BEANet: An Energy-efficient BLE Solution for High-capacity Equipment Area Network

YIFAN XU, School of Software, Tsinghua University, Beijing, China
FAN DANG and KEBIN LIU, Global Innovation Exchange, Tsinghua University, Beijing, China
ZHUI ZHU, Department of Automation, Tsinghua University, Beijing, China
XINLEI CHEN, Shenzhen International Graduate School, Tsinghua University; Pengcheng Laboratory; RISC-V International Open Source Laboratory, Shenzhen, China
XU WANG, Global Innovation Exchange, Tsinghua University, Beijing, China
XIN MIAO, School of Software, Tsinghua University, Beijing, China
HAITIAN ZHAO, Global Innovation Exchange, Tsinghua University, Beijing, China

The digital transformation of factories has greatly increased the number of peripherals that need to connect to a network for sensing or control, resulting in a growing demand for a new network category known as the Equipment Area Network (EAN). The EAN is characterized by its cable-free, high-capacity, low-latency, and low-power features. To meet these expectations, we present **BEANet**, a novel solution designed specifically for EAN that combines a two-stage synchronization mechanism with a time division protocol. We implemented the system using commercially available Bluetooth Low Energy (BLE) modules and evaluated its performance. Our results show that the network can support up to 150 peripherals with a packet reception rate of 95.4%, which is only 0.9% lower than collision-free BLE transmission. When the cycle time is set to 2 s, the average transmission latency for all peripherals is 0.1 s, while the power consumption is 18.9 μW, which is only half that of systems using LLDN or TSCH. Simulation results also demonstrate that BEANet has the potential to accommodate over 30,000 peripherals under certain configurations.

CCS Concepts: • **Networks** → *Network protocol design*; *Cyber-physical networks;*

Additional Key Words and Phrases: Equipment Area Network, Bluetooth Low Energy, industrial internet

This work is supported in part by the National Key R&D Program of China under grant 2021YFB2900100, the National Natural Science Foundation of China under grant 62302259, 62202263, 62332016, 62272320, and 62371269, Guangdong Innovative and Entrepreneurial Research Team Program No. 2021ZT09L197, Shenzhen 2022 Stabilization Support Program No. WDZC20220811103500001, Tsinghua Shenzhen International Graduate School Cross-disciplinary Research and Innovation Fund Research Plan No. JC20220011 as well as Tsinghua University - Architectural Design and Research Institute Joint Research Center for Synergy and Wisdom Creation of Architecture.
Authors' addresses: Y. Xu and X. Miao, School of Software, Tsinghua University, Beijing, China; e-mails: xuyifan20@mails.tsinghua.edu.cn, miaoxin@tsinghua.edu.cn; F. Dang (Corresponding author), K. Liu, X. Wang, and H. Zhao, Global Innovation Exchange, Tsinghua University, Beijing, China; e-mails: dangfan@tsinghua.edu.cn, kebin@greenorbs.com, wangxu.93@icloud.com, zhaohaitian@msn.com; Z. Zhu, Department of Automation, Tsinghua University, Beijing, China; e-mail: 306812417@qq.com; X. Chen, Shenzhen International Graduate School, Tsinghua University; Pengcheng Laboratory; RISC-V International Open Source Laboratory, Shenzhen, China; e-mails: chen.xinlei@sz.tsinghua.edu.cn.

ACM 1550-4859/2024/02-ART52
https://doi.org/10.1145/3641280

ACM Reference Format:
Yifan Xu, Fan Dang, Kebin Liu, Zhui Zhu, Xinlei Chen, Xu Wang, Xin Miao, and Haitian Zhao. 2024. BEANet: An Energy-efficient BLE Solution for High-capacity Equipment Area Network. *ACM Trans. Sensor Netw.* 20, 3, Article 52 (February 2024), 23 pages. https://doi.org/10.1145/3641280

1 INTRODUCTION

Increasing interconnectivity and intelligent automation bring about radical changes in technology, industries, and social patterns as Industry 4.0 advances [20]. Digital transformation has become the trend for traditional industrial enterprises in order to keep up with the times and increase productivity [11, 43]. In a highly automated, intelligent, and data-driven factory, more fine-grained sensing and control are required [6, 21, 28, 40, 44]. A novel networking category appears where massive connections in a restricted area should be set up.

We conducted a field study on one of the most prestigious auto glass companies worldwide (anonymity due to blind review), which provides services for almost all automobile enterprises, including Benz, Toyota, and Tesla. It has an urgent need for networked factories because a large amount of production data is recorded manually by workers, resulting in a high human cost and error rate. In Figure 1, numerous dial indicators are dispersed across a testing fixture. When a piece of glass is produced by an assembly line, workers randomly select it and use the fixture to measure its dimensions and surface contour. To replace human recording with network transmission, a naive but reliable transformation strategy is to equip each detection point with network cables, which ensures on-time, collision-free data transmission. However, due to the frequent movement of the fixture and the replacement of dial indicators, it would be much easier and cheaper to maintain the fixture if each detection point could be wirelessly connected.

