A Comparative Analysis of Time Synchronization Techniques for a Hybrid Multi-Rate Power Monitoring System

Ngo Phuong Le
Institute of Science and Technology for Energy and Environment, Vietnam Academy of Science and Technology, Vietnam | School of Electrical and Electronics Engineering, Hanoi University of Industry, Vietnam
ngophuongle@istee.vast.vn (corresponding author)

Nguyen Binh Khanh
Institute of Science and Technology for Energy and Environment, Vietnam
nguyenbinhkhanh@istee.vast.vn

Luong Ngoc Giap
Institute of Science and Technology for Energy and Environment, Vietnam
luongngocgiap@istee.vast.vn

Truong Nguyen Tuong An
Institute of Science and Technology for Energy and Environment, Vietnam
truongnguyentuongan@istee.vast.vn

Bui Tien Trung
Institute of Science and Technology for Energy and Environment, Vietnam
buitientrung@istee.vast.vn

Received: 11 September 2025 | Revised: 3 October 2025 | Accepted: 14 October 2025

 DOI: https://doi.org/10.48084/etasr.14702

ABSTRACT

This study presents a comparative analysis of time synchronization methodologies for a hybrid distributed power measurement system, specifically engineered to facilitate the creation of a high-resolution Non-Intrusive Load Monitoring (NILM) dataset for Vietnam. The system architecture employs a multi-rate approach, combining high-frequency (6.99 kHz) waveform sampling for transient event detection with low-frequency (~1 Hz) power monitoring for appliance state tracking. The primary technical challenge addressed is the achievement of microsecond-level time synchronization across distributed low-cost Internet of Things (IoT) nodes without resorting to specialized, high-cost timing hardware. Four distinct synchronization architectures are evaluated, from a baseline Internet-based Network Time Protocol (NTP) approach to a proposed software-generated Pulse Per Second (PPS) trigger mechanism. The performance of each method is rigorously assessed based on a synthesis of data from established literature and technical benchmarks, focusing on critical metrics including synchronization accuracy, temporal jitter, per-node deployment cost, and implementation complexity. The analysis validates that the proposed software-generated PPS trigger, orchestrated by an NVIDIA Jetson Nano edge computer, provides an optimal balance between these competing factors. It achieves microsecond-level synchronization accuracy (50-200 µs jitter) sufficient for high-fidelity NILM applications, while maintaining a low-cost hardware profile essential for scalable academic research.

Keywords-NILM; time synchronization; PPS; IoT; Jetson

I. INTRODUCTION

Non-Intrusive Load Monitoring (NILM) has significant potential to enhance energy efficiency and enable intelligent grid management. By disaggregating the total energy consumption measured at a single point—typically the main electrical service entry—into appliance-specific profiles, NILM provides granular insights into energy usage patterns without the need for intrusive sub-metering of every device [1]. Because this disaggregation method avoids the need for multiple data recording sensors, it offers a cost-effective approach to forecasting load and reducing demand [2]. This ability empowers consumers with detailed energy audits, facilitates demand-response programs for utility providers, and supports predictive maintenance by identifying anomalous consumption patterns [3].

However, the efficacy of any NILM system is fundamentally constrained by the quality and resolution of its input data. Traditional NILM approaches have historically relied on low-frequency measurements, typically sampling active and reactive power at rates of 1 Hz or lower [4]. Although effective in identifying high-power two-state appliances (e.g., electric water heaters, incandescent lights), these methods face significant challenges in modern electrical environments [5]. The proliferation of electronic devices with Switched-Mode Power Supplies (SMPS), variable-speed motors, and complex control logic results in overlapping and highly variable power signatures that are difficult to distinguish using low-resolution data alone [6]. This limitation has been a critical bottleneck, affecting the accuracy of disaggregation algorithms and preventing the widespread adoption of NILM technologies [5].

To overcome these challenges, the field has increasingly shifted its focus toward high-frequency data acquisition. By sampling voltage and current waveforms at rates in the kilohertz (kHz) range, it is possible to capture the unique electrical "signatures" that appliances imprint on the power line [4]. These signatures are composed of detailed features, such as transient turn-on events and harmonic distortions, which provide a much richer set of discriminative information for machine learning algorithms to leverage, significantly improving disaggregation accuracy [5]. The development of advanced NILM is thus inextricably linked to the ability to capture and analyze these high-fidelity electrical characteristics.

