GERMAN - MONGOLIAN INSTITUTE
FOR RESOURCES AND TECHNOLOGY

DESIGN AND IMPLEMENTATION OF A DATA ACQUISITION SYSTEM FOR MULTI FAULT DIAGNOSIS ANALYSIS IN INDUCTION MOTORS

Master Thesis

By

Tuvshin Gankhuyag

Master of Science (M.Sc.) in Resources and Technology

Nalaikh, Ulaanbaatar, Mongolia

2024 Spring

GERMAN - MONGOLIAN INSTITUTE
FOR RESOURCES AND TECHNOLOGY

The present master's research thesis was submitted to the Faculty of Mechanical and Electrical Engineering

DESIGN AND IMPLEMENTATION OF A DATA ACQUISITION SYSTEM FOR MULTI FAULT DIAGNOSIS ANALYSIS IN INDUCTION MOTORS

MASTER THESIS

Professional name:	Master of Science (M.Sc.) in Resources and Technology
Supervisor 1 / Examiner 1:	Professor Ariunbolor Purvee (Ph.D.)
Supervisor 2 / Examiner 2:	Turbat Shagdar (Ph.D.)

Submitted date May 20, 2024

STATUTORY DECLARATION

GANKHUYAG, TUVSHIN	M2120578
Last Name, First Name	Student ID Number

I hereby affirm in that I provided the submitted master research thesis

Design And Implementation Of A Data Acquisition System For Multi Fault Diagnosis Analysis In Induction Motors

I did not use any sources other than those stated. In case that the work is additionally submitted on a data medium, I declare that the written and the electronic form are completely identical. The work was not submitted in the same or similar form to any examination authority.

Erdenet, Ulaanbaatar,
20.05.24

Place, Date

Signature

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my supervisors for helping me complete my thesis with valuable advice and guidelines. First of all, I want to express my sincere gratitude to Prof. Ph.D. Ariunbolor Purvee for her outstanding support and encouragement. Without her guidance and supervision, this work would not have been possible.

I wish to express my thanks to my colleagues and engineers at Erdenet Mining Corporation, for providing me with important practical advice and a detailed explanation of the case study.

My grateful thanks go to my co-supervisor Turbat Shgdar (Ph.D.) for his support and instruction during my research thesis. With his guidance, I found what I am capable of doing. Also, I am thankful for his acceptance of being my supervisor.

I would like to thank all the professors of GMIT and GMIT's family, who shared their great knowledge and experience. This work is just a piece of combined all the lessons I learned.

Finally, I would like to express my deepest gratitude to my family and friends for their unconditional support and encouragement during my studies.

ABSTRACT

This Master's thesis focuses on the design of a data acquisition system for detecting motor faults early and implementing smart maintenance strategies. Understanding and diagnosing faults in induction motors is critical for maintaining the reliability and efficiency of industrial operations.

The thesis aims to design a data acquisition system for early detection of motor faults. By developing a novel data acquisition system and comparing its performance with existing systems, this research aims to provide valuable insights and practical solutions to enhance motor fault diagnosis, ultimately improving operational reliability and reducing downtime in industrial settings.

The study begins by introducing the main research topic, discussing its broader context and significance, and emphasizing the need for a reliable data acquisition system for multi-fault diagnosis in induction motors, particularly squirrel cage motors widely used across various industries.

A comprehensive study of key motor elements is provided, including their importance and operating principles. Existing methods for motor fault diagnosis and online condition monitoring are covered. The thesis introduces data acquisition systems and reviews various algorithms for their design, including sensors, signal processing techniques, and data analysis methods.

The research delves deeper into data acquisition systems, reviewing various algorithms for their design, the selection and placement of sensors, signal processing techniques, and data analysis methods. It investigates the development of an algorithm, flow chart, and block diagram for a data acquisition system to study motor fault data from different load motors in the mineral processing plant. The study details the development and design of the data acquisition system, analyses and compares the results obtained from different motor loads, and discusses essential aspects based on these results.

Furthermore, this research discusses developing and evaluating a newly designed data acquisition system, the "Motor Analyzer LEVEL V1.0." It compares it with existing systems in the industry, specifically the Baker EXP3000 Explorer Dynamic Motor Analyzer (99-EXP3000-CE). The Fast Fourier Transform (FFT) converts real-time data from the time domain into the frequency domain, facilitating detailed analysis.

The "Motor Analyzer LEVEL V1.0" was developed based on this research and was tested against the Baker EXP3000 Explorer. Voltage measurements in the time domain from both systems showed very similar results. However, the horizontal axis scales differed due to variations in sampling rates—1000 Hz for the Motor Analyzer LEVEL V1.0 and 10000 Hz for the Baker EXP3000 Explorer. The frequency domain analysis also demonstrated substantial similarity between the two systems, with a validation accuracy of nearly 98% for both current and voltage measurements in both time and frequency domains.

This research marks the first development of the Motor Analyzer LEVEL V1.0 in Mongolia, contributing significantly to advancements in local industrial technology. The thesis summarizes these findings, presents conclusions, and offers further recommendations for future research to enhance industrial operations' reliability and efficiency.

CONTENTS

TABLE

CHAPTER 1. RESEARCH BACKGROUND

1.1. Overview

A Data Acquisition System is a critical tool in modern industrial environments, providing detailed insights into the health and performance of induction motors, facilitating early fault detection, and supporting effective maintenance strategies.

Induction motors are critical components in various industrial applications due to their robustness, simplicity, and cost-effectiveness. However, multiple faults can compromise their performance, such as bearing defects, stator winding issues, rotor bar problems, and eccentricities. Early detection and diagnosis of these faults are essential to prevent severe damage, reduce downtime, and optimize maintenance schedules.

Designing and implementing a Data Acquisition System for multi-fault diagnosis in induction motors involves selecting appropriate sensors, developing reliable hardware and software systems, and employing sophisticated analysis techniques. This approach ensures early fault detection, thereby enhancing the operational efficiency and longevity of induction motors. Research on the design and implementation of data acquisition systems for multi-fault diagnosis analysis in induction motors is a well-explored field, given their critical role in industrial applications.

Studying the design and implementation of a DAS for multi-fault diagnosis analysis in induction motors is essential for enhancing industrial operations' reliability, efficiency, and safety. It brings significant economic benefits, drives technological advancements, and contributes to sustainable practices. Additionally, it advances academic research and addresses complex industry challenges, making it a crucial area of study in modern industrial applications.

Developing a data acquisition systems for multi-fault diagnosis in induction motors is a comprehensive process that involves careful planning, design, and implementation. By integrating advanced sensor technology, robust data acquisition hardware, sophisticated data processing algorithms, and user-friendly interfaces, such a system can significantly enhance induction motor operations' reliability, efficiency, and safety. Continuous advancements in machine learning, edge computing, and sensor technology promise to further improve the capabilities and effectiveness of DAS in industrial applications.

The existing body of research encompasses a variety of approaches, technologies, and methodologies aimed at enhancing fault diagnosis systems' reliability, accuracy, and efficiency.

1.2. Problem Description, Motivation, Research Objectives

AC motors are the primary energy consumers in industrial sectors worldwide. Squirrel cage induction motors constitute approximately 90% of all motors. Their applications span various domains, from household utilities to the demanding environments of petroleum extraction, natural gas production, and mining operations, both open-pit and underground, where spark protection is critical .

However, mechanical and electrical faults challenge the reliability of these motors, often triggered by inrush currents during startup, overloading, and the rigours of operational conditions. Detecting these faults early on is paramount to extending the motor's lifespan, enhancing equipment productivity, and preventing costly downtimes.

Yet, obtaining timely insights into a motor's condition proves challenging. This hinders prompt fault detection and leads to increased repair costs and unplanned downtimes. Implementing technical inspections and services capable of identifying motor damage during operation is thus imperative, promising improved equipment utilization and reduced maintenance expenses.

Developing a robust data acquisition system for multi-fault diagnosis in induction motors, particularly squirrel cage motors, becomes indispensable to address these critical needs. Such a system, capable of identifying motor faults during operation, is critical to minimizing unplanned downtimes, optimizing production schedules, and ultimately lowering operational costs.

The compact Data Acquisition System proposed by this research promises a solution to these challenges. This system offers a cost-effective and efficient alternative to existing solutions by focusing on essential functions tailored to multi-fault diagnosis. Its portability and compact design make it ideal for diverse industrial settings, ensuring timely fault detection and contributing to overall operational excellence.

Although various data acquisition systems are available for recording experimental current and voltage signatures, along with post-processing and analysis, including those with menu-driven interfaces, they come with advantages and limitations. While these systems excel at managing large datasets during measurements, they can be complex, expensive, and require training for effective use. Menu-driven interfaces offer convenience for standard calculations but may lack the flexibility needed for advanced analyses. Additionally, the operations performed behind menu selections may not always be transparent, leading to a lack of traceability in chosen analysis procedures. Despite significant research in developing and designing data acquisition systems for multi-fault diagnosis in induction motors, there has been no such research conducted in Mongolia.Therefore, it may be preferable to design data acquisition systems tailored to specific needs, such as embedded systems.

The development of such a system marks a significant step towards a more reliable and efficient industrial landscape. By addressing the pressing needs of fault

detection in induction motors, this research not only enhances equipment reliability but also paves the way for a more sustainable and cost-effective industrial future.

1.3. Object of the research study

Erdenet Mine significantly contributes to Mongolia's mining sector and economic landscape. Located in the northern region of Mongolia, specifically in the Orkhon Province, the mine plays a pivotal role in extracting copper and molybdenum ores. Established in the 1970s, it has since evolved into one of Asia's largest open-pit copper and molybdenum mines.

A unique aspect of Erdenet Mine lies in its ownership structure, being jointly owned by the Mongolian and Russian governments, with Mongolia holding a majority stake. This collaborative ownership underscores the strategic importance of the mine for both nations, as it serves as a key driver of economic growth and development.

Erdenet Mine provides significant employment opportunities, with thousands of workers directly and indirectly benefiting from its operations. Beyond employment, the mine's economic impact extends to the broader community, contributing substantially to government revenues and serving as a cornerstone of Mongolia's export sector.

However, operating Erdenet Mine is not without its challenges. Environmental concerns, including land degradation and water pollution, present ongoing challenges that require careful management and mitigation efforts. Efforts to balance economic interests with environmental sustainability are thus integral to the mine's long-term viability and success.

Despite these challenges, Erdenet Mine remains vital to Mongolia's mining industry and economy. Its substantial reserves and ongoing investments in infrastructure and technology position it as a critical player in the global mining landscape. Moving forward, responsible resource management and international collaboration will be essential to ensuring the continued success and sustainability of Erdenet Mine for generations to come.

The processing plant at Erdenet Mine relies on a diverse array of equipment and motors to facilitate its operations efficiently. These include mills, crushers, sieves, pumps, ventilators, compressors, and cranes. Mills grind the extracted ore into finer particles, while crushers reduce large chunks of ore into smaller pieces. Sieves or screens separate the ground ore into different size fractions. Pumps transport fluids, such as water and slurries, throughout the processing plant. Ventilators provide ventilation and air circulation within the plant, ensuring a safe working environment. Compressors compress air for various applications, including pneumatic tools and control systems. Cranes are essential for lifting and moving heavy equipment and materials within the processing plant.

In terms of motors, the processing plant primarily utilizes squirrel cage induction motors, which are known for their robustness and reliability. These motors are well-suited for heavy-duty operations and are commonly used in industrial applications.

Additionally, synchronous motors are employed when precise speed control and high efficiency are required. These motors often use variable frequency drives (VFDs) for optimal performance. High-power motors drive equipment such as crushers, mills, and pumps, while low-power motors are used for auxiliary functions such as driving fans, blowers, and smaller pumps.

The equipment and motors utilized in the processing plant are integral to the overall efficiency and productivity of the operation. Proper maintenance and optimization of these assets are crucial for maximizing throughput, minimizing downtime, and ensuring the safety of personnel and equipment. Additionally, advancements in motor technology, such as adopting energy-efficient motors and VFDs, can result in significant cost savings and environmental benefits over the long term.

Erdenet concentration plant, where asynchronous motors, notably squirrel cage induction motors, and synchronous motors power a diverse array of equipment. With over 3,000 electric motors driving crucial machinery, early fault detection is even more pressing to optimize motor performance, reduce maintenance costs, and enhance operational efficiency. These motors are the objective of the research study.

This research's primary contribution lies in pioneering the development of the Motor Analyzer LEVEL V1.0 in Mongolia, marking a significant advancement in local industrial technology. By introducing this innovative system, we address a critical gap in the region's industrial landscape, offering a tailored solution to enhance motor performance monitoring and maintenance practices. The Motor Analyzer LEVEL V1.0 empowers local industries with state-of-the-art diagnostic capabilities, enabling them to detect and address motor faults proactively, thereby minimizing downtime and maximizing operational efficiency.

1.4. Goal & objectives of the study

This study aims to design a data acquisition system for multi-fault diagnosis analysis in induction motors and compare its performance with other existing data acquisition systems in the industry. The specific objectives of this master's research are as follows:

- Firstly, the background and introduction will provide an overview of induction motors and their importance in various industries, explaining their principles of operation and common faults. This section will also review existing methods for fault diagnosis in induction motors, discuss the role of data acquisition systems in fault diagnosis, identify gaps and limitations in current research, and establish a theoretical foundation for multi-fault diagnosis.
- Next, the development of the data acquisition system design will be detailed, outlining the research approach and design, describing the architecture of the data acquisition system, and detailing the selection and placement of sensors. This section will also explain signal processing techniques, develop a multi-fault diagnosis algorithm, integrate sensor data and features, and specify the hardware and software components.

— Following this, the experimental results section will describe the test setup and outline data acquisition and preprocessing methods. It will compare the results with existing data, discuss the interpretation and analysis of the experimental results, evaluate the performance of the data acquisition system and diagnosis algorithm, and compare the results with other relevant research. Additionally, this section will identify challenges and limitations and suggest future research directions.

1.5. Outline

Understanding and diagnosing faults in induction motors is crucial for maintaining the reliability and efficiency of industrial operations. This thesis aims to bridge the gap between existing fault diagnosis methods and the need for more effective and efficient data acquisition systems tailored for multi-fault diagnosis in induction motors. By developing a new data acquisition system and comparing its performance with existing systems, this research seeks to provide valuable insights and practical solutions to enhance motor fault diagnosis and ultimately improve operational reliability and reduce downtime in industrial settings.

This thesis comprises five chapters:

- Chapter 1 introduces the main research topic and discusses the study's purpose and importance in a broader context, setting the stage for its importance and relevance.
- Chapter 2 comprehensively studies key elements related to motors, including their importance and operating principles. It covers existing methods for motor fault diagnosis and online condition monitoring.
- Chapter 3 delves deeper into data acquisition systems, reviewing various algorithms for their design, the selection and placement of sensors, signal processing techniques, and data analysis methods. This chapter also introduces data acquisition systems and reviews various design algorithms, including sensors, signal processing techniques, and data analysis method. This chapter builds on the foundational knowledge provided in Chapter 2.
- Chapter 4 investigates the development of an algorithm, flow chart, and block diagram for a data acquisition system to study motor fault data from different load motors in the mineral processing plant. It determines algorithm blocks based on load characteristics and then explains data acquisition approaches with defined constraints. The chapter then details the development and design of the data acquisition system, analysing and comparing the results obtained from different motor loads. Essential aspects based on the data acquisition system results are discussed.
- Chapter 5 discusses the research results based on the data acquisition system, comparing the results of the newly designed system with existing data acquisition systems in the industry. This chapter summarizes the research

findings, presents conclusions, and provides recommendations for future research.

1.6. Conclusion of the chapter

The conclusion drawn from the extensive discussion of data acquisition systems for multi-fault diagnosis in induction motors underscores their critical role in modern industrial environments. These systems offer invaluable insights into motor health and performance, enabling early fault detection and facilitating effective maintenance strategies. The research presented in this study addresses the pressing need for more efficient and reliable data acquisition systems tailored for multi-fault diagnosis in induction motors.

By introducing the Motor Analyzer LEVEL V1.0, a novel data acquisition system developed in Mongolia, this research contributes significantly to local industrial technology. The system's innovative design empowers industries with state-of-the-art diagnostic capabilities, enabling proactive fault detection and minimizing downtime. The comparison of the Motor Analyzer LEVEL V1.0 with existing systems, such as the Baker EXP3000 Explorer Dynamic Motor Analyzer, highlights its effectiveness in enhancing motor fault diagnosis.

The comprehensive analysis conducted in this study lays the groundwork for future advancements in motor fault diagnosis and maintenance practices. By bridging the gap between existing methods and the evolving needs of industrial operations, this research provides valuable insights and practical solutions for improving operational reliability and reducing downtime. The development of tailored data acquisition systems tailored to specific industrial needs promises to further enhance efficiency and reliability in industrial settings.

In conclusion, the research outlined in this thesis underscores the importance of continuous innovation in data acquisition systems for multi-fault diagnosis in induction motors. By addressing current limitations and challenges, such as those encountered in the Erdenet Mine, this study contributes to the advancement of industrial technology and fosters a more sustainable and cost-effective industrial future.

CHAPTER 2. MOTORS AND MOTOR FAULTS

2.1. Background knowledge of motors and online condition monitoring

There two main types of the motors. One is synchronous and another on is asynchronous motors. There are the four main motor of industries. is shown the motor type of industrial motors (Automation World n.d.).

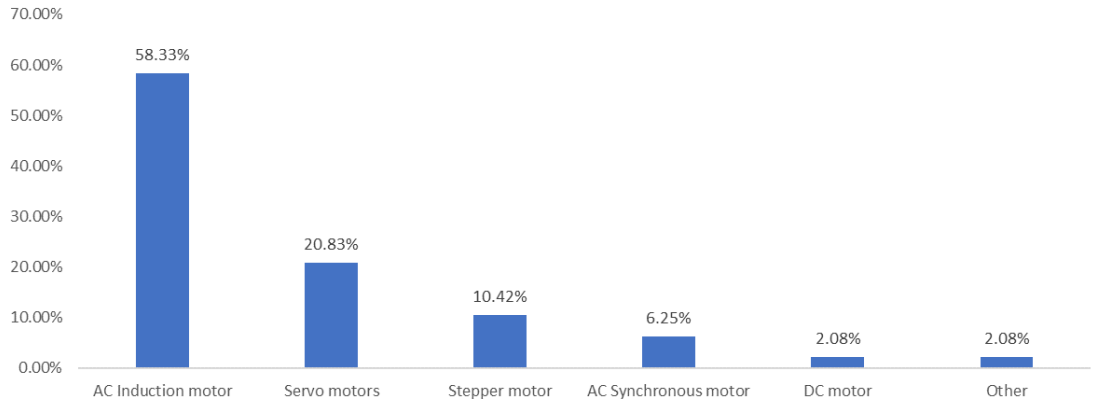

Figure 1: Motor type of industrial motors

Asynchronous motors are classified into two main types: squirrel-cage motors and wound-rotor motors. The squirrel-cage induction motor is simpler, more economical, and more rugged compared to the wound-rotor induction motor. Consequently, about 90% of industrial motors are induction motors, which are integral to the operation of pumps, conveyors, machine tools, centrifugal machines, presses, elevators, excavators, and packaging equipment. These motors are also essential in hazardous environments such as petrochemical and natural gas plants, and the mining industry, as well as in severe conditions like grain elevators, shredders, and coal plants.

According to this graph, the induction motor is the most widely utilized type of motor. Thus, it is extremely important from an economic standpoint to investigate the motor and identify its condition before there is damage .

The economic efficiency of industrial operations heavily depends on machine utilization, as indicated by machine available time. The condition of machines, particularly electrical equipment like motors, continuously changes due to the interaction of various operational factors. This ongoing change can lead to faults in the motors, deteriorating their operational parameters over time. One of the primary reasons for sudden motor breakdowns and unplanned maintenance is these faults. Unplanned maintenance increases the total maintenance cost of industrial operations significantly. Unforeseen breakdowns account for a considerable portion of time lost due to delays and downtime, which is a costly overhead. Therefore, it is crucial to implement maintenance strategies based on online condition monitoring to enhance machine utilization and reduce maintenance costs.

Condition monitoring involves observing machinery parameters to identify significant changes indicative of developing failures. According to a survey by the Electric Power Research Institute (EPRI) as cited in IEEE (1985), a wide range of motors experienced failures due to various reasons: 37% were caused by stator winding failures, 41% by bearing failures, 10% by rotor failures, and 12% by miscellaneous failures, including air-gap eccentricity (Ariunbolor Purvee 2013).

Various research studies have examined the use of different signals—such as vibration, motor current, and leakage flux—to detect electrical faults in motors.

Vibration signals are one of the oldest condition monitoring techniques, widely used to detect mechanical faults like bearing failures or mechanical imbalance. A piezo-electric transducer, which provides a voltage signal proportional to acceleration, is often used. This acceleration signal can be integrated to yield velocity or position.

The most common signal analysis technique in electrical monitoring is Motor Current Signature Analysis (MCSA). Research has shown a relationship between mechanical vibrations of a machine and the magnitude of the stator current components at corresponding harmonics. Increased mechanical vibrations lead to a rise in the magnitude of these stator current harmonic components, as the mechanical vibration modulates the air gap at specific frequencies. These frequency components then appear in the stator inductance and ultimately in the stator current. Therefore, MCSA can be used to detect rotor and bearing faults.

Axial magnetic leakage flux of an induction machine is measured using a circular search coil placed at the non-drive end of the machine, concentric with the shaft. The search coil generates an output voltage proportional to the rate of change of the axial leakage flux. This signal contains many of the same frequency components as the stator current and is particularly useful for estimating speed due to a strong component at the slip frequency.

Temperature sensors monitoring the bearings and stator windings have traditionally been used for condition monitoring. While these sensors provide useful indications of machine overheating, their fault diagnostic capability is limited. Increased operating temperatures can indicate several issues, such as ventilation problems, asymmetric voltage supply, stator winding turn-to-turn failure, severe rotor failure, and bearing lubrication problems.

By implementing comprehensive condition monitoring, including advanced data acquisition systems, it is possible to detect motor faults early, plan smart maintenance, and thus improve the reliability and efficiency of industrial operations.

2.2. Review for the studies faults of the induction motors

Healthy Motor Studies: For effective motor condition monitoring, it is essential to have a comprehensive understanding of healthy machine behaviours. This understanding forms the foundation for reliable diagnostic methods, enabling accurate fault identification by contrasting faulty behaviours with the known characteristics of a healthy machine (Purvee et al. n.d.).

The "d-q" model for dynamic simulation of induction machines assumes that both stator and rotor windings are sinusoidally distributed in space. A.R. Minoz introduced a detailed mathematical derivation of the squirrel-cage induction machine d–q model for studying rotor bar faults and air gap eccentricity. This model is based on coupled magnetic circuit theory and complex space-vector notation, considering the actual non-sinusoidal rotor bar distribution. This level of detail is critical for precisely simulating and understanding how faults develop and manifest in induction motors.

