Department of Information Science Engineering in Association with Elevium

Mini Project Synopsis AY:2025-2026

Name of Team leader	**:**	**SHIVAPRASAD V TOGGI**	**Team No**	**:**	**37**
Project Title	**:**	**Industrial Machine Health Monitoring System using Accelerometers for Predictive Failure Analysis**	**Sem & Sec**	**:**	**3rd C**
Team Name	**:**	**ZODIAC**	**Course Code**	**:**	**IC204**

1. Project Title

Industrial Machine Health Monitoring System using Accelerometers for Predictive Failure Analysis

2. Team Details

- Student Name(s) & USN's: Shivaprasad V Toggi [1CD24IS147]
  Varun Gowda J [1CD24IS187]
  Sneha K M [1CD24IS163]
  Sinchana [1CD24IS158]

- Semester & Branch: 3rd semester Information Science Engineering
- Internal Mentor:
- External Mentor:

3. Problem Statement

Unexpected machine failures in industries lead to costly downtime and expensive repairs. There is a need for a cost-effective, embedded solution that can predict faults early using vibration and thermal analysis.

4. Objectives

- To interface 3-axis accelerometers and environmental sensors with the STM32F446RE.
- To implement signal processing algorithms (Fast Fourier Transform) on the STM32 to analyze vibration frequencies.
- To correlate temperature rises and vibration spikes with specific machine faults.
- To display the health status and generate alerts on a remote dashboard.

5. Literature Review / Background

The Need for Predictive Maintenance

Industries are moving away from "run-to-failure" maintenance (fixing things only after they break) toward "Predictive Maintenance." The goal is to detect faults—like loose bearings or overheating—days or weeks before a machine actually stops working. This is primarily done by analyzing Vibration, Temperature, and Current.

Limitations of Existing Solutions

Based on recent research, existing solutions generally fall into two categories, both of which have drawbacks for this specific university project:

1. **Low-Cost Hobbyist Systems (e.g., Arduino/NodeMCU):**
   Many researchers have used 8-bit microcontrollers (like Arduino UNO) to read sensors. While cheap, these boards lack the processing power to perform complex math (Signal Processing/FFT) efficiently. They often just send raw data to a PC, which causes delays and high bandwidth usage.

2. **High-End Industrial Systems (e.g., PLCs/NI CompactRIO):**
   Other studies utilize industrial PLCs (Programmable Logic Controllers) or SCADA systems. While accurate, these are extremely expensive, proprietary, and bulky, making them unsuitable for cost-effective, embedded applications.

The Proposed Solution (STM32)

This project bridges the gap by using the STM32F446RE. Unlike basic Arduinos, this board has a dedicated Floating Point Unit (FPU) and DSP (Digital Signal Processing) instructions. This allows the system to calculate vibration frequencies and detect faults directly on the board in real-time, offering a balance of low cost and high engineering performance.

Author	Year	Title	Hardware/Software	Pros	Cons
Korkua, S. et al.	2010	Wireless Health Monitoring System for Vibration Detection of Induction Motors	ZigBee (CC2430), 8051 MCU, ADXL330 (Accelerometer), LabVIEW	Wireless implementation; Used FFT and Crest Factor for detailed analysis.	Used older 8051 architecture with low processing power; High latency in ZigBee.
Lin, C. et al.	2014	Structural Health Monitoring of Bridges Using Cost-Effective 1-axis Accelerometers	RM48HDK Gateway, SDI1221 (1-axis Accel), ADC, Custom Packet Software	Demonstrated 64% cost saving by using specific 1-axis sensors; Good data synchronization.	Focused on civil structures (Bridges) rather than rotating machinery; No onboard spectral analysis.
Khan, N. et al.	2019	IoT Based Health Monitoring System for Electrical Motors	Arduino Mega 2560, ESP8266 (WiFi), GSM, ADXL335, ACS712 (Current), LabVIEW	Multi-parameter monitoring (Vibration + Current + Temp); Dual connectivity (WiFi + GSM).	Processing done off-board in LabVIEW; Arduino Mega is limited for complex DSP math.
Alghassi, A. et al.	2020	Machine Performance Monitoring and Fault Classification using Vibration Frequency Analysis	NI Compact RIO, IoT Edge Device, Python, Random Forest/SVM Algorithms	High accuracy using Machine Learning; Real-time frequency domain (FFT) analysis.	Uses expensive proprietary hardware (NI RIO); High power consumption; Complex setup.

