Type of Contribution:
Editorial
Research Paper
Review Paper
Case Study

INTRO: JURNAL INFORMATIKA DAN TEKNIK ELEKTRO
DOI: 10.51747/intro.v5i1.p27-41
ISSN 3025-602X

This article contributes to:

Design and Simulation of an Arduino-Based Safety Device System for AHTS Auxiliary Engine

Baharuddin[1], Basitha Febrinda Hidayatulail[1]*, Abd. Rabi'[1]

[1] Electrical Engineering, University of Merdeka Malang, 65146, **Indonesia**

* basitha@unmer.ac.id

Abstract

Anchor Handling Tug Supply (AHTS) vessels require a reliable auxiliary engine monitoring and protection system to prevent severe damage caused by low oil pressure, high coolant temperature, and overspeed conditions. Conventional safety device systems on some vessels still exhibit limitations in real-time monitoring and alarm history logging. This study aims to design and implement a simulation of an Arduino Mega 2560-based auxiliary engine monitoring and safety device system integrated with a Nextion Human Machine Interface (HMI). The system is equipped with sensor loss detection, alarm history logging based on RTC DS3231 and EEPROM, and an intuitive HMI interface. Tests were conducted on system functionality, sensor accuracy, sensor loss detection, and the protection system. The results show that the system successfully responds to abnormal conditions according to the predetermined setpoints: overspeed shutdown at 1700 RPM, low oil pressure shutdown at 1.0 Bar, and high temperature shutdown at 95°C. Sensor reading accuracy yields average errors of 0.73% for the speed sensor, 0.99% for the temperature sensor, and 1.94% for the oil pressure sensor. The sensor loss detection feature and HMI interface function with a 100% success rate based on 10 trials. A conceptual comparison with conventional protection systems indicates that the proposed system provides integrated monitoring and protection, timestamp-based data logging, sensor fault detection, and flexible setpoint configuration.

Keywords: Arduino, Safety Device, Auxiliary Engine, Monitoring System, AHTS, HMI.

Article Info
Submitted: 2026-04-29
Revised: 2026-06-08
Accepted: 2026-06-20
Published: 2026-06-28

Publisher
Universitas Panca Marga

1. Introduction

Anchor Handling Tug Supply (AHTS) vessels are offshore support vessels that function in anchor handling, logistics distribution, and supporting offshore oil and gas drilling activities. The auxiliary engine on an AHTS vessel serves as the main electrical power source for navigation, communication, lighting, and ship safety systems [1]. The reliability of the auxiliary engine is crucial because a power system failure can cause the cessation of ship operations and potentially pose severe hazards.

According to The Swedish Club report, auxiliary engine damage accounts for approximately 16% of engine damage volume and 13% of total engine claim costs, with an average claim cost reaching USD 345,000 [2]. Data from the Australian Maritime Safety Authority (AMSA) in 2019 showed that the power, propulsion, and steering category was the highest incident at 26%, with 130 incidents related to auxiliary engines [3]. In the field, the author observed cases of cracked engine blocks due to overspeed, jammed crankshafts due to low oil pressure, and engine room fires caused by unprotected overheating.

Several previous studies have explored the application of Arduino-based microcontrollers for various monitoring and protection systems. For instance, research on the design of a three-phase motor protection system against overvoltage using an Arduino Mega 2560 demonstrates the effectiveness of this microcontroller for real-time protection applications with affordable costs [4]. Furthermore, Arduino-based monitoring systems have been developed for various applications, including water level monitoring using LoRa technology and industrial control systems integrating Arduino Mega 2560 with PLCs [5], [6]. Additionally, Arduino-based systems have been successfully implemented for continuous monitoring applications, such as BTS tower tilt monitoring, proving the flexibility of the Arduino platform for integrated monitoring and control systems [7]. The implementation of sensors and Arduino-based systems has also been utilized for automatic environmental applications, demonstrating the platform's versatility [8].

Regarding specific engine protection systems, Several previous studies have discussed microcontroller-based engine protection systems. Wicaksana et al. [9] designed a microcontroller-based safety device to prevent overheating, but it only monitored a single parameter. Hendrawan et al. [10] created an Arduino-based generator set safety device simulator, but it did not integrate RPM and oil pressure monitoring. Mahendra et al. [11] developed an IoT-based monitoring system, but it was not equipped with automatic hardware-level protection. Ahmadjayadi [12] integrated monitoring and protection, but did not utilize an HMI and alarm history logging.

Previous studies have generally focused on monitoring a single parameter or have been limited to monitoring functions without integrating hardware-based protection, HMI, and data logging. This research addresses these limitations by developing an integrated monitoring and protection system for an AHTS auxiliary engine based on the Arduino Mega 2560. The novelty lies in the comprehensive integration of three vital engine parameters—oil pressure, coolant temperature, and RPM—with automatic protection and an intuitive Nextion HMI within a single system. In addition, the system incorporates sensor loss detection to distinguish abnormal engine conditions from sensor damage or disconnection, as well as timestamp-based alarm history logging using the RTC DS3231 and EEPROM. This research also offers a cost-effective and flexible modernization (retrofitting) alternative for ship auxiliary engine protection systems, particularly as a simulation medium and crew training tool.