Toliyat developed a multiple-coupled circuit machine model and a method known as the "Winding Function Approach" to calculate mutual inductances.—researchers such as (Gojko M. Joksimovic 2000), and others have used this approach to study broken rotor bars, end-ring, and turn-to-turn faults. These studies provide insights into the intricate interactions within the motor that lead to fault conditions, enabling more effective monitoring and diagnosis.

Researchers like A. Arkkio, S. Kanerva, A. Belahcen, S.C. Chang, C.C.M. Cunha, M.J. DeBortoli, D.H. Hwang, F. Ishibashi, M.D. Negrea, and others have employed finite element methods for condition monitoring of induction motors. Finite element analysis offers a detailed view of the motor's physical and electromagnetic fields, allowing for precise detection and analysis of potential fault points (Antero Arkkio 2001).

P. Salminen studied the effects of vibration due to torque, while F. Ishibashi developed a two-mass, two-degrees-of-freedom model to investigate the natural frequency and resonance of the stator core with windings . V. Lieven proposed a method for computing the magnetic forces acting on the stator of a squirrel-cage induction motor and the resulting vibrations. These studies contribute to a deeper understanding of how physical and operational factors influence motor health.

Rotor Fault Studies: Rotor faults in squirrel cage motors, including broken rotor bars and end-ring faults, are significant issues that can lead to severe motor damage and operational inefficiencies. The large starting currents in induction motors, usually 5 to 8 times full load currents, occur when cooling is minimal, resulting in maximum thermal and mechanical stresses. Cracks in the rotor bar to end-ring joint are most common during long start-up times and frequent starts in heavy-duty cycles. Broken rotor bars and end-ring faults often develop or worsen due to pulsating loads, frequent starts, or fatigue from reaching the end of their normal life. Understanding these failure mechanisms is crucial for developing effective condition monitoring and maintenance strategies.

When a rotor bar breaks completely, arcing occurs across the break, damaging the rotor laminations around the faulted bar. Neighbouring bars carry increased current is subject to higher stresses, eventually leading to their failure. Broken bars may lift outwards due to centrifugal forces, potentially causing catastrophic damage to the stator windings. Rotor faults commonly cause torque pulsations, speed fluctuations, vibration, and changes in the frequency components of the supply current and magnetic fields. These detailed observations enable the development of specific diagnostic techniques to identify and mitigate rotor faults before they cause extensive damage.

C.K. Mechefske analysed vibrations in motors with broken rotor bars and high spectral response at motor running speed and/or second harmonic. Toliyat and Kokko used axial magnetic flux measurements to design condition monitoring diagnostics systems for motors with broken bars and endring faults. S. Poyhonen utilized a Support Vector Machine for condition monitoring of squirrel-cage motors with broken rotor bars and end ring faults. These advanced diagnostic methods leverage modern technologies to enhance the accuracy and reliability of fault detection.

Several researchers used finite element analysis to simulate induction machines with broken rotor bars and end rings, supporting discussions and simulations with experimental results. W.R. Finley, T. Bishop, N.M. Dhanesh, and A. Alireza observed that broken rotor bars cause vibrations at slip frequency times the number of poles (pole pass frequency) in the stator winding. These findings underscore the importance of detailed modelling and simulation in understanding the complex behaviours associated with rotor faults.

Stator Fault Studies: Stator faults can be broadly classified into two categories:

Laminations (core hot spots, core slackening) and frame (vibration, circulating currents, loss of coolant, earth faults). Stator Windings Defects/Faults: These are related to either the end winding portion (local insulation damage, insulation fretting, contamination, connector damage, insulation cracking, erosion, conductor displacement, turn-to-turn faults) or the slot portion (insulation fretting, conductor displacement).A. Siddique provided a comprehensive review of various stator faults, their causes, detection parameters/techniques, and the latest trends in condition monitoring technology. This review consolidates existing knowledge and highlights areas where further research is needed. M. Aderiano developed a new method for fault diagnosis of broken rotor bars, and inter-turn short-circuits using three-phase stator current envelopes with reconstructed phase space transforms. This innovative approach demonstrates the potential of advanced signal processing techniques in improving fault detection accuracy.

Researchers have studied stator faults using negative sequence supply current, computer-aided monitoring of the stator current Park’s Vector, and Wavelet signal processing methods. These diverse methodologies reflect the complexity of stator fault diagnostics and the need for multiple approaches to achieve robust monitoring solutions. E. Israel, L. Jarmo, and other researchers used Artificial Neural Networks

to detect faults in induction motors. G.M. Joksimovic developed a winding-function-based method for modelling polyphase cage induction motors with inter-turn short circuits in the stator winding. The application of artificial intelligence and advanced modelling techniques represents a significant advancement in motor condition monitoring.

Air Gap Eccentricity Studies: Machine eccentricity is defined as an asymmetric air gap between the stator and rotor, accounting for around 12% of induction motor faults. Due to manufacturing and assembly, an inherent level of static eccentricity exists even in new machines. Excessive eccentricity can lead to significant motor faults, so it is essential to study eccentricities in detail. Static air-gap eccentricity is caused by the fixed position of the minimal radial air-gap length, while dynamic eccentricity results from the centre of the rotor not aligning with the centre of rotation. Static and dynamic eccentricities can coexist and may lead to bent rotor shafts, bearing wear, and other faults. Understanding these types of eccentricity and their causes is critical for developing effective preventive maintenance strategies. Toliyat et al. presented a multiple-coupled circuit machine model and the "Winding Function Approach" for calculating mutual inductances, which Joksimovic et al. used to analyse static and dynamic eccentricity in induction motors. Al-Nuaim and Toliyat applied the "Modified Winding Function Approach" to analyse dynamic eccentricity in synchronous machines. Nandi et al. analysed the static, dynamic, and mixed eccentricity of induction motors. These advanced modelling techniques allow for precise simulations of eccentricity effects, facilitating early detection and intervention. G.M.Joksimovic analyzed the skew effect on inductances using equations developed for the machine with axial uniformity. G. Bossio proposed a method to calculate the inductances of induction motors with non-uniform air gaps, considering radial and axial asymmetries such as skew and eccentricity along the machine's axial length. These contributions help to address the complexities of real-world motor designs, improving the accuracy of fault diagnostics (Gojko M. Joksimovic 2000).

Bearing Fault Studies: According to an IEEE motor reliability study, bearing faults are the most frequent faults in induction machines, accounting for 41% of failures. Though relatively inexpensive, motor bearings cause significant downtime and lost production costs when they fail. Bearing faults are a major cause of eccentricities in induction motors. Fundamental defect frequencies depend on bearing geometry and shaft speed. Fault-free bearings create low vibrations, but faults related to ball diameter, pitch diameter, number of rolling elements, and the angle between rolling element surfaces and races show specific frequencies in vibration signatures. Hasan developed algorithms for estimating running speed and bearing defect frequencies using vibration data. Yang developed a coupled Recirculation-Backpropagation neural network to identify operational states of rotating machinery from spectral features, providing real-time fault identification capabilities. These innovative approaches highlight the potential of combining traditional vibration analysis with advanced computational techniques for more effective bearing fault diagnostics.

Mechanical Fault Studies: Unbalanced rotors are a common cause of machinery malfunction, leading to excessive vibration, accelerated wear, and reduced performance. Unbalance results from the centre of gravity of a rotating member not coinciding with the centre of rotation, causing a heavy spot on the rotor. High vibration amplitude at the 1x frequency in spectral analysis indicates unbalance, which researchers have widely discussed. Summarizing the above research, rotor faults and eccentricities have been studied using stator current analysis, while bearing faults have been studied using vibration analysis in laboratory conditions. Some researchers consider the simulation of bearing faults challenging. M. Blodt developed a new fault model considering bearing fault-related air-gap length variations and load torque changes. Vibration is a primary indicator of machine health, and vibration monitoring has proven effective for overseeing machine health over the years. Implementing advanced vibration analysis techniques and integrating them with modern data acquisition and computational methods can significantly enhance the early detection of motor faults, leading to improved reliability and reduced maintenance costs.

2.3. Conclusion of the chapter

Condition monitoring of asynchronous motors, particularly squirrel-cage induction motors, is essential for maintaining industrial efficiency and reliability. These motors, critical in various applications and harsh environments, are prone to faults that cause sudden breakdowns and increased maintenance costs. Implementing online condition monitoring helps enhance machine utilization and reduce overall maintenance expenses.

Faults in motors, such as stator winding failures, bearing failures, rotor failures, and air gap eccentricity, are significant causes of unplanned maintenance. Techniques like vibration analysis, Motor Current Signature Analysis (MCSA), axial magnetic leakage flux measurement, and temperature monitoring are employed to detect these faults.

Advanced models, such as the "d-q" model and the "Winding Function Approach," aid in understanding motor behavior and developing fault detection techniques. These models enable precise simulations for early fault detection and intervention.

Rotor and stator faults, along with air gap eccentricity and bearing issues, require comprehensive monitoring. Techniques such as finite element analysis, advanced signal processing, artificial intelligence, and detailed modeling are crucial for accurate diagnostics.

In summary, advanced condition monitoring techniques significantly improve the reliability and efficiency of industrial operations by enabling early fault detection and proactive maintenance strategies, reducing unplanned downtime and maintenance costs.

CHAPTER 3. DATA ACQUISITION SYSTEMS AND CONDITION MONITORING

3.1. Evaluation and selection of methods for online condition monitoring

Online motor condition monitoring refers to the continuous and real-time monitoring of various parameters and performance metrics of electric motors while they are in operation. This monitoring is typically conducted using sensors and data acquisition systems that collect data on factors such as temperature, vibration, current, voltage, and other relevant variables.

The primary goal of online condition monitoring is to detect any deviations from normal operating conditions that may indicate potential motor faults or failures. By continuously analysing the collected data, engineers and maintenance personnel can identify early signs of issues such as bearing wear, rotor faults, insulation degradation, imbalance, misalignment, and other mechanical or electrical problems.

Early detection of these issues allows for proactive maintenance and repair interventions, minimizing unplanned downtime and preventing costly equipment failures. Additionally, online condition monitoring enables predictive maintenance strategies, where maintenance activities are scheduled based on the actual condition of the motor rather than predetermined time intervals.

Overall, online motor condition monitoring provides valuable insights into the health and performance of electric motors, helping to optimize their reliability, efficiency, and lifespan while reducing operational costs and minimizing disruptions to industrial processes.

Evaluating and studying the current methods and techniques used in condition monitoring is essential for defining the aims and objectives of research work. By synthesizing and summarizing previous research findings, a functional condition monitoring scheme can be developed, laying the groundwork for innovative approaches to enhancing machine health and performance.

The output signals from accelerometers, commonly used in vibration measurement, manifest as electrical signals in the time domain, which may exhibit periodic or non-periodic behaviour. While these signals can be visualized on a cathode ray oscilloscope, such displays often fail to provide actionable insights due to the combined representation of signals at multiple frequencies. To extract meaningful information, it is crucial to decipher the frequency composition of the signal by isolating individual frequencies.

Determining the frequency content of vibrations represents a critical step in the measurement process. By identifying and analysing individual frequencies, researchers can gain valuable insights into the origin and characteristics of vibrations, enabling targeted interventions to address underlying issues and optimize machine performance.

Understanding the frequency composition of vibrations is akin to unlocking a hidden language that machines use to communicate their health status. Engineers and

researchers can pinpoint potential faults or anomalies by deciphering this language, allowing for proactive maintenance and optimization strategies. This minimizes downtime and maintenance costs, maximizes operational efficiency, and extends the lifespan of critical machinery. Emphasizing the importance of frequency analysis underscores its role as a foundational pillar in condition monitoring, driving continuous innovation and improvement in machine health management practices (АРИУНБОЛОР 2009).

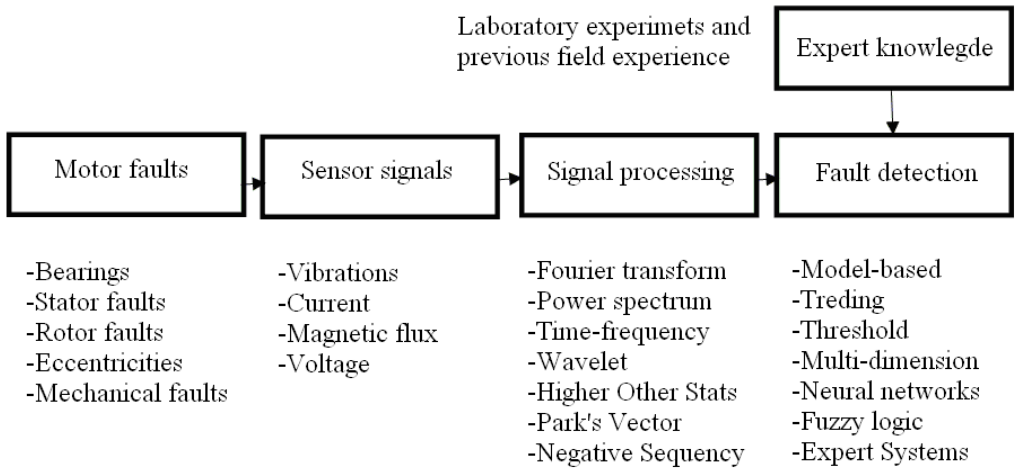

Figure 2. Block diagram of condition monitoring

Numerous signal processing methods have been employed in studies focusing on motor condition monitoring, as illustrated in Figure 2. Fast Fourier transforms, and Power Spectrum Density is commonly utilized in vibration monitoring, while Park's vector and negative sequence methods are prevalent in current tracking. Power Spectrum Density, in particular, stands out for its ability to filter out low-amplitude noise, enhancing the clarity of peaks in spectrum analysis. In this thesis, Fast Fourier transforms are exclusively employed, making them the primary spectrum analysis technique.

However, it's crucial to acknowledge the role of expertise and experience in motor condition monitoring. Experts are tasked with making final decisions, conducting analyses, and drawing conclusions using the methods outlined in Figure 2.

3.2. Signal conditioning and processing for online condition monitoring

Signal conditioning and processing play pivotal roles in online condition monitoring systems, enabling the conversion of raw sensor data into actionable insights for fault detection and diagnosis. Signal conditioning involves preprocessing raw sensor signals to enhance their quality and suitability for analysis (Yuan et al., 2014). This process typically includes amplification, filtering, and analog-to-digital conversion to ensure compatibility with digital processing techniques (Sala et al., 2015). For instance, amplification increases the signal-to-noise ratio, improving the accuracy of subsequent analysis, while filtering removes unwanted noise and artifacts from the sensor data (Garcia-Perez et al., 2020).

Signal processing, on the other hand, involves the application of algorithms and techniques to extract meaningful information from the conditioned sensor signals (Yuan et al., 2014). This may include time-domain analysis, frequency-domain analysis, or advanced signal processing methods such as wavelet transform or machine learning algorithms (Garcia-Perez et al., 2020). Time-domain analysis focuses on analyzing signals in the time dimension, identifying temporal patterns and anomalies indicative of faults (Sala et al., 2015). Frequency-domain analysis, enabled by techniques like Fast Fourier Transform (FFT), decomposes signals into their frequency components, revealing characteristic signatures associated with specific fault conditions (Yuan et al., 2014).

The integration of signal conditioning and processing techniques in online condition monitoring systems enables real-time monitoring of equipment health and performance, facilitating early fault detection and predictive maintenance strategies (Garcia-Perez et al., 2020). By continuously analyzing sensor data, these systems can identify deviations from normal operating conditions, allowing maintenance teams to intervene before faults escalate into costly failures (Sala et al., 2015).

Signal conditioning and processing are essential components of online condition monitoring systems, enabling the conversion of raw sensor data into actionable insights for fault detection and diagnosis. Through techniques such as amplification, filtering, and advanced signal processing algorithms, these systems empower industries to maintain equipment reliability, optimize maintenance schedules, and minimize downtime.

Some signal conditioning and processing methods have been introduced below:

The Fast Fourier Transform (FFT): FFT is a crucial algorithm for converting real-time data from the time domain into the frequency domain. This transformation facilitates the analysis of frequency components within a signal, which is essential in fields such as signal processing, communications, and data analysis (Cooley & Tukey, 1965). In the time domain, signals are represented as functions of time, making it challenging to identify specific frequencies. The FFT efficiently computes the Discrete Fourier Transform (DFT) of a sequence, decomposing the time-domain signal into its constituent frequencies and providing a frequency spectrum that illustrates the presence of various frequencies (Brigham, 1988).

The FFT algorithm reduces computational complexity by dividing the signal into smaller segments, performing the Fourier transform on these segments, and combining the results (Oppenheim & Schafer, 1975). This efficiency makes FFT suitable for real-time applications requiring rapid processing. Utilizing FFT, real-time data can be transformed into the frequency domain to identify dominant frequencies, detect periodic patterns, and analyze signal behavior (Smith, 1997). This capability is particularly useful in audio signal processing, vibration analysis, and mechanical fault detection. By examining the frequency spectrum, engineers and scientists can uncover characteristics of the data that are not easily observable in the time domain.

Overall, FFT is a powerful tool for transforming and analyzing real-time data, providing detailed insights into the frequency content within a signal, and enabling numerous practical applications in science and engineering.

Power spectrum analysis: Power spectrum analysis is a fundamental technique used in online condition monitoring systems to analyze the frequency content of signals acquired from sensors mounted on machinery. It involves the decomposition of a time-domain signal into its frequency components, providing valuable insights into the machinery's operational condition (Li et al., 2019). By employing techniques such as the Fast Fourier Transform (FFT), power spectrum analysis reveals the distribution of signal power across different frequency bands, highlighting characteristic frequencies associated with specific fault modes (Majhi et al., 2018). For example, the presence of peaks or anomalies in the power spectrum at certain frequencies may indicate the occurrence of mechanical faults such as bearing defects or gear damage (Wang et al., 2019).

Power spectrum analysis enables the detection of subtle changes in machinery vibration or acoustic signals, which are often early indicators of impending failures (Majhi et al., 2018). By continuously monitoring the power spectrum of sensor data in real-time, online condition monitoring systems can detect abnormalities and deviations from normal operating conditions, allowing maintenance teams to take timely corrective actions (Li et al., 2019). Moreover, power spectrum analysis facilitates the development of condition-based maintenance strategies by providing quantitative metrics for assessing machinery health and degradation levels (Wang et al., 2019).

In summary, power spectrum analysis is a powerful tool in online condition monitoring, offering insights into machinery health by analyzing the frequency content of sensor signals. Through techniques like FFT, it identifies characteristic frequencies associated with specific fault modes, enabling early fault detection and condition-based maintenance strategies.

Wavelet analys: Wavelet analys is is a powerful technique used in online condition monitoring systems to analyze signals acquired from sensors mounted on machinery. It involves decomposing a time-domain signal into different frequency bands at varying resolutions, allowing for the detection of localized features and transient events (Li et al., 2019). By employing wavelet transforms, online monitoring systems can capture both low-frequency trends and high-frequency oscillations in sensor data, providing detailed insights into machinery health and performance (Antoni & Randall, 2006).

The wavelet transform offers several advantages for online condition monitoring, including the ability to localize both time and frequency information within a signal (Li et al., 2019). This localization property enables the detection of transient faults and incipient failures, which may not be observable in the frequency domain using techniques like Fourier analysis (Antoni & Randall, 2006). Additionally, wavelet analysis facilitates feature extraction and pattern recognition, making it suitable for fault diagnosis and classification tasks (Liu et al., 2016).

In practical applications, wavelet analysis is often combined with machine learning algorithms to develop advanced diagnostic systems for machinery health monitoring (Liu et al., 2016). By extracting relevant features from wavelet-transformed signals and training classification models, these systems can automatically detect, diagnose, and predict impending failures in real-time (Li et al., 2019).

In summary, wavelet analysis is a versatile tool for online condition monitoring, offering localized time-frequency analysis capabilities that enable the detection of transient faults and incipient failures. Its integration with machine learning techniques enhances fault diagnosis and prognostics, contributing to improved machinery reliability and operational efficiency.

Higher-order statistics (HOS): This analysis is a method utilized in online condition monitoring to extract valuable information from signals acquired from sensors installed on machinery. Unlike traditional methods such as Fourier analysis, which focus on linear relationships, HOS examines non-linear properties within the data, providing insights into the system's dynamic behavior (Antoni & Randall, 2006). By analyzing higher-order moments such as skewness and kurtosis, HOS techniques can detect deviations from Gaussian distributions, which often indicate the presence of faults or anomalies (Li et al., 2019).

This approach offers several advantages for online condition monitoring, including the ability to capture non-linear effects and characterize complex signal dynamics associated with machinery health and performance (Antoni & Randall, 2006). HOS analysis is particularly effective for detecting incipient faults and early signs of degradation, allowing for proactive maintenance interventions to prevent costly downtime and failures (Li et al., 2019).

In practical applications, HOS analysis is often integrated into advanced monitoring systems alongside other signal processing techniques and machine learning algorithms (Antoni & Randall, 2006). By combining HOS features with classification models, these systems can automatically identify and diagnose various fault conditions, providing actionable insights for maintenance personnel (Li et al., 2019).

In summary, higher-order statistics analysis is a valuable tool for online condition monitoring, offering the capability to capture non-linear signal characteristics and detect subtle changes indicative of machinery faults or degradation. Its integration with machine learning approaches enhances fault diagnosis and prognostics, contributing to improved reliability and operational efficiency in industrial settings.

Park's Vector approach: This approach is a powerful method used in online condition monitoring to analyze three-phase electrical signals from rotating machinery. It involves transforming the signals from the time domain to the synchronous reference frame, where stationary and dynamic components can be separated (Wang et al., 2020). By decomposing the signals into orthogonal components, Park's Vector facilitates the detection of faults such as rotor bar defects and unbalanced loads (Tirale et al., 2017).

This technique offers several advantages for online condition monitoring, including the ability to accurately represent the rotating machinery's behavior and detect incipient faults at early stages (Wang et al., 2020). Park's Vector analysis enables engineers to identify specific fault signatures within the electrical signals, providing insights into the machine's health and performance (Tirale et al., 2017).

In practical applications, Park's Vector is often integrated into condition monitoring systems alongside other signal processing techniques and machine learning algorithms. By combining Park's Vector features with classification models, these systems can automatically identify and diagnose various fault conditions, enhancing maintenance decision-making (Wang et al., 2020).

In summary, Park's Vector analysis is a valuable tool for online condition monitoring of rotating machinery, allowing for accurate fault detection and diagnosis based on electrical signal characteristics. Its integration with machine learning approaches improves the effectiveness of condition monitoring systems, contributing to enhanced reliability and operational efficiency in industrial settings.

The Fast Fourier Transform (FFT) is utilized in this study to convert real-time domain data into frequency domain data.