In such a scenario, there may be tens or hundreds of wireless peripherals on a piece of equipment within a small area (e.g., several m^2). Such a dense deployment of wireless nodes is of great need for industry. For instance, inspection fixtures for high-precision part production benefit from dense wireless nodes to enable precise and real-time measurements on each workpiece, ensuring high-quality standards in manufacturing. Robotic arms, when equipped with wireless edge nodes at multiple points, allow for real-time monitoring and control of movement, position, and performance, contributing to enhanced automation efficiency and safety. Conveyor belts, with strategically placed wireless nodes, facilitate continuous tracking of products or materials, enabling efficient logistics management and immediate identification of any anomalies or disruptions in the production line. These applications highlight the need for deploying a large number of wireless edge nodes on various equipment within the factory to achieve detailed monitoring and control, optimizing the performance and reliability of industrial processes. Different from **personal area networks (PANs)** or **local area networks (LANs)**, we refer to this type of network as **Equipment Area Networks (EANs)**, which possess the following characteristics:

- **Cable Free.** No network cable or electrical wire is allowed since the equipment needs dense nodes on the cable for sensing and motion control, as well as minimal maintenance costs and simple construction.
- **High Capacity.** The equipment domain requires the dense deployment of a large number of wireless connections, typically greater than 100 points per square meter.
- **Low Overall Latency.** The EAN must provide low overall latency. To keep up with production, a cycle of sensing measurements and the uploading of data from all connection points must be finished within a few seconds.
- **Low Power.** As each access point is cable-free, it must be powered by a battery. To reduce the frequency of battery replacement, peripherals should operate with low power.

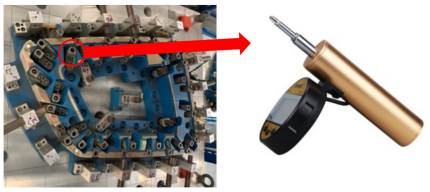


Fig. 1. Testing fixture with dial indicators.

Unfortunately, there is no out-of-the-box solution targeting EANs. Typical commercial low-power wireless technologies include LoRa, NB-IoT, IEEE 802.15.4, and BLE. LoRa achieves low power by reducing the communication frequency [3, 5]. Once we reduce the transmission period, the power consumption also increases [27, 35]. Besides, the capacity of LoRa is also limited [41]. Although NB-IoT provides massive capacity with greater than 52K devices per cell [23], a licensed spectrum and a highly expensive base station are required. Lots of standards and protocols have been put forward to support low-latency deterministic end-to-end communication based on IEEE 802.15.4 or BLE. For instance, Glossy [13] and BlueFlood [26] propose an efficient network flooding and time synchronization mechanism for the mesh networks. However, nodes in the network should pay extra effort for packet relaying, which introduces heavy energy consumption overhead. Beyond that, IEEE 802.15.4e provides time-critical MAC protocols: **Low Latency and Deterministic Networks (LLDNs)** and **Time Synchronous Channel Hopping (TSCH)** [19]. Nevertheless, the frequent synchronization behavior brings considerable additional power consumption, as we discuss later. Therefore, we try to design a more power-saving protocol that fulfills the EAN's requirement for large capacity and low latency as well. Considering that BLE achieves the lowest power consumption and a reasonable data rate, we believe that BLE is a possible candidate for the Equipment Area Network implementation.

To set up an Equipment Area Network with **commercial off-the-shelf (COTS)** BLE devices, we settle the following challenges. **(1) A typical connection-based BLE system suffers from low capacity and relatively high power consumption.** Up to 10 simultaneous connections are usually limited in a COTS system [38]. Moreover, the connection event mode, i.e., the central device polls data, also requires the peripherals to enable their radios and listen before actually sending data. Such extra power consumption makes it less practical in EANs. To overcome this problem, we adopt connection-free communication. **(2) However, self-initiated broadcasting may cause heavy collisions and increase latency.** Experiments show that less than 15% of packets are received when 150 nodes transmit data together every 2 s, causing a high collection latency of 10 s, detailed in Section 5.1. With a **time division multiple access (TDMA)** link layer protocol, we propose BEANet. Each peripheral is precisely synchronized with the central node. A short time slot ensures that hundreds of peripherals transmit data within 1 second, achieving a large capacity. The design also ensures a high **packet reception ratio (PRR)**, the same as a collision-free transmission, and guarantees low latency. **(3) While decreasing the frequency of synchronization to reduce power consumption, high synchronization precision must be maintained.** BLE devices in BEANet rely on 32,768 Hz RC oscillators as their clock source for low-power mode. While RC oscillators offer an advantage in terms of power consumption and cost compared to crystal oscillators, they are not as precise. However, our proposed two-stage synchronization mechanism allows for minimal overhead and compatibility with low-power peripherals. Otherwise, an increase in synchronization frequency is required to improve accuracy and introduces heavy overhead, as with the naive periodic synchronization mechanism used in LLDN and TSCH. In addition to the proposed EAN, the method we designed is equally applicable to

collaborative **Internet of Things (IoT)** scenarios involving multiple devices, such as **Unmanned Aerial Vehicles (UAVs)** and so on [7–9, 33, 42, 46].

To sum up, the contributions are as follows:

- We propose BEANet based on COTS devices, achieving large capacity, low power, and low latency. BEANet is a first step toward the low-power network with small data volume but dense peripheral deployment.
- Instead of increasing the frequency of synchronization, we reconsider the relationship between the frequency and the precision of synchronization in the low-duty cycle scenario. The proposed two-stage synchronization mechanism decreases the frequency by over 40×. It is also practical for other low-power protocols and SoCs with an inaccurate RC oscillator as the clock source.
- We implement BEANet with COTS BLE **Systems-on-Chip (SoCs)**, which achieve hundreds of peripherals to transmit data within seconds. The power of the peripherals in BEANet is 7.8 μW (29.2%) less than the connection-based solution, and only half of those adopting LLDN or TSCH. BEANet has been successfully deployed at the aforementioned auto glass manufacturer. It helps fulfill continuous and flexible data acquisition on the production line while achieving low-power consumption.
- We conduct abundant simulations exploring the performance boundaries of BEANet. Setting the transmission duty cycle to one packet per 64 s, BEANet can potentially accommodate over 30,000 peripherals.