2. Research and Methods

2.1 System Desain

The system is designed using the Arduino Mega 2560 as the central controller with the following main component configuration: DS18B20 temperature sensor (One-Wire protocol), analog VDO oil pressure sensor 0–10 bar (resistive type, calibrated with a Look-Up Table), FC-03 infrared speed sensor, RTC DS3231 (I^2C), Nextion HMI NX3224T024 (serial UART), 1-channel relay module, and an active buzzer. The VDO pressure sensor is manually calibrated using a 10-point Look-Up Table (LUT) method with linear interpolation due to its non-linear resistance characteristics against pressure. Data conversion is performed in three stages: (1) ADC to voltage, (2) voltage to sensor resistance, and (3) linear interpolation from the LUT to obtain the pressure value in Bar.

2.2 Block Diagram

The block diagram of the safety device system designed in Figure 1 shows that the microcontroller will read data in the form of analog signals from each sensor and convert them into the required units (temperature, pressure, and speed). The system initialization results will be displayed on the HMI, and the system will be ready. If the start command from the HMI is executed, the microcontroller will give an ON command and check the threshold of the sensor signal data. If it exceeds the setpoint, the microcontroller will trigger the alarm and shutdown, display the alarm on the HMI, and save the history in the data logging.

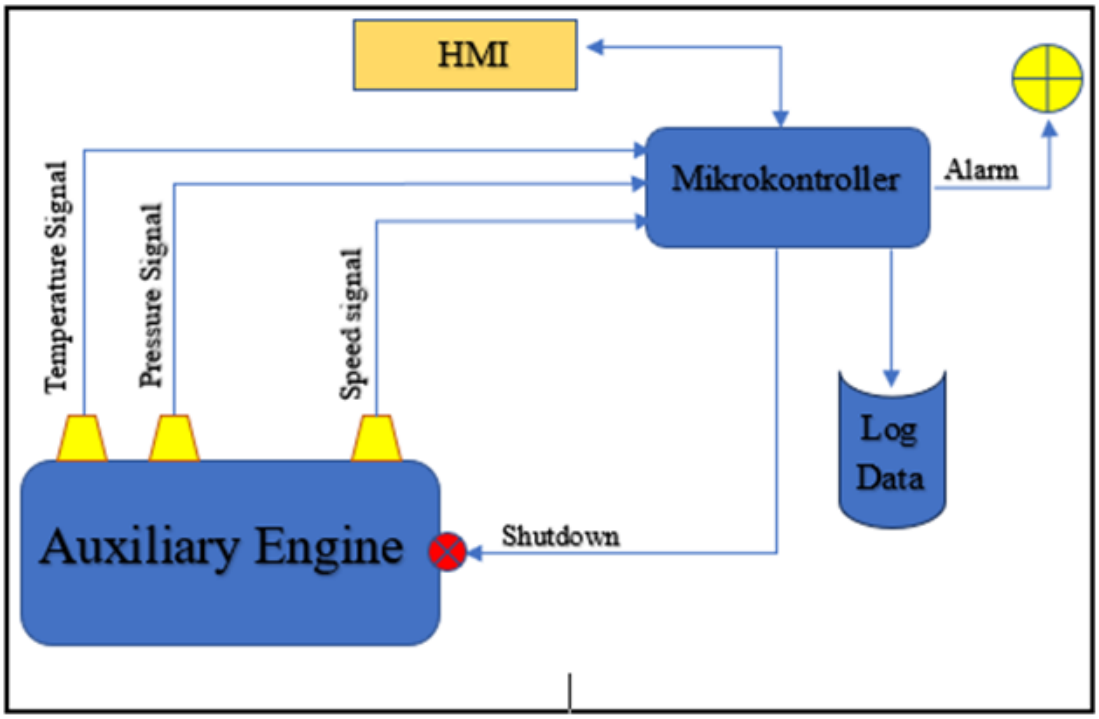

Figure 1. System Block Diagram

2.3 Schematic Diagram

The collected data were analyzed using descriptive statistical methods. The fluid pressure and vacuum pressure values were compared for both impeller configurations. The efficiency of the pump was calculated by analyzing the hydraulic power and mechanical power delivered by the pump system, with special attention given to the relationship between the impeller blade count and the overall performance metrics.