3.3. Approaches of motor online condition monitoring

Online condition monitoring involves continuously surveilling machinery and equipment during their operation to detect faults and assess performance in real time. This proactive approach helps prevent unexpected failures and optimize maintenance schedules. Several approaches are used in online condition monitoring:

- Vibration Analysis,
- Thermography,
- Oil Analysis,
- Current Signature Analysis,
- Acoustic Emission Monitoring,
- Ultrasonic Testing,
- Condition Monitoring with IoT (Internet of Things),
- Machine Learning and AI-Based Approaches,
- Fiber Optic Sensing.

Vibration Analysis: Vibration analysis is widely used in online condition monitoring. Sensors like accelerometers are attached to critical machinery components to measure vibration levels. The collected data is analyzed to identify patterns and anomalies that indicate mechanical issues like imbalance, misalignment, bearing faults, or gear defects. Fast Fourier Transform (FFT) and Power Spectral Density (PSD) are signal processing methods used in vibration analysis.

Thermography: Thermography involves using infrared cameras to monitor the temperature of machine components. It helps identify hotspots that may indicate overheating, which can be a sign of electrical faults, friction, or inadequate lubrication. Continuous monitoring of temperature variations helps in the early detection of potential issues, ensuring timely intervention.

Oil Analysis: Oil analysis monitors the condition of lubricants in machinery. By analyzing the physical and chemical properties of oil samples, including viscosity, contamination, and the presence of wear particles, it's possible to assess the machinery's health. Online oil sensors can provide real-time data on oil quality, facilitating prompt maintenance actions.

Current Signature Analysis: Current signature analysis involves monitoring the electrical parameters of motors and generators, such as current, voltage, and power. This approach helps detect electrical faults like insulation breakdown, rotor bar issues, and phase imbalances. Techniques such as Motor Current Signature Analysis (MCSA) and Park's vector approach are often used.

Acoustic Emission Monitoring: Acoustic emission monitoring captures high-frequency sound waves emitted by machinery components under stress. Sensors detect these emissions, and the data is analyzed to identify cracks, leaks, and other structural defects. This approach efficiently monitors pressure vessels, pipelines, and rotating machinery.

Ultrasonic Testing: Ultrasonic testing uses high-frequency sound waves to detect surface and subsurface defects in materials. Ultrasonic sensors can provide real-time data on components' thickness, density, and structural integrity, making them a valuable tool for detecting corrosion, cracks, and other anomalies.

Condition Monitoring with IoT (Internet of Things): Integrating IoT technologies in condition monitoring systems allow for collecting, transmitting, and analyzing vast amounts of data from multiple sensors in real time. IoT-enabled devices can communicate wirelessly, providing continuous monitoring and enabling predictive maintenance through advanced data analytics and machine learning algorithms.

Machine Learning and AI-Based Approaches: Machine learning and artificial intelligence (AI) significantly enhance online condition monitoring. By using historical data to train models, these algorithms can accurately predict future failures and anomalies. Techniques such as neural networks, support vector machines, and deep learning are commonly used to analyze complex datasets and identify patterns indicative of potential faults.

Fiber Optic Sensing: Fiber optic sensors are used to monitor various parameters such as temperature, strain, and pressure. They offer high sensitivity and can be embedded within structures for real-time monitoring. This approach is beneficial in harsh environments and for long-term health monitoring of critical infrastructure.

Wearable and Remote Monitoring Systems: Wearable sensors and remote monitoring systems allow for continuous tracking of equipment and operator conditions. These systems can provide real-time feedback on the operational status and health of machinery and alert operators to potential hazards or needed maintenance activities.

Each of these approaches offers unique benefits and can be selected based on the specific requirements of the machinery, operational environment, and desired outcomes of the condition monitoring program. Combining multiple approaches often results in a more robust and comprehensive monitoring system.

Current Signature Analysis is used in research studies because MCSA offers a noninvasive, cost-effective, and comprehensive approach to motor condition monitoring with real-time capabilities. It complements other methods and can be integrated into a multifaceted monitoring system to enhance reliability and efficiency.

3.4. Algorithms of data acquisition systems

Designing algorithms for data acquisition systems involves a nuanced understanding of various methodologies to ensure efficient and accurate data collection, processing, and analysis. Here's a comprehensive review of key design algorithms:

- Sampling Algorithms,
- Filtering Algorithms,
- Signal Conditioning Algorithms,
- Data Compression Algorithms,
- Feature Extraction Algorithms,
- Machine Learning Algorithms,
- Data Fusion Algorithms,
- Real-Time Processing Algorithms,
- Fault Detection and Diagnostics Algorithms.

Sampling Algorithms: These determine how data is collected from sensors over time. Techniques include periodic sampling (at fixed intervals) and event-based sampling (triggered by specific events). The choice depends on data nature, required frequency, and power constraints. Sampling algorithms play a crucial role in data acquisition systems by determining the frequency at which data is collected from sensors. Periodic sampling, one of the fundamental methods, ensures that data is collected at regular, fixed intervals, providing a consistent and uniform dataset over time. This method is particularly useful for applications where continuous monitoring is required, such as in environmental sensing or industrial process control. Event-based sampling, on the other hand, activates the data collection process only when specific conditions or events occur, which helps in conserving power and reducing unnecessary data collection. This approach is beneficial in applications where changes are infrequent or where power efficiency is a priority, such as in battery-operated devices and remote monitoring systems. Both sampling methods aim to balance the need for timely and relevant data collection with considerations of resource utilization and operational efficiency.

Filtering Algorithms: Essential for noise reduction and signal enhancement, filters include Finite Impulse Response (FIR) and Infinite Impulse Response (IIR). FIR provides linear-phase response and stability, while IIR offers efficient implementation. Filtering algorithms are essential in data acquisition systems for removing noise and unwanted components from collected data, thereby enhancing the quality and accuracy of the signals being analyzed. Common filtering algorithms include low-pass, high-pass, band-pass, and band-stop filters, each designed to allow specific frequency ranges to pass through while attenuating others. Low-pass filters permit signals below a certain cutoff frequency to pass, effectively removing high-frequency noise. High-pass filters, conversely, allow signals above a certain frequency to pass, eliminating low-frequency noise. Band-pass filters enable signals within a specific frequency range to pass, useful for isolating particular frequency

components of interest, while band-stop filters do the opposite by rejecting a specific frequency band, thus removing unwanted interference within that range. Advanced filtering techniques, such as adaptive filtering, dynamically adjust filter parameters based on real-time signal characteristics, further improving the accuracy of data interpretation by continuously optimizing the noise reduction process.

Signal Conditioning Algorithms: These preprocess raw sensor data to make it suitable for analysis. Techniques involve amplification, attenuation, and offsetting to match sensor ranges and eliminate bias.

Signal conditioning algorithms are crucial in data acquisition systems for preparing raw sensor data for accurate digitization and analysis. These algorithms include amplification, where weak signals are boosted to a level suitable for further processing; attenuation, which reduces the amplitude of signals that are too strong and could potentially saturate the data acquisition system; and isolation, which protects both the data acquisition hardware and the sensors from potential damage due to high voltage or current spikes. Additionally, signal conditioning often involves linearization algorithms that convert non-linear sensor outputs into a linear form, making the data easier to interpret and analyze. Temperature compensation is another important aspect of signal conditioning, adjusting the signal to account for variations in sensor readings due to temperature changes. These algorithms ensure that the signals entering the analog-to-digital converter (ADC) are within the optimal range, accurately representing the measured phenomena and minimizing errors in the subsequent digital processing stages.

Data Compression Algorithms: Reduce data volume without sacrificing critical information. Techniques include run-length encoding, delta encoding, and lossy compression like JPEG for images. Data compression algorithms are essential in data acquisition systems for reducing the size of data files, which conserves storage space and bandwidth without significantly compromising the integrity of the information. These algorithms can be lossless, where the original data can be perfectly reconstructed from the compressed data, or lossy, where some amount of data is lost, but the loss is usually imperceptible or acceptable for the intended application. Lossless compression algorithms, such as Huffman coding, Lempel-Ziv-Welch (LZW) compression, and run-length encoding (RLE), are often used when it is critical to retain the original data, such as in medical imaging or scientific measurements. On the other hand, lossy compression algorithms, such as those used in JPEG and MP3 formats, significantly reduce data size by discarding less critical information, making them suitable for applications like multimedia streaming and storage where some data loss is tolerable. The choice of compression algorithm depends on the specific requirements of the data acquisition system, balancing the need for data fidelity, compression ratio, and processing speed to ensure efficient and reliable data management.

Feature Extraction Algorithms: Identify relevant patterns or features from raw data for analysis. Methods range from statistical analysis (mean, variance) to advanced techniques like wavelet transforms or Principal Component Analysis (PCA). Feature extraction algorithms are crucial components of data acquisition systems, tasked

with identifying and extracting relevant information or features from raw data to facilitate subsequent analysis and interpretation. These algorithms aim to transform complex, high-dimensional data into a reduced set of meaningful features that capture essential characteristics or patterns. Various techniques are employed for feature extraction, including statistical methods like principal component analysis (PCA), which identifies the most significant dimensions of variation in the data, and linear discriminant analysis (LDA), which maximizes the separability between different classes or categories in the data. Additionally, signal processing techniques such as wavelet transforms and Fourier analysis can extract frequency or time-domain features from signals, revealing underlying patterns or structures. Machine learning approaches, such as deep learning and convolutional neural networks (CNNs), can automatically learn and extract features directly from the data, enabling more complex feature representations. The choice of feature extraction algorithm depends on the nature of the data, the specific objectives of the analysis, and the computational resources available, with the goal of capturing relevant information while minimizing redundancy and computational complexity.

Machine Learning Algorithms: Utilized for predictive analytics and anomaly detection. Examples include Support Vector Machines (SVM), Random Forests, and Deep Learning models like Convolutional Neural Networks (CNN) or Recurrent Neural Networks (RNN). Machine learning algorithms are computational models that enable systems to automatically learn and improve from experience without being explicitly programmed, leveraging patterns and relationships within data to make predictions or decisions. These algorithms encompass a broad range of techniques, including supervised learning, where models are trained on labeled data to predict or classify new instances based on learned patterns; unsupervised learning, which involves discovering hidden structures or patterns in unlabeled data, such as clustering similar data points together; and reinforcement learning, where agents learn to interact with an environment by receiving feedback in the form of rewards or penalties, aiming to maximize cumulative rewards over time. Within these categories, various algorithms exist, such as decision trees, random forests, support vector machines (SVM), k-nearest neighbors (KNN), neural networks, and deep learning models like convolutional neural networks (CNN) and recurrent neural networks (RNN). These algorithms play a crucial role in extracting insights, detecting patterns, and making predictions from complex datasets across diverse domains, including image recognition, natural language processing, and predictive maintenance in industrial settings.

Data Fusion Algorithms: Combine information from multiple sensors to enhance accuracy and reliability. Techniques include Kalman filtering for state estimation and Bayesian networks for probabilistic reasoning. Data fusion algorithms integrate information from multiple sources or sensors to provide a comprehensive and accurate representation of the underlying system or phenomenon. These algorithms combine data from diverse sensors or data streams, often of different types or modalities, to enhance the overall understanding or inference about the target system. By integrating data from various sources, such as sensors measuring

different aspects of a system or data collected at different times or locations, data fusion algorithms aim to overcome limitations or uncertainties associated with individual sources of data. Common techniques in data fusion include fusion at the sensor level, where raw sensor measurements are combined, fusion at the feature level, where extracted features from multiple sources are merged, and fusion at the decision level, where decisions or outputs from individual sources are combined. Data fusion algorithms are widely used in applications such as target tracking, surveillance, environmental monitoring, and healthcare, where combining information from multiple sources can improve the accuracy, reliability, and robustness of the overall system.

Real-Time Processing Algorithms: Execute computations on-the-fly to enable immediate decision-making. These algorithms prioriize low latency and may include Fast Fourier Transforms (FFT), Kalman filters, or recursive estimators. Real-time processing algorithms are designed to handle data and perform computations with minimal delay, ensuring that results are produced within a specified time frame or deadline. These algorithms are optimized for processing data as it arrives, often in streaming or continuous fashion, without significant buffering or latency. They are essential for applications where timely responses or decisions are critical, such as in control systems, telecommunications, and signal processing. Real-time processing algorithms typically prioritize efficiency and speed, utilizing techniques like parallel processing, optimized data structures, and predictive modeling to meet stringent time constraints. By processing data in real-time, these algorithms enable immediate feedback, response, or action based on the incoming data, facilitating rapid decision-making and enhancing system responsiveness.

Fault Detection and Diagnostics Algorithms: Identify anomalies or malfunctions in the system. Techniques range from rule-based systems to model-based approaches like fault tree analysis or Bayesian networks.

Each algorithm type serves a specific purpose within the data acquisition process, ensuring robust performance and meaningful insights from collected data. The choice depends on system requirements, data characteristics, and computational resources available. Fault detection and diagnostics algorithms are essential components of condition monitoring systems, enabling the identification of anomalies or malfunctions in machinery. These algorithms leverage sensor data, such as vibration patterns, temperature readings, or electrical signals, to detect deviations from normal operating conditions. They employ various statistical, mathematical, and machine learning techniques to differentiate between typical and atypical behavior, facilitating the early detection of potential faults or failures. Threshold-based methods are commonly used in fault detection algorithms, wherein predefined thresholds or criteria are established to signal deviations from expected performance, thereby initiating diagnostic processes to pinpoint the root cause of the issue.

Signal conditioning, real-time processing, fault detection and diagnostics algorithms are used research study.

3.5. Conclusion of the chapter

In Chapter 3, we thoroughly examined the essential components of data acquisition systems and their role in online condition monitoring of electric motors. Online condition monitoring facilitates continuous, real-time assessment of motor performance through the collection and analysis of parameters such as temperature, vibration, current, and voltage. This proactive approach aids in the early detection of potential faults, enabling timely maintenance and minimizing unplanned downtime. Various monitoring techniques, including vibration analysis, thermography, oil analysis, and current signature analysis, offer unique benefits for identifying mechanical and electrical issues. Evaluating and synthesizing these methods underscore the importance of selecting appropriate monitoring techniques to optimize motor health and performance.

The Fast Fourier Transform (FFT) is utilized in this study to convert real-time domain data into frequency domain data, playing a pivotal role in analyzing vibration data by filtering noise and highlighting significant frequencies. Expertise and experience are crucial in interpreting these analyses to make informed maintenance decisions. The chapter also delves into the design of algorithms necessary for data acquisition systems, covering aspects such as sampling, filtering, signal conditioning, data compression, feature extraction, machine learning, data fusion, real-time processing, and fault detection and diagnostics. These algorithms ensure efficient, accurate data collection, processing, and analysis, forming the backbone of effective condition monitoring systems.

Current Signature Analysis (CSA) is emphasized in research studies due to its noninvasive, cost-effective, and comprehensive approach to motor condition monitoring with real-time capabilities. It complements other methods and can be integrated into multifaceted monitoring systems to enhance reliability and efficiency.

Signal conditioning, real-time processing, and fault detection and diagnostics algorithms are crucial in research studies to ensure robust and effective monitoring systems.

CHAPTER 4. DESIGNING AND DEVELOPING DATA ACQUISITION SYSTEM

4.1. Development hardware

The data acquisition system developed by the research was named "Motor Analyzer LEVEL V1.0" data acquisition system.

The hardware of the Motor Analyzer LEVEL V1.0 data acquisition system consists of several key components that work together to collect, process, and store data from various sensors. The central component is the STM32F407VET microcontroller, which acts as the main processing unit, interfacing with and managing the data from connected sensors. These sensors include a voltage sensor for measuring electrical voltage, a current sensor for monitoring current flow, a temperature sensor for detecting ambient temperature, a humidity sensor for measuring moisture levels, and an accelerometer for capturing motion data.

Supporting components include a real-time clock (RTC) that provides accurate time-stamping for the data collected, ensuring that each data point is recorded with precise timing information. A microSD card is used for data storage, allowing the system to log large amounts of data for later retrieval and analysis. A USB port facilitates data transfer to other devices or for system programming and updates.

The system is powered by a 4800 mAh battery, providing a reliable power source that enables the system to operate independently of external power. The battery ensures that the data acquisition system can function in various environments, including remote or portable applications. All these components are interconnected through appropriate wiring and communication interfaces such as I2C, SPI, or analog connections, forming a cohesive and functional data acquisition system that can effectively monitor and log various types of environmental and operational data.

The block diagram for designing a data acquisition system to detect motor faults at an early stage and plan smart maintenance is shown Figure 3.

The block diagram of the data acquisition system includes the following sensors:

- Voltage sensor,
- Current sensor,
- Temperature sensor,
- Ambition and humidity sensor,
- Accelerometer.

The data acquisition system also includes the following elements:

- Real time clock,
- USB port,
- STM32F407VET microcontroller,
- Micro SD card, and
- 4800 mA battery.

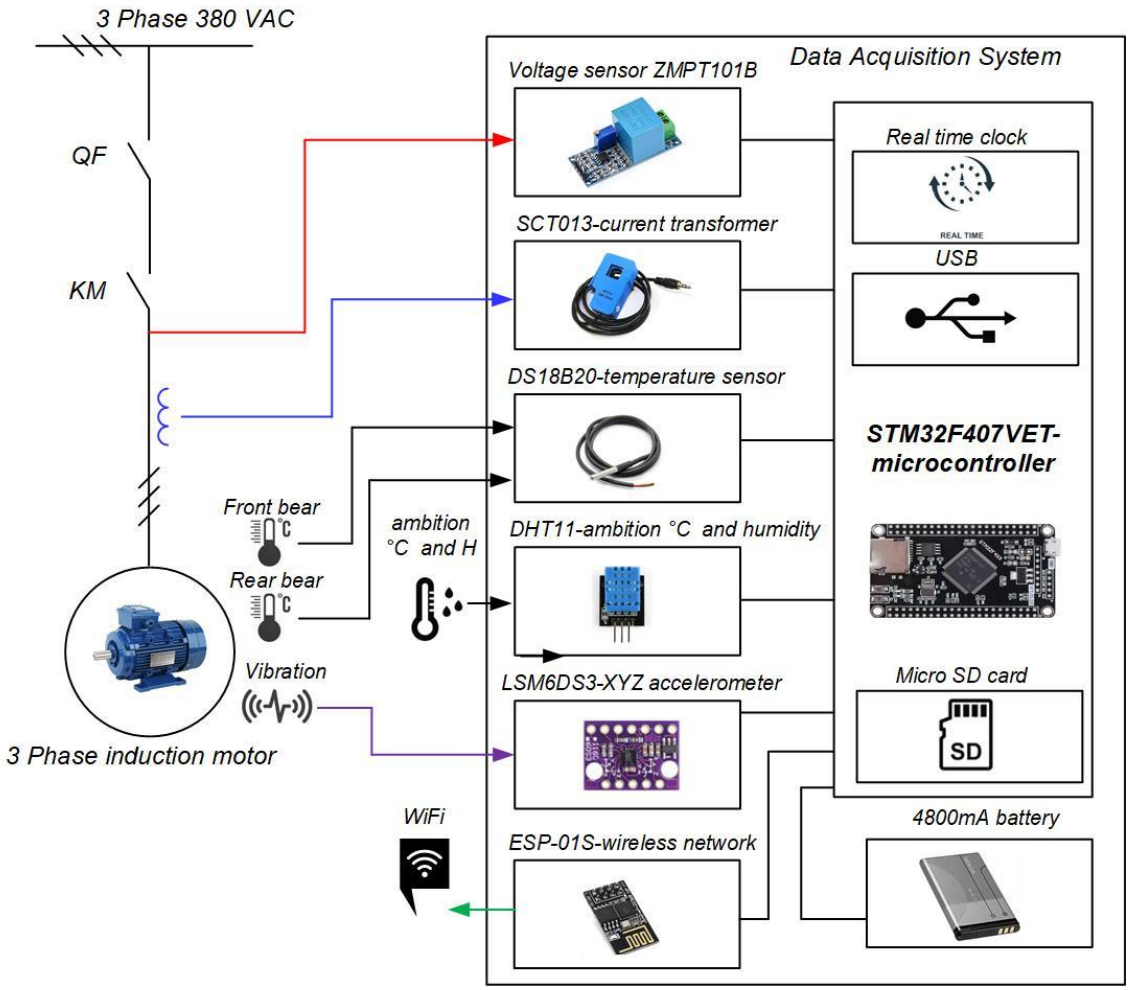

Figure 3. The block diagram of designing Motor Analyzer LEVEL V1.0 data acquisition system

Microcontroller: The data acquisition system is designed and developed based on STM32F407VET microcontroller shown in Figure *4*.

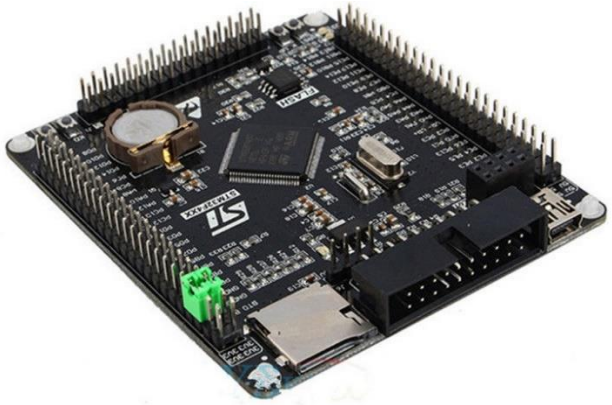

Figure 4. STM32F407VET microcontroller(github.com n.d.)

The STM32F407VET microcontroller is part of the STM32F4 series, developed by STMicroelectronics, and is based on the ARM Cortex-M4 32-bit RISC core. It operates at a frequency of up to 168 MHz, providing high performance and efficiency for various applications. This microcontroller includes 512 KB of Flash memory and

192 KB of SRAM, offering ample storage and memory for complex programs and data.

One of the key features of the STM32F407VET is its extensive range of peripherals and interfaces. It includes up to 17 timers, three 12-bit ADCs, two DACs, and numerous communication interfaces such as USART, SPI, I2C, and CAN. Additionally, it supports USB 2.0 full-speed and high-speed, and Ethernet MAC interfaces, making it suitable for communication-intensive applications. The microcontroller also includes advanced control capabilities with a floating-point unit (FPU) and a digital signal processing (DSP) unit, enhancing its ability to handle complex mathematical computations and signal processing tasks.

The STM32F407VET is designed with power efficiency in mind, featuring multiple low-power modes to optimize energy consumption. It supports a wide operating voltage range from 1.8V to 3.6V, making it adaptable to various power supply conditions. Its operating temperature ranges from -40°C to 85°C, suitable for industrial and consumer applications.

The microcontroller is housed in a 100-pin LQFP package, offering a compact form factor with extensive I/O capabilities. It includes advanced safety and security features, such as a CRC calculation unit, a hardware random number generator, and support for secure firmware updates, enhancing the reliability and security of applications.

