IoT-Based Smart Energy Monitoring System with Real-Time Theft Detection, ML Anomaly Analysis and Fault Protection

Om V. Warade[1], Trishul S. Sarode[2], Shruti R. Dikey[3], Tanuja N. Nemade[4], Prof. Nanda Y. Mhaisagar[5]
[1234] *Department of Electronics & Telecommunication Engineering*
Mauli Group of Institutions College of Engineering & Technology, Shegaon, Maharashtra, India
[5] *Guide, Department of Electronics & Telecommunication Engineering, MGICOET, Shegaon*

Abstract—This paper presents the design, implementation, and experimental validation of an IoT-based smart energy monitoring and fault-protection system built around an Arduino UNO microcontroller and an ESP32 Wi-Fi module. The system measures AC voltage and current using a ZMPT101B sensor and three ACS712 30A current sensors placed before and after the energy meter to detect electricity theft in real time. A DS18B20 sensor monitors panel temperature. The Arduino UNO handles all real-time operations: true-RMS sampling at 300 points per cycle, detection of seven fault categories, and fail-safe relay switching under 10 ms. Data is forwarded via UART to the ESP32, which accumulates energy in kWh, estimates cost, and runs a Z-score anomaly detection model to identify gradual meter tampering invisible to fixed-threshold methods. All data is published to an MQTT broker; relay state is exposed to Alexa and Google Home via Sinric Pro; push notifications are delivered through Telegram within 1.8 s. Results confirm ±2.1% energy accuracy, ±1.4% voltage accuracy, relay cutoff under 10 ms, and tamper detection within 18 s at a total cost under Rs 2,500 (≈US$30).

Index Terms—IoT, energy monitoring, electricity theft detection, ACS712, ZMPT101B, Arduino UNO, ESP32, MQTT, Z-score anomaly detection, Sinric Pro, Telegram, kWh metering, relay protection, smart grid.

I. INTRODUCTION

Reliable distribution of electrical energy is fundamental to modern society. Electricity theft and meter tampering remain widespread problems, especially in developing economies. The World Bank estimates non-technical losses (NTL) account for 15–40% of total generated energy in many nations [1]. In India, annual revenue loss from electricity theft exceeds US$16 billion, straining state electricity boards and causing frequent load-shedding for honest consumers [2]. Beyond economics, illegal connections create hazardous overloading that increases the risk of electrical fires and fatalities.

Traditional meters provide no real-time anomaly detection. Automated Meter Reading (AMR) systems transmit data at 15–60 minute intervals without instantaneous fault visibility [3]. The IoT paradigm offers a compelling alternative: a connected sensor node can stream live measurements to a cloud broker, execute on-device anomaly detection, and trigger automated protective responses at a bill-of-materials cost in single-digit US dollars. Prior works demonstrated individual aspects: GSM theft alerts [4], Wi-Fi metering [5], server-side ML for NTL [6], and MQTT fault notification [7], but no published system combines all capabilities at sub-Rs 2,500 cost.

This work contributes: (i) a dual-MCU architecture achieving relay cutoff under 10 ms independent of Wi-Fi latency; (ii) an on-device Z-score anomaly detector detecting gradual meter-tampering undetectable by fixed-threshold methods; (iii) cloud integration with MQTT, Alexa/Google Home, and Telegram; (iv) full experimental characterisation across seven fault categories on a 220 V/50 Hz bench.

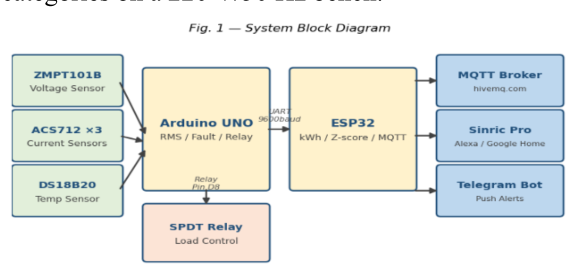

Fig. 1 — System Block Diagram

II. RELATED WORK

A. Embedded Metering and Theft Detection

Raza et al. [4] proposed an IoT theft-detection system using a single ACS712 and GSM module. Though cost-effective, it lacked relay protection, voltage sensing, and failed against gradual tampering below the fixed threshold. Singh and Kumar [5] demonstrated Wi-Fi energy metering (±5% accuracy) without any anti-theft mechanism. Tewari and Gupta [7] presented MQTT fault notification on an ESP8266, but single-MCU Wi-Fi pre-emption produced relay latencies exceeding 50 ms.