The rest of the article is organized as follows. Section 2 reviews the previous related work. Section 3 puts forward the two-stage synchronization mechanism. We introduce the thorough protocol design in Section 4. Through extensive experimental results, we evaluate the performance of BEANet in Section 5. Finally, we leave a couple of discussions in Section 6 and conclude the article in Section 7.

2 RELATED WORK

Several standards and protocols have been proposed targeting low-power real-time wireless communication.

Standards. WirelessHART [39], ISA SP100.11a [37], and WIA-PA [47] are three standards for industrial wireless communications. They are based on the IEEE 802.15.4 physical layer and adopt a channel-hopping mechanism to ensure the low latency of wireless communication among devices [39]. WirelessHART relies on a centralized scheduling method to enhance transmission dependability, which restricts network capacity to less than 100 peripherals. The IEEE 802.15.4e [18] was proposed as an amendment of the legacy IEEE 802.15.4-2011 standard to satisfy the emerging IoT applications in the industrial domain. The proposed LLDN protocol uses the deterministic TDMA technique to guarantee low-latency deterministic data transmission [29]. However, before transmitting packets in the dedicated time slot, the devices should synchronize to a beacon at the start of the superframe, which results in extra considerable power consumption overhead that is almost as much as sending data packets. TSCH in IEEE 802.15.4e is put forward to operate over a schedule that is based on **Frequency Division Multiple Access (FDMA)** and TDMA modes. Nonetheless, the devices should wait for ACK after data transmission in designated time slots, which causes unnecessary waste of energy for periodical data uploading applications. Other than that, its synchronization frequency is affected by clock error [30]. Implementing TSCH on the SoCs with a less accurate oscillator will result in unacceptable synchronization power overhead, as we analyze in Section 3.3.

Table 1. Comparison among Different Standards and Protocols

Protocol	High Capacity	Low Latency	Low Power	High Precision Clock	Topology
WirelessHART [39]	**Yes**	No guarantee	No guarantee	Yes	Mesh
SP100.11a [37]	**Yes**	No guarantee	No guarantee	Yes	Mesh
WIA-PA [47]	**Yes**	No guarantee	No guarantee	Yes	Mesh
LLDN [18]	**Yes**	**Yes**	**Yes**	Yes	Single Hop
TSCH [18]	**Yes**	**Yes**	**Yes**	Yes	Single Hop
WiDom [31]	No	**Yes**	No	**No**	Single Hop
RT-Link [34]	**Yes**	**Yes**	No guarantee	Yes	Mesh
Flammini [14]	No	**Yes**	**Yes**	Yes	Mesh
WISA [36]	**Yes**	**Yes**	No	Yes	Single Hop
Glossy [13]	**Yes**	**Yes**	No guarantee	Yes	Mesh
BlueFlood [26]	**Yes**	**Yes**	No guarantee	Yes	Mesh
BEANet	**Yes**	**Yes**	**Yes**	**No**	Single Hop

Protocols. WiDom [31] assigns to each device a unique priority. Devices should compete for the channel by sending carrier signals and listening to the channel at the same time before transmission, and the one with the highest priority wins the chance to send data. RT-Link [34] adopts a TDMA-based link layer protocol and specific synchronization hardware to enable collision-free medium access. WiDom [31] and RT-Link [34] require specific hardware design, which increases the cost as well as power consumption. Flammini et al. designed a hybrid MAC layer combining TDMA and **Carrier Sense Multiple Access with collision avoidance (CSMA/CA)** in [14]. It supports at most 16 peripherals, which is far too little for a large-capacity requirement. WISA [36] combines the scheme of TDMA and frequency hopping on the basis of IEEE 802.15.1 PHY. It introduces a heavy synchronization overhead and increases the power consumption to over 1 mW and disobeys the low-power demand. For mesh networks, in order to make network flooding and time synchronization more efficient, Glossy [13] and BlueFlood [26] make concurrent transmissions of the same packet interfere constructively, allowing a receiver to decode the packet even in the absence of capture effects. Other than transmitting their own sensing data, the devices have to spend extra energy on relaying packets, causing a heavy power consumption overhead for the network.

In summary, we have consolidated the existing methodologies and their key attributes in Table 1, shedding light on their limitations while emphasizing the specific areas where our proposed solution excels. Most protocols designed for mesh networks face challenges in simultaneously ensuring low latency and low power, as median devices often incur additional energy consumption or time delays in packet relay processes. Although LLDN and TSCH have demonstrated the ability to deliver high capacity, low latency, and low power concurrently, they rely on precise external crystal oscillators with clock skews below 20 ppm or 40 ppm. In contrast, contemporary low-power chips, such as BLE chips, typically employ less precise RC oscillators as clock sources for their advantages in reduced power consumption and cost. Our proposed system is tailored to be compatible with such devices, aiming to achieve large capacity, low power, and low latency even with cheaper and less high-performance devices.

