International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 10 | Oct 2026 www.irjet.net p-ISSN: 2395-0072

Real-Time Multi-Sensor Edge Condition Monitoring Architecture with Adaptive Thresholding on STM32

Sachin BM[1], Parshw Sangame[2], Sugureshwara H[3] , Siddharth N[4], Sumit S[5]

[1]*Department of ECE,Bangalore Institute of Technology,Bengaluru,Karnataka,India*
[2345]*VTU,ECE,BIT,Bengaluru,Karnataka,India*

Abstract - *This paper documents a four-channel condition-monitoring prototype built around an STM32 Nucleo-F446RE microcontroller. The prototype interfaces with an SCT-013-030 current transformer and a MAX4466 microphone amplifier through ADC1, an ADXL345 accelerometer through SPI1, and a DS18B20 temperature sensor through a software-driven 1-Wire interface. The firmware emits comma-separated current, vibration, acoustic, and temperature fields over USART2 at a nominal 115200 bit/s for a Python serial-to-browser bridge. Inspection of the supplied firmware shows that the current field is calculated using an unverified ADC-to-current scale factor, the vibration field is the standard deviation of acceleration-magnitude register counts rather than calibrated acceleration in g, and the acoustic field is an ADC-code RMS-like feature rather than voltage or sound-pressure level. Temperature conversion timing is not explicitly checked against the DS18B20 conversion time. The browser dashboard applies its own thresholds and learning calculations; it does not implement physical motor control. These limitations are stated explicitly. No controlled accuracy, fault-detection, false-alarm, or end-to-end latency results are claimed.*

Key Words: Condition monitoring, embedded systems, STM32, sensor acquisition, RMS, current transformer, vibration sensing, edge computing.

1. INTRODUCTION

Condition monitoring uses measured quantities such as electrical current, vibration, acoustic activity, and temperature to observe the operating state of machinery. These measurements can support maintenance decisions when interpreted with suitable operating context and validated against known machine conditions [1]-[3]. A microcontroller-based node can reduce the amount of raw sensor data sent to a host by computing features locally.

This work presents an embedded prototype built around an STM32 Nucleo-F446RE board. Four sensing channels are integrated: current measurement through an SCT-013-030 current transformer, three-axis acceleration through an ADXL345, acoustic signal measurement through a MAX4466 microphone amplifier, and temperature measurement through a DS18B20. The firmware performs acquisition and feature processing on the microcontroller and transmits telemetry over a UART connection to a Python host-side bridge.

The contribution described here is the integration of sensor interfaces, simple time-domain calculations, UART telemetry, and a browser-based display. The implementation is not a validated failure-prediction model. In particular, the names shown in the dashboard should not be interpreted as calibrated engineering units for all channels, and the browser's motor graphic is a visualization rather than a physical motor controller.

2. SYSTEM ARCHITECTURE

The prototype uses an STM32 Nucleo-F446RE development board based on the STM32F446RE microcontroller. The sensor interfaces use analog-to-digital conversion for the current and microphone channels, SPI for the ADXL345, and a GPIO-based 1-Wire interface for the DS18B20. Processed measurements are transmitted over USART2 at a configured baud rate of 115200 bit/s to a Python bridge for host-side display or logging.

The supplied source configures peripheral registers directly and does not configure a sampling timer for the ADC acquisition loops. The UART baud-rate register and microsecond-delay timer also depend on clock assumptions that are not established by the supplied source alone. Therefore, sampling frequency, loop period, and deterministic execution timing are not claimed.

2.1 Current Sensor Interface

The source identifies an SCT-013-030 connected to ADC1 channel 0 (PA0). The code estimates a DC level from one set of 500 ADC conversions, then takes a separate set of 500 conversions and calculates deviations from that earlier mean. It converts the resulting ADC-code RMS-like quantity using ((r_c / 4095) * 3.3) * 30. The supplied material does not include the complete analog schematic, so it is not possible to verify that this scale factor matches the CT sensitivity and any front-end attenuation or gain. For the SCT-013-030 nominal rating of 1 V RMS at 30 A RMS, a calibrated conversion must account for the actual ADC input voltage and analog path. The current output is therefore treated as an unvalidated estimate, not a calibrated current measurement.