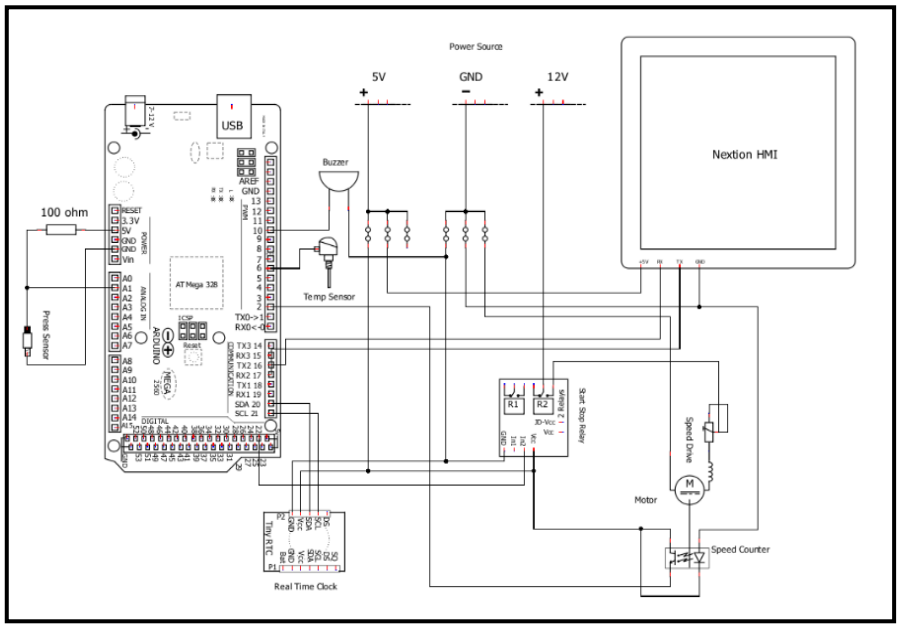

Figure 2. System Schematic Diagram

Figure 2 shows the schematic diagram of the hardware implementation of the safety device system centered on the Arduino Mega 2560 microcontroller. This system is designed with a dual power supply configuration to separate the logic circuit and load power; a 5V DC source is used to supply the microcontroller, sensors, and HMI module, while a 12V DC source is allocated to drive the simulation DC motor and relay module. Three vital engine parameters are monitored through specific input interfaces: the Oil Pressure Sensor (VDO type) is connected to an analog pin to read resistance changes, the Temperature Sensor (DS18B20) is connected to a digital pin using the One-Wire protocol, and the Speed Sensor (FC-03) is connected to an interrupt pin to count pulses from the Speed Counter precisely. The RTC DS3231 module is integrated via the I2C bus (SDA and SCL pins) to provide real-time data for alarm history logging.

On the output and control side, the Nextion HMI communicates with the Arduino via a serial interface (UART) for data visualization and receiving user commands. The physical protection mechanism is implemented through a Relay module installed in series on the 12V power line to the DC Speed Drive and Motor. This relay functions as the main shutdown actuator controlled by the Arduino digital pin. An active buzzer is also connected to the output pin to provide an auditory warning. This schematic represents a closed-loop system where the Arduino monitors feedback from the sensors while controlling the load. When the program logic detects abnormal conditions such as overspeed, overheating, or low oil pressure, the microcontroller will activate the buzzer and cut the signal to the relay, so the 12V power flow to the motor is cut off, simulating the emergency shutdown process of the auxiliary engine automatically and safely.

2.4 Flowchart

The flowchart in **Figure 3** explains the program logic flow of the Arduino-based AHTS vessel auxiliary engine safety device system. The process begins with system initialization, where I/O pins, HMI serial communication, RTC, and sensors are configured. Next, the system continuously checks the engine speed status. If the speed is not detected (speed inactive), the system will update the HMI status to STOP. However, if the speed is active, the system will wait for a 10-second delay to ensure the engine is in a stable condition before protection is activated

After the delay, the system reads the oil pressure and coolant temperature sensor values in real-time, then displays this data on the HMI interface and saves it to EEPROM memory for data logging purposes. The system then evaluates whether the engine parameter values are still within the established normal threshold limits. If the values are still within normal limits, the system will repeatedly check the speed. Conversely, if values exceeding safe limits are detected (such as overspeed, low oil pressure, or overtemperature), the system will activate the alarm and siren,

perform a shutdown by deactivating the relay, change the HMI status to FAULT/STOP, and record the event in the alarm log. This mechanism guarantees automatic protection against abnormal engine conditions.