Khandelwal, V. et al.	2022	Sensor Based Vibration Analysis of Motor Using MATLAB Software	Arduino UNO, ADXL335, MATLAB	Simple and low-cost implementation; Good for basic visualization.	Relies on MATLAB for processing (not real-time on chip); Only Time-domain analysis (no FFT).
Mastang et al.	2023	Machine Condition Monitoring System Based on IoT Platform for Intelligent Maintenance	Omron NX-102 PLC, Python, SQL Database, Dashboard	Industrial grade reliability; Uses Linear Regression for trend prediction.	Requires expensive PLC hardware; Not a dedicated embedded system solution.
Banerjee, A. et al.	2024	Fault Detection and Machine Health Monitoring in Cylinder Block Production Lines...	Siemens S7-300 PLC, Ignition SCADA, OPC UA Server	Integrated into actual manufacturing line (Honing Machine); Real-time SCADA visualization.	High complexity; Proprietary software license costs; Focus on servo drives rather than general vibration.

6. Proposed Methodology & Architecture (Block Diagram)

A. Methodology

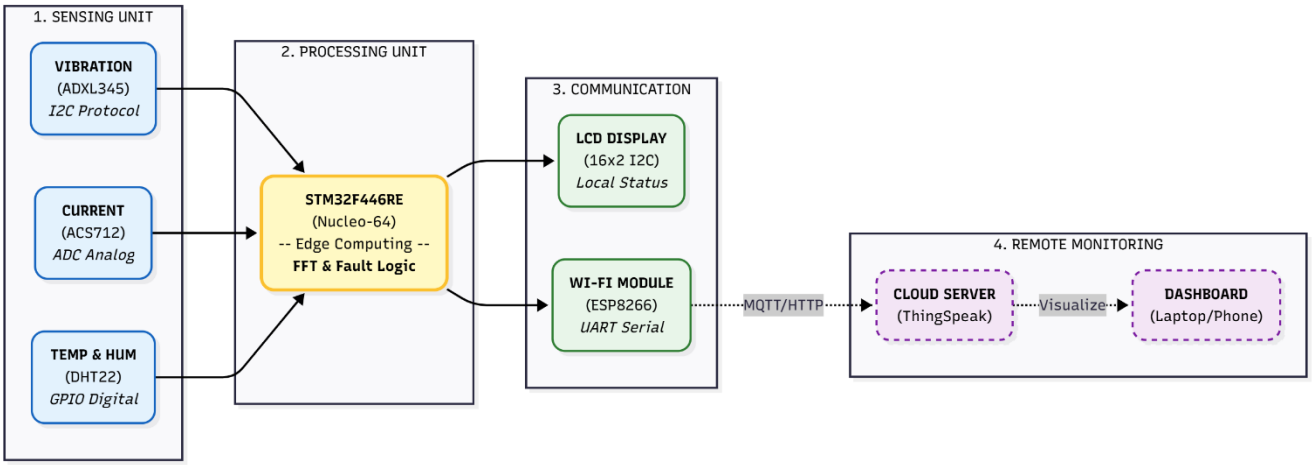
The project aims to replicate the robust monitoring framework but enhances it by using a high-performance 32-bit STM32F446RE microcontroller to perform real-time signal processing. The methodology is divided into three distinct phases:

1. **Data Acquisition (Sensing Layer):**
    - **Vibration:** A MEMS Accelerometer (ADXL345/335) measures acceleration in X, Y, and Z axes to detect mechanical looseness or bearing faults.
    - **Environmental:** A digital sensor (DHT22/BME280/DHT11) monitors Temperature and Humidity, as overheating is a precursor to electrical failure.
    - **Current Sensing:** An ACS712 sensor measures the load current to detect electrical anomalies (overloading).
2. **Edge Processing & Predictive Analysis (Processing Layer):**
    - The STM32F446RE will perform "Edge Computing".
    - The microcontroller collects raw vibration data and applies the Fast Fourier Transform (FFT) algorithm using the ARM CMSIS-DSP library.
    - This converts time-domain vibration signals into frequency domains to identify specific fault signatures (e.g., spikes at specific frequencies indicating rotor imbalance)
3. **IoT Communication & Visualization (Application Layer):**

- The processed health status is transmitted via an ESP8266 Wi-Fi module (or HC-05 Bluetooth) to a cloud platform (like ThingSpeak) or a local PC Dashboard.
- If critical thresholds (high temp or abnormal vibration) are exceeded, the system triggers a local alert (Buzzer/LED) and updates the remote dashboard in real-time.