Overall, the STM32F407VET microcontroller is a versatile and powerful component, ideal for applications requiring high performance, extensive peripherals, and robust communication capabilities. Its advanced features and power efficiency make it suitable for a wide range of applications, from industrial automation and motor control to consumer electronics and IoT devices.

Real time clock: A real-time clock (RTC) shown in Figure 5 is a specialized integrated circuit (IC) used to keep track of the current time and date. It maintains accurate timekeeping even when the main system power is turned off, typically powered by a small backup battery or a supercapacitor. The RTC operates independently of the main system's microcontroller and continues to count time accurately in low-power modes.

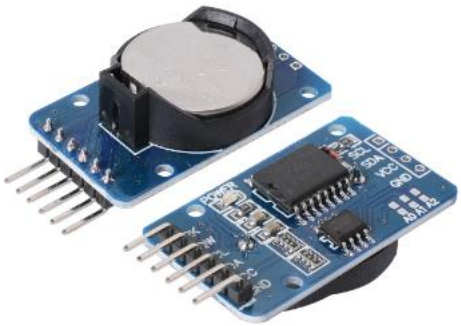

Figure 5. A real-time clock (RTC)

RTCs are essential in applications where accurate timekeeping is critical, such as in computers, embedded systems, and consumer electronics. They provide time information in hours, minutes, and seconds, as well as date information including day, month, and year, often accounting for leap years and different month lengths. This time and date information can be accessed by the system's microcontroller or processor through standard communication interfaces like I2C, SPI, or even simple parallel interfaces.

One common example of an RTC is the DS3231, which is a highly accurate RTC with an integrated temperature-compensated crystal oscillator (TCXO). It provides timekeeping accuracy within ±2 minutes per year and operates over a wide voltage range from 2.3V to 5.5V. The DS3231 includes features such as two programmable time-of-day alarms and a programmable square-wave output.

Another example is the PCF8523, a low-power RTC with a battery backup switch-over circuit. It operates with a supply voltage from 1.8V to 5.5V and provides a timekeeping accuracy of ±5 ppm at room temperature. The PCF8523 supports clock and calendar functions, an alarm, and a timer with an interrupt function.

RTCs are widely used in applications such as time-stamping data logs, scheduling tasks, and triggering events at specific times. They are critical in systems that need to maintain accurate time over long periods, such as servers, network devices, and security systems. The integration of an RTC in a system ensures that timekeeping functions continue uninterrupted and accurate, contributing to the overall reliability and functionality of the system.

USB port: A USB port, short for Universal Serial Bus port, serves as a standardized interface for connecting various peripherals and devices to a host computer or digital system. It facilitates both data transfer and power supply, making it a versatile and widely used connector. USB ports come in different types, including Type-A, Type-B, Type-C, and micro-USB, each with unique shapes and sizes to accommodate different devices. They support plug-and-play functionality, enabling devices to be connected and recognized without the need for manual configuration. USB ports have evolved through multiple versions, with each iteration offering faster data transfer rates and increased power delivery capabilities. They play a crucial role in modern computing and consumer electronics, serving as the primary interface for connecting peripherals such as keyboards, mice, printers, storage devices, and smartphones to host systems.

Micro SD card: A microSD card is a small, portable storage device commonly used in electronic devices such as smartphones, tablets, digital cameras, and portable gaming consoles. It is a type of Secure Digital (SD) card, but smaller in size, making it ideal for compact devices where space is limited. MicroSD cards are available in different storage capacities, ranging from a few gigabytes to several terabytes, allowing users to expand the storage capacity of their devices. They use flash memory technology, which enables fast read and write speeds, making them suitable for storing large files such as photos, videos, music, and apps. MicroSD cards are often used to store media files, documents, and applications, providing users with additional storage space and flexibility in managing their data. They are compatible with a wide range of devices that support the microSD format and can be easily inserted and removed from devices for data transfer or storage expansion. Overall, microSD cards are an essential accessory for many electronic devices, offering convenient and portable storage solutions for users' digital content.

4800 mA battery: A 4800 mAh battery is a type of rechargeable battery commonly used in portable electronic devices such as smartphones, tablets, and portable chargers. The "mAh" stands for milliampere-hour, which indicates the capacity of the battery to store electrical charge. In this case, a 4800 mAh battery can deliver a current of 4800 milliamperes for one hour before needing to be recharged. The higher the milliampere-hour rating, the longer the battery can typically last between charges. 4800 mAh batteries provide a balance between capacity and size, offering a decent amount of power while still being relatively compact and lightweight. They are often used in devices that require extended battery life or higher power

consumption, such as smartphones with large screens and high-performance processors. Overall, 4800 mAh batteries provide a reliable power source for portable devices, allowing users to stay connected and productive throughout the day without constantly needing to recharge.

Voltage sensor: The voltage sensor used is the ZMPT101B shown in Figure 6, a popular module that provides an easy interface for microcontrollers, with specifications including an input voltage range of AC 0-250V and a high-precision analog output signal. Another example is the INA219, a high-side current and voltage sensor with an I2C interface, capable of measuring voltage ranges from 0-26V and current ranges up to ±3.2A, providing a digital output.

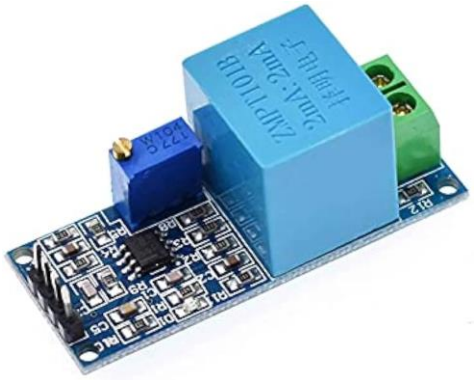

Figure 6. Voltage sensor ZMPT101B

A voltage sensor is a device used to measure and monitor voltage levels in an electrical circuit, converting the voltage into a readable signal for a microcontroller or other data acquisition systems. These sensors are crucial in various applications such as motor fault detection, power supply monitoring, and battery management systems. Voltage sensors are characterized by their high accuracy, wide range of measurable voltages, and the ability to provide either analog or digital output signals. They often include isolation features to protect the measurement system from high voltage levels and have fast response times suitable for dynamic systems.

There are several types of voltage sensors, including analog voltage sensors that provide a continuous output signal proportional to the input voltage, digital voltage sensors that provide discrete outputs often through interfaces like I2C or SPI, and Hall effect voltage sensors that measure voltage indirectly by detecting magnetic fields generated by current. Common applications of voltage sensors include monitoring voltage levels to detect motor faults, ensuring stable voltage levels in power supplies, managing battery charge and discharge cycles in battery management systems, and maintaining proper voltage levels in industrial automation systems.

Integrating voltage sensors with data acquisition systems involves connecting them to microcontrollers such as the STM32F407VET through analog inputs or digital interfaces. These sensors require calibration to ensure accuracy, especially in critical applications, and the data they generate can be logged to storage devices like a Micro SD card for further analysis and monitoring. Voltage sensors play a significant role in systems requiring precise and reliable voltage measurements, essential for predictive maintenance and fault detection.

Current Transformer: The current transformer is used SCT013 model. The SCT013 Current Transformer shown in Figure 7 is a type of sensor used to measure alternating current (AC) in electrical circuits.

Figure 7. SCT013 Current Transformer

It is commonly used in energy monitoring systems, power meters, and renewable energy applications. The SCT013 operates based on the principle of electromagnetic induction, where the current flowing through a conductor induces a magnetic field around it. The transformer's core captures this magnetic field and produces a proportional output voltage or current, which can be measured by a connected device such as a microcontroller or data acquisition system. This allows for non-invasive current measurement, meaning the SCT013 can be easily clamped onto a wire without interrupting the circuit. The SCT013 comes in different models with varying current ratings and output types to suit different applications. Overall, the SCT013 Current Transformer provides a convenient and accurate method for measuring AC currents in electrical systems, making it a valuable tool for energy monitoring and management.

Temperature Sensor: The Temperature Sensor us used DS18B20 model shown in Figure 8 is a digital temperature sensor manufactured by Maxim Integrated.

Figure 8. DS18B20 Temperature Sensor

It is capable of measuring temperatures with a high degree of accuracy and precision. One of its notable features is that it operates as a digital sensor, utilizing the 1-Wire communication protocol, which allows multiple DS18B20 sensors to be connected to a single microcontroller pin. The DS18B20 provides temperature readings in digital format, making it easy to interface with various microcontrollers and digital systems. Additionally, it has a wide temperature measurement range and can be calibrated for specific applications. The sensor is commonly used in applications such as environmental monitoring, HVAC systems, industrial automation, and consumer electronics, where accurate temperature sensing is essential. Overall, the DS18B20 Temperature Sensor offers a reliable and versatile solution for temperature measurement in a wide range of applications.

DHT11 Ambition temperature and humidity: Ambition temperature and humidity uses DHT11 model shown in Figure 9 and Humidity Sensor is a low-cost digital sensor commonly used for measuring temperature and humidity levels in the surrounding environment.

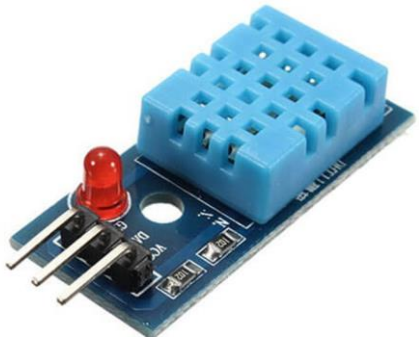

Figure 9. DHT11 Ambition temperature and humidity

It consists of a capacitive humidity sensor and a thermistor for temperature sensing, all housed in a compact package. The sensor communicates with a microcontroller through a single-wire digital interface, making it easy to integrate into various projects. The DHT11 provides accurate readings of temperature within a range of 0°C to 50°C with a precision of ±2°C and humidity within a range of 20% to 80% with a precision of ±5%. Its simplicity, affordability, and ease of use make it popular for applications such as weather stations, home automation systems, and greenhouse monitoring. Overall, the DHT11 Ambient Temperature and Humidity Sensor offers a cost-effective solution for measuring ambient conditions in a wide range of settings.

Accelerometer: The LSM6DS3-XYZ accelerometer uses LSM6DS3-XYZ model shown in Figure 10 is a highly versatile sensor capable of measuring acceleration along three axes: X, Y, and Z. It is manufactured by STMicroelectronics and is commonly used in various applications such as motion sensing, inertial navigation systems, and gesture recognition.

Figure 10. LSM6DS3-XYZ accelerometer

The sensor combines a 3-axis accelerometer and a 3-axis gyroscope in a single package, providing accurate motion detection and orientation tracking capabilities. With its compact size and low power consumption, the LSM6DS3-XYZ accelerometer is suitable for integration into portable devices, wearable technology, and Internet of Things (IoT) devices. It communicates with a microcontroller or other digital system through standard communication interfaces such as I2C or SPI, enabling easy integration into electronic projects. Overall, the LSM6DS3-XYZ accelerometer offers precise and reliable motion sensing capabilities, making it an essential component in various motion-related applications.

Wireless network: The wireless network module is used ESP-01S module shown in Figure *11* is a compact and versatile device developed by Espressif Systems, based on the ESP8266 chipset. It is designed for Internet of Things (IoT) applications and offers Wi-Fi connectivity in a small form factor.

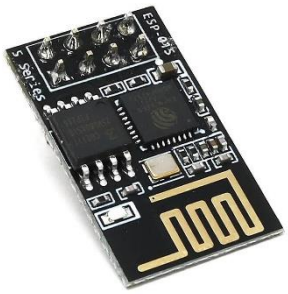

Figure 11. The ESP-01S wireless

The ESP-01S module provides a convenient and cost-effective solution for adding wireless networking capabilities to electronic projects, allowing devices to connect to local Wi-Fi networks and communicate with other devices or servers over the internet. It features a built-in TCP/IP protocol stack, making it easy to establish connections and transfer data over Wi-Fi networks. The ESP-01S module is commonly used in applications such as home automation, smart devices, remote monitoring, and IoT prototypes. It can be programmed using the Arduino IDE or other development environments, allowing users to customize its functionality according to their specific requirements. Overall, the ESP-01S wireless network module offers a reliable and accessible means of adding wireless connectivity to IoT projects, enabling seamless communication and control over Wi-Fi networks.

The schematic of the data acquisition system outlines the connections and interactions between various components used to collect and process data from sensors. It typically includes multiple sensors such as voltage sensors, current sensors, temperature sensors, humidity sensors, and accelerometers, each connected to a central microcontroller. In this system, the microcontroller, such as the STM32F407VET, serves as the core processing unit that receives analog or digital signals from the sensors. Additional components like a real-time clock (RTC) ensure accurate time-stamping of the collected data, while a microSD card provides storage for logging the data. A USB port may be included for data transfer or system programming. Power is supplied by a battery, such as a 4800 mAh battery, ensuring the system operates independently of an external power source. The schematic diagram visually represents how these components are wired together, highlighting power connections, data lines, and communication interfaces, thus providing a comprehensive overview of the system's architecture and functionality.

The schematic of the data acquisition system is shown in Figure 12.

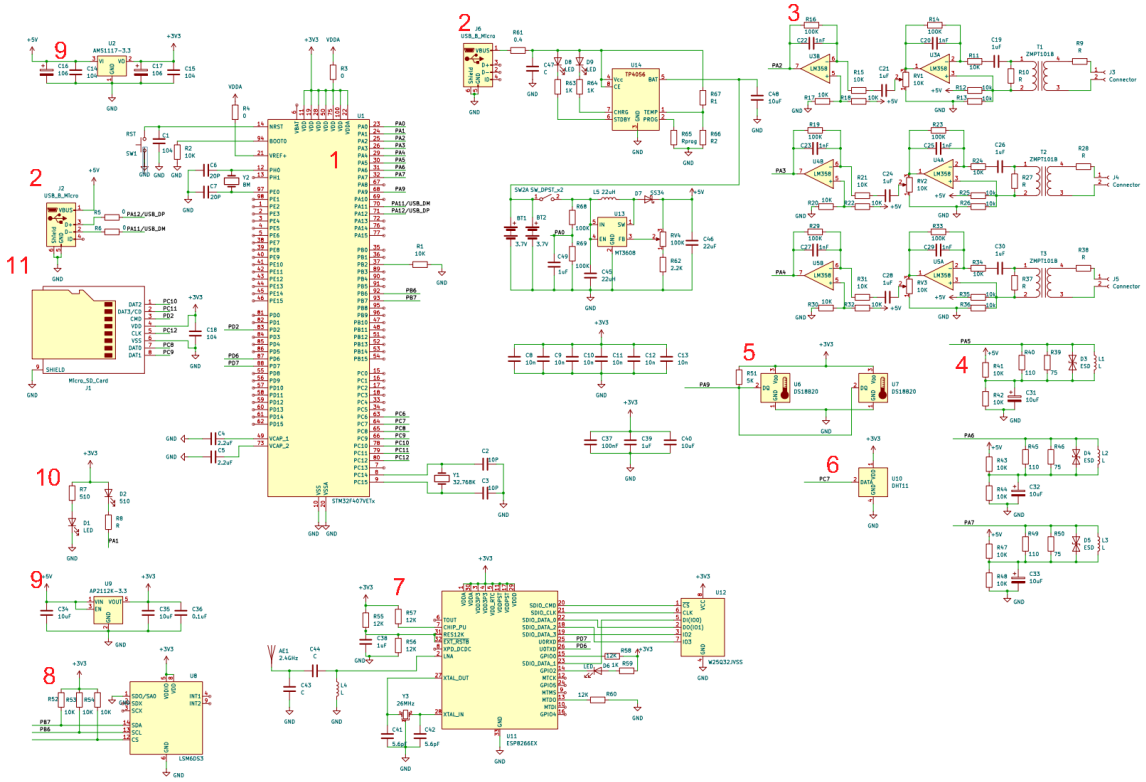

Figure 12. Schematic of Motor Analyzer LEVEL V1.0 data acquisition system data acquisition system

Motor Analyzer LEVEL V1.0 data acquisition system have the following peripherals of 1-11(in red) in Figure 12.

1 is the schematic of STM32F407VET microcontroller and shown in Figure *13*.

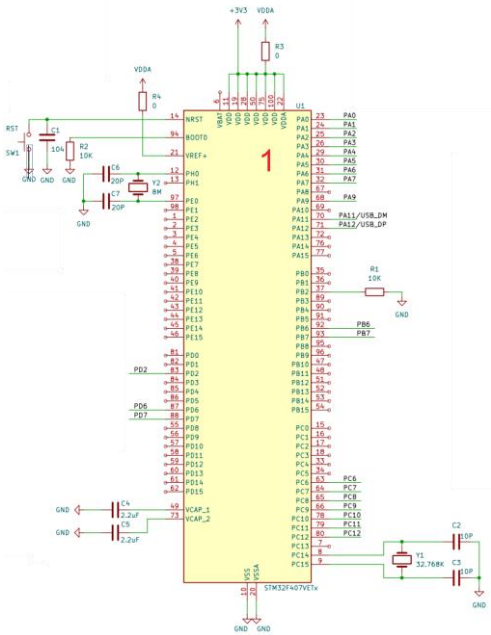

Figure 13. Schematic of STM32F407VET microcontroller

2 is the schematic of USB port, charger of li-ion battery (U13) and converter (U13) and shown in Figure 14.

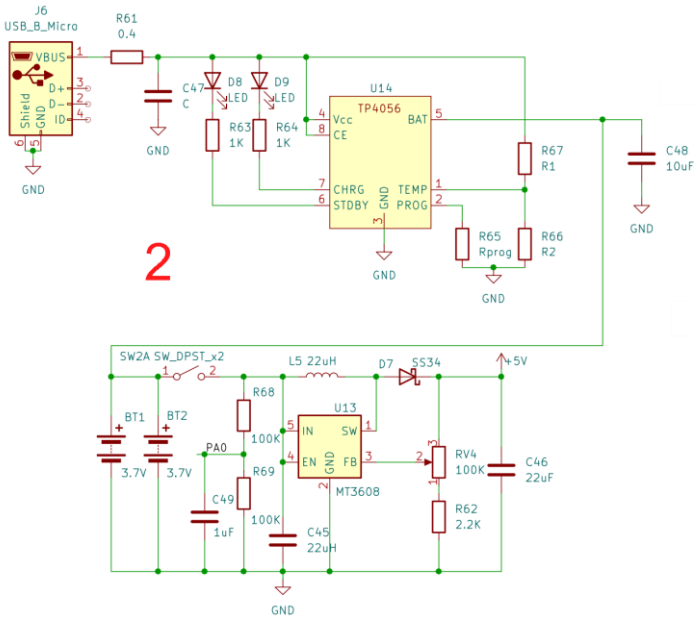

Figure 14. Schematic of USB port, charger of li-ion battery and converter

3 is the schematic of stepdown transformer to reduce voltage from 380V shown in Figure 15.

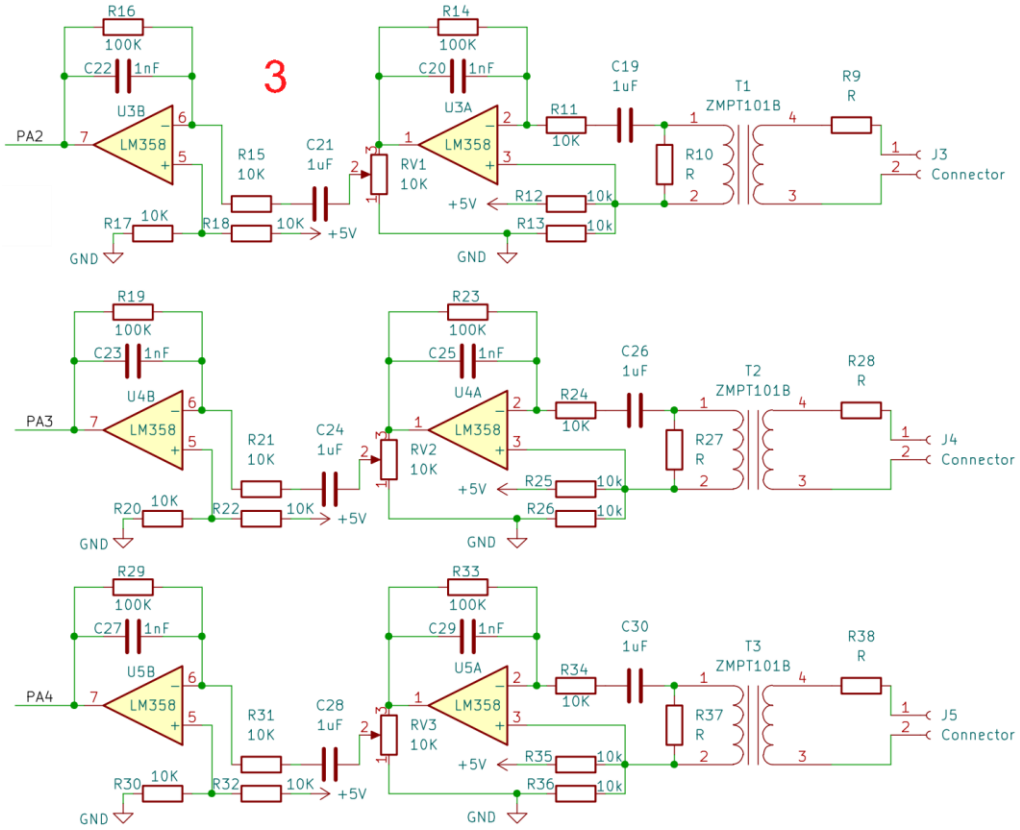

Figure 15. Schematic of stepdown transformer

4 is the schematic of current transformer to measure current shown in Figure 16.

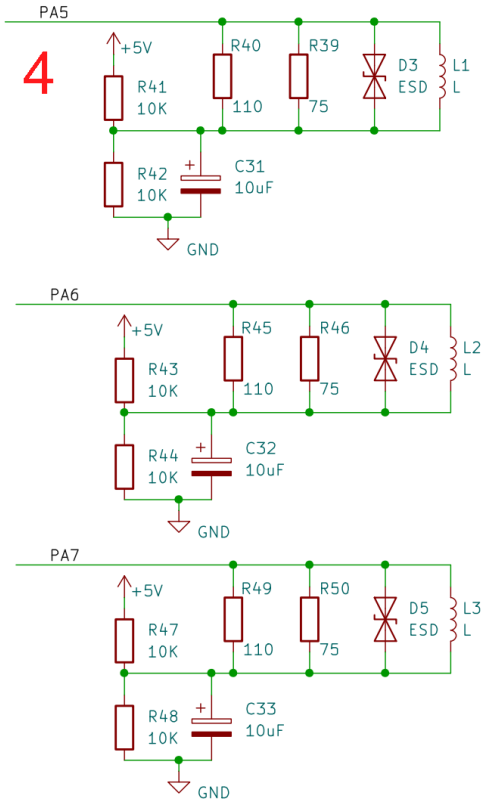

Figure 16. Schematic of stepdown transformer

5 is the schematic of current transformer to measure current shown in Figure 17.