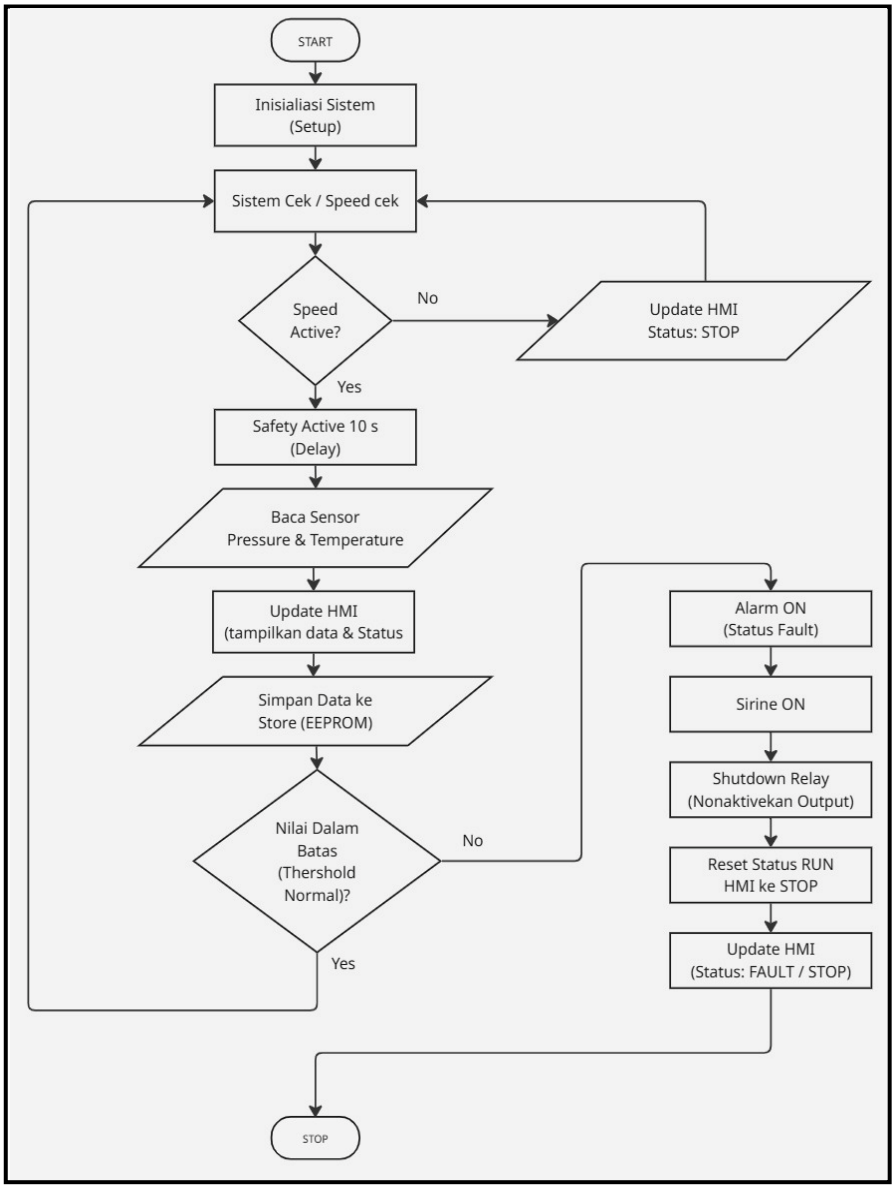

Figure 3. System Flowchart

2.5 Software Design

The software is developed using the Arduino IDE with a modular approach comprising: sensor reading, RPM interrupt, fault detection, EEPROM data logging, HMI serial communication, state management, and protection logic. The protection system is configured with the following setpoints as per **Table 1**.

Table 1. Protection Setpoints

Parameter	Alarm Threshold	Shutdown Threshold
Overspeed	≥ 1600 RPM	≥ 1700 RPM
Low oil pressure	≤ 1.5 Bar	≤ 1.0 Bar
Over temperature	≥ 85°C	≥ 95°C
Loss sensor	Loss of Connection	-

Table 1 presents the protection setpoints used as reference limits in the monitoring and protection system for the AHTS auxiliary engine. Each parameter is assigned two protection levels, namely the alarm threshold and shutdown threshold. For the overspeed parameter, an alarm is activated when the engine speed reaches ≥1600 RPM, while an automatic shutdown is triggered at ≥1700 RPM. For low oil pressure, the alarm is activated when the pressure decreases to ≤1.5 bar, whereas shutdown occurs when the pressure reaches ≤1.0 bar. For coolant overtemperature, an alarm is generated at ≥85°C, followed by an automatic shutdown when the temperature reaches ≥95°C. In addition, the system is equipped with a sensor loss detection feature to identify loss of connection or sensor failure. Low oil pressure and overtemperature protection are activated after a 5-second delay once the engine is idle to prevent false shutdowns caused by temporary parameter fluctuations during the initial operating condition. In contrast, overspeed protection remains active at all times because excessive engine speed can occur suddenly and may cause serious engine damage. These protection setpoints enable the system to function not only as a monitoring device but also as an early warning and automatic protection system against potentially hazardous engine operating conditions.