2.2 Vibration, Acoustic, and Temperature Interfaces

The ADXL345 communicates over SPI1 and is configured with data-format value 0x0B [4]. The supplied code reads raw axis counts, computes their Euclidean magnitude, and reports the population standard deviation of that magnitude over 128 reads. It does not convert counts to g or subtract a stationary per-axis gravity baseline. The MAX4466 is sampled through ADC1 channel 1 (PA1); the output is an ADC-code RMS-like feature, not voltage or calibrated sound-pressure level.

The DS18B20 is accessed through a software-driven 1-Wire interface on PC0 [5], [6]. The code starts a conversion and attempts to read the scratchpad on the next main-loop iteration, but it does not explicitly wait for conversion completion or verify the required conversion interval. Temperature validity and freshness are therefore not established by the source alone.

3. SIGNAL PROCESSING IN FIRMWARE

3.1 Current Channel

The firmware first calculates a mean ADC code x_bar_1 from N1 = 500 conversions. It then obtains a separate group of N2 = 500 conversions, x2[k], and computes:

$$r_c = \text{sqrt}((1 / N2) * \text{sum}((x2[k] - x_bar_1)^2))$$

The implemented output is:

$$I_{out} = ((r_c / 4095) * 3.3) * 30$$

with values below an ADC-code threshold of 35 forced to zero. This describes the source code, not a verified conversion from amperes to ADC codes. The factor of 30 and the multiplication by 3.3 V require an explicit analog-front-end transfer function and calibration to establish physical units. The two groups are acquired separately, and the firmware does not pace ADC samples with a timer; the effective sampling frequency and captured mains cycles are therefore unknown.

3.2 Vibration Channel

For each of N = 128 samples, the firmware reads raw ADXL345 axis counts X[k], Y[k], and Z[k] and calculates the magnitude:

$$M[k] = \text{sqrt}(X[k]^2 + Y[k]^2 + Z[k]^2)$$

It then computes the mean magnitude M_bar and the feature:

$$V_{feat} = \text{sqrt}((1 / N) * \text{sum}((M[k] - M_bar)^2))$$

This is the population standard deviation of the raw magnitude counts. The code does not convert the axis values to g and does not subtract a per-axis stationary baseline before computing magnitude.

3.3 Acoustic Channel

The acoustic path estimates a mean from the first group of 1000 ADC conversions and computes a root-mean-square deviation over a second group of 1000 conversions relative to that mean. The output is an ADC-code feature. The supplied code does not convert ADC counts to voltage, and no microphone sensitivity or sound-level calibration is provided.

3.4 Temperature Channel

The firmware issues a DS18B20 Convert T command and attempts to read the scratchpad on the next pass through the main loop. Although the source comments describe this as non-blocking, the code does not check conversion completion or guarantee that the required conversion interval has elapsed. It also retains the previous temperature value if a scratchpad CRC check fails.

4. DASHBOARD THRESHOLDING

The supplied HTML dashboard applies threshold logic in the browser to the telemetry values. In manual mode, it estimates a current threshold from a user-entered motor horsepower, nominal 230 V supply, assumed power factor 0.8, and the configured 30 A CT rating. This is a display heuristic: the formula omits motor efficiency and does not establish a valid motor-protection threshold.

The dashboard's learning mode accumulates browser-received values and calculates per-channel mean-plus-three-standard-deviation warning limits and mean-plus-five-standard-deviation critical limits. These are browser-side heuristic thresholds, not validated fault-detection limits. The motor graphic changes its displayed state according to browser-side thresholds. It does not send a stop command to a physical motor or operate a contactor.

TABLE I: IMPLEMENTED INTERFACES & LIMITS

Table -1: Interfaces and Features Implemented in Firmware

Channel / Interface	Firmware Calculation / Acquisition	Interpretation Limit
Current (SCT-013-030, ADC1 PA0)	Two 500-conversion groups; ADC-code deviation RMS-like	Current scale factor and analog front end are unverified

	calculation	
Vibration (ADXL345, SPI1)	128 raw XYZ reads; standard deviation of magnitude in register counts	Not converted to g; no per-axis static-baseline subtraction
Acoustic (MAX4466, ADC1 PA1)	Two 1000-conversion groups; ADC-code RMS-like calculation	Not voltage-calibrated or sound-pressure calibrated
Temperature (DS18B20, 1-Wire PC0)	Start conversion, attempt scratchpad read on next loop iteration	Conversion completion interval is not checked

5. HOST TELEMETRY

The firmware emits four comma-separated numeric fields in this order: current estimate, vibration feature, acoustic feature, and temperature, followed by a newline. The Python bridge opens COM5 at 115200 bit/s, accepts lines containing exactly four comma-separated values that parse as floats, and publishes the most recently parsed values as JSON at /data. The dashboard polls that endpoint approximately every 500 ms.