3 SYNCHRONIZATION FOR LOW-DUTY-CYCLE WIRELESS NETWORKS

To ensure deterministic transmission without conflict of many devices in a short time, we should coordinate them with high synchronization precision. Considering the energy consumption and

Table 2. Important Notations and Descriptions

Notation	Description
G_{1s}	A global second referenced to the broadcaster (central) clock
c_{M1s}	The tick count of the broadcaster clock for one G_{1s}
c_{S1s}	The tick count of the observer (peripheral) clock for one G_{1s}
T_{drift}	The clock drift offset between observer and broadcaster for one G_{1s}
Err_{limit}	The clock drift offset limitation. The observer gets out of sync against the broadcaster if the offset exceeds Err_{limit}
$T_{out\_of\_sync}$	The time it takes for the observer to get out of sync after the last synchronization
T	Stage I of the synchronization process lasts T G_{1s}
e_{Δ}	The frequency jitter of the observer clock
$T_{interval}$	The synchronization interval in Stage II

latency, clock synchronization methods based on message exchange have certain drawbacks. Hence, we design a two-stage synchronization mechanism with zero-message exchange. Table 2 summarizes the essential notations.

3.1 Motivation for Synchronizing with Zero-message Exchange

The synchronization of clocks serves as the foundation for various network applications [45]. Numerous renowned clock synchronization protocols, leveraging message exchange, have been proposed to cater to diverse scenarios' specific requisites, spanning global network synchronization (NTP [25]) to high-precision synchronization (PTP [17]) and resource-limited sensor networks (TPSN [15] and RBS [12]).

However, they share intrinsic disadvantages in energy consumption and latency due to message exchange [10]. On the one hand, it requires the transmission and reception of additional messages between devices, which significantly contributes to energy consumption. In low-power wireless devices where energy efficiency is crucial, minimizing unnecessary message exchanges becomes imperative. Zero-message exchange methods eliminate the need for transmitting extra synchronization messages, resulting in reduced energy consumption and improved battery life. On the other hand, message exchange-based methods introduce additional delays due to the transmission, reception, and processing of synchronization messages. These delays can adversely affect real-time applications that rely on accurate and timely synchronization. Zero-message exchange methods, by their nature, minimize latency by avoiding the overhead associated with message exchange, enabling faster and more efficient clock synchronization.

We are therefore exploring the possibility of designing a synchronization mechanism that involves zero-message exchange. This mechanism contributes significantly to the advancement of clock synchronization techniques and guides the development of efficient and reliable synchronization methods in wireless communication systems.

3.2 Clock Skew in BLE

Before discussing our proposed synchronization mechanism, we first analyze the clock skew, which is a formidable obstacle for precise synchronization in a low-power wireless network due to manufacturing errors [2]. Modern BLE SoCs (e.g., TI CC25xx, Nordic nRF5, Silicon Labs EFR32, Dialog DA14xxx) usually use two different clocks simultaneously to achieve the lowest possible power consumption.

The high-speed clock, typically driven by an external crystal oscillator, is used for processing the BLE stack, while the low-speed clock, typically driven by an internal RC oscillator with an

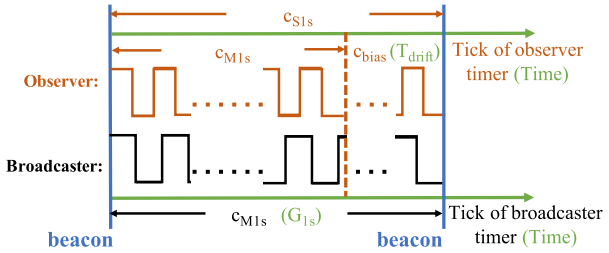


Fig. 2. Clock rate discrepancy between devices: observers' clock runs faster than broadcaster's, resulting in clock drift.

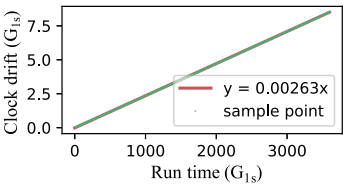


Fig. 3. Clock drift and skew of observer and broadcaster.

ideal frequency of 32,768 Hz, is used for sleeping in a low-power mode. Due to a combination of factors in fabrication, different RC oscillators output frequencies with small offsets from the desired value. Clocks run at slightly different speeds and accumulate offsets among themselves, causing the clock drift phenomenon. The slope of change in drift offset is defined as clock skew. Using the EFR32BG22 BLE SoC that we use in BEANet as an example, we randomly select two BLE modules as the Broadcaster (the device that sends packets) and the Observer (the device that discovers broadcasters). Figure 2 shows the case of an observer with a faster oscillator. The broadcaster periodically sends a beacon every c_{M1s} = 32,768 ticks (blue solid lines), i.e., almost 1 second, and the period of this interval is denoted by G_{1s}(i.e., global 1 second). The observer scans for beacons and records the ticks (in Figure 2) using its own timer when each beacon arrives. The difference between the recorded tick and the broadcaster's tick is denoted by $c_{bias} = c_{S1s} - c_{M1s}$. Accordingly, the clock drift offset for one G_{1s} is $T_{drift} = \frac{c_{bias}}{c_{bias}+c_{M1s}} G_{1s}$. By measurement, the accumulative clock drift is shown in Figure 3. The average clock skew is 2,360 ppm (parts per million), and $T_{drift} = 0.00236 G_{1s}$.