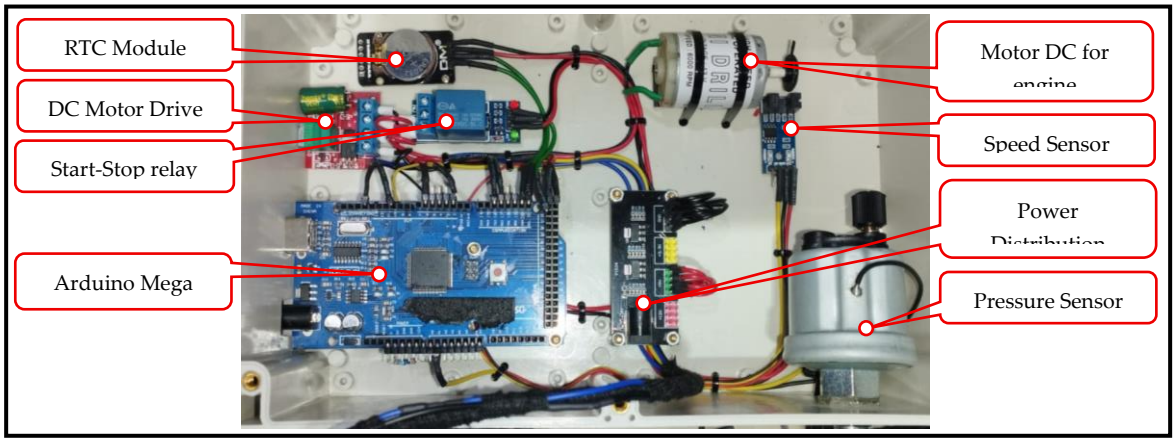

Figure 4. Physical Implementation of the System

2.6 Testing Method

Testing was conducted in 10 trials for each parameter, including: (1) HMI-Arduino communication, (2) DS18B20 temperature sensor accuracy, (3) VDO pressure sensor accuracy, (4) RPM sensor accuracy, (5) RTC DS3231 module, (6) sensor loss detection, (7) EEPROM storage, and (8) the main protection system. The physical implementation of the system is shown in **Figure 4**.

3.Results and Discussion

3.1 Speed Sensor (RPM) Testing

The FC-03 RPM sensor testing was conducted at 10 rotation points. In the initial stage, noise was found due to optical switch bouncing and electromagnetic interference (EMI) from the DC motor. Mitigation was performed using interrupt debounce, a 250 ms sampling time setting, and physical sensor trigger improvement. The test results are shown in **Table 2**.

Table 2. RPM Sensor Test Results

No	RPM On HMI	RPM Comparison	Error
1	443	437.7	1.21%
2	600	593.3	1.13%
3	707	702.4	0.65%
4	904	904.5	0.06%
5	1024	1022.4	0.16%
6	1205	1197.0	0.67%
7	1311	1301.2	0.75%
8	1420	1407.9	0.86%
9	1513	1497.7	1.02%
10	1692	1679.1	0.77%

The stability of RPM readings is inseparable from the implementation of the interrupt debounce method in the software. In the initial testing, the pulse signal from the FC-03 sensor experienced noise due to electromagnetic interference (EMI) emitted by the DC motor simulator. By setting the data update sampling time to the HMI at a 250 ms interval, the system successfully filtered out false pulses without compromising real-time responsiveness, so the RPM numbers displayed on the screen did not flicker and were easy for the operator to read. The average error is 0.73%. The readings are stable and linear from low to high speeds, proving the system can work reliably despite environmental disturbances during testing.

3.2 DS18B20 Temperature Sensor Testing

Testing was conducted by comparing HMI readings with a digital thermometer at 10 temperature points (30–100°C). The average error is 0.99%. Higher errors at low temperatures (31.2°C) were caused by differences in probe measurement points and the 10-bit resolution of the sensor. A 500 ms sampling time was chosen according to the internal conversion characteristics of the DS18B20, which requires up to 187 ms for 10-bit resolution. The results are shown in **Table 3**.

3.3 Oil Pressure Sensor Testing

The VDO sensor was calibrated using a 10-point LUT with a linear interpolation system. Testing was conducted at 10 pressure points with an average

error of 1.94% (excluding the 0.5 Bar point). High error at 0.5 Bar is a normal characteristic of resistive sensors in the deadband region near 0 Bar, with an absolute difference of only 0.2 Bar—still within simulation tolerance. A 500 ms sampling time with a moving average algorithm was applied to filter ADC noise. The results are shown in **Table 4.**

Table 3. Temperature Sensor Test Results

No	Temperature Reference	Temperature On HMI	Error
1	100.0°C	100.3°C	0.30%
2	95.0°C	95.8°C	0.84%
3	90.0°C	90.5°C	0.56%
4	85.0°C	85.8°C	0.94%
5	80.0°C	80.8°C	1.00%
6	70.0°C	70.3°C	0.43%
7	60.1°C	60.8°C	1.16%
8	50.4°C	50.3°C	0.20%
9	40.1°C	40.0°C	0.25%
10	31.2°C	29.5°C	5.44%

The percentage error reaching 40% at the 0.5 Bar point is a normal mathematical anomaly for resistive-type sensors. In the low-pressure range (approaching 0 Bar), the resistance characteristic curve of the VDO sensor is highly non-linear (deadband region). Although the percentage error appears large, the absolute difference is only 0.2 Bar, which is still within the safe tolerance limit for simulation and early monitoring purposes. To mitigate noise on the Arduino ADC pin, the system applies a moving average algorithm at every 500 ms sampling time.