High-frequency sampling reveals two classes of features, invisible to low-frequency meters but critical to distinguishing modern electronic loads. First, transient events capture short-lived behaviors during state changes, such as inrush currents, waveform shapes in the first AC cycles, and Electromagnetic Interference (EMI) from switching. These millisecond-scale signatures, unique to the device topology, enable event-based NILM where changes in the aggregate waveform trigger classification [4]. Resolving them requires kHz-level sampling [5]. Second, harmonic content provides steady-state fingerprints. Linear loads draw sinusoidal currents, while nonlinear loads with SMPS (e.g., computers, TVs, LED lighting) draw distorted pulses. Fourier decomposition reveals harmonics (3rd, 5th, 7th, etc.), whose amplitudes and phases form stable identifiers for appliance types and even models [5, 7]. Thus, NILM has evolved from detecting power steps to recognizing high-frequency spectral and temporal patterns, demanding both high sampling rates and precise timing.

The advancement of NILM algorithms is highly dependent on large, high-quality, publicly available datasets for training and validation. Although several such datasets exist worldwide, there is a notable lack of a comprehensive high-resolution NILM dataset specifically for Vietnam. A localized dataset is critical for developing algorithms tailored to the unique mix of household appliances, power grid characteristics, and energy consumption patterns prevalent in Vietnamese homes. The creation of such a dataset is a key step in fostering NILM research and development within the country.

Although state-of-the-art research has demonstrated high-precision synchronization using hardware-based methods such as GNSS-disciplined clocks or Precision Time Protocol (PTP), these solutions present a significant cost barrier for scalable academic research. In contrast, common low-cost methods based on standard NTP over local networks fail to provide the necessary microsecond-level accuracy. This study addresses this specific methodological gap by identifying a cost-effective yet highly precise synchronization architecture.

The primary obstacle to building a Vietnamese NILM dataset is the significant technical challenge and cost barrier of achieving microsecond-level time synchronization across distributed low-cost IoT measurement devices. This level of synchronization is essential for accurately labeling ground-truth data, where high-frequency waveform samples from an aggregate meter must be precisely correlated with events from individual appliance sub-meters. Any significant time error corrupts the dataset, making it impossible to accurately label transient events or correlate harmonic signatures, which are vital for training advanced machine learning models.

This study aimed to systematically present, evaluate, and compare the performance of four synchronization methods to facilitate the creation of an affordable high-fidelity Vietnamese NILM dataset. The objective was to identify and validate an optimal solution that balances microsecond-level synchronization accuracy with low deployment cost and implementation complexity. The final output is a practical, validated recommendation and blueprint for the data acquisition architecture, specifically tailored to the goal of building the first high-resolution NILM dataset for Vietnam.

II. A HYBRID, MULTI-RATE ARCHITECTURE FOR POWER MONITORING

This work employs a hybrid multi-rate data acquisition architecture to address the dual requirements of capturing high-fidelity waveform data and monitoring multiple appliances in a cost-effective manner, as shown in Figure 1. The system is designed as a hierarchical network with a central time master providing a unified time reference to two distinct types of edge nodes: a high-frequency node for detailed waveform analysis and multiple low-frequency nodes for steady-state appliance monitoring.

On the left, the high-frequency NILM node is responsible for capturing detailed waveform data. It consists of a power meter built with an ADE7953 sensor and an STM32 microcontroller, which interfaces with an NVIDIA Jetson Nano. The Jetson Nano processes the data and provides a Pulse Per Second (PPS) signal to the STM32 for precise synchronization. Data is then forwarded from the Jetson Nano to the central server.