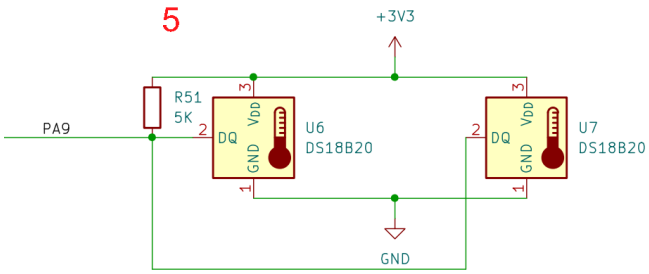

Figure 17. Schematic of current transformer

6 is the schematic of ambition and humidity sensor shown in Figure 18.

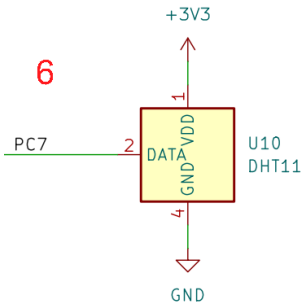

Figure 18. Schematic of ambition and humidity sensor

7 is the schematic of wireless network shown in Figure 19.

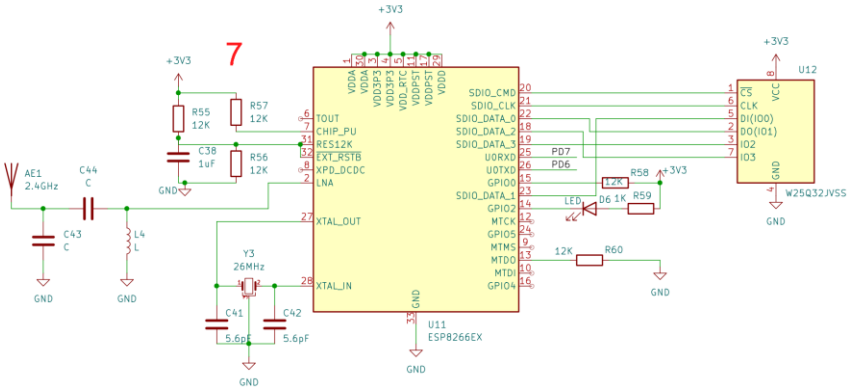

Figure 19. Schematic of wireless network

8 is the schematic of accelerometer shown in Figure 20Figure *19*.

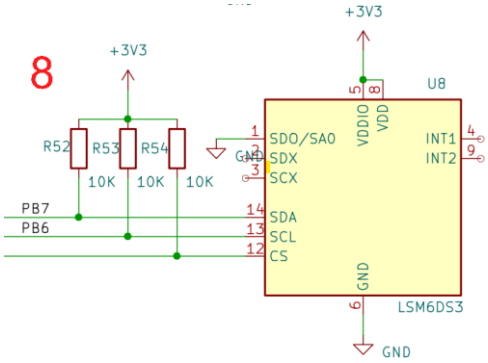

Figure 20. Schematic of accelerometer

11 is the schematic of the mini-SD card shown in Figure 21Figure *19*.

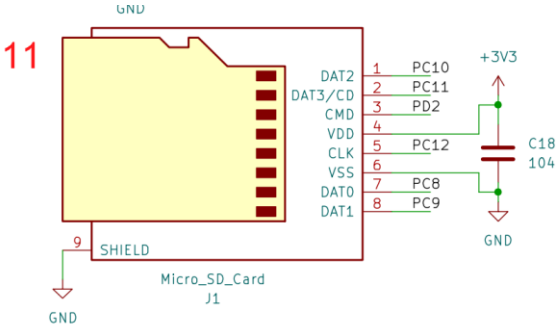

Figure 21. Schematic of mini-SD card

11 is the schematic of the USB port shown in Figure 22Figure *21*Figure *19*.

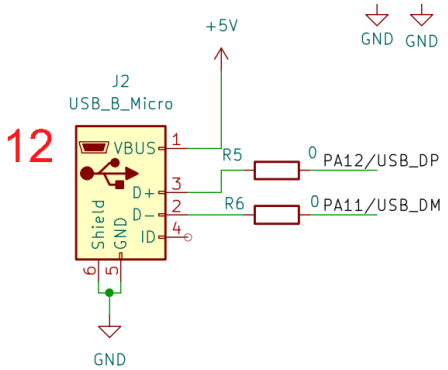

Figure 22. Schematic of the USB port

The "Motor Analyzer LEVEL V1.0" data acquisition system's industrial measurement capabilities are designed to ensure precise monitoring and analysis of various parameters critical to industrial operations. This system is adept at measuring electrical and environmental parameters such as voltage, current, temperature, humidity, and acceleration. It is an invaluable tool for predictive maintenance and fault detection in industrial settings.

The ZMPT101B voltage sensor achieves the voltage measurement, which accurately captures AC voltage levels. This is crucial for monitoring the electrical supply to motors and other machinery, ensuring they operate within safe and efficient voltage ranges. The current measurement is handled by the SCT013 current transformer, which provides non-invasive monitoring of AC current, essential for assessing the load and performance of electrical equipment.

The DS18B20 temperature sensor and the DHT11 humidity sensor monitor temperature and humidity, respectively. These sensors help maintain optimal environmental conditions for industrial equipment, prevent overheating and moisture damage, and ensure safe operating conditions.

The LSM6DS3-XYZ accelerometer measures acceleration along three axes, providing detailed data on machinery's motion and vibration. This is particularly useful for detecting imbalances or unusual vibrations indicating mechanical faults or wear.

All these sensors are integrated with the STM32F407VET microcontroller, which processes the data in real time and logs it for further analysis. The system's real-time clock (RTC) ensures that all data is accurately time-stamped, vital for tracking performance trends and diagnosing issues over time.

Data is stored on a microSD card, allowing large amounts of data to be recorded and retrieved for detailed analysis. The USB port facilitates easy data transfer to other devices for further processing or reporting. The ESP-01S wireless network module also provides Wi-Fi connectivity, enabling remote monitoring and data access.

The system is powered by a 4800 mAh battery, ensuring it can operate independently in various industrial environments, including remote or mobile applications. This independence from external power sources enhances its versatility and reliability.

In summary, the "Motor Analyzer LEVEL V1.0" data acquisition system offers comprehensive industrial measurement capabilities, combining accurate sensor data, robust processing, and reliable data storage and communication, making it an effective tool for maintaining and optimizing industrial equipment performance.

4.2. Software development

The software flowchart of Motor Analyzer LEVEL V1.0 data acquisition system is shown in Figure 23.

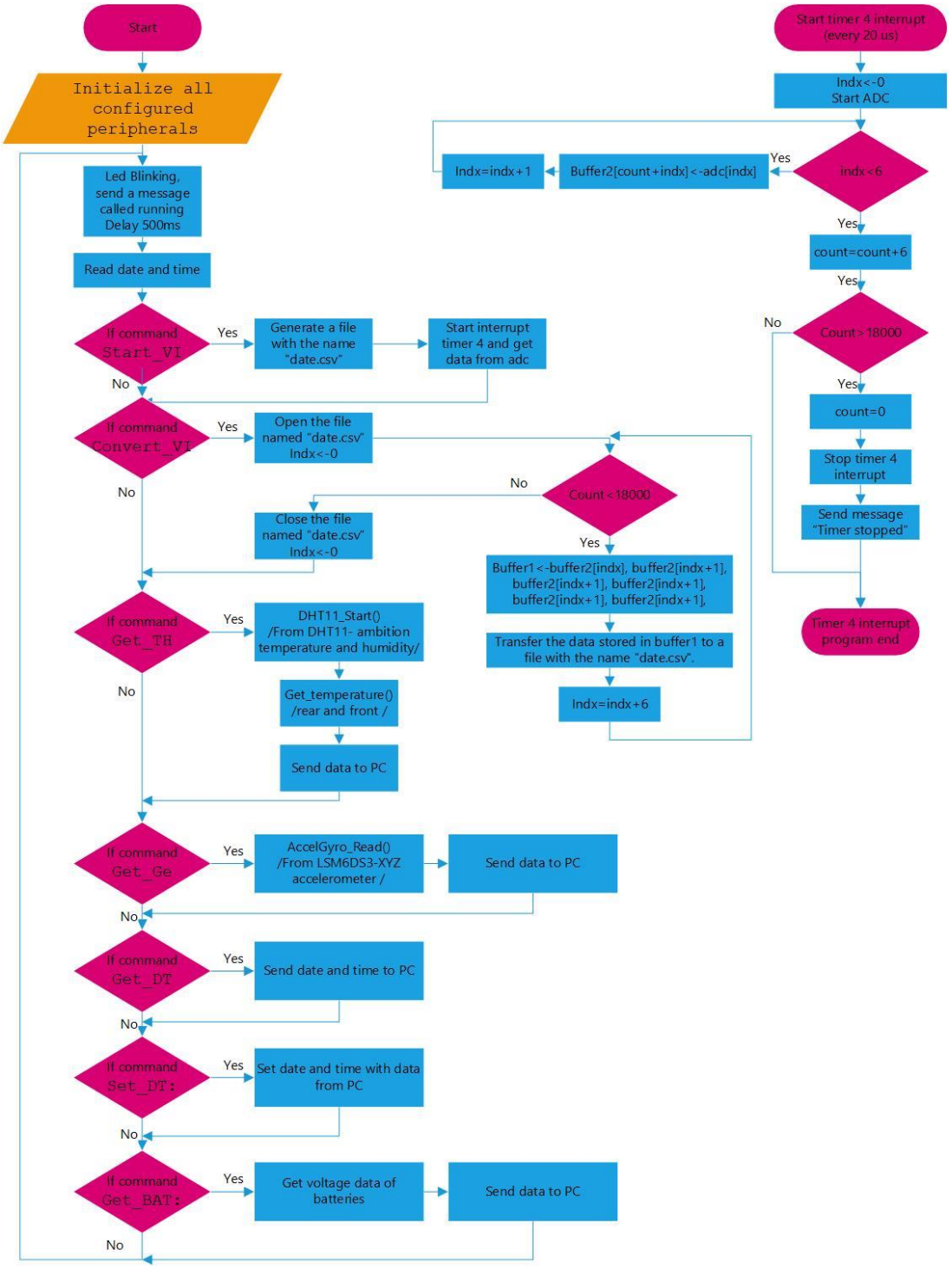

Figure 23. Software flowchart of Motor Analyzer LEVEL V1.0 data acquisition system

The software development of the Motor Analyzer LEVEL V1.0 data acquisition system involves creating programs and algorithms that enable the system to collect, process, and store data from various sensors. This includes writing firmware for the microcontroller, such as the STM32F407VET, which manages sensor data acquisition, real-time processing, and communication with other components. The

software must handle the input from sensors like voltage, current, temperature, humidity, and accelerometer, converting raw data into meaningful information.

Additionally, the software integrates the real-time clock (RTC) to timestamp data accurately and manages data logging to the microSD card for storage. Communication protocols such as I2C, SPI, and UART are implemented to facilitate data exchange between the microcontroller and sensors, as well as to interface with external devices via the USB port.

User interface software might also be developed to allow users to interact with the system, configure settings, and retrieve data. This can include creating a graphical user interface (GUI) for easier data visualization and analysis. Overall, the software development ensures that the data acquisition system operates efficiently, reliably, and meets the specific requirements of the application, enabling comprehensive monitoring and analysis of motor performance and other parameters.

The code for the software is shown and in Appendix 1.

```
CDC_Transmit_FS((uint8_t*)"Running\n", 10);
HAL_Delay(500);
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_SET) ;
HAL_Delay(500);
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_RESET);

HAL_RTC_GetTime(&hrtc, &sTime, RTC_FORMAT_BIN);
HAL_RTC_GetDate(&hrtc, &sDate, RTC_FORMAT_BIN);
sprintf(date,"Date: %02d.%02d.%02d\t",sDate.Year,sDate.Month,sDate.Date);
sprintf(time,"Time: %02d.%02d.%02d\t",sTime.Hours,sTime.Minutes,sTime.Seconds);


if(!strcmp(buffer, "Start_VI"))
{
    memset (buffer, '\0', 64);
    Mount_SD("/");
    //sprintf(FileName1, (uint8_t*)"%02d-%02d-%02d-%02d-%02d-%02d.CSV",sDate.Year,sDa
    sprintf(FileName1, "VI%02d%02d%02d.CSV",sTime.Hours,sTime.Minutes,sTime.Seconds);
    Create_File(FileName1);
    HAL_Delay(10);
    Update_File(FileName1, "Va,Vb,Vc,Ia,Ib,Ic\n");
    HAL_Delay(10);

    CDC_Transmit_FS((uint8_t*)"Timer starting\n", 15);
    HAL_TIM_Base_Start_IT(&htim4);
}
if(!strcmp(buffer, "Convert_VI"))
{
    memset (buffer, '\0', 64);
    CDC_Transmit_FS((uint8_t*)"Starting convert\n", 17);
    Mount_SD("/");
      /* Open the file with write access */
      fresult1 = f_open(&fil1, FileName1, FA_OPEN_EXISTING | FA_READ | FA_WRITE);
      /* Move to offset to the end of the file */
      fresult1 = f_lseek(&fil1, f_size(&fil1));
      if (fresult1== FR_OK) CDC_Transmit_FS((uint8_t*)"About to update the file1\n",
    for (indx=0;indx<18000;indx+=6)
    {
        HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_0);
        sprintf(buffer1,"%4d,%4d,%4d,%4d,%4d,%4d\n", buffer2[indx], buffer2[indx+1],
        fresult1 = f_puts(buffer1, &fil1);
        memset (buffer1, '\0', 64);
    }
    f_close (&fil1);
    Unmount_SD("/");
```

Figure 24. Motor Analyzer LEVEL V1.0 data acquisition system

4.3. Results of Motor Analyzer LEVEL V1.0 data acquisition system

Motor Analyzer LEVEL V1.0 data acquisition system is shown in Figure 25 the industrial field.

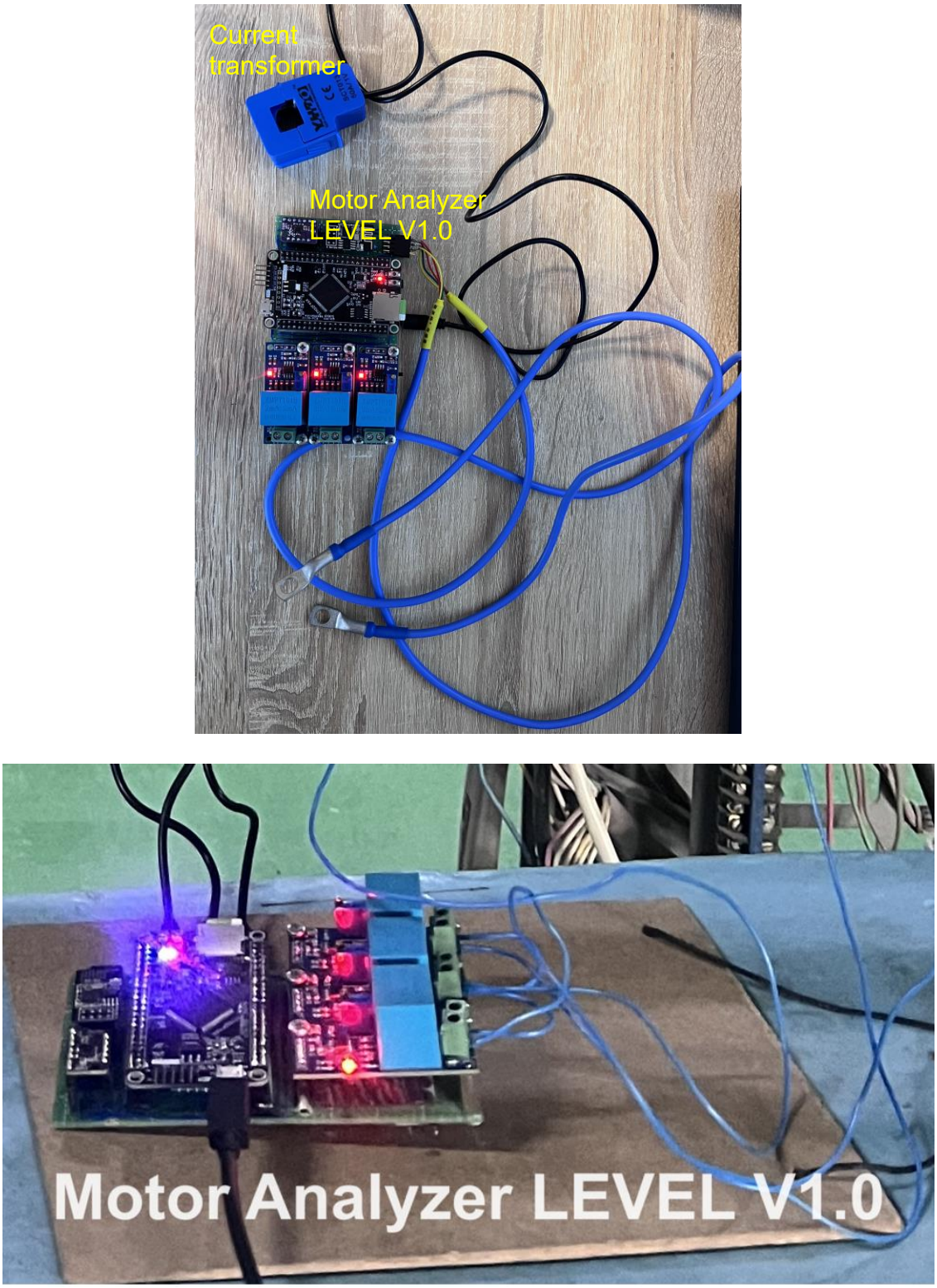

Figure 25. Motor Analyzer LEVEL V1.0 data acquisition system

The current of the mill's cooling ventilation motor in the processing plant was measured by the "Motor Analyzer LEVEL V1.0" data acquisition system shown in Figure *26*.

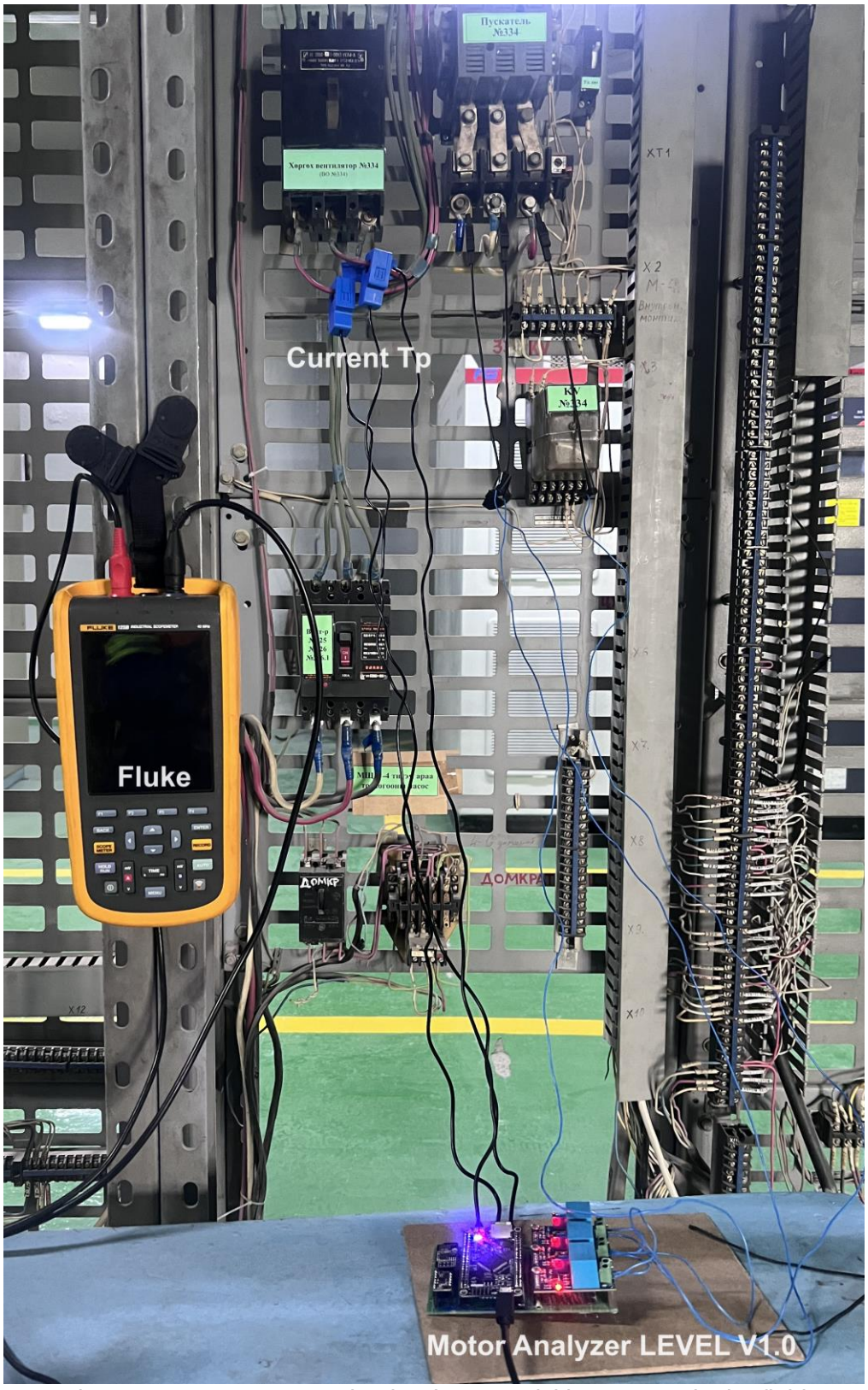

Figure 26. Measurement by the data acquisition system in the field

4.4. Conclusion of Chapter

The "Motor Analyzer LEVEL V1.0" is a data acquisition system designed to monitor and log various types of environmental and operational data. It is built around the STM32F407VET microcontroller, which processes data from voltage, current, temperature, humidity, and accelerometer sensors. Key components include:

- Real-Time Clock (RTC): Ensures accurate time-stamping.
- MicroSD Card: Provides large data storage.
- USB Port: Facilitates data transfer and system updates.
- 4800 mAh Battery: Enables independent operation.

The system uses various sensors, such as the ZMPT101B voltage sensor, SCT013 current transformer, DS18B20 temperature sensor, DHT11 humidity sensor, and LSM6DS3-XYZ accelerometer. It also includes the ESP-01S wireless network module for Wi-Fi connectivity.

