systems.

1.2 Problem statement

Healthcare monitoring systems are inherently time-sensitive because physiological abnormalities must often be detected and communicated within strict timing constraints. Many low-cost implementations rely on simple sequential software architectures that employ blocking delays and polling mechanisms. While such approaches simplify implementation, they may introduce significant response delays during emergency situations.

The ATmega328P-based healthcare monitoring device investigated in this study follows a super-loop execution model in which sensing, display updates, blood pressure measurements, and GSM communication tasks are executed sequentially. This architecture may lead to increased emergency notification latency, particularly when lengthy monitoring or communication operations are in progress.

Consequently, there is a need to analyze the timing behavior of such systems and evaluate whether the introduction of real-time scheduling mechanisms can improve responsiveness and predictability.

1.3 Research Motivation

The motivation behind this work stems from the growing importance of reliable and responsive healthcare monitoring systems. Although considerable research has focused on sensor accuracy, wireless communication, and IoT integration, relatively fewer studies

have investigated the real-time characteristics of low-cost healthcare monitoring platforms based on resource-constrained microcontrollers [6], [8].

Analyzing timing requirements and scheduling behavior is essential for ensuring that healthcare monitoring systems can react promptly to critical physiological events. This study therefore aims to bridge the gap between embedded healthcare monitoring and real-time systems engineering by evaluating the timing performance of a practical healthcare monitoring implementation.

1.4 Research Objectives

The primary objectives of this research are:

1. To analyze the real-time requirements of an implemented ATmega328P-based healthcare monitoring device.
2. To identify system tasks and derive their timing characteristics.
3. To estimate Worst-Case Execution Time (WCET) parameters for the identified tasks.
4. To evaluate system schedulability using Rate Monotonic Scheduling (RMS).
5. To evaluate system schedulability using Earliest Deadline First (EDF).
6. To propose a hybrid real-time architecture that improves emergency responsiveness.
7. To quantify the impact of the proposed architecture using simulation-based performance evaluation.

1.5 Research Contributions

This paper makes the following contributions:

- Analysis of a practical healthcare monitoring device implemented using the ATmega328P microcontroller.
- Derivation of a real-time task model from the implemented software architecture.
- Application of WCET estimation and response-time analysis techniques.
- Comparative evaluation of RMS and EDF scheduling approaches.
- Proposal of a hybrid interrupt-assisted real-time architecture.
- Quantitative evaluation of emergency alert latency reduction.

1.6 Paper Organization

The remainder of this paper is organized as follows. Section 2 reviews related studies in healthcare monitoring and real-time scheduling. Section 3 presents the system overview and hardware architecture. Section 4 describes the software architecture of the implemented device. Section 5 presents the real-time requirements analysis and task characterization. Sections 6 and 7 discuss WCET estimation and scheduling analysis using RMS and EDF. Section 8 introduces the proposed hybrid architecture. Section 9 presents the simulation framework and experimental methodology. Section 10 discusses the obtained results. Finally, Sections 11 and 12 present the conclusion and future research directions.

2 Related Work

2.1 Smart Healthcare Monitoring Systems

Smart healthcare monitoring systems have received significant attention in recent years due to their ability to support continuous physiological observation, remote patient supervision, and early detection of medical emergencies. Advances in embedded systems, wireless communication, and biomedical sensing technologies have enabled the development of low-cost monitoring platforms suitable for home-based and mobile healthcare applications.

[6] developed a portable emergency health monitoring system based on an ATmega328P microcontroller and multiple biomedical sensors, including the MAX30100 pulse oximeter, MLX90614 infrared temperature sensor, and MPS20N0040D pressure sensor. The system was capable of monitoring heart rate, oxygen saturation, body temperature, and blood pressure while transmitting alerts through a GSM communication module. Experimental validation was conducted using ten participants and demonstrated acceptable measurement accuracy compared with hospital equipment. However, the study primarily focused on hardware implementation and measurement validation and did not investigate real-time scheduling behavior, response-time guarantees, or schedulability analysis. Their study demonstrated the feasibility of collecting and transmitting patient data using compact hardware architectures. However, the work primarily focused on system functionality and did not investigate real-time schedulability or timing guarantees.

Iqbal et al. [7] reviewed recent developments in wearable healthcare devices and highlighted the growing importance of sensor miniaturization, communication reliability, and power efficiency. Although their review provided a comprehensive overview of wearable healthcare technologies, limited attention was given to task scheduling behavior and response-time performance.

Khan et al. [8] examined IoT-based healthcare monitoring architectures and emphasized the role of connectivity and remote accessibility in modern healthcare services. Their findings demonstrated the benefits of integrating sensing devices with communication infrastructures; however, deterministic timing analysis and emergency response latency were not considered.

Several healthcare monitoring systems have successfully demonstrated physiological data acquisition and remote reporting capabilities. Nevertheless, most existing studies evaluate system functionality, measurement accuracy, and communication performance while providing limited analysis of temporal behavior, deadline satisfaction, and scheduling efficiency in resource-constrained embedded environments.

Therefore, despite the growing adoption of smart healthcare technologies, the application of formal real-time analysis techniques remains relatively limited. This observation motivates the present work, which combines healthcare monitoring implementation analysis with schedulability evaluation, response-time analysis, and emergency-alert optimization.