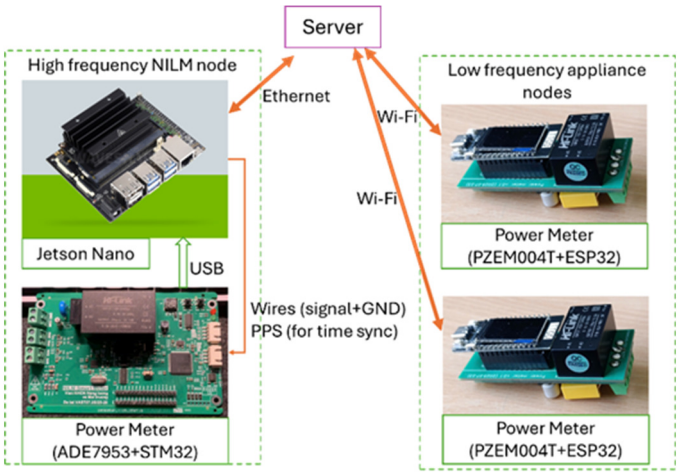

Fig. 1. The hybrid multi-rate power monitoring system

On the right, multiple low-frequency appliance nodes are deployed to monitor individual appliances. Each of these cost-effective nodes consists of a PZEM-004T power meter integrated with an ESP32 microcontroller. These nodes periodically sample power consumption and transmit the data directly to the central server via Wi-Fi. This hybrid approach allows for targeted high-fidelity data collection where it is most needed, while using scalable, low-cost nodes for broader appliance monitoring.

A. System Topology and Rationale for a Hybrid Approach

As shown in Figure 1, the system topology is structured to balance performance with economic pragmatism. The core of the design philosophy is to allocate measurement resources according to the temporal demands of the specific monitoring task. High-frequency sampling is computationally intensive and generates large volumes of data, making it expensive to deploy at every measurement point. In contrast, many appliances have long stable operational states in which their power consumption changes slowly, if at all. For these devices, continuous high-frequency monitoring is unnecessary and inefficient [8].

The proposed hybrid architecture resolves this trade-off. A single, more powerful high-frequency NILM node is deployed at the main electrical input to capture the aggregate signal rich in transient and harmonic information. Concurrently, multiple low-cost, low-power, and low-frequency appliance nodes are deployed on individual circuits or appliances to provide ground-truth labeling and track steady-state power consumption. This approach optimizes the use of hardware, reduces the overall cost of the system, and minimizes data storage and transmission overhead.

Time synchronization for this distributed system is managed hierarchically. The entire local network is synchronized to a Central Time Master, which is synchronized to public Network Time Protocol (NTP) servers. This creates a consistent and accurate time base for all subsequent measurements.

B. The Central Time Master and Data Aggregation Hub

A dedicated server functions as the nexus of the system, performing two critical roles: Time Master and Data Aggregator. The server runs a Chrony NTP daemon as the Time Master. Chrony was selected over the more traditional ntpd due to its demonstrated superior performance under a variety of conditions. It exhibits faster clock convergence, can better compensate for clock frequency drift (slew rate), and is more resilient to variable network latency and intermittent connectivity [9]. These characteristics make it exceptionally well-suited for establishing a highly accurate and stable local Stratum 1 time source, which is essential for achieving microsecond-level synchronization at the edge nodes. The central server synchronizes its clock with a pool of public Stratum 2 servers and then serves as the authoritative time reference for all other devices on the local network.

The server hosts a simple HTTP endpoint as a Data Aggregator. Each edge node, after collecting and timestamping its data, transmits the information to this central server for logging, storage, and subsequent offline analysis. This centralized approach simplifies the data management pipeline and decouples the data acquisition process from the data storage infrastructure.

C. High-Frequency Waveform Acquisition Node

This node is the core of the high-fidelity measurement system, designed specifically to capture the nuanced electrical signatures discussed in Section I. It comprises a sensor, a real-time microcontroller, and an edge computing gateway:, as follows.

- The ADE7953 Energy Metering IC is the front-end of the data acquisition chain, chosen for its ability to perform high-accuracy measurements of active, reactive, and apparent energy, as well as provide access to the raw, sampled voltage and current waveforms [10]. The system leverages it at a high sampling rate of 6.99 kHz to capture a rich set of harmonic data.
- An STM32F407 series microcontroller serves as the real-time data acquisition engine. This component is critical for its deterministic performance. It interfaces directly with the ADE7953 via a Serial Peripheral Interface (SPI) bus, continuously reading waveform samples. The key feature leveraged in this design is its low-latency external interrupt capability. The ARM Cortex-M core can respond to an external interrupt signal in as few as 12 clock cycles, which at a 168 MHz clock speed translates to a response time of approximately 72 ns [11]. This near-instantaneous response is essential for the proposed trigger-based synchronization method. The STM32's role is to buffer a batch of samples upon receiving an external trigger and forward it to the

Jetson Nano via a UART interface (Universal Asynchronous Receiver-Transmitter).