The software development involved creating software for sensor data acquisition, real-time processing, and communication using protocols like I2C, SPI, and UART. The system effectively measured the current of a mill's cooling ventilation motor, demonstrating its capability in monitoring and analyzing motor performance for predictive maintenance and fault detection.

The software and hardware of "Motor Analyzer LEVEL V1.0" data acquisition system had developed based on the research work.

The current of the mill's cooling ventilation motor in the processing plant was measured by the "Motor Analyzer LEVEL V1.0" data acquisition system, demonstrating its effectiveness in monitoring and analyzing motor performance and other parameters.

CHAPTER 5. RESULTS AND DISCUSSIONS

5.1. Setup data acquisition systems

The two different data acquisition systems used in this study are the newly designed "Motor Analyzer LEVEL V1.0" and the Baker EXP3000 Explorer Dynamic Motor Analyzer (99-EXP3000-CE), which is already established in the industrial field, as shown in Figure 27. The Baker EXP3000 Explorer Dynamic Motor Analyzer is used to validate the data from the newly designed "Motor Analyzer LEVEL V1.0," developed based on this research work.

Figure 27. Voltage measurement by Baker EXP3000 Explorer in the field

The Baker EXP3000 Explorer Dynamic Motor Analyzer 99-EXP3000-CE is an advanced diagnostic tool used for comprehensive analysis of electric motors. It is designed to perform detailed assessments of motor condition and performance by measuring and recording various parameters such as voltage, current, and power.

The EXP3000 provides real-time data and diagnostic capabilities, enabling users to detect and diagnose potential issues with motor operation before they lead to failures. It supports both static and dynamic testing, allowing for thorough examination of motor health under different operating conditions. This analyzer is widely used in industrial settings to monitor motor efficiency, predict maintenance needs, and enhance overall reliability and performance of motor-driven systems. The EXP3000 is known for its accuracy, ease of use, and robust features, making it a valuable tool for maintenance professionals and engineers in the field.

The Baker EXP3000 Explorer Dynamic Motor Analyzer has own software that shows measurements shown in Figure 28.

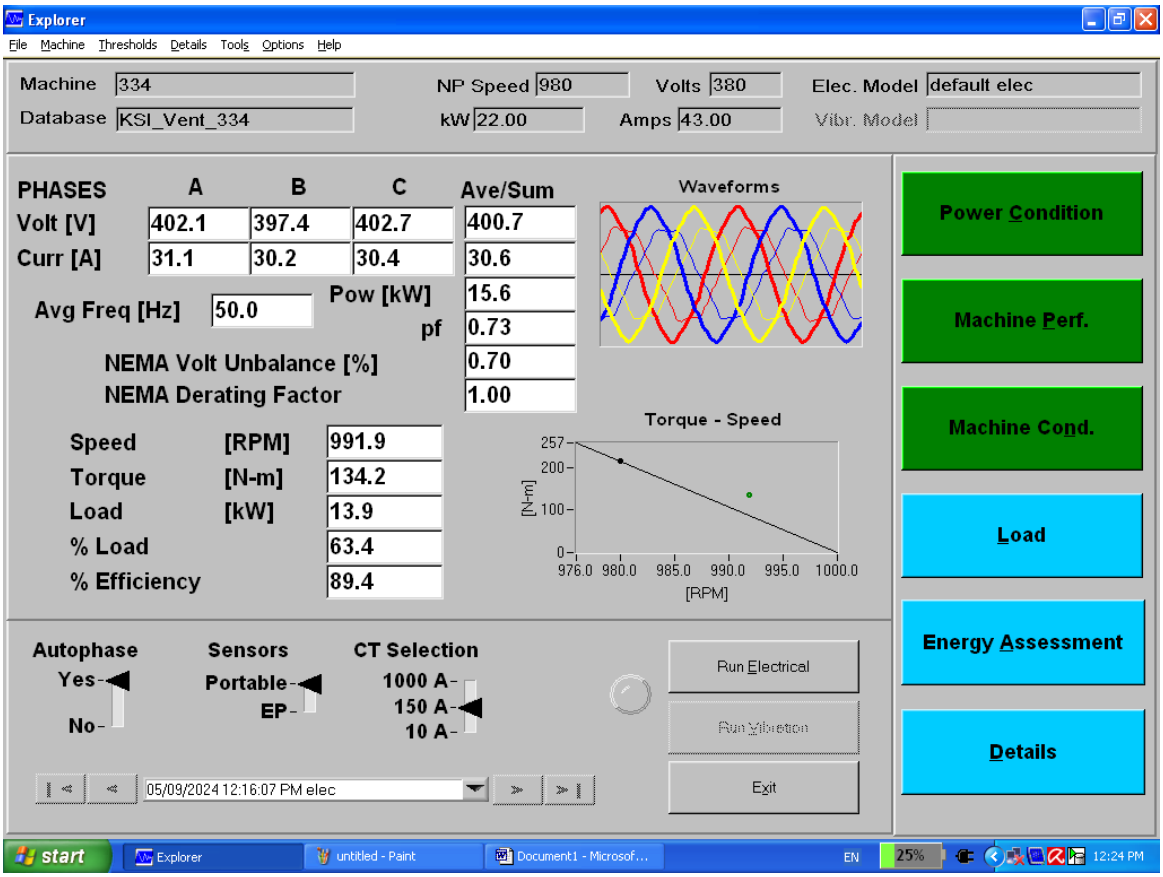

Figure 28. Voltage Measurement by *Baker EXP3000 Explorer*

The SKF Dynamic Motor Analyzer EXP4000 is the next version of Baker EXP3000 Explorer Dynamic Motor Analyzer.

The SKF Dynamic Motor Analyzer EXP4000 is a sophisticated diagnostic tool used for testing and analyzing electric motors. It measures various parameters like current, voltage, power, speed, and torque, providing comprehensive insights into motor performance. With dynamic testing capabilities, it can assess motor behavior under actual operating conditions, aiding in the detection of hidden issues. Equipped with advanced software, it allows real-time data acquisition and analysis, facilitating instant diagnostics. Supporting methods such as motor current signature analysis (MCSA), it helps identify electrical and mechanical problems like bearing faults and rotor bar issues. Overall, the EXP4000 enhances motor maintenance by offering user-friendly interfaces and comprehensive reporting tools, ensuring improved operational reliability and efficiency in industrial settings.

5.2. Time domain analysis of Motor Analyzer LEVEL V1.0

The voltage measurement in the time domain of the "Motor Analyzer LEVEL V1.0" data acquisition system is shown in Figure 29.

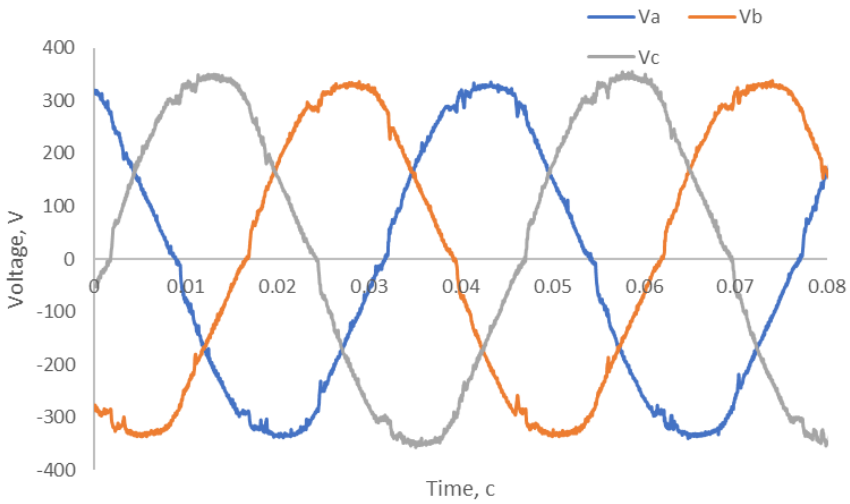

Figure 29. Voltage measurement by Motor Analyzer LEVEL V1.0 data acquisition system in the field

The current measurement in the time domain of the "Motor Analyzer LEVEL V1.0" data acquisition system is shown in Figure 30.

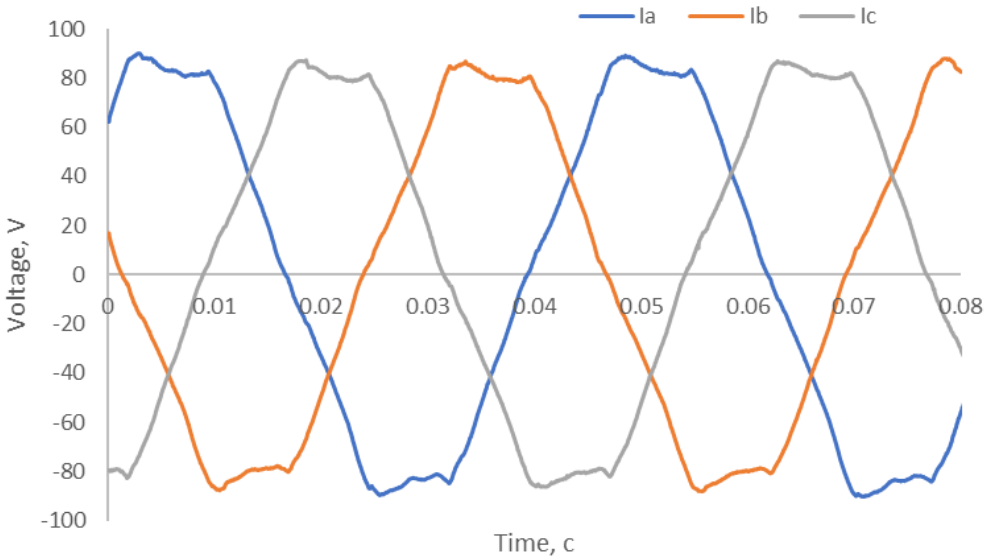

Figure 30. Current measurement by Motor Analyzer LEVEL V1.0 data acquisition system in the field

The voltage measurement in the time domain of the "Explorer" data acquisition system that is available in the field is shown in Figure 31.

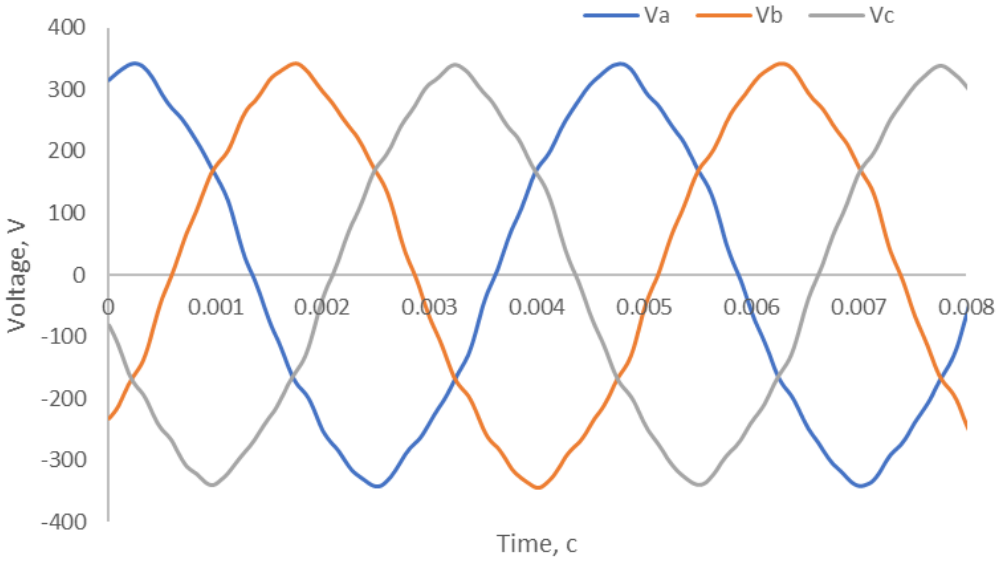

Figure 31. Voltage Measurement by Explorer acquisition system in the field

The current measurement in the time domain of the "Explorer" data acquisition system that is available in the field is shown in Figure *32*.

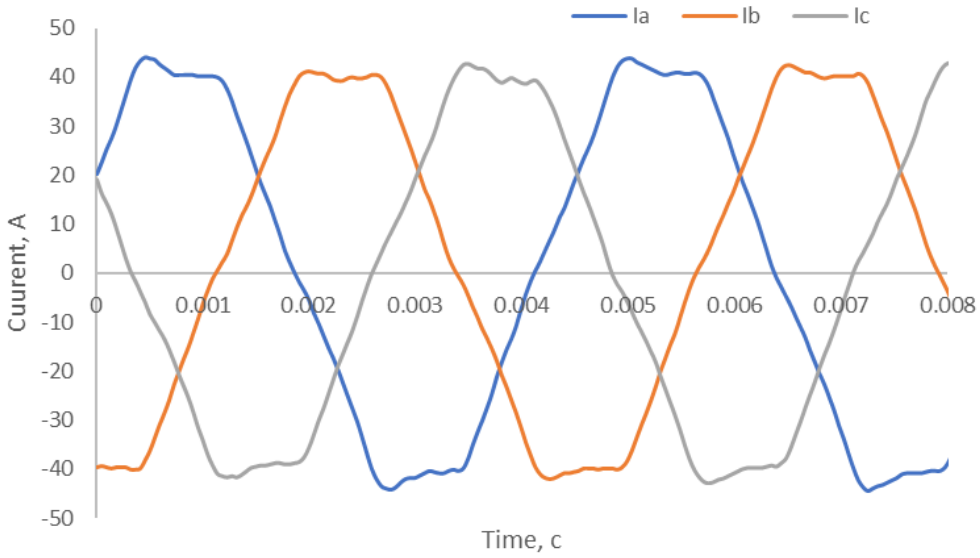

Figure 32. Current Measurement by Explorer acquisition system in the field

The graphics of the voltage measurements in the time domain of the "Motor Analyzer LEVEL V1.0." Explorer are very similar, but the scales of the horizontal axes are different.

From the graphics of the voltage measurements in the time domain of the "Motor Analyzer LEVEL V1.0" and Baker EXP3000 Explorer Dynamic Motor Analyzer, the results are very similar. However, the scales of the horizontal axes are different.

The voltage and current graphs from the Motor Analyzer LEVEL V1.0 and the Baker EXP3000 Explorer Dynamic Motor Analyzer are quite similar. However, their graphs need to be validated. Therefore, the correlation coefficient of their graphs needs to be calculated for validation. If the correlation coefficient of LEVEL and Explorer is

close to 1, it indicates a strong similarity. Only one period of the sinusoid of LEVEL and Explorer are focused on. Only one period of the voltage sinusoid from LEVEL and Explorer is focused on (Figure 33).

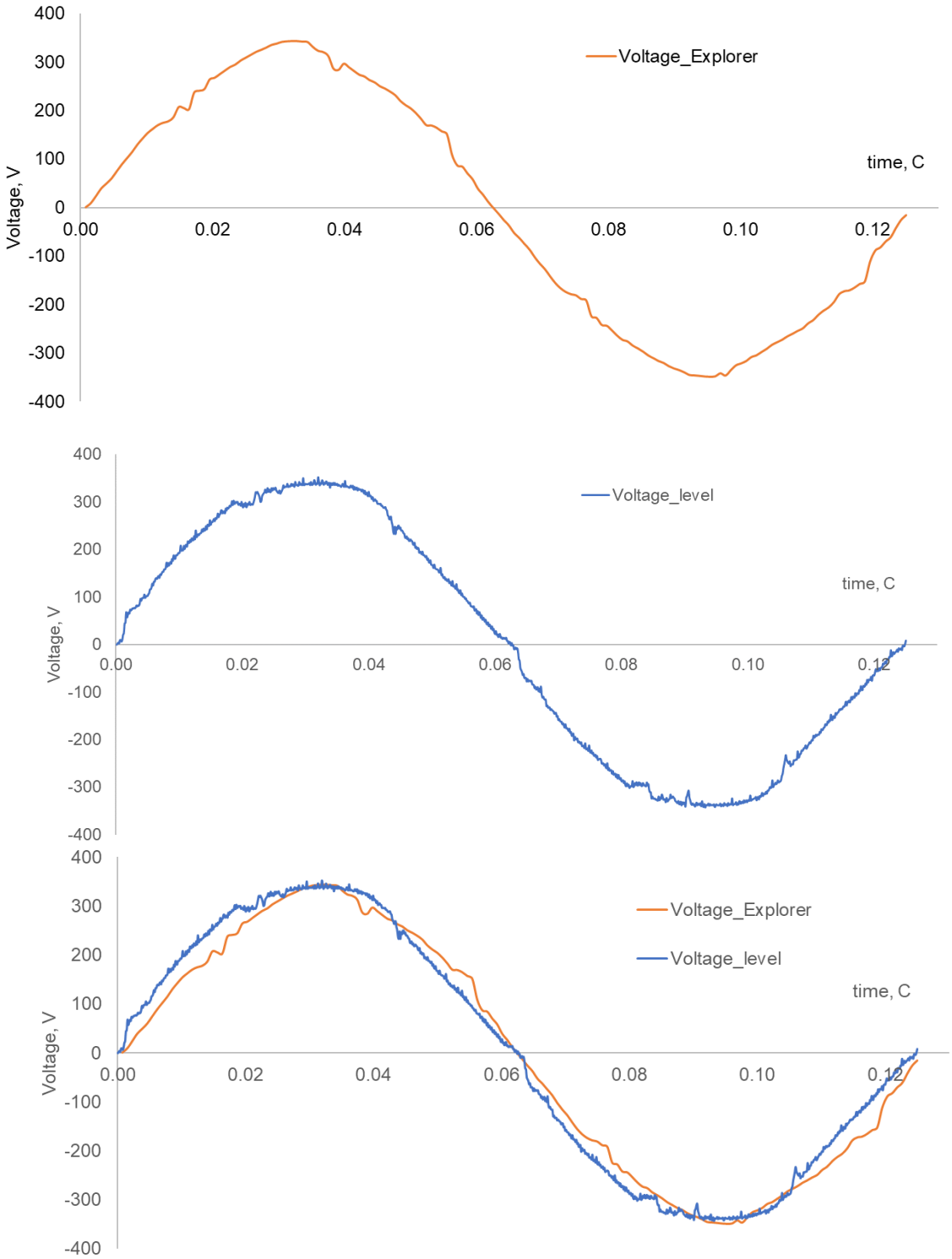

Figure 33. Voltage measurement of one period

Current Measurement of one period are shown in Figure 34.

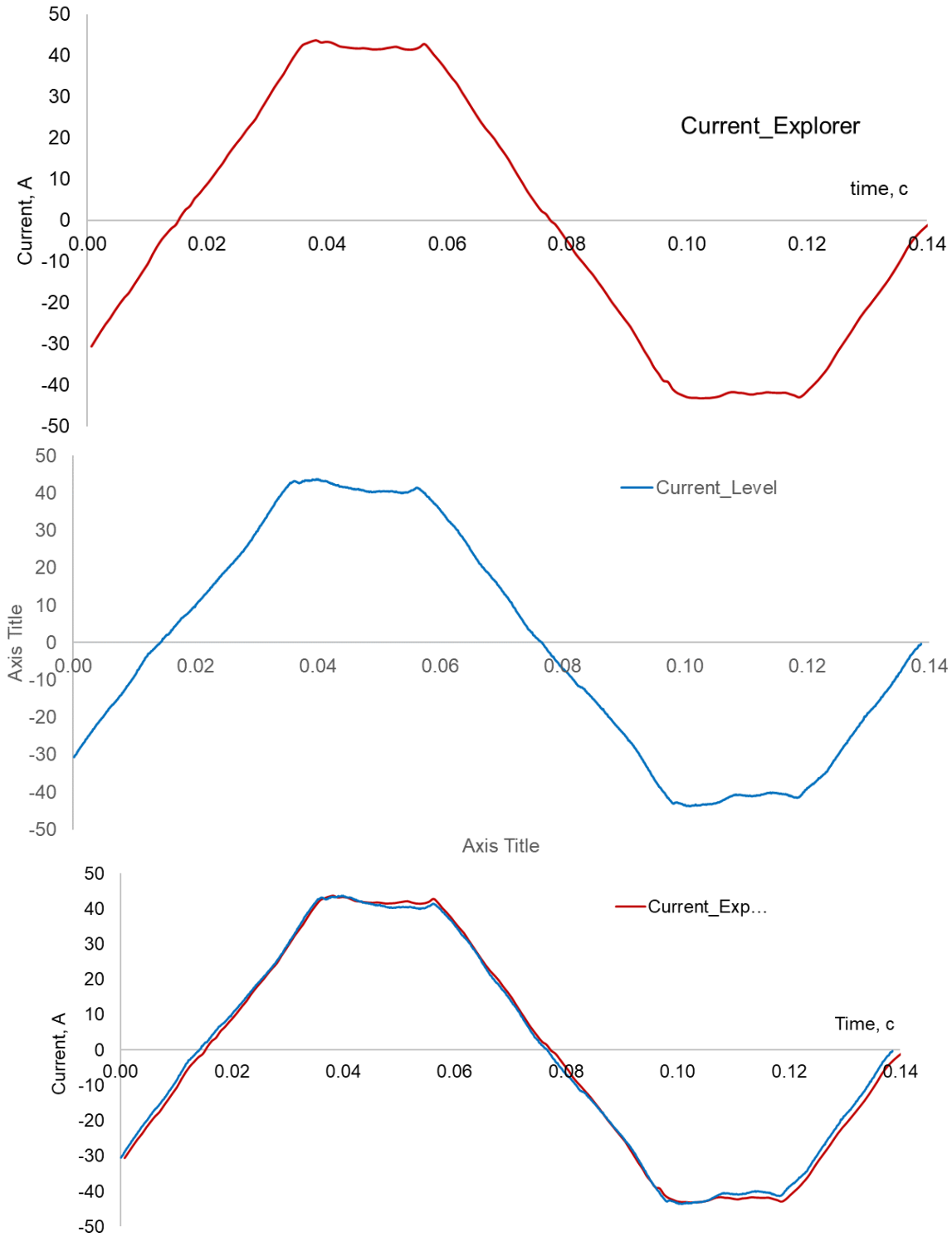

Figure 34. Current Measurement of one period

The correlation coefficients of LEVEL and Explorer are 1 and 2, respectively. A correlation coefficient close to 1 indicates a strong similarity between the two datasets, suggesting that the measurements from both devices are highly correlated. To validate the similarity of their sinusoidal outputs, focusing on only one period of the sinusoid can provide a clear comparison of their performance.

The correlation coefficients of LEVEL and Explorer are shown in Figure 35 and Figure 36.