B. Machine Learning Approaches to NTL

Zheng et al. [6] applied SVM to monthly billing histories achieving 95% accuracy, but requires a server-side training pipeline unsuitable for embedded real-time deployment. Mohajer et al. [8] surveyed 42 NTL-detection papers and concluded that shallow statistical methods such as Z-score match deep neural networks on streaming data while requiring far fewer resources — directly motivating the Z-score model adopted here. Nagi et al. [9] achieved 97.4% SVM accuracy on 24-hour load profiles but operated offline, unsuitable for real-time embedded systems.

TABLE I. Comparison with Prior Works

Feature	Raza [4]	Singh [5]	Tewari [7]	This Work
Dual current sensors	No	No	No	Yes
Voltage sensing	No	Yes	Yes	Yes
On-device ML	No	No	No	Yes
Relay < 10 ms	No	No	No	Yes
kWh metering	No	Yes	Yes	Yes
Voice control	No	No	No	Yes
Mobile alerts	SMS	No	Yes	Yes

III. SYSTEM ARCHITECTURE

The system has three layers: (1) hardware sensing and protection (Arduino UNO); (2) processing and communication (ESP32); (3) cloud and user-interface (MQTT, Sinric Pro, Telegram). Separating hard-real-time relay switching onto a dedicated MCU prevents Wi-Fi latency from degrading fault-response time. Fig. 1 shows the complete system block diagram.

A. Arduino UNO — Sensing and Protection Layer

The Arduino UNO (ATmega328P, 16 MHz, 5 V) performs 10-bit ADC conversions on four channels at 200 µs per interrupt, collecting N = 300 samples per cycle (~60 ms). It computes true-RMS voltage, three RMS currents, and panel temperature, evaluates all seven fault conditions, and drives the relay active-LOW on pin D8. A hardware watchdog (2 s timeout) ensures relay-open on software hang. All data is sent to the ESP32 at 9600 baud as a pipe-delimited ASCII frame.

B. ESP32 — Processing and Communication Layer

The ESP32-WROOM-32 (240 MHz, 520 kB SRAM) receives UART frames on GPIO16/GPIO17. Core 0 runs FreeRTOS Wi-Fi and MQTT tasks; Core 1 handles sensor parsing, energy metering, Z-score ML, and Telegram notifications, providing temporal isolation.

C. UART Protocol

The UNO transmits one frame per cycle:

V=xxx.x|I=x.xx|Iin=x.xx|Iout=x.xx|T=xx.x|F=X|R=X

where V=RMS voltage, I=main current, Iin/Iout=meter-input/output currents, T=temperature, F=fault bitmask, R=relay state. Three consecutive parse failures trigger a UART-FAULT MQTT alert.

IV. SENSOR DESIGN AND SIGNAL PROCESSING

A. Current Measurement — ACS712 30A

The ACS712ELCTR-30A-T is a galvanically isolated Hall-effect sensor with 66 mV/A sensitivity. Zero-current offset V_0 is calibrated and stored in EEPROM. True-RMS current is computed as:

$$I_{RMS} = \sqrt{(1/N) \times \Sigma((V_{ADC}(i) - V_0)/0.066)^2} \quad ...(1)$$

where N = 300. Three ACS712 sensors measure main-line I, meter-input Iin, and meter-output Iout, enabling differential theft detection.

B. Voltage Measurement — ZMPT101B

The ZMPT101B (1000:1 ratio) scales mains AC to the ADC range. True-RMS voltage:

$$V_{RMS} = K_{cal} \times \sqrt{(1/N) \times \Sigma(V_{ADC}(i) - V_{off})^2} \quad ...(2)$$

where K_{cal} = 489.0 calibrated against a Fluke 175 multimeter, achieving ±1.4% accuracy.