2.2 Physiological Monitoring Technologies

Pulse oximetry and heart-rate monitoring technologies have become essential components of modern healthcare devices. The MAX30100 integrated sensor provides a compact solution for simultaneous heart-rate and SpO_2 measurement and has been widely adopted in wearable healthcare applications [9].

Body temperature monitoring is another important aspect of patient observation. Infrared temperature sensors such as the MLX90614 provide non-contact temperature measurement and have been employed extensively in healthcare applications requiring rapid and convenient temperature acquisition [10].

Blood pressure monitoring remains one of the most important physiological measurements in healthcare systems. Shaltis et al. [11] investigated wearable blood pressure monitoring technologies, while Yilmaz et al. [12] explored methods for detecting vital signs using wearable sensing platforms.

2.3 Real-Time Scheduling in Embedded Systems

Real-time scheduling algorithms play a critical role in ensuring that timing constraints are satisfied in embedded systems. Liu and Layland [1] introduced the foundational concepts of Rate Monotonic Scheduling (RMS) and Earliest Deadline First (EDF), which remain among the most widely studied scheduling techniques in real-time systems research.

Liu [2] provided a comprehensive trea of real-time system analysis, incl schedulability testing, response analysis, and timing verification methods. Buttazzo [3] further extended these concepts by discussing practical scheduling techniques for modern embedded applications.

Although real-time scheduling has extensively studied in industrial autom aerospace, and safety-critical domains, its application to low-cost healthcare monitoring systems remains relatively limited.

2.4 Research Gap

Existing healthcare monitoring studies primarily focus on sensor integration, communication technologies, and IoT connectivity. Relatively few investigations provide detailed analysis of timing requirements, schedulability, response-time behavior, and emergency alert latency in resource-constrained healthcare monitoring platforms.

Furthermore, most studies evaluate healthcare devices from a functional perspective without systematically examining the impact of software architecture on real-time responsiveness. This gap motivates the present work, which combines healthcare monitoring implementation analysis with formal real-time scheduling evaluation and architectural optimization.

Table 1-Comparison with Previous Studies

Study	Vital Signs Monitoring	Embedded Platform	Real-Time Analysis	RMS/EDF Analysis	Emergency Alert Optimization
Noh and Na [6]	✓	✓	✗	✗	✗
Iqbal et al. [7]	✓	✓	✗	✗	✗
Khan et al. [8]	✓	✓	✗	✗	✗
Shaltis et al. [11]	✓	✓	✗	✗	✗
Yilmaz et	✓	✓	✗	✗	✗

al. [12]					
Proposed Work	✓	✓	✓	✓	✓

3 System Overview

3.1 System Description

The healthcare monitoring system considered in this study is a low-cost embedded platform designed for continuous monitoring of multiple physiological parameters. The system is based on the ATmega328P microcontroller and integrates several biomedical sensors to acquire patient vital signs in real time.

The device is capable of monitoring:

- Body temperature
- Heart rate
- Blood oxygen saturation (SpO_2)
- Blood pressure

The measured physiological parameters are displayed locally through an LCD module and can also be transmitted remotely using GSM communication. In addition, the system supports emergency notification functionality when abnormal physiological conditions are detected.

The implemented platform was originally developed as a standalone healthcare monitoring device and subsequently analyzed in this work from a real-time systems perspective.

3.2 Operational Workflow

The system operation consists of five major stages:

1. Physiological data acquisition.
2. Signal processing and parameter extraction.
3. Local display of measurements.
4. Detection of abnormal physiological conditions.
5. Transmission of emergency alerts through GSM communication.

During operation, physiological measurements are continuously collected from the connected sensors and processed by the ATmega328P microcontroller. The measured values are displayed on the LCD screen and evaluated against predefined medical thresholds.

When an abnormal condition is detected, the system generates an emergency notification containing the patient's physiological information and transmits it to a predefined contact number.

3.3 Functional Components

The system consists of the following functional modules:

Sensing Module

Responsible for acquiring physiological parameters.

Processing Module

Responsible for signal processing, threshold evaluation, and decision making.

User Interface Module

Responsible for displaying measured parameters on the LCD screen.

Communication Module

Responsible for GSM-based emergency notification.

Emergency Management Module

Responsible for identifying abnormal physiological conditions and initiating emergency alerts.

3.4 System Architecture

Figure 1 illustrates the overall architecture of the healthcare monitoring system.

The sensing layer acquires physiological data from the MAX30100 pulse oximeter, MLX90614 infrared temperature sensor, and MPS20N0040D pressure sensor. These measurements are processed by the ATmega328P microcontroller, which performs parameter evaluation and emergency detection. The processed information is displayed locally on the Nokia 5110 LCD and transmitted remotely through the SIM900A GSM module when necessary.

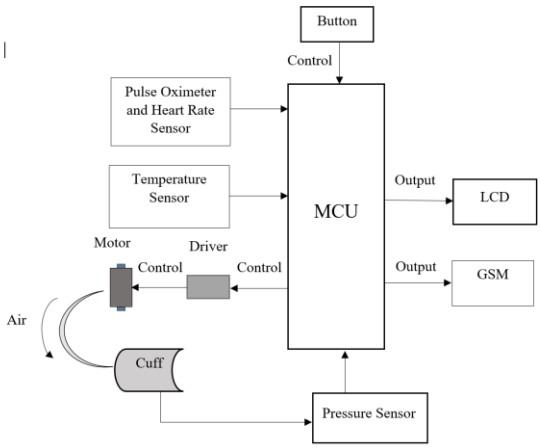

Figure 1-System Architecture Diagram (Block Diagram)

4 Hardware Architecture

4.1 ATmega328P Microcontroller

The ATmega328P serves as the central processing unit of the healthcare monitoring platform. It is an 8-bit AVR microcontroller widely used in embedded applications because of its low cost, low power consumption, and sufficient computational capabilities for sensor-based systems [5].