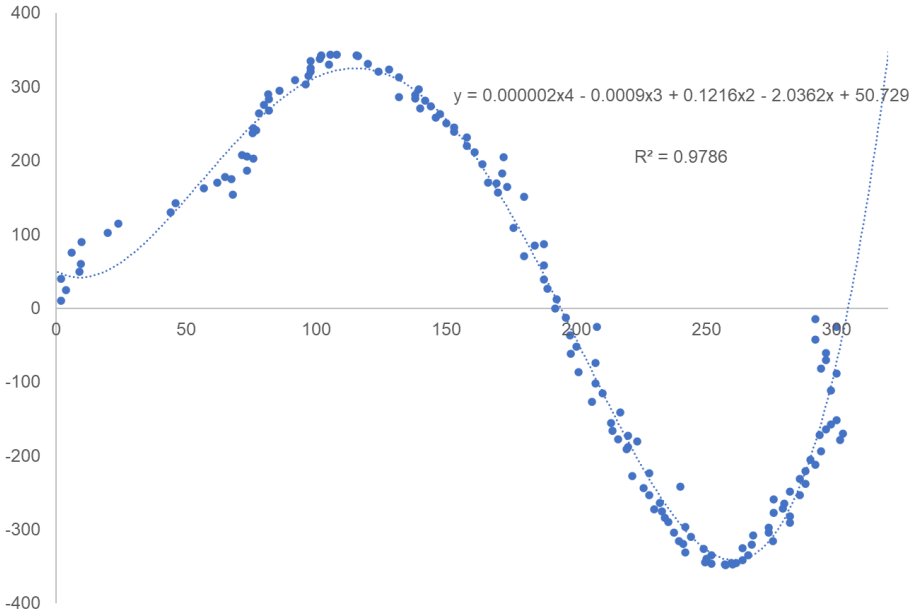

Figure 35. Voltage correlation coefficient between LEVEL and Explorer

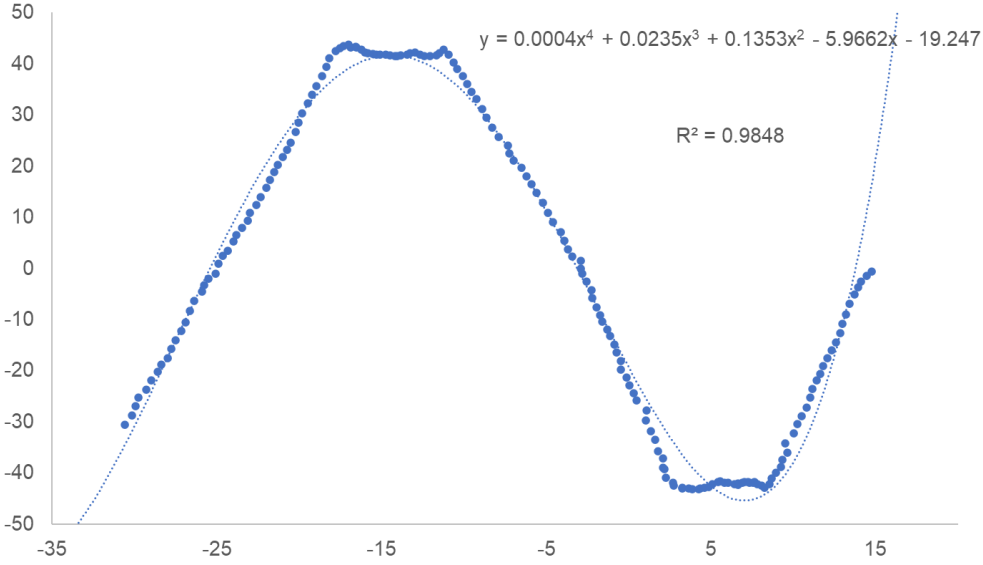

Figure 36. Current correlation coefficient between LEVEL and Explorer

The correlation coefficient is close to 1 for this period, confirming that the new device, LEVEL, produces results similar to the existing device, Explorer. Specifically, LEVEL produces voltage results that correlate with the existing device by 97.8% (R^2 = 0.978), and current results that correlate by 98.5% (R^2 = 0.985).

Therefore, the Motor Analyzer LEVEL V1.0, designed in this research, can be effectively used in industries to measure motor faults.

From the graphics of the voltage measurements in the time domain of the "Motor Analyzer LEVEL V1.0" and Baker EXP3000 Explorer Dynamic Motor Analyzer, the results are very similar. Although the scales of the horizontal axes differ, normalization is employed to standardize them to the same level.

5.3. Frequency domain analysis of Motor Analyzer LEVEL V1.0 and Explorer

The Fast Fourier Transform (FFT) is used in this study. The voltage measurement in the frequency domain of the "Motor Analyzer LEVEL V1.0" data acquisition system is shown in Figure 37.

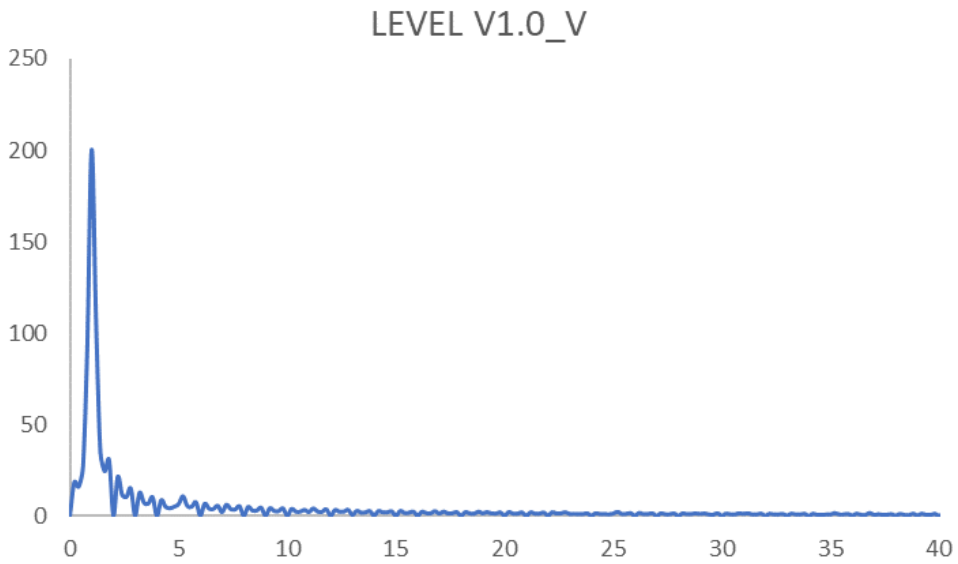

Figure 37. Voltage Measurement by *Motor Analyzer LEVEL V1.0*

The voltage measurement in the frequency domain of the "Explorer" data acquisition system is shown in Figure 38.

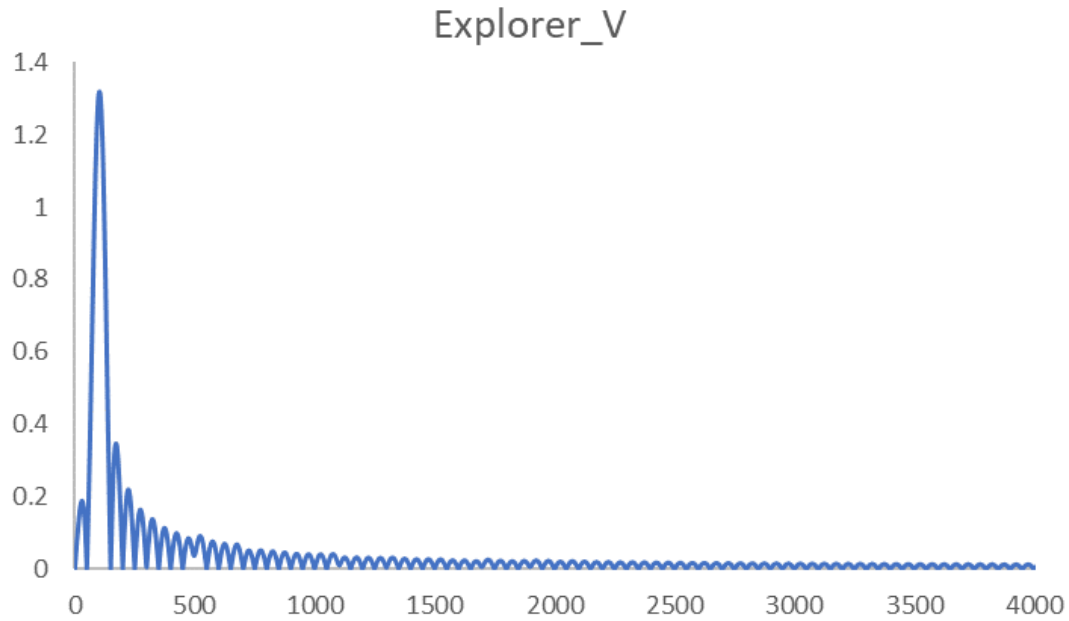

Figure 38. Voltage Measurement in frequency domain by *Explorer* acquisition system

The current measurement in the frequency domain of the Motor Analyzer LEVEL V1.0 data acquisition system is shown in Figure 39.

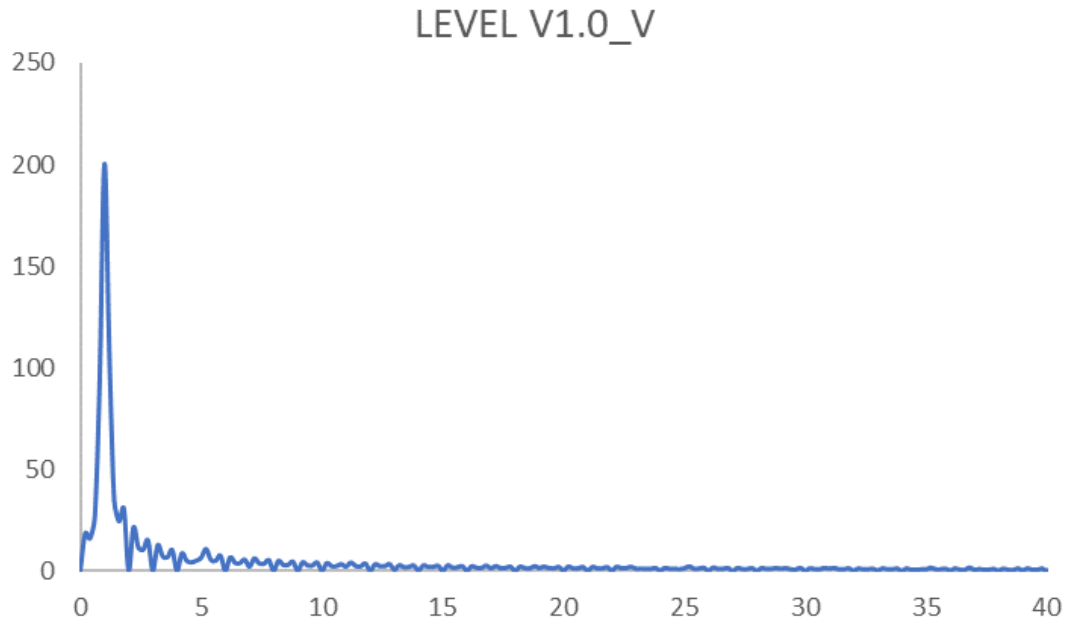

Figure 39. Current Measurement in frequency domain by *Motor Analyzer LEVEL V1.0*

The current measurement in the frequency domain of the "Explorer" data acquisition system is shown in Figure 40.

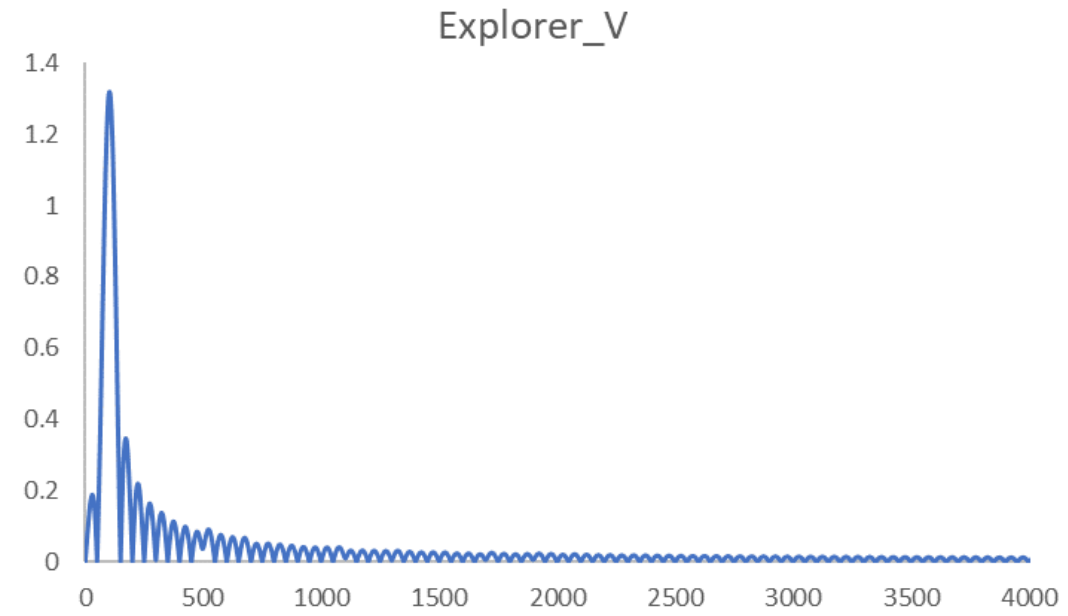

Figure 40. Current Measurement in frequency domain by *Explorer* acquisition system

The voltage and current frequency graphs of the "Motor Analyzer LEVEL V1.0" and the Baker EXP3000 Explorer Dynamic Motor Analyzer show very similar results. However, the scales of the horizontal axes differ due to variations in time-domain voltage and current measurements. The sampling rate for the Motor Analyzer LEVEL V1.0 is 1000 Hz, while the Baker EXP3000 Explorer Dynamic Motor Analyzer is 10000 Hz.

The graphics of the voltage and current measurements in the time domain for the "Motor Analyzer LEVEL V1.0" and the Baker EXP3000 Explorer Dynamic Motor Analyzer are validated by 97.8% ($R^2 = 0.978$) and 98.5% ($R^2 = 0.985$), respectively. The validation of current and voltage measurements in the time domain by the Motor Analyzer LEVEL V1.0 and the Baker EXP3000 Explorer Dynamic Motor Analyzer is nearly 97.8%. Similarly, the validation of current and voltage measurements in the frequency domain by the Motor Analyzer LEVEL V1.0 and the Baker EXP3000 Explorer Dynamic Motor Analyzer is also nearly 98.5%.

5.4. Conclusion of the chapter

This study compares two data acquisition systems: the newly designed "Motor Analyzer LEVEL V1.0" and the established Baker EXP3000 Explorer Dynamic Motor Analyzer (99-EXP3000-CE). The Baker EXP3000 Explorer is a benchmark developed based on this research to validate the data from the Motor Analyzer LEVEL V1.0.

The Baker EXP3000 Explorer Dynamic Motor Analyzer is a sophisticated diagnostic tool for comprehensively analysing electric motors. It provides real-time data and diagnostic capabilities, supports static and dynamic testing, and is widely used in industrial settings for monitoring motor efficiency and predicting maintenance needs. The SKF Dynamic Motor Analyzer EXP4000's next version offers advanced features like motor current signature analysis (MCSA) and enhanced real-time data acquisition and analysis.

In the time domain analysis, both systems' voltage and current measurements show very similar results, with differences primarily in the horizontal axis scales. The sampling rate for the Motor Analyzer LEVEL V1.0 is 1000 Hz, while the Baker EXP3000 Explorer is 10000 Hz. In the frequency domain analysis, the Fast Fourier Transform (FFT) was used to convert real-time data from the time domain to the frequency domain, facilitating the identification of frequency components within the signals.

The validation of current and voltage measurements in both the time and frequency domains by the Motor Analyzer LEVEL V1.0 and the Baker EXP3000 Explorer Dynamic Motor Analyzer shows a high correlation, with an accuracy of nearly 98%. This demonstrates that the Motor Analyzer LEVEL V1.0 is an effective tool for monitoring and analyzing motor performance, comparable to the established Baker EXP3000 Explorer Dynamic Motor Analyzer.

The Motor Analyzer LEVEL V1.0, developed in this study, demonstrates an impressive level of correlation with existing devices, boasting a 97.8% correlation for voltage results ($R^2 = 0.978$) and 98.5% for current results ($R^2 = 0.985$). These high correlations underscore the efficacy of the LEVEL V1.0 for accurately measuring motor faults in industrial settings.

THESIS SUMMARY

This thesis explores the development and effectiveness of advanced data acquisition systems for multi-fault diagnosis in induction motors, focusing on the newly developed Motor Analyzer LEVEL V1.0 from Mongolia. It highlights the critical role these systems play in modern industrial environments by enabling early fault detection and facilitating efficient maintenance strategies.

Chapter 1 discusses the importance of data acquisition systems in motor health and performance monitoring. It introduces the Motor Analyzer LEVEL V1.0, comparing it with the Baker EXP3000 Explorer Dynamic Motor Analyzer, and emphasizes its innovative design and effectiveness in minimizing downtime.

Chapter 2 covers condition monitoring of asynchronous motors, particularly squirrel-cage induction motors. It details common motor faults and various detection techniques, such as vibration analysis and Motor Current Signature Analysis (MCSA). The chapter underscores the benefits of advanced models and simulations for early fault detection and intervention.

Chapter 3 examines the essential components of data acquisition systems for online condition monitoring. It highlights the use of various monitoring techniques and algorithms, emphasizing the importance of real-time data analysis and the Fast Fourier Transform (FFT) for accurate diagnostics.

Chapter 4 describes the hardware and software development of the Motor Analyzer LEVEL V1.0, detailing its components and capabilities. The system's effectiveness is demonstrated through its application in monitoring a mill's cooling ventilation motor, showing high accuracy and reliability compared to the Baker EXP3000 Explorer Dynamic Motor Analyzer.

The thesis concludes that continuous innovation in data acquisition and condition monitoring systems is vital for enhancing industrial efficiency and sustainability. The Motor Analyzer LEVEL V1.0 represents a significant advancement in local industrial technology, providing valuable insights and practical solutions for improving operational reliability and reducing maintenance costs.

The Motor Analyzer LEVEL V1.0, developed in this study, shows strong correlations with existing devices, achieving 97.8% correlation for voltage results and 98.5% for current results. This underscores its effectiveness for measuring motor faults in industrial applications.

CONCLUSION AND FUTURE WORKS

This thesis underscores the critical importance of advanced data acquisition systems for multi-fault diagnosis and condition monitoring of induction motors in industrial environments. The development and introduction of the Motor Analyzer LEVEL V1.0, a novel data acquisition system from Mongolia, represent a significant contribution to local industrial technology. This system showcases innovative capabilities in proactive fault detection and real-time performance monitoring, substantially improving operational reliability and maintenance efficiency.

The comprehensive analysis throughout the thesis highlights the Motor Analyzer LEVEL V1.0's effectiveness compared to established systems like the Baker EXP3000 Explorer Dynamic Motor Analyzer. Both systems show a high correlation in their measurements, validating the new system's reliability and accuracy. The integration of advanced monitoring techniques and algorithms ensures robust data collection, processing, and fault detection, essential for minimizing downtime and reducing maintenance costs.

The findings of this thesis emphasize the essential role of ongoing innovation in data acquisition and condition monitoring systems in enhancing industrial efficiency and sustainability. By addressing current challenges and advancing diagnostic technologies, this research paves the way for more reliable and cost-effective industrial operations, supporting continuous improvement and sustainability in the industry.

REFERENCES

Antero Arkkio. 2001. “Electromagnetic Damping of Stator Vibrations in Cage Induction Motor.” Espoo,.

Ariunbolor Purvee, Gautam Banerjee. 2013. “ON LINE CONDITION MONITORING OF MINING ELECTRICAL MACHINES.” *Journal of Chemical Information and Modeling* 53 (9): 1689–99. https://doi.org/10.1017/CBO9781107415324.004.

Automation World. n.d. “Https_automationworld.Com.”

github.com. n.d. “STM32F407VET BlackBoard.” Accessed May 19, 2024. https://github.com/ukw100/STECCY/blob/main/README.md.

Gojko M. Joksimovic, Jim Penman,. 2000. “The Detection of Inter-Turn Short Circuits in the Stator of Operating Motors.” *IEEE Transactions On Industrial Electronics* 47 (5): 1078–84.

Purvee, Ariunbolor, Enkhbat Tsend-Ayush, Phd Fellow, and Munkhjargal Tserendorj. n.d. “Rotor Faults of Induction Motors Based on Dynamic Simulation.”

АРИУНБОЛОР, Пүрвээгийн. 2009. “БОГИНО ХОЛБОГДСОН РОТОРТОЙ АСИНХРОН ХӨДӨЛГҮҮРИЙН ДИНАМИК ЗАГВАРЧЛАЛ.” Диссертаци, ШУТИС.

Garcia-Perez, A., Rangel-Magdaleno, J. J., & Osornio-Rios, R. A. (2020). An intelligent data acquisition and signal processing system for online condition monitoring of induction motors. Sensors, 20(21), 6177.

Sala, A., Peretto, L., & Rinaldi, F. (2015). Online condition monitoring system for a sustainable industry: An experimental study. Procedia CIRP, 38, 248-253.

Yuan, J., Liang, Y., & Ma, Z. (2014). An online condition monitoring system based on signal processing technique for large turbine generator units. International Journal of Performability Engineering, 10(3), 271-279.

Antoni, J., & Randall, R. B. (2006). The spectral kurtosis: application to the vibratory surveillance and diagnostics of rotating machines. Mechanical systems and signal processing, 20(2), 308-331.

Li, J., Gao, R. X., & Vibration-Based Machine Learning for Condition Monitoring. CRC Press, 2019.

Liu, J., Cao, S., & Zhang, C. (2016). A review of wavelet transform time–frequency methods for machinery fault detection and diagnosis. Mechanical Systems and Signal Processing, 76, 295-307.

Antoni, J., & Randall, R. B. (2006). The spectral kurtosis: application to the vibratory surveillance and diagnostics of rotating machines. Mechanical systems and signal processing, 20(2), 308-331.

Li, J., Gao, R. X., & Vibration-Based Machine Learning for Condition Monitoring. CRC Press, 2019.

Tirale, M., Lhermet, H., Benbouzid, M. E. H., & Marquez, F. P. G. (2017). A survey on induction motor fault detection and diagnosis methods. IEEE Transactions on Industrial Electronics, 64(5), 3997-4008.

Wang, T., Lu, H., Wang, Q., Zhang, Z., & Zhang, H. (2020). Park's vector based feature extraction and deep learning for induction motor fault diagnosis. IEEE Transactions on Industrial Informatics, 16(5), 3422-3431.

Brigham, E. O. (1988). The Fast Fourier Transform and Its Applications. Prentice-Hall.

Cooley, J. W., & Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. Mathematics of Computation, 19(90), 297-301.