- The NVIDIA Jetson Nano single-board computer acts as the intelligent gateway for the high-frequency node. Its powerful quad-core ARM Cortex-A57 CPU and substantial memory allow it to run a full Linux operating system, manage the NTP client synchronization with the central server, handle the high-throughput data stream from the STM32, and forward the data to the central aggregator. In the proposed synchronization method, the Jetson Nano is responsible for generating the high-precision software-based PPS trigger signal on one of its GPIO pins. The Jetson Nano also serves as the embedded computer for executing the NILM algorithm.

D. Low-Frequency Appliance Monitoring Node

This node is designed as a minimal, low-cost, and easily deployable solution for monitoring the steady-state power consumption of individual appliances, providing the essential ground-truth data to train the NILM algorithm. PZEM-004T is a cost-effective module that measures key electrical parameters, including RMS voltage, RMS current, active power, and cumulative energy. It provides these readings through a simple serial TTL interface at a rate of approximately 1 Hz [12]. Although it cannot provide high-frequency waveform data, its 0.5% accuracy is more than sufficient for tracking the steady-state power draw of most household appliances.

ESP32 is an ideal choice for this application due to its powerful dual-core processor, low cost, and integrated Wi-Fi connectivity [13]. It runs a lightweight client that periodically polls the PZEM-004T sensor. The ESP32's networking stack includes built-in support for SNTP (Simple Network Time Protocol) client functionality, allowing it to easily synchronize its internal Real-Time Clock (RTC) with the central Chrony server over Wi-Fi [14]. Once a measurement is taken and timestamped, the ESP32 transmits the data packet directly to the central data aggregator.

III. COMPARATIVE ANALYSIS OF SYNCHRONIZATION ARCHITECTURES

This section details the four distinct time synchronization methods evaluated in this study. The primary focus is on their ability to meet the stringent microsecond-level requirements of the high-frequency NILM node. For each method, the operational principles are described, followed by a theoretical analysis of the primary sources of timing error and jitter.

A. Method 1: Internet-Based NTP Synchronization

This method is the most straightforward and common approach to time synchronization in non-critical IoT applications. Each edge node (the Jetson Nano and all ESP32 nodes) independently functions as an NTP client, configured to synchronize its system clock directly with public NTP server pools, such as pool.ntp.org, bypassing the local time master. The accuracy of this method is fundamentally limited by the unpredictable nature of the Internet. The NTP algorithm is designed to mitigate network latency by measuring the round-trip time of packets, but it cannot fully compensate for the two primary sources of error over a Wide Area Network (WAN).

- High and Variable Latency: Packets traveling across the Internet traverse numerous routers and switches, each introducing processing and queuing delays. Physical distance alone contributes to a significant and irreducible delay. This results in round-trip times that are typically in tens to hundreds of milliseconds [15].
- Asymmetric Jitter: More problematic than the latency itself is its variability, or jitter. Network congestion, dynamic route changes, and varying queue depths in intermediary network hardware cause the latency of successive NTP packets to fluctuate unpredictably [15]. NTP's filtering algorithms can average out some of this noise over time, but large, sudden variations remain.

Empirical studies of NTP performance across the Internet consistently show that while average accuracy can often be maintained within a few tens of milliseconds, the instantaneous error can be much larger [16]. Given that the interval between consecutive samples at 6.99 kHz is approximately 143 µs, a timing error of several milliseconds makes it impossible to meaningfully correlate waveform data from different nodes. Therefore, this method serves as a crucial baseline to demonstrate the necessity of a local, high-precision time source for this application.