The microcontroller is responsible for:

- Sensor interfacing
- Data acquisition
- Signal processing
- Threshold evaluation
- LCD control
- GSM communication

The device provides digital and analog input/output interfaces, timers, interrupt capabilities, and serial communication peripherals, making it suitable for healthcare monitoring applications [5].

4.2 MAX30100 Pulse Oximeter and Heart Rate Sensor

The MAX30100 is an integrated pulse oximeter and heart rate monitoring sensor developed for wearable and healthcare applications [9].

The sensor combines:

- Infrared LED
- Red LED
- Photodetector
- Signal processing circuitry

Using photoplethysmography (PPG), the sensor estimates both heart rate and blood oxygen saturation levels. Its compact design and low power consumption make it suitable for portable healthcare monitoring systems [9].

In the implemented platform, the MAX30100 is used to continuously monitor heart rate and SpO_2 levels.

4.3 MLX90614 Infrared Temperature Sensor

The MLX90614 is a non-contact infrared temperature sensor capable of measuring object temperatures without direct physical contact [10].

The sensor offers several advantages:

- Fast response time
- Non-invasive measurement
- High measurement convenience
- Digital communication interface

The MLX90614 communicates through the I^2C protocol and is widely used in healthcare applications requiring rapid body temperature measurement [10].

Within the proposed healthcare monitoring system, the sensor provides continuous body temperature monitoring functionality.

4.4 MPS20N0040D Pressure Sensor

Blood pressure estimation in the implemented system is achieved using the MPS20N0040D pressure sensor.

The sensor converts applied pressure into corresponding electrical signals that can be processed by the microcontroller. Combined with an inflatable cuff and air pump mechanism, the sensor enables blood pressure measurement through pressure monitoring techniques [11].

The measured pressure values are processed using calibration equations implemented within the embedded software.

4.5 Nokia 5110 LCD Display

The Nokia 5110 LCD serves as the primary user interface of the healthcare monitoring device.

The display provides local visualization of:

- Body temperature
- Heart rate
- Oxygen saturation
- Blood pressure

The module offers low power consumption and simple SPI-based communication, making it suitable for embedded healthcare applications.

4.6 SIM900A GSM Communication Module

The SIM900A GSM module provides wireless communication capabilities for emergency notification and remote monitoring functions.

- The module supports:
- SMS transmission
- Serial communication
- Mobile network connectivity

When abnormal physiological conditions are detected, the module transmits emergency messages containing patient vital signs to predefined contacts.

This communication capability extends the functionality of the healthcare monitoring platform beyond local monitoring and supports remote emergency response.

5 Software Architecture

5.1 Original Software Design

The implemented healthcare monitoring device utilizes a super-loop software architecture.

In this design, all tasks are executed sequentially within a continuous loop. Sensor acquisition, signal processing, display updates, blood pressure measurement, emergency detection, and GSM communication are performed one after another.

Although this architecture simplifies implementation, it introduces limitations related to responsiveness and timing predictability.

5.2 Sensor Acquisition Process

The software periodically acquires measurements from the physiological sensors.

The acquisition process includes:

- Reading heart rate data.
- Reading oxygen saturation values.
- Reading body temperature measurements.
- Acquiring blood pressure data.

Each measurement is processed before being displayed and evaluated.

5.3 Emergency Detection Logic

The system continuously compares measured physiological values against predefined thresholds.

Emergency conditions include:

Elevated Body Temperature

Temperature > 37°C

Low Oxygen Saturation

$SpO_2 < 60\%$

High Blood Pressure

Blood Pressure ≥ 140/95 mmHg

Low Blood Pressure

Blood Pressure ≤ 100/65 mmHg

When one or more of these conditions occur, the system classifies the situation as abnormal and initiates emergency notification procedures.

5.4 GSM Alert Generation

Emergency notifications are transmitted through SMS messages.

The transmitted message contains:

- Body temperature
- Heart rate
- Oxygen saturation
- Blood pressure readings

This information enables healthcare providers or family members to assess the patient's condition and take appropriate action.

5.5 Limitations of the Original Architecture

Analysis of the implementation revealed several limitations.

Sequential Execution

Tasks are executed one after another without prioritization.

Blocking Delays

The extensive use of delay () functions prevents concurrent processing.

Lack of Preemption

Critical events cannot interrupt currently executing tasks.

Increased Emergency Latency

Emergency notifications may be delayed while the system completes ongoing monitoring operations.

These limitations motivated the real-time analysis and architectural improvements proposed in this study.

6 Real-Time Requirements Analysis

6.1 Importance of Real-Time Requirements in Healthcare Systems

Healthcare monitoring systems are inherently time-sensitive because physiological abnormalities may require immediate intervention. Unlike conventional embedded systems that primarily focus on functional correctness, healthcare applications must satisfy both functional and temporal requirements. A correct result delivered too late may significantly reduce the effectiveness of emergency response procedures and potentially compromise patient safety [2], [3].