Oppenheim, A. V., & Schafer, R. W. (1975). Digital Signal Processing. Prentice-Hall.

Smith, S. W. (1997). The Scientist and Engineer's Guide to Digital Signal Processing. California Technical Publishing

APPENDIX: MAIN SCRIPT CODES

```
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file           : main.c
  * @brief          : Main program body
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2024 STMicroelectronics.
  * All rights reserved.
  *

  *
  ******************************************************************************
  */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"
#include "fatfs.h"
#include "usb_device.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include "usbd_cdc_if.h"
#include "string.h"
#include "stdio.h"
#include "stdlib.h"
#include "File_Handling.h"
#include "OneWire.h"
#include "DHT11.h"
#include "lsm6ds3.h"

/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */
char time[30];
char date[30];
RTC_TimeTypeDef sTime;
RTC_DateTypeDef sDate;
/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */

/* USER CODE END PD */

/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/
ADC_HandleTypeDef hadc1;
ADC_HandleTypeDef hadc2;
DMA_HandleTypeDef hdma_adc1;

I2C_HandleTypeDef hi2c1;
```

```
RTC_HandleTypeDef hrtc;

SD_HandleTypeDef hsd;

TIM_HandleTypeDef htim1;
TIM_HandleTypeDef htim4;

UART_HandleTypeDef huart1;

/* USER CODE BEGIN PV */
void microDelay (uint16_t delay)
{
  __HAL_TIM_SET_COUNTER(&htim1, 0);
  while (__HAL_TIM_GET_COUNTER(&htim1) < delay);
}
//RTC_TimeTypeDef gTime;
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_DMA_Init(void);
static void MX_SDIO_SD_Init(void);
static void MX_TIM4_Init(void);
static void MX_ADC1_Init(void);
static void MX_USART1_UART_Init(void);
static void MX_TIM1_Init(void);
static void MX_I2C1_Init(void);
static void MX_RTC_Init(void);
static void MX_ADC2_Init(void);
/* USER CODE BEGIN PFP */

FATFS fs1;  // file system
FIL fil1; // File
FILINFO fno1;
FRESULT fresult1;  // result
UINT br1, bw1;  // File read/write count
/* USER CODE END PFP */

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
//uint8_t *data = "Hello World from USB CDC\n";
char *data = "Hello World from USB CDC\n";

char buffer[64];
char buffer1[64];
uint16_t buffer2[18000];
uint8_t result;
uint16_t indx = 0, m=0, count=0;
uint16_t adc[6];
uint16_t battery;
extern float Temp[MAXDEVICES_ON_THE_BUS];
////////////DHT11////////////
uint8_t RHI, RHD, TCI, TCD, SUM;
float tCelsius = 0;
float tFahrenheit = 0;
float RH = 0;
int c;
char FileName1[30]; //24-05-11 18:16:54.CSV

/* USER CODE END 0 */
```

```
/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_DMA_Init();
  MX_SDIO_SD_Init();
  MX_FATFS_Init();
  MX_USB_DEVICE_Init();
  MX_TIM4_Init();
  MX_ADC1_Init();
  MX_USART1_UART_Init();
  MX_TIM1_Init();
  MX_I2C1_Init();
  MX_RTC_Init();
  MX_ADC2_Init();
  /* USER CODE BEGIN 2 */
  extern float Temp[MAXDEVICES_ON_THE_BUS];
  HAL_Delay(2000);
  Mount_SD("/");
  Format_SD();
  //HAL_ADC_Start_DMA(&hadc1, (uint32_t *) adc, 6); // start adc in DMA mode
  HAL_TIM_Base_Stop_IT(&htim4);
 // HAL_TIM_Base_Start_IT(&htim4);

  HAL_Delay(1000);
  get_ROMid();
 // HAL_TIM_Base_Start(&htim1);
  Accel_Gyro_Ini();
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
          CDC_Transmit_FS((uint8_t*)"Running\n", 10);
```

```
        HAL_Delay(500);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_SET) ;
        HAL_Delay(500);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_RESET);

        HAL_RTC_GetTime(&hrtc, &sTime, RTC_FORMAT_BIN);
        HAL_RTC_GetDate(&hrtc, &sDate, RTC_FORMAT_BIN);
        sprintf(date,"Date: %02d.%02d.%02d\t",sDate.Year,sDate.Month,sDate.Date);
        sprintf(time,"Time: %02d.%02d.%02d\t",sTime.Hours,sTime.Minutes,sTime.Seconds);


        if(!strcmp(buffer, "Start_VI"))
        {
          memset (buffer, '\0', 64);
          Mount_SD("/");
          //sprintf(FileName1,
(uint8_t*)"%02d-%02d-%02d-%02d-%02d-%02d.CSV",sDate.Year,sDate.Month,sDate.Date,sTime.
Hours,sTime.Minutes,sTime.Seconds);
          sprintf(FileName1,
"%02d-%02d-%02d.CSV",sTime.Hours,sTime.Minutes,sTime.Seconds);
          Create_File(FileName1);
          HAL_Delay(10);
          Update_File(FileName1, "Va,Vb,Vc,Ia,Ib,Ic\n");
          HAL_Delay(10);

          CDC_Transmit_FS((uint8_t*)"Timer starting\n", 15);
          HAL_TIM_Base_Start_IT(&htim4);
        }
        if(!strcmp(buffer, "Convert_VI"))
        {
          memset (buffer, '\0', 64);
          CDC_Transmit_FS((uint8_t*)"Starting convert\n", 17);
          Mount_SD("/");
                /* Open the file with write access */
                fresult1 = f_open(&fil1, FileName1, FA_OPEN_EXISTING | FA_READ |
FA_WRITE);
                /* Move to offset to the end of the file */
                fresult1 = f_lseek(&fil1, f_size(&fil1));
                if (fresult1== FR_OK) CDC_Transmit_FS((uint8_t*)"About to update the file1\n", 32);
          for (indx=0;indx<18000;indx+=6)
          {
                  HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_0);
                  sprintf(buffer1,"%4d,%4d,%4d,%4d,%4d,%4d\n", buffer2[indx], buffer2[indx+1],
buffer2[indx+2], buffer2[indx+3], buffer2[indx+4], buffer2[indx+5]);
                  fresult1 = f_puts(buffer1, &fil1);
                  memset (buffer1, '\0', 64);
          }
          f_close (&fil1);
          Unmount_SD("/");
          CDC_Transmit_FS((uint8_t*)"Stopped convert\n", 16);
        }
        if(!strcmp(buffer, "Get_TH"))
        {
          memset (buffer, '\0', 64);
          if(DHT11_Start())
          {
                  RHI = DHT11_Read();
                  RHD = DHT11_Read();
                  TCI = DHT11_Read();
                  TCD = DHT11_Read();
                  SUM = DHT11_Read();
                  if (RHI + RHD + TCI + TCD == SUM)
```

```
                {
                        tCelsius = (float)TCI + (float)(TCD/10.0);
                  tFahrenheit = tCelsius * 9/5 + 32;
                  RH = (float)RHI + (float)(RHD/10.0);
                }
            }
            get_Temperature();
            sprintf(buffer1,"F:%.f\nR:%.f\nA:%.f\nH:%.f\n", Temp[0],Temp[1],tCelsius,RH);
            CDC_Transmit_FS((uint8_t*)buffer1,48);
            memset (buffer1, '\0', 64);
          }
          if(!strcmp(buffer, "Get_Ge"))
          {
            memset (buffer, '\0', 64);
            while(strcmp(buffer, "Stop_Ge"))
                    AccelGyro_Read();
            memset (buffer, '\0', 64);
          }
          if(!strcmp(buffer, "Get_DT"))
          {
            memset (buffer, '\0', 64);

            CDC_Transmit_FS((uint8_t*)date,sizeof(date));
            HAL_Delay(50);
            CDC_Transmit_FS((uint8_t*)time,sizeof(time));
          }
          if(!strncmp(buffer, "Set_DT:",6))    //Set_DT:24-05-07 16:47:48
          {
            sDate.Year=CopyStringInteger((uint8_t*)buffer,7);
            sDate.Month=CopyStringInteger((uint8_t*)buffer,10);
            sDate.Date=CopyStringInteger((uint8_t*)buffer,13);
            HAL_RTC_SetDate(&hrtc, &sDate, RTC_FORMAT_BIN);

            sTime.Hours=CopyStringInteger((uint8_t*)buffer,16);
            sTime.Minutes=CopyStringInteger((uint8_t*)buffer,19);
            sTime.Seconds=CopyStringInteger((uint8_t*)buffer,22);
            HAL_RTC_SetTime(&hrtc, &sTime, RTC_FORMAT_BIN);
            memset (buffer, '\0', 64);
          }
          if(!strcmp(buffer, "Get_BAT"))
            {
                    char temp[15];
                    memset (buffer, '\0', 64);
                    HAL_ADC_Start(&hadc2);
                    HAL_ADC_PollForConversion(&hadc2,300);
                    battery = HAL_ADC_GetValue(&hadc2);
                    float vin = battery*(3.3/4096);
                    sprintf(temp,"Battery: %.2f\r\n", vin);
                    CDC_Transmit_FS((uint8_t*)temp,sizeof(temp));
            }

  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
```

```
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Configure the main internal regulator output voltage
  */
  __HAL_RCC_PWR_CLK_ENABLE();
  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);

  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType                                                    =
RCC_OSCILLATORTYPE_HSE|RCC_OSCILLATORTYPE_LSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.LSEState = RCC_LSE_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
  RCC_OscInitStruct.PLL.PLLM = 4;
  RCC_OscInitStruct.PLL.PLLN = 168;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
  RCC_OscInitStruct.PLL.PLLQ = 7;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV4;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV2;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_5) != HAL_OK)
  {
    Error_Handler();
  }
}

/**
  * @brief ADC1 Initialization Function
  * @param None
  * @retval None
  */
static void MX_ADC1_Init(void)
{

  /* USER CODE BEGIN ADC1_Init 0 */

  /* USER CODE END ADC1_Init 0 */

  ADC_ChannelConfTypeDef sConfig = {0};

  /* USER CODE BEGIN ADC1_Init 1 */

  /* USER CODE END ADC1_Init 1 */

  /** Configure the global features of the ADC (Clock, Resolution, Data Alignment and number of
conversion)
  */
  hadc1.Instance = ADC1;
```

```
  hadc1.Init.ClockPrescaler = ADC_CLOCK_SYNC_PCLK_DIV4;
  hadc1.Init.Resolution = ADC_RESOLUTION_12B;
  hadc1.Init.ScanConvMode = ENABLE;
  hadc1.Init.ContinuousConvMode = DISABLE;
  hadc1.Init.DiscontinuousConvMode = DISABLE;
  hadc1.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
  hadc1.Init.ExternalTrigConv = ADC_SOFTWARE_START;
  hadc1.Init.DataAlign = ADC_DATAALIGN_RIGHT;
  hadc1.Init.NbrOfConversion = 6;
  hadc1.Init.DMAContinuousRequests = DISABLE;
  hadc1.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
  if (HAL_ADC_Init(&hadc1) != HAL_OK)
  {
    Error_Handler();
  }

  /** Configure for the selected ADC regular channel its corresponding rank in the sequencer and its
sample time.
  */
  sConfig.Channel = ADC_CHANNEL_2;
  sConfig.Rank = 1;
  sConfig.SamplingTime = ADC_SAMPLETIME_15CYCLES;
  if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }

  /** Configure for the selected ADC regular channel its corresponding rank in the sequencer and its
sample time.
  */
  sConfig.Channel = ADC_CHANNEL_3;
  sConfig.Rank = 2;
  if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }

  /** Configure for the selected ADC regular channel its corresponding rank in the sequencer and its
sample time.
  */
  sConfig.Channel = ADC_CHANNEL_4;
  sConfig.Rank = 3;
  if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }

  /** Configure for the selected ADC regular channel its corresponding rank in the sequencer and its
sample time.
  */
  sConfig.Channel = ADC_CHANNEL_5;
  sConfig.Rank = 4;
  if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }

  /** Configure for the selected ADC regular channel its corresponding rank in the sequencer and its
sample time.
  */
  sConfig.Channel = ADC_CHANNEL_6;
  sConfig.Rank = 5;
```

```
  if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }

  /** Configure for the selected ADC regular channel its corresponding rank in the sequencer and its
sample time.
  */
  sConfig.Channel = ADC_CHANNEL_7;
  sConfig.Rank = 6;
  if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN ADC1_Init 2 */

  /* USER CODE END ADC1_Init 2 */

}

/**
  * @brief ADC2 Initialization Function
  * @param None
  * @retval None
  */
static void MX_ADC2_Init(void)
{

  /* USER CODE BEGIN ADC2_Init 0 */

  /* USER CODE END ADC2_Init 0 */

  ADC_ChannelConfTypeDef sConfig = {0};

  /* USER CODE BEGIN ADC2_Init 1 */

  /* USER CODE END ADC2_Init 1 */

  /** Configure the global features of the ADC (Clock, Resolution, Data Alignment and number of
conversion)
  */
  hadc2.Instance = ADC2;
  hadc2.Init.ClockPrescaler = ADC_CLOCK_SYNC_PCLK_DIV4;
  hadc2.Init.Resolution = ADC_RESOLUTION_12B;
  hadc2.Init.ScanConvMode = DISABLE;
  hadc2.Init.ContinuousConvMode = DISABLE;
  hadc2.Init.DiscontinuousConvMode = DISABLE;
  hadc2.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
  hadc2.Init.ExternalTrigConv = ADC_SOFTWARE_START;
  hadc2.Init.DataAlign = ADC_DATAALIGN_RIGHT;
  hadc2.Init.NbrOfConversion = 1;
  hadc2.Init.DMAContinuousRequests = DISABLE;
  hadc2.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
  if (HAL_ADC_Init(&hadc2) != HAL_OK)
  {
    Error_Handler();
  }

  /** Configure for the selected ADC regular channel its corresponding rank in the sequencer and its
sample time.
  */
  sConfig.Channel = ADC_CHANNEL_0;
```

```
  sConfig.Rank = 1;
  sConfig.SamplingTime = ADC_SAMPLETIME_3CYCLES;
  if (HAL_ADC_ConfigChannel(&hadc2, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN ADC2_Init 2 */

  /* USER CODE END ADC2_Init 2 */

}

/**
  * @brief I2C1 Initialization Function
  * @param None
  * @retval None
  */
static void MX_I2C1_Init(void)
{

  /* USER CODE BEGIN I2C1_Init 0 */

  /* USER CODE END I2C1_Init 0 */

  /* USER CODE BEGIN I2C1_Init 1 */

  /* USER CODE END I2C1_Init 1 */
  hi2c1.Instance = I2C1;
  hi2c1.Init.ClockSpeed = 400000;
  hi2c1.Init.DutyCycle = I2C_DUTYCYCLE_2;
  hi2c1.Init.OwnAddress1 = 0;
  hi2c1.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
  hi2c1.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
  hi2c1.Init.OwnAddress2 = 0;
  hi2c1.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
  hi2c1.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
  if (HAL_I2C_Init(&hi2c1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN I2C1_Init 2 */

  /* USER CODE END I2C1_Init 2 */

}

/**
  * @brief RTC Initialization Function
  * @param None
  * @retval None
  */
static void MX_RTC_Init(void)
{

  /* USER CODE BEGIN RTC_Init 0 */

  /* USER CODE END RTC_Init 0 */

  RTC_TimeTypeDef sTime = {0};
  RTC_DateTypeDef sDate = {0};

  /* USER CODE BEGIN RTC_Init 1 */
```

```
  /* USER CODE END RTC_Init 1 */

  /** Initialize RTC Only
  */
  hrtc.Instance = RTC;
  hrtc.Init.HourFormat = RTC_HOURFORMAT_24;
  hrtc.Init.AsynchPrediv = 127;
  hrtc.Init.SynchPrediv = 255;
  hrtc.Init.OutPut = RTC_OUTPUT_DISABLE;
  hrtc.Init.OutPutPolarity = RTC_OUTPUT_POLARITY_HIGH;
  hrtc.Init.OutPutType = RTC_OUTPUT_TYPE_OPENDRAIN;
  if (HAL_RTC_Init(&hrtc) != HAL_OK)
  {
    Error_Handler();
  }

  /* USER CODE BEGIN Check_RTC_BKUP */

  /* USER CODE END Check_RTC_BKUP */

  /** Initialize RTC and set the Time and Date
  */
  sTime.Hours = 0x0;
  sTime.Minutes = 0x0;
  sTime.Seconds = 0x0;
  sTime.DayLightSaving = RTC_DAYLIGHTSAVING_NONE;
  sTime.StoreOperation = RTC_STOREOPERATION_RESET;
  if (HAL_RTC_SetTime(&hrtc, &sTime, RTC_FORMAT_BCD) != HAL_OK)
  {
    Error_Handler();
  }
  sDate.WeekDay = RTC_WEEKDAY_MONDAY;
  sDate.Month = RTC_MONTH_JANUARY;
  sDate.Date = 0x1;
  sDate.Year = 0x0;

  if (HAL_RTC_SetDate(&hrtc, &sDate, RTC_FORMAT_BCD) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN RTC_Init 2 */

  /* USER CODE END RTC_Init 2 */

}

/**
  * @brief SDIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_SDIO_SD_Init(void)
{

  /* USER CODE BEGIN SDIO_Init 0 */

  /* USER CODE END SDIO_Init 0 */

  /* USER CODE BEGIN SDIO_Init 1 */

  /* USER CODE END SDIO_Init 1 */
```

```
  hsd.Instance = SDIO;
  hsd.Init.ClockEdge = SDIO_CLOCK_EDGE_RISING;
  hsd.Init.ClockBypass = SDIO_CLOCK_BYPASS_DISABLE;
  hsd.Init.ClockPowerSave = SDIO_CLOCK_POWER_SAVE_DISABLE;
  hsd.Init.BusWide = SDIO_BUS_WIDE_1B;
  hsd.Init.HardwareFlowControl = SDIO_HARDWARE_FLOW_CONTROL_DISABLE;
  hsd.Init.ClockDiv = 4;
  /* USER CODE BEGIN SDIO_Init 2 */

  /* USER CODE END SDIO_Init 2 */

}

/**
  * @brief TIM1 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM1_Init(void)
{

  /* USER CODE BEGIN TIM1_Init 0 */

  /* USER CODE END TIM1_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};

  /* USER CODE BEGIN TIM1_Init 1 */

  /* USER CODE END TIM1_Init 1 */
  htim1.Instance = TIM1;
  htim1.Init.Prescaler = 83;
  htim1.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim1.Init.Period = 65535;
  htim1.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim1.Init.RepetitionCounter = 0;
  htim1.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim1) != HAL_OK)
  {
    Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim1, &sClockSourceConfig) != HAL_OK)
  {
    Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim1, &sMasterConfig) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN TIM1_Init 2 */

  /* USER CODE END TIM1_Init 2 */

}

/**
  * @brief TIM4 Initialization Function
  * @param None
```

```
  * @retval None
  */
static void MX_TIM4_Init(void)
{

  /* USER CODE BEGIN TIM4_Init 0 */

  /* USER CODE END TIM4_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};

  /* USER CODE BEGIN TIM4_Init 1 */

  /* USER CODE END TIM4_Init 1 */
  htim4.Instance = TIM4;
  htim4.Init.Prescaler = 42-1;
  htim4.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim4.Init.Period = 40-1;
  htim4.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim4.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim4) != HAL_OK)
  {
    Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim4, &sClockSourceConfig) != HAL_OK)
  {
    Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim4, &sMasterConfig) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN TIM4_Init 2 */

  /* USER CODE END TIM4_Init 2 */

}

/**
  * @brief USART1 Initialization Function
  * @param None
  * @retval None
  */
static void MX_USART1_UART_Init(void)
{

  /* USER CODE BEGIN USART1_Init 0 */

  /* USER CODE END USART1_Init 0 */

  /* USER CODE BEGIN USART1_Init 1 */

  /* USER CODE END USART1_Init 1 */
  huart1.Instance = USART1;
  huart1.Init.BaudRate = 115200;
  huart1.Init.WordLength = UART_WORDLENGTH_8B;
  huart1.Init.StopBits = UART_STOPBITS_1;
  huart1.Init.Parity = UART_PARITY_NONE;
```

```
  huart1.Init.Mode = UART_MODE_TX_RX;
  huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;
  huart1.Init.OverSampling = UART_OVERSAMPLING_16;
  if (HAL_HalfDuplex_Init(&huart1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN USART1_Init 2 */

  /* USER CODE END USART1_Init 2 */

}

/**
  * Enable DMA controller clock
  */
static void MX_DMA_Init(void)
{

  /* DMA controller clock enable */
  __HAL_RCC_DMA2_CLK_ENABLE();

  /* DMA interrupt init */
  /* DMA2_Stream0_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(DMA2_Stream0_IRQn, 0, 0);
  HAL_NVIC_EnableIRQ(DMA2_Stream0_IRQn);

}

/**
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};
/* USER CODE BEGIN MX_GPIO_Init_1 */
/* USER CODE END MX_GPIO_Init_1 */

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOD_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOC, GPIO_PIN_0|GPIO_PIN_1|GPIO_PIN_7, GPIO_PIN_RESET);

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_RESET);

  /*Configure GPIO pins : PC0 PC1 PC7 */
  GPIO_InitStruct.Pin = GPIO_PIN_0|GPIO_PIN_1|GPIO_PIN_7;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);

  /*Configure GPIO pin : PA1 */
  GPIO_InitStruct.Pin = GPIO_PIN_1;
```

```
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

/* USER CODE BEGIN MX_GPIO_Init_2 */
/* USER CODE END MX_GPIO_Init_2 */
}

/* USER CODE BEGIN 4 */
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
  HAL_ADC_Start_DMA(&hadc1,(uint32_t *)adc,6);
  if(htim->Instance == TIM4)
  {
   // HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_0);
  //  HAL_GPIO_WritePin(GPIOC, GPIO_PIN_1, GPIO_PIN_SET);
    for (indx=0;indx<6;indx++)
    {
        buffer2[count+indx]=adc[indx];
    }
    count=count+6;
    if (count>18000)
    {
        count=0;
        HAL_TIM_Base_Stop_IT(&htim4);
        CDC_Transmit_FS((uint8_t*)"Timer stopped\n", 14);
    }
  //  HAL_GPIO_WritePin(GPIOC, GPIO_PIN_1, GPIO_PIN_RESET);

  }
}

void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc1)
{
        UNUSED(hadc1);
        //HAL_GPIO_WritePin(DebugOut_GPIO_Port, DebugOut_Pin, GPIO_PIN_RESET);
}

int CopyStringInteger(uint8_t* Buf, uint16_t Len)
{
        char* substr = malloc(2);
        strncpy(substr, Buf+Len, 2);
        strncat(substr, "\0", 1);
        return atoi(substr);
}
/* USER CODE END 4 */

/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}
```

```
#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```