B. Method 2: Local NTP with Gateway Timestamping

This method introduces a local time server to eliminate the unpredictability of the Internet. The Jetson Nano's clock is accurately synchronized to the local Chrony server, achieving a stable and precise time reference with sub-millisecond accuracy relative to the master clock. The High-Frequency NILM Node operates as follows: the STM32 continuously samples data from the ADE7953, batches it, and transmits it via a UART serial connection to the Jetson Nano. An application running on the Jetson Nano reads the incoming serial data and immediately applies a system timestamp using a function such as clock_gettime().

Although the Jetson Nano's internal clock is highly accurate, this method is fundamentally flawed for high-frequency signals because it introduces a significant non-deterministic data path latency. The timestamp applied in the Jetson's user application does not reflect the time the measurement was taken by the sensor, but rather the time it was processed by the application after a long and variable journey. The total end-to-end timing error is a summation of several unpredictable delays, including:

- STM32 Processing Latency: This is the time taken by the microcontroller to read from the SPI bus and place the data in its UART transmit buffer, which is generally small and deterministic.
- UART Transmission Delay: This is the time to physically transmit the bits over the wire, which is deterministic and calculated based on the baud rate.
- Jetson Kernel Latency: This is the largest source of non-determinism. When data arrives at the Jetson's UART, it is

first placed in a kernel buffer. An interrupt is generated, but the Linux kernel's scheduler must decide when to wake up the user application process for this data. This delay is subject to OS scheduler jitter and can be influenced by other running processes, system load, and other interrupts.

- User's code Latency: Once the user process is scheduled to run, there is an additional delay before it executes the read system call and then the clock_gettime() system call.

The cumulative effect of these kernel and user code latencies introduces a jitter that can easily range from tens of microseconds to several milliseconds, especially under system load. This unpredictable jitter completely negates the benefit of having an accurate clock on the Jetson Nano, as the timing relationship between the actual measurement and the applied timestamp is effectively lost. This method is therefore unsuitable for timestamping high-frequency data streams originating from an external, unsynchronized source.

C. Method 3: Software-Generated PPS Trigger Mechanism

This proposed method represents a fundamental architectural shift from the reactive timestamping of Method 2 to a proactive, trigger-based approach. It decouples the task of high-precision timekeeping from the task of real-time data acquisition, assigning each to the hardware component best suited for it. Synchronization is achieved through the following sequence of operations:

- Timekeeping: The Jetson Nano, acting as an NTP client, maintains a highly accurate system clock synchronized to the local Chrony time master.
- Trigger Generation: A dedicated, high-priority user application runs on the Jetson Nano. This application is designed to toggle a GPIO pin at a precise interval, typically on the exact boundary of each second, creating a software-generated PPS signal. The application records the system time (T_{pps}) just before initiating the GPIO toggle.
- Hardware Triggering: The Jetson's PPS GPIO pin is physically wired to an external interrupt-capable pin on the STM32 microcontroller.
- Real-Time Acquisition: The rising edge of the PPS signal triggers an Interrupt Service Routine (ISR) on the STM32. Because the STM32 is a bare-metal/RTOS environment, its response to this hardware interrupt is extremely fast and deterministic [11]. The ISR immediately initiates a pre-configured, high-speed data acquisition cycle, capturing a fixed-size batch of samples from the ADE7953 via SPI at the 6.99 kHz rate.
- Data Transfer: Once the batch is collected, the STM32 transmits the entire block of raw sample data to the Jetson Nano over the UART interface.
- Timestamp Reconstruction: The user application on the Jetson receives the data batch. Crucially, because it knows the precise time the acquisition was triggered (T_{pps}) and the fixed sampling rate ($f_s = 6990\,Hz$), it can accurately calculate the timestamp for every individual sample in the batch using a simple arithmetic formula: $T_{sample\_n} = T_{pps} + \frac{n}{f_s}$, where n is the sample index within the batch (starting from 0).

This architecture elegantly sidesteps the entire data path latency problem of Method 2. The timing of the UART data transfer is no longer critical, as the temporal reference is established by the PPS trigger itself. The dominant remaining source of error is the OS scheduler jitter on the Jetson Nano during the generation of the PPS signal. When the high-priority application attempts to toggle the GPIO pin at a specific moment, its execution can be delayed by the Linux scheduler due to other interrupts or kernel tasks. This means that the actual rising edge of the PPS signal will have a small, random offset from the intended time. The magnitude of this jitter on a standard, non-real-time Linux kernel is the primary performance characteristic to be determined by the empirical evaluation in this study. Based on benchmarks of similar embedded Linux systems, this jitter is expected to be 50–200 µs [17].