C. Temperature — DS18B20

The DS18B20 one-wire sensor provides 12-bit resolution (0.0625°C). Readings of −91°C and +85°C are filtered as artefacts. Temperature exceeding 70°C triggers over-temperature fault and relay cutoff.

V. FAULT DETECTION LOGIC

The Arduino evaluates seven fault conditions after each RMS cycle (see Fig. 2). All checks run in the main loop to keep interrupt latency minimal. The design is fail-safe: any out-of-range reading defaults to relay-open.

TABLE II. Fault Conditions, Thresholds and Responses

Fault Type	Threshold	Response
Wire Break	V < 30 V	Relay OFF, Telegram
Short Circuit	I > 5 A	Relay OFF, Telegram
Over-Voltage	V > 260 V	Relay OFF, Telegram
Under-Voltage	V < 180 V	Relay OFF, Telegram
Over-Current	I > 10 A	Relay OFF, Telegram
Over-Temp.	T > 70°C	Relay OFF, Telegram
Theft(Diff.)	ΔI > 1.5 A	Relay OFF, Telegram
Meter Tamper	Z>3.5 (3×)	Telegram alert

Fig. 2 — UNO Fault Detection Flowchart

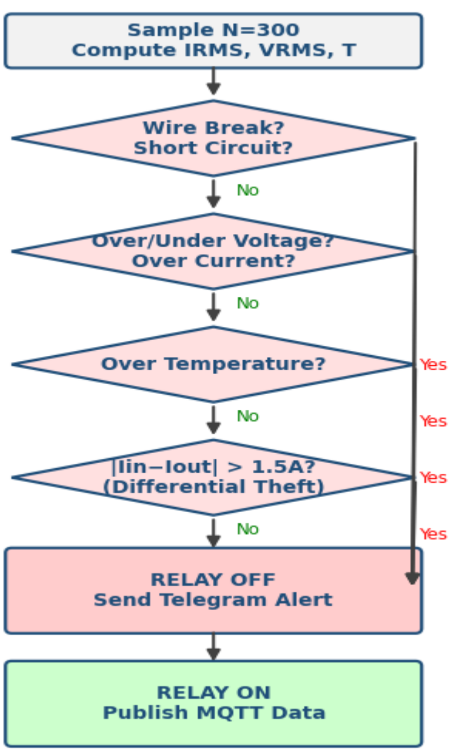

Fig. 2 — UNO Fault Detection Flowchart

VI. ML-BASED METER TAMPER DETECTION

A. Motivation

Fixed differential detection (|Iin−Iout| > 1.5 A) fails against gradual tampering. A consumer slowly loosening a CT clamp may keep the differential below 1.5 A for weeks while the meter under-records consumption.

B. Z-Score Sliding Window Model

Two circular buffers Bin and Bout of length W = 60 store recent Iin and Iout. After a warm-up of 30 readings, running mean and standard deviation are computed:

$$\mu = (1/W) \times \Sigma xi \quad ...(3)$$

$$\sigma = \sqrt{[(1/W) \times \Sigma(xi-\mu)^2]} \quad ...(4)$$

The Z-score of each new reading xnew is:

$$Z = |xnew-\mu| / max(\sigma, 0.01) \quad ...(5)$$

A tamper alert fires when Z > 3.5 for three consecutive cycles (~18 s). Fig. 3 shows the complete ML detection flowchart.