Table 4. Oil Pressure Sensor Test Results

No	Pressure Reference	Pressure On HMI	Error
1	6.7 Bar	6.8 Bar	1.49%
2	6.0 Bar	6.3 Bar	5.00%
3	5.0 Bar	5.1 Bar	2.00%
4	4.0 Bar	4.0 Bar	0.00%
5	3.0 Bar	3.1 Bar	3.33%
6	2.5 bar	2.7 Bar	8.00%
7	2.0 Bar	2.0 Bar	0.00%
8	1.5 bar	1.5 Bar	0.00%
9	1.0 bar	1.0 Bar	0.00%
10	0.5 Bar	0.7 Bar	40.00%

3.4 RTC, EEPROM, and Sensor Loss Testing

The RTC DS3231 successfully maintained accurate timekeeping after being powered off for 2–5 minutes without any discrepancy, thanks to the CR2032 backup battery. The EEPROM successfully stored 10 permanent alarm history entries even

when the system lost power. The system successfully detected sensor loss on all three sensors with a 100% success rate out of 10 trials, including: the speed sensor was not given a signal during run, the temperature sensor was disconnected, and the pressure sensor was disconnected.

3.5 Main Protection System Testing

Table 5. Main Protection System Test Results

No	Alarm Type	Limit Setpoint	Result
1	Over Speed Alarm	≥ 1600 RPM	Alarm Active
2	Over Speed Shutdown	≥ 1700 RPM	Alarm + Shutdown
3	Low Oil Pressure Alarm	≤ 1.5 Bar	Alarm Active
4	Low Oil Pressure Shutdown	≤ 1.0 Bar	Alarm + Shutdown
5	Over Temperature Alarm	≥ 85°C	Alarm Active
6	Over Temperature Shutdown	≥ 95°C	Alarm + Shutdown

The **Table 5** presents the results of testing the alarm and shutdown functions based on the predefined protection setpoints of the AHTS auxiliary engine monitoring and protection system. The results demonstrate that each monitored parameter has two levels of protection: an alarm threshold and a shutdown threshold. For the overspeed condition, the alarm is activated when the engine speed reaches ≥1600 RPM, while at ≥1700 RPM, the system activates both the alarm and automatic shutdown function. For low oil pressure, the alarm is activated when the pressure decreases to ≤1.5 bar, whereas at ≤1.0 bar, the system activates the alarm and performs an automatic shutdown. Similarly, for overtemperature, the alarm is activated when the temperature reaches ≥85°C, while at ≥95°C, the system activates the alarm and shutdown function. These results indicate that the developed system can provide an early warning when engine parameters approach critical conditions and can automatically initiate protective shutdown when the predefined critical limits are reached. Therefore, the test confirms that the monitoring and protection system responds appropriately to abnormal engine conditions and provides a multi-level protection mechanism to reduce the risk of engine damage caused by overspeed, low oil pressure, and excessive temperature.

Based on the results in **Table 6** the protection system worked according to logic with a 100% success rate. The shutdown relay and buzzer activated according to the setpoint. The HMI interface display is shown in **Figure 5**.

3.6 Analysis of Test Results

The choice of the laboratory simulation method in this research is based on three crucial technical and maritime regulatory considerations. First, operational safety factors and blackout risks. Testing the safety device response directly on an operating ship's auxiliary engine carries fatal risks if a bug occurs in the prototype logic or a relay failure triggers a false shutdown. Simulation allows for testing extreme scenarios such as overspeed and loss of oil pressure safely, repeatedly, and

in a controlled manner without damaging the original engine block. Second, compliance with classification society regulations (BKI, ABS, DNV) which require marine-grade certification for safety-critical devices. This Arduino prototype functions as a Proof of Concept to validate the control logic before migrating to a marine-grade PLC. Third, the flexibility of fault injection which allows researchers to inject disturbances precisely at the sensor signal level to verify system reliability.