Consequently, evaluating timing constraints is an essential step in the design and verification of healthcare monitoring platforms.

6.2 Task Identification

The original software implementation was analyzed and decomposed into a set of independent real-time tasks. Each task represents a specific system functionality that must be executed periodically or in response to particular events.

Table 2-Real-Time Task Set

Task	Description
Heart Rate	Heart-rate acquisition and processing
SpO_2	Blood oxygen saturation monitoring
Temperature	Body temperature monitoring
Blood Pressure	Blood pressure acquisition and calculation
LCD	Display update operations
Emergency Detection	Detection of abnormal physiological conditions
GSM Alert	Emergency notification transmission

6.3 Task Classification

The identified tasks can be categorized according to their operational behavior.

Periodic Tasks

These tasks are executed repeatedly at predefined intervals:

- Heart Rate
- SpO_2
- Temperature
- Blood Pressure
- LCD

Event-Driven Tasks

These tasks are carried out when specific conditions occur:

- Emergency Detection
- GSM Alert

Event-driven tasks are particularly important because they directly influence emergency response performance.

6.4 Timing Parameters

Three primary timing parameters were assigned to each task:

Worst-Case Execution Time (WCET)

The maximum execution time required by a task under worst-case conditions.

Period

The interval between successive task activations.

Deadline

The maximum allowable completion time for a task.

These parameters form the basis for schedulability analysis and response-time evaluation.

6.5 Real-Time Constraints

The healthcare monitoring system must satisfy several timing requirements.

Continuous Physiological Monitoring

Sensor measurements must be updated periodically to maintain accurate patient status information.

Timely Emergency Detection

Abnormal physiological conditions should be identified immediately after measurement acquisition.

Rapid Emergency Notification

Emergency alerts should be transmitted with minimal delay.

Deadline Satisfaction

All tasks must be completed before their corresponding deadlines.

Failure to satisfy these requirements may reduce system responsiveness and reliability.

7 Worst-Case Execution Time (WCET) Analysis

7.1 WCET Analysis Methodology

Worst-Case Execution Time (WCET) represents the maximum amount of processor time required by a task under worst-case operating conditions [2].

Since direct hardware timing measurements were not available during this study, WCET values were derived from detailed analysis of the implemented source code, sensor communication characteristics, processing operations, and communication delays.

The estimation process considered:

Sensor reading operations.

Data processing computations.

LCD update activities.

Emergency evaluation logic.

GSM communication overhead.

The resulting values provide conservative timing estimates suitable for schedulability evaluation.

7.2 Derived WCET Parameters

Table 2 presents the WCET values used throughout this study.

Table 3-Derived Task Parameters

Task	WCET (ms)	Period (ms)	Deadline (ms)
Heart Rate	15	1000	1000
SpO_2	15	1000	1000

Temperature	10	1000	1000
Blood Pressure	50	10000	10000
LCD	30	1000	1000
Emergency Detection	10	500	500
GSM Alert	150	3000	3000

7.3 Processor Utilization Calculation

Processor utilization is calculated using:

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i} \tag{1}$$

where $C_i$represents the worst-case execution time of task i, and $T_i$denotes its corresponding period

Substituting the task parameters from Table 2 yields:

$$U = \frac{15}{1000} + \frac{15}{1000} + \frac{10}{1000} + \frac{50}{10000} + \frac{30}{1000} + \frac{10}{500} + \frac{150}{3000} \tag{2}$$

$$U = 0.145 \tag{3}$$

Therefore:

$$U = 14.5\% \tag{4}$$

The resulting utilization indicates that the processor workload remains relatively low and that substantial processing capacity remains available for future extensions.

7.4 Significance of WCET Results

The obtained WCET values provide the foundation for subsequent schedulability analysis. They demonstrate that the ATmega328P possesses sufficient computational resources to execute all identified healthcare monitoring tasks while maintaining a comfortable utilization margin.

Furthermore, the relatively low utilization level suggests that timing violations are unlikely when appropriate scheduling mechanisms are employed.

8 Rate Monotonic Scheduling (RMS) Analysis

8.1 Overview of RMS

Rate Monotonic Scheduling (RMS) is a fixed-priority scheduling algorithm introduced by Liu and Layland [1] which remains one of the most widely adopted fixed-priority scheduling frameworks in real-time systems research [17], [18].

Under RMS:

Tasks with shorter periods receive higher priorities.

Priorities remain constant throughout execution.

Scheduling decisions are deterministic and predictable.

Because healthcare monitoring systems frequently involve periodic sensing activities, RMS provides a suitable framework for evaluating schedulability.

8.2 Priority Assignment

Based on task periods, priorities were assigned as follows.

Table 4-RMS Priority Assignment

Priority	Task
Highest	Emergency Detection
High	Heart Rate

High	Temperature
High	SpO_2
Medium	LCD
Low	GSM Alert
Lowest	Blood Pressure

The emergency detection task receives the highest priority because it has the shortest period and directly affects patient safety.

8.3 RMS Schedulability Test

According to Liu and Layland [1], a task set is guaranteed schedulable if:

$$U \leq n(2^{1/n} - 1) \tag{5}$$

where: (U) = processor utilization. (n) = number of tasks.