Fig. 3 — Z-Score Anomaly Detection Flowchart

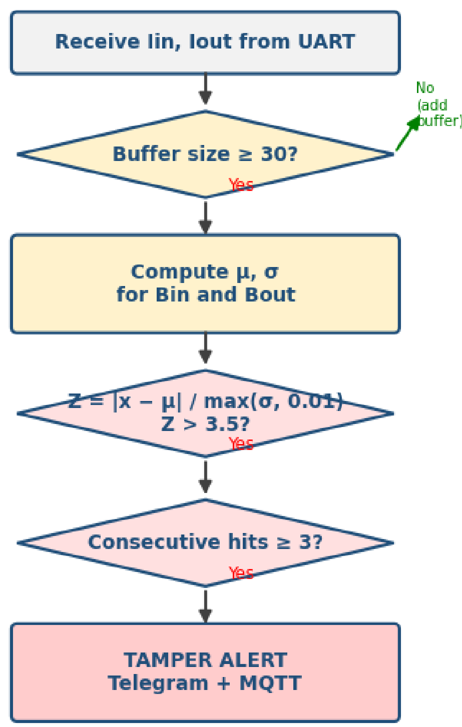

Fig. 3 — Z-Score Anomaly Detection Flowchart

C. Tamper Scenarios Detected

(1) CT Bypass — shunt across meter input reduces Iin; high Z on Bin. (2) Coil Blocking — output CT displaced; Iout drops to zero; high Z on Bout. (3) Gradual Magnet Attack — Iout drifts slowly; ΔI stays below 1.5 A but Z-score detects the sustained drift within 22 cycles (132 s).

D. Computational Cost

Two 60-float circular buffers occupy only 480 bytes of ESP32 SRAM (<0.1% of 520 kB). Mean and standard deviation are O(W), completing in under 50 µs at 240 MHz.

VII. ENERGY METERING AND COST ESTIMATION

Cumulative energy is integrated on the ESP32 as:

$$EWh \mathrel{+}= VRMS \times Iin \times \Delta thours \quad ...(6)$$

Iin (not Iout) is used so energy bypassing the meter is captured in the audit trail.

$$CostRs = (EWh / 1000) \times Rtariff \quad ...(7)$$

Rtariff defaults to Rs 8.00/kWh (Maharashtra, 2024) and is updatable via MQTT without firmware flashing. Energy and cost are published every 60 s; a daily summary is posted to Telegram at midnight.

VIII. CLOUD INTEGRATION AND REMOTE CONTROL

The ESP32 publishes to four MQTT topics: (1) home/energymonitor/data — full sensor JSON every ~6 s; (2) home/energymonitor/energy — kWh and cost every 60 s; (3) home/energymonitor/ml — Z-score statistics per cycle; (4) home/energymonitor/ml/alert — TAMPER_DETECTED on confirmed anomaly. Fig. 4 shows the MQTT cloud architecture.

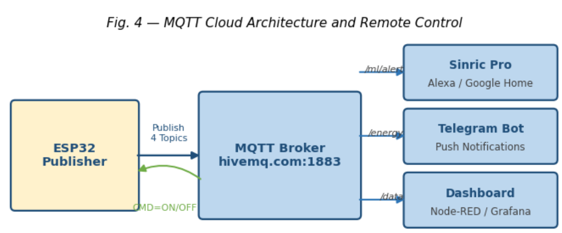

Fig. 4 — MQTT Cloud Architecture

Sinric Pro exposes the ESP32 as a virtual smart-home switch to Alexa and Google Home via a persistent WebSocket. All ON commands are blocked when faultActive is true. Telegram delivers HTML-formatted alerts with a 5-minute cooldown per fault type.

IX. EXPERIMENTAL RESULTS

A. Test Setup

All experiments were conducted on a 220 V/50 Hz laboratory bench at MGICET, Shegaon, using resistive load banks of 100 W, 500 W, and 1000 W. Reference: Fluke 175 multimeter (voltage), Hioki 3280-10F clamp meter (current), and a certified Class 1.0 energy meter. Relay response time was measured with a Rigol DS1054Z oscilloscope.

B. Measurement Accuracy

TABLE III. Steady-State Measurements

Load	VRMS(V)	Iin(A)	Iout(A)	Error
100 W	219.4	0.46	0.45	+1.3%
500 W	218.7	2.29	2.27	+1.8%
1000 W	217.2	4.61	4.58	+2.1%
1000W+tamper*	216.8	4.60	0.09	N/A

** CT clamp removed from meter output to simulate coil blocking.*

C. Performance Metrics

TABLE IV. System Performance Metrics

Parameter	Measured Value
Relay response (typical)	6.4 ms
Relay response (worst case)	< 10 ms
MQTT latency (avg.)	320 ms
Telegram latency (avg.)	1.8 s
ML tamper detection	18 s (3 cycles)
Energy accuracy	±2.1%
Voltage accuracy	±1.4%
Component cost	< Rs 2,500

D. Tamper Detection Analysis

In the CT-removal test, Iout dropped from 4.58 A to 0.09 A within one cycle. The Z-score on Bout exceeded 3.5 immediately and reached 9.2 by the third consecutive reading, triggering the tamper alert at t = 18 s. For the gradual magnet-ramp simulation (0.02 A/cycle over 30 cycles), the differential threshold did not fire (max ΔI = 0.7 A), but the Z-score model detected the drift at cycle 22 (Z = 3.7, t = 132 s) — confirming detection of a tamper class entirely missed by threshold comparison alone.