3.3 Naive Mechanism with Zero-message Exchange

A naive mechanism to perform synchronization without message exchange is for peripherals to periodically synchronize their timelines by utilizing beacons broadcast by the central device. In star-topology systems such as the **Code Division Multiple Access (CDMA)** cellular system and the **Global Navigation Satellite System (GNSS)**, it is common for the central device to broadcast its time to the peripherals for synchronization. Continuous synchronizations mitigate clock drift offset. The more frequent the synchronization, the more precise the clock. To guarantee precision, some protocols like LLDN perform synchronization operations shortly before sending each packet, as we mention in Section 2, which introduces great overhead. The TSCH protocol utilizes the same periodic synchronization mechanism as well. The SoCs it targets are equipped with an accurate crystal oscillator with a small clock skew of ±20 ppm, and thus the synchronization interval can be as large as over 20 seconds [30]. However, as demonstrated by our experiment in Section 3.2, the clock skew of the RC oscillator can reach as high as 2,360 ppm, 128 times the crystal oscillator. Suppose the clock drift offset is restricted within Err_{limit}, for instance, $2.5 \times 10^{-3}\ G_{1s}$, and describes the observer getting out of sync with the broadcaster if the offset exceeds Err_{limit}. Then the out-of-sync duration is $T_{out\_of\_sync} = \frac{Err_{limit}}{T_{drift}/G_{1s}} = 1.06 G_{1s}$ and its synchronization interval should be even shorter. Such frequent synchronization (almost as frequent as the rate at which we send data) results in an unacceptable level of power consumption.

3.4 Two-stage Synchronization Mechanism

Synchronization requires the radio on. Therefore, we need to minimize the synchronization frequency as low as feasible via a two-stage process. The broadcaster, i.e., the device with the clock

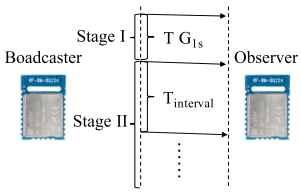


Fig. 4. Overview of two-stage synchronization.

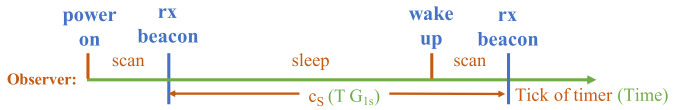


Fig. 5. Observer's timeline during Stage I.

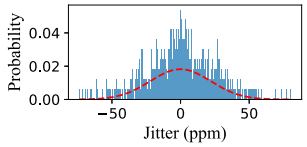


Fig. 6. Distribution of jitter.

source, periodically broadcasts a beacon with a sequence number every c_{M1s} ticks. The synchronization for an observer, i.e., the device to be synchronized, is divided into two stages (Figure 4): Stage I measures the frequency skew with relatively high power consumption and achieves the initial precision. It only needs to be executed once at the start of the observers' running procedure. Then Stage II continuously calibrates the clock with low power consumption and ensures precision.

Stage I. As shown in Figure 5, when an observer powers on, it listens for a coming beacon. Once a beacon is received, it sleeps for a while and scans for the next beacon. The time between these two beacons can be computed from the sequence number, denoted as $T\,G_{1s}$. Assuming that the observer counts c_S ticks during the interval, the observer's clock can be computed as

$$f_{Scomputed} = \frac{c_S}{T}\mathrm{Hz}_G, \tag{1}$$

where Hz_G is the frequency relative to the broadcaster; i.e., we regard G_{1s} as 1 second.

However, Stage I may potentially have small inaccuracies in the calculation. Given that the tick count is an integer, the actual tick count for the interval should fall between $[c_S, c_S + 1)$. In addition, the internal RC oscillator exhibits frequency jitter owing to white noise and temperature fluctuations. Therefore, the actual clock frequency should be

$$f_{Sreal} = \frac{c_S + 0.5}{T} + \frac{c_S}{T} \times e_\Delta \mathrm{Hz}_G \tag{2}$$

on average, where e_Δ describes the frequency jitter with reference to the broadcaster's clock during runtime. The jitter is limited to a modest value of tens of ppm over a short duration, such as dozens of seconds [16]. We measure the frequencies of the observer using a frequency meter with the gate time of 40 s. As depicted in Figure 6, the jitter follows a normal distribution, where the bars represent the probabilities of jitter and the dashed line represents the fitted normal distribution with $\mu = -0.058, \sigma = 21.041$; 99.73% of jitter falls within the range of $(\mu - 3\sigma, \mu + 3\sigma)$. Consequently, each second of the observer's timer sec_S can be denoted by

$$sec_S = f_{Scomputed} \times \frac{1}{f_{Sreal}} G_{1s} = \frac{1}{1 + e_\Delta + 0.5/c_S} G_{1s}. \tag{3}$$

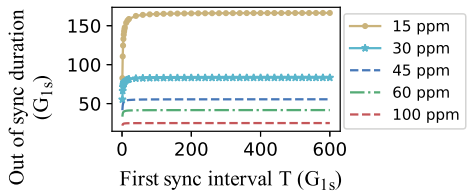


Fig. 7. The simulated out-of-sync duration increases with Stage I interval T, when $Err_{limit} = 2.5 \times 10^{-3}\ G_{1s}$.

The clock drift offset between the observer and the broadcaster for 1 G_{1s} should be

$$\begin{aligned} bias &= |G_{1s} - sec_S| \\ &= \left| \frac{1}{1 + 1/(e_\Delta + 0.5/c_S)} \right| \approx \frac{1}{1 + 1/(|e_\Delta| + 0.5/c_S)} G_{1s}. \end{aligned} \tag{4}$$

Since c_S is of a big value, the offset should be tiny after adjusting the observer's clock. It can be as small as 0.064 ms when $T = 39\ G_{1s}$ and $|e_\Delta| = 63ppm$.

With a relatively high power consumption of 32.4 mW in dozens of seconds (detailed in the "Power and Cost" in Section 5.1), the sub-millisecond synchronization between the observer and the broadcaster is realized.