For: [n=7] the RMS utilization bound becomes:

$$U_{RMS} = 7(2^{1/7} - 1) \tag{6}$$

$$U_{RMS} = 0.7286 \tag{7}$$

Or: $U_{RMS} = 72.86\%$ (8)

8.4 RMS Evaluation

The calculated processor utilization was:

$$U = 14.5\%$$

Since:

14.5% < 72.86%

The task set satisfies the RMS schedulability condition.

8.5 RMS Results

The analysis indicates that all healthcare monitoring tasks can be scheduled successfully under RMS without violating deadlines.

The substantial gap between actual utilization and theoretical RMS bound suggests that the system possesses a considerable timing safety margin.

9 Earliest Deadline First (EDF) Analysis

EDF scheduling has been widely recognized for its optimal processor utilization characteristics under preemptive uniprocessor environments [23].

9.1 Overview of EDF

Earliest Deadline First (EDF) is a dynamic-priority scheduling algorithm in which task priorities change during runtime according to their deadlines [1], [3].

The task with the earliest absolute deadline always receives the highest execution priority.

EDF is widely recognized for achieving higher processor utilization than fixed-priority scheduling algorithms.

9.2 EDF Schedulability Condition

For independent preemptive tasks, EDF guarantees schedulability whenever:

$$U \leq 1 \tag{9}$$

$$U \leq 100\% \tag{10}$$

This utilization bound is significantly higher than the RMS bound.

9.3 EDF Evaluation

Using the utilization previously derived:

$U = 14.5\%$

Since:

14.5%<100%

The task set satisfies the EDF schedulability condition.

9.4 Comparison between RMS and EDF

Both scheduling algorithms successfully satisfy the timing requirements of the healthcare monitoring system.

- However, EDF provides:
- Higher theoretical processor utilization.
- Better resource utilization efficiency.

Greater flexibility under increasing workloads.

In contrast, RMS offers:

- Simpler implementation.
- Lower runtime overhead.
- Greater predictability.

Because processor utilization remains relatively low in the studied system, both scheduling approaches provide satisfactory performance.

EDF Results

The results obtained confirm that the healthcare monitoring task set remains schedulable under EDF without deadline violations.

Therefore, both fixed-priority and dynamic-priority scheduling techniques can be considered viable solutions for the proposed healthcare monitoring platform.

10 Proposed Hybrid Real-Time Architecture

10.1 Motivation for Architectural Improvement

Analysis of the original software implementation revealed that the healthcare monitoring system relies on sequential super-loop architecture. While this approach is simple and easy to implement, it introduces several limitations, including blocking delays, lack of task prioritization, and increased emergency response latency.

The blood pressure measurement routine and GSM communication operations represent the most time-consuming activities in the system. During execution of these operations, critical events cannot immediately interrupt ongoing processing, resulting in delayed emergency handling.

To overcome these limitations, hybrid real-time architecture is proposed.

10.2 Design Objectives

The proposed architecture was designed to achieve the following objectives:

- Improve emergency response performance.
- Reduce notification latency.
- Increase timing predictability.
- Support priority-based execution.
- Maintain compatibility with the ATmega328P microcontroller.
- Preserve low implementation complexity.

10.3 Hybrid Scheduling Concept

The proposed architecture combines:

- **Periodic Scheduling**

Periodic healthcare monitoring tasks execute according to predefined timing parameters.

Examples include:

- Heart-rate monitoring
- SpO_2 monitoring
- Temperature monitoring
- LCD updates
- Blood pressure monitoring

Event-Driven Processing

Emergency events are handled immediately after abnormal physiological conditions are detected.

Examples include:

- Fever detection
- Low oxygen saturation detection
- Abnormal blood pressure detection

Interrupt-Assisted Activation

Instead of waiting for the completion of all scheduled activities, emergency processing can be activated immediately through interrupt-assisted mechanisms.

This approach significantly reduces waiting time and improves responsiveness.

10.4 Proposed Task Structure

The healthcare monitoring application is decomposed into independent real-time tasks.

Table 5-Proposed Task Structure

Task	Type
Heart Rate	Periodic
SpO_2	Periodic
Temperature	Periodic
Blood Pressure	Periodic
LCD	Periodic
Emergency Detection	Event-Driven
GSM Alert	Event-Driven

The Emergency Detection task is assigned to be the highest priority because it directly influences patient safety.

10.5 Priority Assignment Strategy

The proposed architecture assigns priorities according to task criticality.

Critical Tasks

- Emergency Detection
- GSM Alert

Monitoring Tasks

- Heart Rate
- SpO_2
- Temperature

Background Tasks

- LCD
- Blood Pressure

This hierarchy ensures that life-critical activities receive processor access before non-critical operations.

10.6 Expected Benefits

The proposed hybrid architecture is expected to provide:

- Reduced emergency notification latency.
- Faster abnormal-condition detection.
- Improved timing predictability.

- Better processor utilization.
- Enhanced reliability for healthcare applications.

These expected improvements are evaluated through simulation in the following sections.

11 Python Simulation Framework

11.1 Simulation Objectives

A Python-based simulation framework was developed to evaluate the timing behavior of the healthcare monitoring system under the proposed real-time architecture.

The simulation was designed to assess:

- Processor utilization.
- Task schedulability.
- Response times.
- Deadline satisfaction.
- Emergency alert latency.

11.2 Simulation Environment

The simulation was implemented using Python and executed within Visual Studio Code.