Figure 5. HMI Interface Display

Based on all the tests conducted, the monitoring and protection system based on the Arduino Mega 2560 and Nextion HMI is capable of working according to the design. All hardware components are well integrated and communicate optimally with each other. In the RPM sensor testing using the FC-03 sensor with the interrupt method, the system is capable of reading the rotor speed in real-time. Although noise was initially found due to optical switch bouncing and electromagnetic interference (EMI) from the simulator motor, after mitigation through sensor trigger improvement, sampling time adjustment, and interrupt debounce implementation, RPM readings became stable with an average error of 0.73% (maximum 1.21%). This proves the system can work reliably despite environmental disturbances during testing.

The DS18B20 temperature sensor showed good performance with the ability to read temperature in real-time and display it on the HMI. The resulting error rate is very reasonable with a maximum of 1.16% in the 30–100°C testing range. The overtemperature protection system also functioned well when the temperature exceeded the specified limit.

In the VDO oil pressure sensor testing, the system successfully converted ADC data into pressure units (Bar) using a non-linear Look-Up Table (LUT) method with linear interpolation. Manual calibration is required because the VDO sensor does not have a standard Arduino library. The system is capable of detecting low oil pressure conditions and executing alarm and shutdown functions according to the program logic. An average error of 1.94% is still within reasonable tolerance limits for a resistive sensor, although at the 0.5 Bar point a 40% error is seen, which is a normal

characteristic of the sensor's deadband region with an absolute difference of only 0.2 Bar.

Testing of the RTC DS3231 module shows that this module is capable of storing and maintaining time data accurately using a CR2032 backup battery, so the time continues to run even when the system is turned off. This feature is very important for recording timestamps in alarm history and data logging.

In determining the sampling time, the system applies a strategy adapted to the physical response characteristics of each parameter. For the RPM sensor, the refresh rate to the HMI is set at a 250 ms interval so that the display does not flicker and is easy for the operator to read, while also preventing serial UART communication bottlenecks. The DS18B20 temperature sensor uses a 500 ms sampling time, considering the internal ADC conversion time of the sensor which requires up to 187 ms for 10-bit resolution. The oil pressure sensor also uses a 500 ms interval, during which the microcontroller performs hundreds of ADC readings filtered using a moving average algorithm to ensure the displayed pressure value is stable, precise, and free from noise.

The alarm and protection system successfully worked according to the designed program logic. When abnormal conditions occur such as overspeed (≥1700 RPM), low oil pressure (≤1.0 Bar), overtemperature (≥95°C), or sensor loss, the system is capable of providing an alarm via the buzzer and HMI display, as well as performing an automatic shutdown by cutting the START relay. Based on 10 repetitive trials, the system showed a 100% success rate with consistent and fast responses.

Overall, the analysis of the test results shows that the designed system has a high level of accuracy, good stability, and reliability in detecting and responding to abnormal conditions. The errors produced by all sensors are in the small category and can be relied upon as a reference in the simulation of the AHTS vessel auxiliary engine safety device system.

3.7 Comparison with Conventional Systems

Table 6. Conceptual Comparison of Conventional and Arduino-Based Systems

No	Aspect	Conventional System	Arduino System
1	Monitoring Interface	Gauge / Indicator lamp	HMI
2	Alarm Logging	Manual (Log Book)	Automatic + Timestamp
3	Sensor Fault Detection	None	Available
4	Setpoint Flexibility	Limited	Flexible (Via software)
5	System Integration	Separated	Centralized Integration
6	Cost & Availability	Expensive, Scarce Components	*Cost-effective, open-source*

The conceptual comparison suggests that the proposed system offers greater flexibility in monitoring, alarm logging, sensor fault detection, and setpoint

configuration. In conventional marine-grade systems, changing protection setpoints may require dedicated hardware or configuration procedures, whereas the proposed Arduino-based prototype allows these parameters to be modified through software. However, these observations are based on system characteristics and conceptual comparison rather than direct experimental benchmarking; therefore, claims regarding superiority in maintenance cost, performance, or operational reliability should be validated through future head-to-head testing.

The Arduino-based prototype integrates monitoring and protection functions in a single microcontroller, with an intuitive HMI interface, RTC-based historical data logging, sensor loss detection, and flexible setpoint configuration. Based on the present laboratory validation, the prototype is suitable as a simulation medium and crew training aid, while its applicability as a marine retrofit solution requires further field validation and compliance assessment.