D. Method 4: Hardware-Based GNSS PPS Synchronization

This method serves as the benchmark for the highest achievable synchronization accuracy and is considered the "gold standard" against which other methods are compared. It utilizes a dedicated hardware time source: a Global Navigation Satellite System (GNSS) receiver module (e.g., GPS). These modules, in addition to providing location data, output a highly precise PPS signal. The rising edge of this pulse is aligned with the start of the Coordinated Universal Time (UTC) second with nanosecond-level accuracy. In this architecture, the hardware PPS signal is distributed to both the Central Time Master (as a reference clock to discipline Chrony) and directly to the external interrupt pin of each High-Frequency NILM Node's STM32 microcontroller. The timing error in this system is exceptionally low. The primary sources of error are:

- GNSS PPS Jitter: Commercial-grade GNSS receivers typically specify a PPS jitter of less than 100 ns, with many modern modules achieving better than 30 ns RMS accuracy [18].
- Signal Propagation Delay: The delay for the electrical signal to travel from the GNSS module to the STM32 pin. This is deterministic and can be calibrated out.
- STM32 Interrupt Latency: As previously noted, this is a small and highly deterministic delay of ~72 ns.

The cumulative, non-deterministic jitter of this system is therefore dominated by the GNSS receiver itself and is well within the nanosecond range. This provides a virtually perfect trigger for synchronized data acquisition. However, although technically superior, this method's primary disadvantages are its increased per-node cost and implementation complexity. Each high-frequency node requires its own GNSS receiver and antenna, which adds a cost of $11 to over $60 per node. Furthermore, practical deployment requires careful consideration of antenna placement to ensure a clear view of the sky for satellite reception, which can be challenging in indoor or dense urban environments.

IV. PERFORMANCE EVALUATION AND DISCUSSION

A. Quantitative Comparison of Synchronization Methods

The central hypothesis of this study is that Method 3 (Software-Generated PPS Trigger) provides a practical, low-cost approach to microsecond-level synchronization. The dominant error source is the OS scheduler jitter on the Jetson Nano. To quantify this, an end-to-end measurement method was used, which avoids unreliable self-reporting and instead uses external instrumentation, a standard practice for evaluating kernel and interrupt latencies. The primary limitation of the proposed software-PPS method is its reliance on a general-purpose operating system's scheduler, which introduces inherent, non-deterministic jitter.

A dual-channel digital oscilloscope serves as the primary measurement tool. Channel 1 is connected to the Jetson Nano GPIO programmed to output the software PPS signal; the scope is triggered on the rising edge. Channel 2 is connected to a debug GPIO on the STM32, which is set high at the very start of the PPS-triggered ISR.

The Jetson generates a 1 Hz PPS signal. In the Jetson's software, there is a loop that tries to execute a GPIO toggle command at a perfect, regular interval. However, the Jetson is running a non-real-time Linux OS, and the scheduler's job is to juggle hundreds of tasks. Therefore, there can be a variable delay between the intent time and the actual execution time due to the OS scheduler jitter.

The oscilloscope's channels trigger on the rising edge of this physical signal, not measuring the scheduler directly but measuring the timing inconsistency of the physical output, which is the direct result of that scheduler jitter.

Table II summarizes the results from existing methods (1, 2, and 4) and experiments (Method 3), providing an overview of the performance and practical trade-offs associated with each of the four synchronization architectures.

TABLE I. COMPARATIVE EVALUATION OF SYNCHRONIZATION METHODS

Method	Synchronization accuracy (mean error)	Jitter (st.dev. of error)	Implementation complexity
1. Internet NTP	> 10,000 µs (10 ms)	> 5,000 µs (5 ms) [19]	Low
2. Local NTP + gateway timestamping	> 1,000 µs (1 ms)	> 500 µs	Medium
3. Software-generated PPS	< 200 µs [17]	50-200 µs	Medium
4. Hardware-based GNSS PPS	< 0.2 µs (200 ns)	< 0.1 µs (100 ns) [20]	High

B. Discussion

Method 3 delivers a synchronization jitter (50-200 µs) that is worse than Method 4 but significantly better than the flawed gateway timestamping approach (Method 2). This level of accuracy is suitable for NILM research, as it represents a small fraction of a single 50 Hz AC power cycle (20 ms). The ability to achieve this performance without any additional hardware cost makes Method 3 a highly compelling solution, as it dramatically reduces the financial barrier to entry for researchers, enabling the deployment of larger and more comprehensive data acquisition systems and facilitating the creation of rich datasets needed to advance the field of load disaggregation.