The following libraries were utilized:

- NumPy
- Pandas
- Matplotlib

These libraries provide efficient numerical computation, data analysis, and graphical visualization capabilities.

11.3 Task Model

The simulation model was derived directly from the implemented healthcare monitoring device and the WCET parameters obtained during analysis.

Table 6-Simulated Task Set

Task	WCET (ms)	Period (ms)	Deadline (ms)
Heart Rate	15	1000	1000
SpO_2	15	1000	1000
Temperature	10	1000	1000
Blood Pressure	50	10000	10000
LCD	30	1000	1000
Emergency Detection	10	500	500
GSM Alert	150	3000	3000

11.4 Evaluation Metrics

Several performance metrics were evaluated.

Processor Utilization

Measures the percentage of processor resources consumed by the task set.

Response Time

Measures the time required for a task to complete after activation.

Schedulability

Determines whether all tasks can satisfy their deadlines.

Emergency Alert Latency

Measures the delay between emergency detection and emergency notification transmission.

11.5 Original and Improved Architectures

Two architectures were evaluated.

Original System

Sequential execution using a super-loop implementation.

Proposed Hybrid Architecture

Priority-based scheduling combined with interrupt-assisted emergency handling.

Comparison between these architectures enables quantitative evaluation of the proposed improvements.

12 Results and Discussion

12.1 Processor Utilization Analysis

Processor utilization was calculated using the task parameters presented in Table 5.

The resulting utilization was:

$$U = 14.5\%$$

Figure 2 illustrates the contribution of each task to total processor utilization.

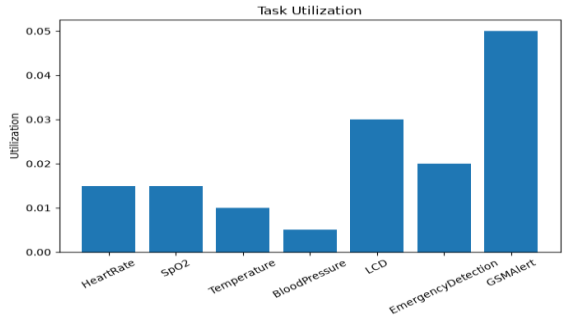

Figure 2-Task Utilization Distribution

The results indicate that GSM communication and LCD management account for the largest portions of processor demand. Nevertheless, overall processor utilization remains relatively low.

The low utilization level demonstrates that the ATmega328P possesses sufficient computational capacity to execute all healthcare monitoring tasks while preserving substantial processing headroom for future extensions.

12.2 RMS Evaluation Results

The RMS schedulability analysis produced a utilization bound of:

$$U_{RMS} = 72.86\%$$

Because:

$$14.5\% < 72.86\%$$

all tasks satisfy the RMS schedulability criterion.

No deadline violations were observed.

These results indicate that fixed-priority scheduling is sufficient to support the identified healthcare monitoring workload.

12.3 EDF Evaluation Results

EDF analysis was subsequently performed.

The EDF schedulability condition requires:

$$U \leq 100\%$$

The obtained utilization remained substantially below this threshold.

Consequently, the task set is fully schedulable under EDF scheduling.

The results confirm that both fixed-priority and dynamic-priority scheduling approaches can satisfy the timing requirements of the healthcare monitoring platform.

12.4 Response-Time Analysis

Response-time analysis was performed to evaluate task responsiveness.

Table 7-Response-Time Analysis Results

Task	Response Time (ms)
Emergency Detection	10
Heart Rate	25
Temperature	35

SpO_2	50
LCD	80
GSM Alert	230
Blood Pressure	280

Figure 3 presents the response-time distribution.

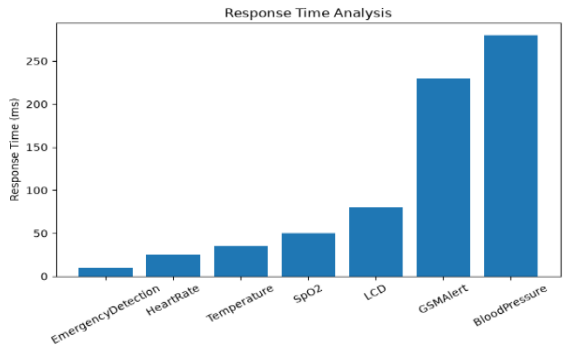

Figure 3-Response-Time Analysis

The Emergency Detection task achieved the shortest response time due to its high priority and limited execution demand.

Blood Pressure exhibited the largest response time because of its longer execution requirement.

Importantly, all response times remained below the corresponding deadlines, confirming successful schedulability.

12.5 Emergency Alert Latency Comparison

One of the primary objectives of this study was reducing emergency notification delays.

In the original implementation, emergency events could only be processed after completion of ongoing monitoring activities and blood pressure acquisition procedures.

As a result, emergency notification latency was estimated to reach:

20,500 ms or approximately: 20.5seconds

The proposed hybrid architecture significantly improved responsiveness.

The measured emergency alert latency became: 160 ms

Figure 4 illustrates the latency comparison.