6. EVALUATION AND LIMITATIONS

Source review identifies several issues that must be resolved before telemetry can be interpreted quantitatively: current scaling is unverified, vibration remains in raw counts, acoustic output remains in ADC codes, ADC sampling is not timer-paced, and DS18B20 conversion completion is not checked. No reproducible reference measurements or labeled fault dataset were supplied. Accordingly, this paper reports no current-accuracy result, calibrated vibration or acoustic result, latency figure, false-alarm rate, or fault-detection accuracy.

7. CONCLUSIONS

This paper documents the supplied STM32 Nucleo-F446RE sensor node, serial bridge, and browser dashboard. Code inspection establishes the four-channel data path but shows key implementation limitations. Future work should correct acquisition and scaling issues, validate each channel against reference instruments, and evaluate threshold logic using labeled healthy and fault conditions. Additional improvements should include noise reduction, reliable sensor calibration, and long-duration testing to assess system stability, measurement accuracy, and fault detection performance under varying operating conditions..

8. ACKNOWLEDGEMENT

The authors acknowledge Sachin BM, Associate Professor, Department of Electronics and Communication Engineering, Bangalore Institute of Technology, for guidance during the major project.

REFERENCES

[1] S. Nandi, H. A. Toliyat, and X. Li, "Condition monitoring and fault diagnosis of electrical motors-a review," IEEE Trans. Energy Convers., vol. 20, no. 4, pp. 719-729, Dec. 2005.

[2] A. K. S. Jardine, D. Lin, and D. Banjevic, "A review on machinery diagnostics and prognostics implemented in condition-based maintenance," Mech. Syst. Signal Process., vol. 20, no. 7, pp. 1483-1510, Oct. 2006.

[3] R. K. Mobley, An Introduction to Predictive Maintenance, 2nd ed. Oxford, U.K.: Butterworth-Heinemann, 2002.

[4] "ADXL345: 3-Axis Digital Accelerometer Data Sheet," Analog Devices, Rev. E, 2015.

[5] "DS18B20: Programmable Resolution 1-Wire Digital Thermometer Data Sheet," Maxim Integrated, Rev. 6, 2019.

[6] "Application Note 126: 1-Wire Communication Through Software," Maxim Integrated, 2002.

[7] ISO 20816-1:2016, Mechanical Vibration - Measurement and Evaluation of Machine Vibration, ISO, 2016.

[8] W. A. Shewhart, Economic Control of Quality of Manufactured Product. New York: Van Nostrand, 1931.

[9] D. C. Montgomery, Introduction to Statistical Quality Control, 7th ed. Hoboken, NJ: Wiley, 2012.

[10] NFPA 70: National Electrical Code, 2023, Table 430.248.

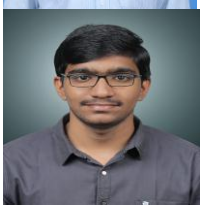
[11] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," ACM Comput. Surv., vol. 41, no. 3, 2009.

[12] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in Proc. IEEE ICDM, 2008, pp. 413-422.

BIOGRAPHIES



"Sachin B. M. is an Assistant Professor in the Department of Electronics and Communication Engineering at Bangalore Institute of Technology, Bengaluru, Karnataka, India.



"Parshw Sangame (1BI23EC140) is an undergraduate student in the Department of Electronics and Communication Engineering at Bangalore Institute of Technology, Bengaluru, Karnataka, India."



"Sugureshwara H (1BI23EC162) is an undergraduate student in the Department of Electronics and Communication Engineering at Bangalore Institute of Technology, Bengaluru, Karnataka, India."



"Sumit S(1BI24EC414) is an undergraduate student in the Department of Electronics and Communication Engineering at Bangalore Institute of Technology, Bengaluru, Karnataka, India."



"Siddharth N(1BI23EC155) is an undergraduate student in the Department of Electronics and Communication Engineering at Bangalore Institute of Technology, Bengaluru, Karnataka, India."