B. System Architecture

The system is built around the STM32 Nucleo-F446RE, acting as the central processing unit. It interfaces with sensors via ADC/I2C protocols and communicates with the external world via UART.



7. Expected Outcome

The primary outcome of this project is a fully functional, low-cost IoT-based Condition Monitoring System capable of detecting faults in rotating machinery. Specifically, the project will deliver:

1. Functional Hardware Prototype

- A standalone embedded system based on the STM32 Nucleo-F446RE controller.
- Integration of a multi-sensor array similar to the reference study, including:
  - Vibration Sensing: 3-Axis MEMS Accelerometer (ADXL345) to detect mechanical instability.
  - Electrical Sensing: Current Sensor (ACS712) to detect motor overloading.
  - Thermal Sensing: Temperature sensor to detect overheating.

2. Real-Time Fault Detection (Edge Processing)

- The system will successfully identify and display the machine's status in real-time:
  - Normal Condition: Vibration and current within baseline limits.
  - Faulty Condition**:** Detection of anomalies (e.g., high vibration amplitude or over-current) triggering an immediate local alert.

3. IoT Data Logging and Visualization

- Implementation of a wireless dashboard (using ThingSpeak or a similar IoT platform) to visualize the data remotely.
- Generation of Time-Domain graphs (Amplitude vs. Time) and Frequency-Domain indicators (FFT peaks) to visualize the severity of the fault, allowing for predictive maintenance planning.

4. Comparative Analysis

- A validation report comparing data collected from a "Healthy Motor" versus a "Faulty Motor" (simulated imbalance), demonstrating the system's ability to distinguish between the two states effectively.

8. Applications / Scope

A. Applications

This system is designed for industries that rely heavily on rotating machinery. The primary applications include:

1. **Manufacturing Industries (CNC & Lathes):**
    - Monitoring the spindle vibration of CNC machines to detect tool wear or misalignment before it damages the workpiece.
2. **Water Pumping Stations:**
    - Predicting "cavitation" or bearing failure in large induction motor pumps used in municipal water supplies or agriculture.
3. **HVAC Systems:**
    - Monitoring the motors in large air conditioning units and ventilation fans in commercial buildings to prevent unexpected cooling failures.
4. **Conveyor Belt Systems:**
    - Detecting motor overload or jamming in assembly lines or mining operations (using the current and vibration sensors).

B. Scope (Industry Relevance & Future Potential)

1. **Transition to Industry 4.0:**
    - This project demonstrates the shift from "Reactive Maintenance" (fixing after breakage) to "Predictive Maintenance". By implementing Edge Computing on the STM32F446RE, it aligns with modern Smart Factory standards.
2. **Cost & Safety Benefits:**
    - Economic: Reduces costly downtime by alerting operators *before* a catastrophic failure occurs.
    - Safety: Prevents dangerous accidents caused by motors exploding or disintegrating due to undetected mechanical faults.
3. **Future Scalability:**
    - While the current prototype monitors one machine via Wi-Fi, the scope extends to creating a "Mesh Network" where hundreds of machines can be monitored simultaneously using a central server, integrating Machine Learning for automated fault classification

9. Resources Required

A. Hardware Requirements

1. **Microcontroller (Development Board):**
    - **STM32 Nucleo-F446RE:** The core processing unit (ARM Cortex-M4) used for data acquisition and FFT signal processing.
2. **Sensors:**
    - **MEMS Accelerometer:** ADXL345 (Digital) or MPU6050. (Used for 3-axis vibration detection).
    - **Current Sensor: ACS712** (20A or 30A module). (Used to measure motor load current).
    - **Environmental Sensor:** DHT22 or BME280 or DHT11. (Measures both Temperature and Humidity, per mentor's requirement).
3. **Communication & Display:**
    - **Wi-Fi Module:** ESP8266 (ESP-01). (For transmitting data to the cloud, replacing the costly GSM module from the reference paper).
    - **Display:** 16x2 LCD (with I2C Backpack for easier wiring).
4. **Miscellaneous:**
    - Breadboard and Jumper Wires (Male-Male, Male-Female).
    - 5V Power Supply (for the motor and sensors).
    - USB Cable (Mini-B or Micro-B) for programming the Nucleo board.