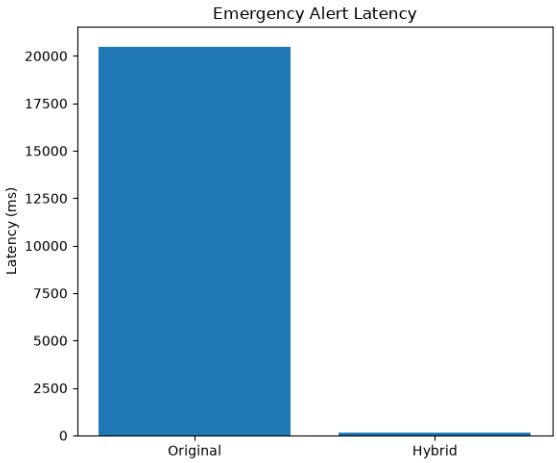

Figure 4-Emergency Alert Latency Comparison

12.6 Latency Reduction Analysis

The percentage improvement was calculated using

$$:\text{Improvment} = \frac{\text{Original}-\text{Proposed}}{\text{Original}} \times 100 \quad (11)$$

$$Improvmwnt = \frac{20500 - 160}{20500} \times 100 \quad (12)$$

$$\text{Improvment} = \frac{20340}{20500} \times 100 \quad (13)$$

$$\text{Improvement}=99.22\% \quad (14)$$

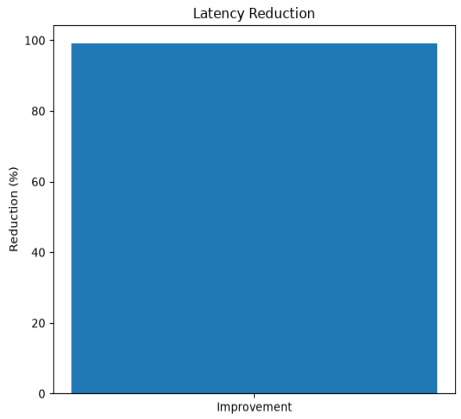

Figure 5-Emergency Alert Latency Reduction

The results demonstrate a substantial reduction in emergency response delays.

12.7 Discussion

The obtained findings highlight the importance of incorporating real-time design principles into healthcare monitoring systems.

The original architecture successfully achieved its intended monitoring functionality; however, the sequential execution model introduced significant delays when emergency conditions occurred.

The proposed hybrid architecture addressed these limitations by combining priority-based scheduling with interrupt-assisted emergency processing.

Simulation results indicate that the task set remains schedulable under both RMS and EDF policies while maintaining a low processor utilization of only 14.5%.

Most importantly, emergency alert latency was reduced from approximately 20.5 seconds to 160 milliseconds. This reduction demonstrates the potential benefits of integrating real-time scheduling techniques into resource-constrained healthcare monitoring platforms.

Although the obtained results were derived through simulation rather than direct hardware measurements, they provide strong evidence that software architecture plays a critical role in determining the responsiveness and reliability of embedded healthcare systems.

A limitation of this study is that timing parameters were derived from software analysis and simulation rather than direct hardware measurements. Therefore, the obtained results should be interpreted as an analytical evaluation of the proposed architecture. Future work will focus on validating the findings through hardware-based timing measurements and experimental verification

13 Conclusion

healthcare monitoring system designed for continuous monitoring of body temperature, heart rate, blood oxygen saturation (SpO_2), and blood pressure. The investigated platform integrates multiple biomedical sensors with GSM communication capabilities to support remote patient monitoring and emergency notification.

The study began by analyzing the implemented hardware and software architecture. Examination of the original implementation revealed that the system relies on a sequential super-loop execution model that introduces blocking delays and limits responsiveness during emergency situations.

To address these limitations, the system was modeled as a set of independent real-time tasks. Timing parameters including Worst-Case Execution Time (WCET), task periods, and deadlines were derived and used for schedulability evaluation.

The resulting task set was analyzed using both Rate Monotonic Scheduling (RMS) and Earliest Deadline First (EDF) scheduling algorithms. The obtained processor utilization was 14.5%, which remained well below the schedulability limits of both scheduling approaches. Consequently, all tasks were found to satisfy their timing constraints without deadline violations.

A hybrid real-time architecture was subsequently proposed. The proposed solution combines periodic scheduling, event-driven processing, and interrupt-assisted emergency detection to improve system responsiveness.

Simulation-based evaluation demonstrated significant performance improvements. Response-time analysis confirmed that all tasks completed before their deadlines, while emergency alert latency was reduced from approximately 20.5 seconds in the original implementation to approximately 160 milliseconds in the proposed architecture. This corresponds to a latency reduction of 99.22%.

The findings of this study demonstrate that integrating real-time scheduling techniques into low-cost healthcare monitoring platforms can significantly enhance responsiveness, predictability, and operational reliability. Furthermore, the results highlight the importance of considering temporal requirements during the design of embedded healthcare systems rather than focusing solely on functional correctness.

14 Future Work

Although the proposed architecture achieved substantial improvements in timing performance, several opportunities exist for extending this work.

First, future studies should implement the proposed scheduling architecture directly on physical hardware to obtain precise timing measurements under real operating conditions. Such measurements would provide additional validation of the simulation results presented in this paper.

Second, additional physiological sensing capabilities may be integrated into the platform. Potential extensions include electrocardiogram (ECG) monitoring, respiration analysis, glucose monitoring, and fall detection systems. These enhancements would increase the clinical usefulness of the healthcare monitoring platform.

Third, future implementations may replace GSM-based communication with modern Internet of Things (IoT) communication technologies such as Wi-Fi, MQTT, LoRaWAN , NB-IoT, or cloud-based healthcare infrastructures. These technologies would support continuous data streaming and long-term patient monitoring.