Stage II. In Stage II, the observer should periodically wake up, listen for beacons, and synchronize to the broadcaster's time. Although the time drift is initially small following the last synchronization in Stage I, it accumulates significantly over an extended period. Furthermore, clock frequency jitter can reach up to 400 ppm over a prolonged duration. Both factors contribute to unsynchronization between the observer and the broadcaster. Thus, the observer has to occasionally synchronize with the broadcaster's clock in Stage II.

Given the limitation of Err_{limit}, the value of out-of-sync duration be computed as

$$\begin{aligned} T_{out\_of\_sync} &= \frac{Err_{limit}}{bias} G_{1s} \\ &= Err_{limit} \left(\frac{1}{|e_\Delta| + \frac{0.5}{c_S}} + 1 \right), \end{aligned} \tag{5}$$

which determines how long the observers should try to synchronize with the broadcaster once again. The smaller the absolute value of jitter $|e_\Delta|$ is, or the longer the Stage I interval T is, the less frequently the observer should conduct synchronizations. Figure 7 shows the relationship between the out-of-sync duration $T_{out\_of\_sync}$ and T with different $|e_\Delta|$, where $Err_{limit} = 2.5 \times 10^{-3}\ G_{1s}$. The out-of-sync duration increases dramatically when T grows and is mainly limited by the jitter. Typically, when T is set to $39\ G_{1s}$ with $|e_\Delta| = 63$ ppm, the observer shall try to synchronize again within $39\ G_{1s}$ after Stage I synchronization, which is much longer than the synchronization interval required in the naive mechanism. To guarantee the precision as well as to reduce the duty cycle, we set $T_{interval}$; i.e., the synchronization interval in Stage II equals $T_{out\_of\_sync}$ according to Equation (5). Once a new beacon is received in Stage II, we readjust the observer's timer according to Equation (1), where c_S refers to the timer count since the last beacon, i.e., ticks during $T_{interval}$.

The synchronization interval is defined by the jitter e_Δ when setting the parameters Err_{limit} and c_S. Assuming a brief period of stable temperature, which is a reasonable expectation in a factory setting, jitter is influenced by both white and 1/f flicker noise sources [16]. Consequently, by employing identical 32,768 Hz RC oscillators as the clock source across all devices in the BEANet, the jitter is anticipated to be within the range of tens of ppm. With our proposed settings, this results in a synchronization interval of dozens of seconds during Stage II. Given the common use

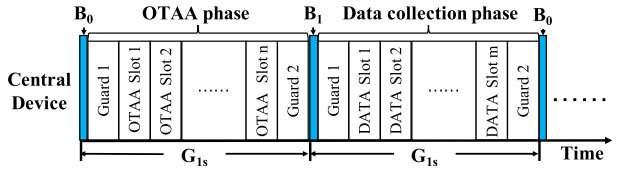


Fig. 8. Overview of communication protocol.

of RC oscillators in clocking low-cost, low-power devices, our proposed synchronization approach demonstrates generalizability.

We then analyze the power consumption overhead brought by the synchronization. In Stage I, the observer does not know the rough beacon interval. Therefore, it has to be in a high-power mode and scan for the beacon for a few seconds twice as depicted in Figure 5. Once Stage I is finished, the observer can predict the next beacon. As Equation (5) has ensured that the clock drift offset is no more than Err_{limit}, the observer only needs to wake up for at most Err_{limit} (milliseconds) before it receives the next beacon in Stage II, which significantly reduces the power consumption.

In contrast to the naive mechanism, the observer initially trades off power consumption but attains initial synchronization precision and clock adjustment in Stage I. Subsequently, the synchronization frequency decreases to 2.6% in Stage II, compared to the naive approach. Consequently, our proposed mechanism significantly reduces the overall synchronization overhead while upholding a consistent level of precision.

4 COMMUNICATION PROTOCOL DESIGN

Based on the synchronization approach, we elaborate on the design of the communication protocol in this section. The network encompasses several peripheral devices (i.e., the observers in synchronization) that report data and a central device (i.e., the broadcaster in synchronization) that provides a global timer, responds to the network access request, and collects data packets from peripheral devices. The communication protocol focuses on the link layer, which incorporates the BLE link layer and thus fits almost all COTS BLE devices. In order to accomplish this, it is mainly concerning channels, packets, and communication methods, including synchronization, **over-the-air activation (OTAA)**, and data collection.

4.1 Overview

Figure 8 depicts the overall protocol design. The central device periodically broadcasts a beacon in two types with an interval of G_{1s}. One of them (B_0) represents the start of the OTAA phase, while the other (B_1) signals the beginning of the data collection phase. Each phase starts and ends with a period of guard time for the central device's advertising/scanning mode transformation. The rest time is divided into n OTAA slots and m DATA slots. The central device communicates with peripheral devices in a TDMA mode.

As we design in Section 3, each peripheral device scans for two beacons with an interval of T G_{1s} at first to adjust its clock frequency according to the global timer. Then it keeps in a cycle of $T_{interval}$ to synchronize to the global timer by scanning for another beacon later on. After receiving the second beacon during the synchronization Stage I, the peripheral device chooses a proper OTAA slot to exchange packets with the central device to get access to the network. The central device should tell the peripheral one the exact DATA slot it should utilize during the data collection phase to transmit data packets in cycles.

4.2 Channel Selection

Since the advertising channels of BLE (channel 37, 38, 39) have been assigned center frequencies that minimize the overlap with the most common 802.11 channel [24], all communication procedures in our system operate on these three channels to mitigate collisions with IEEE 802.11 packets. Generally, data packets should be transmitted continuously in the three channels during each advertisement event. Receiving each of them can be regarded as a successful transmission.