4. Conclusion

Based on the research results and testing, it can be concluded: (1) The simulation of the AHTS vessel auxiliary engine safety device system based on the Arduino Mega 2560 was successfully designed and integrated with the VDO oil pressure sensor, DS18B20 temperature sensor, FC-03 speed sensor, RTC DS3231, and Nextion HMI. The system is capable of reading operational parameters in real time with average errors of 0.73% (RPM), 0.99% (temperature), and 1.94% (pressure). (2) The system successfully detected and responded to abnormal conditions automatically: overspeed shutdown at 1700 RPM, low oil pressure shutdown at 1.0 Bar, and overtemperature shutdown at 95°C, with a 100% success rate across 10 trials. The sensor loss detection feature and HMI interface also functioned with a 100% success rate. (3) Based on the conceptual comparison presented in this study, the Arduino-based system provides integrated monitoring and protection, an intuitive HMI interface, RTC-based timestamp data logging, sensor loss detection, and flexible setpoint configuration. These features indicate potential advantages over conventional systems; however, no direct head-to-head experimental comparison was conducted in this study. The proposed system is therefore considered suitable as a simulation medium and crew training aid, while its application as an alternative modernization (retrofitting) solution for ship auxiliary engine protection systems requires further validation. However, this prototype has been validated only through laboratory-based simulation and has not yet been tested under the extreme vibration, temperature variation, electromagnetic interference, and other environmental conditions encountered in actual shipboard operation. Future work should therefore focus on field testing and validation under real marine

operating conditions, as well as further development toward marine-grade hardware and classification requirements.

Authors' Declaration

Authors' contributions and responsibilities - The author would like to express gratitude to the First Supervisor, Basitha Febrinda Hidayatulail, SS.T., M.T., and the Second Supervisor, Ir. Abd. Rabi', M.Kom., for their guidance in the preparation of this research, as well as the Electrical Engineering Study Program, Faculty of Engineering, Universitas Merdeka Malang, which has supported the implementation of this research.

Funding - No funding information from the authors.

Availability of data and materials - All data is available from the authors.

Competing interests - The authors declare no competing interest.

Additional information - No additional information from the authors.

References

[1] T. Lamb, Ship Design and Construction. New York, NY, USA: Society of Naval Architects and Marine Engineers, 2003.

[2] The Swedish Club, "Machinery Damage Report 2022," The Swedish Club Report, Gothenburg, Sweden, 2022.

[3] Australian Maritime Safety Authority (AMSA), "Technical Incidents Report 2019," AMSA Marine Safety Report, Canberra, Australia, 2019.

[4] T. Maulidan, Subairi, and B. F. Hidayatulail, "Desain Sistem Perlindungan Motor Tiga Fasa dari Tegangan Lebih Berbasis Mikrokontroler Arduino Mega 2560," Jurnal Teknik Elektro, vol. 15, no. 1, pp. 46–53, 2026, doi: 10.26740/jte.v15n1.p46-53.

[5] M. Y. Firmansyah, A. B. Setiawan, and B. F. Hidayatulail, "Prototipe Alat Monitoring Ketinggian Air Tambak dan Sungai Berbasis LoRa (Long Range)," Metrotech (Journal of Mechanical and Electrical Technology), vol. 4, no. 3, pp. 270–276, 2025, doi: 10.70609/metrotech.v4i3.7675.

[6] S. Santika, N. Nachrowie, D. A. Prasetya, and B. F. Hidayatulail, "Mini Plant Sistem Pengendali Berat Limestone Pada PLTU Tanjung Jati B Unit #3&4 Berbasis PLC Dan Arduino Mega 2560," JASIEK (Jurnal Aplikasi Sains, Informasi, Elektronika dan Komputer), 2019.

[7] S. Benyamin, N. Nachrowie, and W. Dirgantara, "Purwarupa Sistem Monitoring Kemiringan Menara Base Transceiver Station (BTS) Berbasis Arduino sebagai Solusi Maintenance Menara," JASIEK (Jurnal Aplikasi Sains, Informasi, Elektronika dan Komputer), vol. 4, no. 2, pp. 85–92, 2022, doi: 10.26905/jasiek.v4i2.9174.

[8] M. I. Ashari and A. H. Yuwono, "Application Of Photodiodes As Sensors In Arduino-Based Automatic River Garbage Cleaner," JEEMECS (Journal of Electrical Engineering, Mechatronic and Computer Science), 2025.

[9] M. A. Wicaksana, N. N. Farida, Santoso, and M. A. Rizza, "Perancangan Alat Safety Device untuk Mencegah Kerusakan Komponen Akibat Engine Overheat," Mars Jurnal Teknik Mesin, Industri, Elektro Dan Ilmu Komputer, vol. 3, no. 4, pp. 74–87, 2025.

[10] A. Hendrawan, M. Hasbi, and N. Rahman, "Simulasi Safety Device Overheat

Generator Set Engine Berbasis Arduino," Jurnal INTEKA: Informasi Teknik dan Niaga, vol. 22, no. 1, pp. 18–24, 2022.

[11] A. P. Mahendra, S. M. Herlambang, and P. Y. Yudianto, "Rancang bangun sistem monitoring kinerja main engine kapal berbasis Internet of Things (IoT)," Globe: Publikasi Ilmu Teknik, vol. 2, no. 4, pp. 110–118, 2024.

[12] C. Ahmadjayadi, "Sistem monitoring dan proteksi mesin berbasis mikrokontroler," Jurnal Teknik Mesin, vol. 5, no. 2, pp. 50–58, 2023.