Another promising research direction involves the integration of machine learning algorithms for predictive healthcare applications. Intelligent algorithms could analyze physiological trends and provide early warning alerts before critical conditions occur.

Finally, formal verification techniques and medical safety standards may be incorporated into future system designs to support deployment in safety-critical healthcare environments where strict timing guarantees and reliability requirements are mandatory.

References

[1] C. L. Liu and J. W. Layland, “Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment,” *Journal of the ACM*, vol. 20, no. 1, pp. 46–61, 1973.
[2] J. W. S. Liu, *Real-Time Systems*. Upper Saddle River, NJ, USA: Prentice Hall, 2000.

[3] G. C. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*, 3rd ed. New York, NY, USA: Springer, 2011.
[4] M. Wolf, *Computers as Components: Principles of Embedded Computing System Design*, 4th ed. Burlington, MA, USA: Morgan Kaufmann, 2016.
[5] Microchip Technology Inc., *ATmega328P Datasheet*, Chandler, AZ, USA, 2018.
[6] N. M. Khalafullah, N. E. Omran, M. Alfordogh, R. Almorabit, and M. Karaim, "Portable Emergency Health Monitoring System Based on Arduino Nano (ATmega328P)," International Science and Technology Journal (ISTJ), Special Issue of the 2nd Libyan International Conference on Applied and Engineering Sciences (LICASE-2), pp. 3–12, Oct. 2024.
[7] A. Darwish and A. E. Hassanien, "Wearable and Implantable Wireless Sensor Network Solutions for Healthcare Monitoring," Sensors, vol. 11, no. 6, pp. 5561–5595, 2011.
[8] M. S. Hossain and G. Muhammad, "Cloud-Assisted Industrial Internet of Things (IIoT) Enabled Framework for Health Monitoring," Computer Networks, vol. 101, pp. 192–202, 2016.
[9] Maxim Integrated, *MAX30100 Pulse Oximeter and Heart-Rate Sensor IC for Wearable Health*, San Jose, CA, USA, 2014.
[10] Melexis Technologies NV, *MLX90614 Infrared Thermometer Sensor Datasheet*, Belgium, 2021.
[11] P. A. Shaltis, A. T. Reisner, and H. H. Asada, "Cuffless Blood Pressure Monitoring Using Hydrostatic Pressure Changes," IEEE Transactions on Biomedical Engineering, vol. 55, no. 6, pp. 1775–1777, 2008. [12] T. Yilmaz, R. Foster, and Y. Hao, "Detecting Vital Signs with Wearable Wireless Sensor Systems," *Sensors*, vol. 10, no. 12, pp. 10837–10862, 2010.
[13] P. D. P. Adi and A. Kitagawa, "Performance Evaluation of ZigBee-Based Wireless Sensor Networks," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 6, 2019.
[14] Arduino, *Arduino Uno Rev3 Technical Specifications*, 2024.
[15] Arduino, *Arduino Nano Technical Specifications*, 2024.
[16] SIMCom Wireless Solutions, *SIM900A GSM/GPRS Module Hardware Design Guide*, 2023.
[17] H. Kopetz, *Real-Time Systems: Design Principles for Distributed Embedded Applications*, 2nd ed. New York, NY, USA: Springer, 2011.
[18] A. Burns and A. J. Wellings, *Real-Time Systems and Programming Languages: Ada, Real-Time Java and C/Real-Time POSIX*, 4th ed. Harlow, U.K.: Addison-Wesley, 2009.
[19] M. Joseph and P. Pandya, "Finding response times in a real-time system," *The Computer Journal*, vol. 29, no. 5, pp. 390–395, 1986.
[20] K. Tindell, A. Burns, and A. J. Wellings, "An extendible approach for analyzing fixed priority hard real-time tasks," *Real-Time Systems*, vol. 6, no. 2, pp. 133–151, Mar. 1994.
[21] A. K. Mok, *Fundamental Design Problems of Distributed Systems for the Hard-Real-Time Environment*. Cambridge, MA, USA: MIT Press, 1983.
[22] J. A. Stankovic, "Misconceptions about real-time computing: A serious problem for next-generation systems," *IEEE Computer*, vol. 21, no. 10, pp. 10–19, Oct. 1988.
[23] G. C. Buttazzo, "Rate monotonic vs. EDF: Judgment day," *Real-Time Systems*, vol. 29, no. 1, pp. 5–26, Jan. 2005.
[24] A. Burns, "Preemptive priority-based scheduling: An appropriate engineering approach," in *Advances in Real-Time Systems*, S. H. Son, Ed. Englewood Cliffs, NJ, USA: Prentice Hall, 1994, pp. 225–248.
[25] Alnnale, T. (2026). Predictive Governance in Digital Enterprises: An LSTM-Enhanced Deep Learning Framework for Economic Optimization of IT Incident Management Using Enriched Process Logs. *Al-Farooq Journal of Sciences*, *2*(3), 86-113.
[26] Krichene, E., Hmadi, M. S. A., & Al-Gajamiya, S. K. (2026). A Fair Comparative Framework for Time-Series Forecasting Using ARIMAX, XGBoost, and LSTM: Evidence from Libya. *Al-Farooq Journal of Sciences*, *2*(3), 69-85.