To receive broadcast packets, we set the device to scanner mode with a scanning interval *s_interval*. It cycles through all three advertising channels in a round-robin fashion according to the specification [4]. To reduce the time overhead of the beacon and guarantee synchronization precision, we limit the broadcast of the beacon to channel 37. Even though the packet reception ratio decreases with such a strategy compared with broadcasting on three channels continuously, the peripheral devices can still catch up with the central device soon, since the beacon is broadcast for every G_{1s}.

4.3 Slot Selection

At last, we discuss the slot selection strategy for OTAA and the data collection phase.

OTAA. Assuming that there are n OTAA slots during each phase, the peripheral device selects the ith slot, which is computed as $i = Saddr[0] \pmod{n} + 1$, where $Saddr[0]$ denotes the last byte of its 6-byte device address. If the peripheral device fails to acquire an ACK after sending a request to access the network, it should perform random backoff for some time like CSMA/CA does [48] and try to communicate with the central device in the same OTAA slot during another OTAA phase.

Data Collection. Since the network is pre-configured at the very beginning, the central device has knowledge of the number of peripherals num and their transmission intervals T_{trans}, and the number of DATA slots m for each data collection phase can be calculated as $m = \frac{num \times 2}{T_{trans}}$. As for the jth peripheral that is willing to join the network, the central device should allocate the kth DATA slot in the lth data collection phase for it to upload data during the OTAA procedure, where

$$k = \begin{cases} j + i \times num, i = 0, 1, \ldots, \frac{m}{num} - 1 & T_{trans} < 2 \\ j\%m & T_{trans} \geq 2 \end{cases}$$

and

$$l = \begin{cases} i, i = 1, 2, \ldots & T_{trans} < 2 \\ (j \mid m) + 1 + i \times T_{trans}, i = 0, 1, \ldots & T_{trans} \geq 2. \end{cases}$$

5 EVALUATION

In this section, we present our testing approach to assess the effectiveness of our solution. We begin by implementing our approach on a small set of COTS BLE devices to evaluate its feasibility, as detailed in Section 5.1. Subsequently, we conduct several simulations to explore the performance boundaries of BEANet, as described in Section 5.2. Finally, we provide a comparison between BEANet and previous transmission protocols to demonstrate its superiority.

5.1 Implementation

The following experiments are conducted in our indoor office, which experiences temperature fluctuations throughout the day and has numerous devices, such as laptops and smartphones, transmitting Bluetooth and Wi-Fi packets. Both the central and peripheral devices utilize Silicon Labs EFR32BG22C112F352GM32 [1] SoCs. The central device, benefiting from continuous power, faces no constraints on energy consumption. It remains active throughout, periodically broadcasting

beacons and scanning for incoming packets during the OTAA and data collection phases. Our primary focus is on optimizing the performance of the extensive array of peripherals. The BLE SoCs operate using a 1 Mbps PHY, and the distance between the peripherals and the central device was approximately 2 meters, consistent with factory scenarios.

In the factory, up to 150 dial indicators may be present on a testing fixture, and collecting the measurement data of all indicators within 3 seconds is required to match the production rhythm. Additionally, minimizing the power consumption of each indicator is essential. To meet these industrial requirements, we configure up to 150 peripheral devices to upload data in a cycle of 2 G_{1s}, and each packet carries 9-byte data. As we discuss in Section 4.3, the number of DATA slots for each data collection phase is set to 150. We commence our evaluation by assessing the synchronization precision and versatility of BEANet, followed by a comparative analysis of its performance against two other protocols: self-initiated and naive-sync. In the self-initiated protocol, peripherals autonomously upload data packets to the central device in an ad hoc manner, with a random delay introduced to each 2 s transmission interval to mitigate collisions. For the naive-sync protocol, peripherals employ the naive synchronization mechanism outlined in Section 3.3. To uphold synchronization precision, peripherals scan for beacons every G_s and subsequently adopt the same TDMA data-uploading protocol as described in Section 4. Consequently, the primary distinction between BEANet and naive-sync lies in the synchronization mechanisms they employ. We exclude the typical connection-based BLE system as it cannot handle more than 10 peripherals simultaneously connecting to the central device. We prioritize several performance metrics, including PRR for peripherals, network throughput, transmission latency, overall collection latency, and power consumption. A comprehensive evaluation demonstrates that BEANet excels in achieving a combination of large capacity, low power consumption, and low overall latency. The experiment results are evaluated as follows.

Synchronization Precision

We begin by presenting the synchronization precision of BEANet using the same central device (Broadcaster) and peripheral (Observer) pair as in Section 3. There are 150 DATA slots, and each peripheral's data packet transmission lasts around 0.0016 ms, so the clock drift offset between the central and peripheral devices must be limited to $Err_{limit} = \frac{1/(150+2)-0.0016}{2} \approx 0.0025\ G_{1s}$.

For the experiment setup, the peripheral continues broadcasting in a 2 G_{1s} cycle after Stage I. We then calculate the time bias between the first and subsequent reception points on the central device and identify the out-of-sync moment when the bias exceeds Err_{limit}.

We collect the out-of-sync duration under various T (Stage I duration) settings, as shown in Figure 9. The minimum duration corresponds to the simulation with $|e_\Delta| = 42$ ppm, which accords with the 2-sigma point of the normal distribution [32] in Figure 6. The average duration agrees with the simulation where $|e_\Delta| = 15$ ppm is computed with Equation (5). The minimum duration slightly increases as T grows, but the difference is small under different T configurations. For instance, when T is set to 600 G_{1s}, the minimum out-of-sync duration is 62 G_{1s}, only 20 G_{1s} higher than when T is equivalent to 6 G_{1s}. This indicates that the clock frequency jitter during runtime is the main factor causing peripherals to fall out of sync with the central clock.