B. Software Requirements

1. **Development Environment (IDE):**
    - **STM32CubeIDE:** For writing and debugging the C code.
    - **STM32CubeMX:** For configuring the microcontroller pins and clock speed.
2. **Libraries:**
    - **CMSIS-DSP Library:** An official ARM library required to perform the Fast Fourier Transform (FFT) math efficiently.
    - **HAL Drivers:** Hardware Abstraction Layer for sensor communication (I2C/UART/ADC).
3. **IoT & Visualization:**
    - **ThingSpeak:** Open-source cloud platform for real-time data logging and graph visualization.
    - **CoolTerm / Putty:** For debugging Serial data on the laptop.
4. **Design Tools:**
    - **MS Visio / Draw.io:** For creating Block Diagrams and Flowcharts.
    - **Proteus / Fritzing:** For circuit diagram simulation.

5. **Version Control:** GitHub (for source code management, version history tracking, and team collaboration).

C. Other Requirements

- **Operating System:** Windows 10/11 (for running STM32 tools).
- **Internet Connection:** Wi-Fi Hotspot (for the ESP8266 module to send data).

10. Project Timeline

- **Phase 1: Literature Survey & Requirement Analysis (Weeks 1-2)**
    - Study of reference papers and STM32 documentation.
    - Procurement of hardware components (STM32 Nucleo, Sensors, ESP8266).
    - Finalizing the system block diagram and circuit design.
- **Phase 2: Hardware Interfacing & Driver Development (Weeks 3-5)**
    - Interfacing the ADXL345 (Vibration), ACS712 (Current), and DHT22 (Temp) sensors with the STM32F446RE.
    - Writing low-level drivers (HAL Code) to read raw data via I2C and ADC.
    - Milestone**:** Successful display of raw sensor values on the serial monitor.
- **Phase 3: Core Software & Signal Processing (Weeks 6-8)**
    - Implementation of the Fast Fourier Transform (FFT) algorithm using the ARM CMSIS-DSP library.
    - Developing the logic to detect specific fault frequencies (Edge Computing).
    - Setting up threshold values for "Healthy" vs "Faulty" states.
- **Phase 4: IoT Integration & System Integration (Weeks 9-10)**
    - Configuring the ESP8266 Wi-Fi module for UART communication.
    - Setting up the ThingSpeak dashboard and transmitting processed data to the cloud.
    - Creating the local alert system (LEDs/Buzzer) and integrating all code modules.
- **Phase 5: Testing, Validation & Documentation (Weeks 11-14)**
    - **Testing:** Running the system on the induction motor test rig; simulating faults (unbalanced rotor) to verify detection accuracy.
    - **Calibration:** Fine-tuning sensor sensitivity based on real-world noise.
    - **Final Output:** Writing the project report, preparing the presentation, and finalizing the prototype enclosure.

11. References

1. (**Korkua, S., Jain, H., Lee, W. J., & Kwan, C. (2010).** Wireless Health Monitoring System for Vibration Detection of Induction Motors. *IEEE*.

2. **Lin, C. H. et al. (2014).** Structural Health Monitoring of Bridges Using Cost-Effective 1-axis Accelerometers. *IEEE*.

3. **Khan, N., Rafiq, F., Abedin, F., & Khan, F. U. (2019).** IoT Based Health Monitoring System for Electrical Motors. *IEEE*.

4. **Alghassi, A., Yu, Z., & Farbiz, F. (2020).** Machine Performance Monitoring and Fault Classification using Vibration Frequency Analysis. *IEEE*.

5. **Aqueveque, P. et al. (2021).** Data-Driven Condition Monitoring of Mining Mobile Machinery in Non-Stationary Operations Using Wireless Accelerometer Sensor Modules. *IEEE Access*.

6. **Khandelwal, V. et al. (2022).** Sensor Based Vibration Analysis of Motor Using MATLAB Software. *IEEE*.

7. **Mastang, M. et al. (2023).** Machine Condition Monitoring System Based on IoT Platform for Intelligent Maintenance. *IEEE*.

8. **Banerjee, A., Bhushan, R., & Mukherjee, S. (2024).** Fault Detection and Machine Health Monitoring in Cylinder Block Production Lines through Ignition in Industry 4.0. *IEEE*.