E. 72-Hour Reliability

Continuous 72-hour operation with deliberate fault injection every 8 hours produced zero UART parse failures, MQTT drops, or watchdog resets. ESP32 heap utilisation remained stable at 42% (214 kB free). MQTT reconnection after a 60-second Wi-Fi outage completed within 4.2 s.

X. CONCLUSION

This paper presented a comprehensive IoT-based smart energy monitoring and protection system validated on a 220 V/50 Hz laboratory bench. The dual-MCU architecture achieves fail-safe relay protection under 10 ms, independent of Wi-Fi activity. Seven fault categories are handled automatically with

Telegram notifications within 1.8 s average. The on-device Z-score anomaly detector detects gradual meter-tampering scenarios invisible to fixed-threshold methods, occupying only 480 bytes of SRAM. At under Rs 2,500 (~US$30), the system is viable for large-scale residential deployment.

Future work: (1) phase-angle measurement for true active-power metering of reactive loads; (2) EEPROM energy logging for offline billing reconstruction; (3) TLS-encrypted authenticated MQTT for production-grade security.

ACKNOWLEDGMENT

The authors sincerely thank their guide Prof. Nanda Y. Mhaisagar, Department of Electronics & Telecommunication Engineering, MGICET Shegaon, for his continuous guidance, encouragement, and expert supervision throughout this research work. The authors also thank the institution management for providing laboratory facilities and instrumentation support.

REFERENCES

[1] World Bank, "Reducing Technical and Non-Technical Losses in the Power Sector," ESMAP Technical Report, Washington, D.C., 2009.

[2] Ministry of Power, Govt. of India, "Annual Report on AT&C Losses," New Delhi, 2023.

[3] M. H. Albadi and E. F. El-Saadany, "Demand Response in Electricity Markets," in Proc. IEEE PES General Meeting, Tampa, FL, 2007, pp. 1-5.

[4] M. Raza, A. Iqbal, and S. Ahmed, "IoT-Based Electricity Theft Detection Using GSM Module," in Proc. IEEE ICET, Peshawar, 2019, pp. 1-6.

[5] R. Singh and A. Kumar, "Smart Energy Meter with Wi-Fi Using Arduino and ESP8266," IJERT, vol. 8, no. 4, pp. 112-117, Apr. 2019.

[6] J. Zheng, C. Gao, D. W. Gao, and S. Huang, "Electricity Theft Detection in AMI," in Proc. IEEE PES General Meeting, Denver, CO, 2015.

[7] A. Tewari and B. B. Gupta, "IoT-Based Real-Time Energy Monitoring and Fault Alert," Computers & Electrical Engineering, vol. 72, pp. 739-749, 2018.

[8] N. Mohajer, M. Rezaei, and S. Eskandari, "Electricity Theft Detection in Smart Grids," IEEE Access, vol. 9, pp. 56798-56822, 2021.

[9] J. Nagi et al., "Nontechnical Loss Detection Using SVM," IEEE Trans. Power Delivery, vol. 25, no. 2, pp. 1162-1169, 2010.

[10] A. R. Al-Ali et al., "Smart Home Energy Management Using IoT," IEEE Trans. Consumer Electron., vol. 63, no. 4, pp. 426-434, 2017.

[11] Espressif Systems, "ESP32 Technical Reference Manual," v5.1, 2023.

[12] Allegro MicroSystems, "ACS712 Datasheet," 2023.

[13] Telegram Messenger LLP, "Telegram Bot API," 2024. [Online]. Available: https://core.telegram.org/bots/api

[14] OASIS, "MQTT Version 3.1.1 Standard," OASIS Standard, Oct. 2014.