Although Method 3 is three orders of magnitude worse than the nanosecond-level jitter of a hardware PPS signal (Method 4), it is crucial to evaluate this performance in the context of actual needs. The primary goal of high-frequency NILM is to analyze phenomena related to the 50/60 Hz AC power line, where a 50 Hz cycle has a period of 20 ms. A worst-case timing jitter of 200 µs corresponds to a phase uncertainty of only $\left(\frac{200\ \mu s}{20000\ \mu s}\right) \times 360° = 3.6°$. This small phase error is more than acceptable for most transient analysis and harmonic decomposition algorithms.

For low-frequency monitoring nodes, temporal requirements are minimal. These nodes track steady-state appliance consumption, which remains stable for minutes or hours [8], with sampling at ~1 Hz (1-second intervals). Standard NTP synchronization over Wi-Fi provides millisecond-level accuracy, easily achieved by an ESP32 with a Chrony server. Even a 10 ms error is only 1% of the sampling interval and has a negligible effect on determining appliance activity or average power.

Using advanced methods such as PPS triggers would be over-engineering—adding cost and complexity without improving data quality. Instead, the use of different synchronization schemes for high- and low-frequency nodes directly reflects the principle of matching synchronization to measurement needs. Thus, the proposed low-cost synchronization approach has some practical implications for NILM research, as it can lower the barrier to entry for innovation in the energy monitoring sector by allowing for rapid and affordable validation of NILM concepts before scaling to more expensive enterprise-grade hardware.

V. CONCLUSION AND FUTURE DIRECTIONS

This study evaluated four time-synchronization methods for hybrid multi-rate power monitoring in NILM, addressing the key challenge of achieving microsecond-level sync for high-frequency (6.99 kHz) data on low-cost hardware. The software-generated PPS trigger (Method 3), using a Jetson Nano to drive an STM32, achieves jitter of 50–200 µs without extra hardware cost, which suffices for transient and harmonic analysis, unlike NTP or gateway timestamping, which suffer from excessive jitters. GNSS PPS offers nanosecond accuracy but is too costly for scalable deployments. Thus, the proposed software PPS approach provides a validated, cost-effective blueprint for high-fidelity NILM research systems.

The main limitation of the software PPS method is scheduler jitter from the standard Linux kernel. Applying the PREEMPT_RT patch could reduce latencies, potentially lowering jitter from 50–200 µs to below 40 µs. Building and deploying PREEMPT_RT on the Jetson Nano is a logical next step, promising near-hardware-level precision without added

cost. For large-scale NILM deployments, distributing physical PPS signals is impractical. PTP (IEEE 1588) enables sub-microsecond sync over networks using hardware timestamping and PTP-aware switches. Although more complex and costly than NTP, PTP offers a scalable path forward. Future work should assess (i) software-only PTP on Jetson Nano and (ii) the cost-benefit relationship of low-cost hardware PTP solutions. This progression from the current validated method to a software-optimized real-time kernel and protocol-level evolution with PTP is a roadmap for future development.

ACKNOWLEDGMENT

This work was funded by the Vietnam Academy of Science and Technology (VAST), code VAST07.05/25-26.

REFERENCES

[1] Y. Liu, Y. Wang, and J. Ma, "Non-Intrusive Load Monitoring in Smart Grids: A Comprehensive Review." arXiv, 2024, https://doi.org/10.48550/ARXIV.2403.06474.

[2] S. Wali, M. H. U. Haq, M. Kazmi, and S. A. Qazi, "An End-to-End Machine Learning based Unified Architecture for Non-Intrusive Load Monitoring," *Engineering, Technology & Applied Science Research*, vol. 11, no. 3, pp. 7217–7222, June 2021, https://doi.org/10.48084/etasr.4142.

[3] P. A. Schirmer and I. Mporas, "Non-Intrusive Load Monitoring: A Review," *IEEE Transactions on Smart Grid*, vol. 14, no. 1, pp. 769–784, Jan. 2023, https://doi.org/10.1109/TSG.2022.3189598.

[4] D. P. B. Renaux *et al.*, "A Dataset for Non-Intrusive Load Monitoring: Design and Implementation," *Energies*, vol. 13, no. 20, Oct. 2020, Art. no. 5371, https://doi.org/10.3390/en13205371.

[5] F. Dinar, S. Paris, and É. Busvelle, "Capturing High-Frequency Harmonic Signatures for NILM: Building a Dataset for Load Disaggregation," *Sensors*, vol. 25, no. 15, July 2025, Art. no. 4601, https://doi.org/10.3390/s25154601.

[6] J. Huchtkoetter and A. Reinhardt, "On the Impact of Temporal Data Resolution on the Accuracy of Non-Intrusive Load Monitoring," in *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, Nov. 2020, pp. 270–273, https://doi.org/10.1145/3408308.3427974.

[7] J. R. Herrero *et al.*, "Non Intrusive Load Monitoring (NILM): A State of the Art," in *Trends in Cyber-Physical Multi-Agent Systems. The PAAMS Collection - 15th International Conference, PAAMS 2017*, Porto, Portugal, 2018, pp. 125–138, https://doi.org/10.1007/978-3-319-61578-3_12.

[8] C. Athanasiadis, D. Doukas, T. Papadopoulos, and A. Chrysopoulos, "A Scalable Real-Time Non-Intrusive Load Monitoring System for the Estimation of Household Appliance Power Consumption," *Energies*, vol. 14, no. 3, Feb. 2021, Art. no. 767, https://doi.org/10.3390/en14030767.

[9] "Key Differences between NTP and Chrony," *Infotechys.com*, Apr. 11, 2025. https://infotechys.com/key-differences-between-ntp-and-chrony/.

[10] "ADE7953 (Rev. C)." [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/ade7953.pdf.

[11] "Measuring Interrupt Latency," *NXP Products.* Available: https://www.nxp.com/docs/en/application-note/AN12078.pdf.

[12] "PZEM-004T-Datasheet-user-manual," *Innovators Guru.* Available: https://innovatorsguru.com/wp-content/uploads/2019/06/PZEM-004T-V3.0-Datasheet-User-Manual.pdf.

[13] Espressif Documentation,' *Espressif.*' https://documentation.espressif.com/esp32_datasheet_en.pdf.

[14] "System Time - ESP32 — ESP-IDF Programming Guide v5.5.1 documentation." *Espressif.* https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/system/system_time.html.

[15] "Jitter vs Latency: Unraveling the Nuances in Network Performance," *LiveAction*. https://www.liveaction.com/resources/blog-post/jitter-vs-latency-unraveling-the-nuances-in-network-performance/.

[16] D. L. Mills, "Measured performance of the Network Time Protocol in the Internet system," RFC Editor, RFC1128, Oct. 1989. https://doi.org/10.17487/rfc1128.

[17] G. K. Adam, "Real-Time Performance and Response Latency Measurements of Linux Kernels on Single-Board Computers," *Computers*, vol. 10, no. 5, May 2021, Art. no. 64, https://doi.org/10.3390/computers10050064.

[18] "GPS Pulse per Second Accuracy", May 24, 2025, https://static.fmad.io/blog-accuracy-of-pps-pulse-per-seccond.html.

[19] D. L. Mills, "Internet Timekeeping Around the Globe," in *Proceedings of the 29th Annual Precise Time and Time Interval Systems and Applications Meeting*, Dec. 1997, pp. 365–372.

[20] X. Niu, K. Yan, T. Zhang, Q. Zhang, H. Zhang, and J. Liu, "Quality evaluation of the pulse per second (PPS) signals from commercial GNSS receivers," *GPS Solutions*, vol. 19, no. 1, pp. 141–150, Jan. 2015, https://doi.org/10.1007/s10291-014-0375-7.