In addition, our experiment highlights the advantage of our synchronization mechanism over the naive approach. As described in Section 3.3, the out-of-sync duration in the naive approach is primarily affected by the large clock skew between the central and peripheral devices, which can reach 2,360 ppm for the selected device pair. By contrast, the main influential factor in BEANet, $|e_\Delta|$, is at most around 42 ppm, which is only 19.07% of the devices' original frequency variance. Therefore, the synchronization cycle with our proposed mechanism can be much less frequent, reducing the power consumption overhead of synchronization operations and benefiting long-running low-power devices.

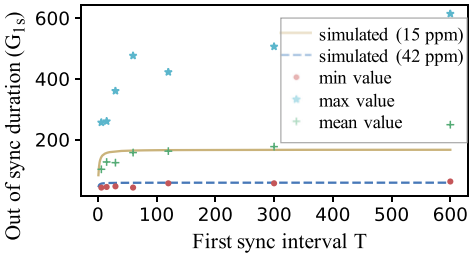


Fig. 9. The out-of-sync duration measured with different Stage I interval settings.

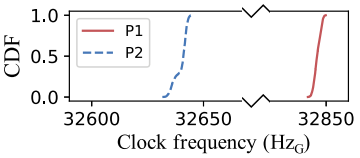


Fig. 10. Distribution of different peripherals' clock frequency.

Specifically, when $T = 39\ G_{1s}$, we set the $T_{interval}$ to be 39 G_{1s}, calculated with $|e_\Delta| = 63$ ppm in Equation (5), which accords with the three-sigma rule [32] for the normal distribution in Figure 6, ensuring that almost all the advertised data packets during the following interval lie within the desired DATA slot. Compared with the 1.06 G_{1s} out-of-sync duration with the naive synchronization mechanism in Section 3.3, BEANet achieves a synchronization precision increase of around 40×, reducing the synchronization frequency. If the inherent gap of clock frequency between the central device and peripherals increases, the strength in synchronization precision of our solution should be highlighted, which is elaborated in the following subsection.

Versatility of BEANet

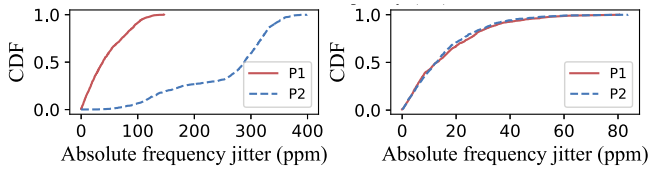
We conduct an experiment to validate the effectiveness of our synchronization solution on various peripherals. Using the settings of $T = 39\ G_{1s}$, $T_{interval} = 39\ G_{1s}$, and $Err_{limit} = 0.0025\ G_{1s}$, we randomly select a peripheral (P2) different from the one (P1) used in previous sections to communicate with the central device. After a 12 h experiment, we found that 99.7% of the data packets broadcast by both peripherals were within the exact DATA slot. The configuration derived from the experiment with P1 is also validated for another randomly picked peripheral.

To gain a deeper understanding of the superiority and versatility of our synchronization mechanism, we analyze the frequency and jitter of the peripheral clocks during runtime. We collect the adjusted frequency computed by the peripherals in each synchronization interval and calculate the long-term jitter between the frequency of the current interval and the first one, as well as the frequency fluctuation between adjacent synchronization intervals as short-term jitter.

Figure 10 presents the peripheral clock frequency during runtime with reference to G_{1s}. P2 had a much slower clock of 32,640 Hz_G on average, with a gap between its clock and the central clock even larger than that of P1. With a naive synchronization mechanism (Section 3.3), P2 would require synchronization every $0.0025/\frac{32,768-32,640}{32,640} = 0.64\ G_{1s}$, which is around 60% of the synchronization cycle required for P1. However, with BEANet, the synchronization cycle we select based on P1 also works for P2, reducing the synchronization overhead by around 66×.

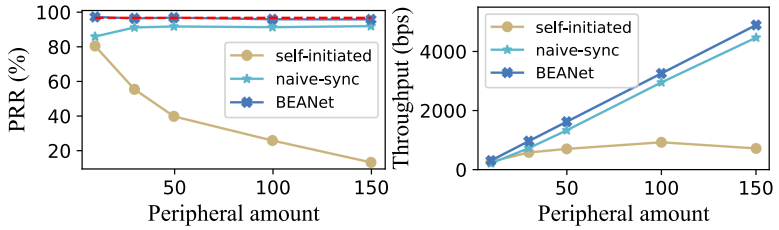
To understand why our solution works, we plot the long-term and short-term clock frequency jitter in Figures 11(a) and 11(b), respectively. Figure 11 shows that in the long run, the frequency fluctuation varies among different peripherals and times, ranging up to 400 ppm due to clock errors or temperature changes. Therefore, we should calibrate the frequency of peripherals occasionally. However, both peripherals exhibit a similar short-term clock frequency jitter distribution, as shown in Figure 11(b), which supports our analysis in Section 3.4.

Our synchronization method can be used in any system where the devices have relatively stable clocks with small jitter and the temperature doesn't change significantly all the time, regardless of the clock skew of different devices.



(a) Long-term frequency jitter (b) Short-term frequency jitter

Fig. 11. Distribution of peripherals' clock frequency jitter, which shows a great difference in the long-term one but little variance in the short-term one.



(a) PRR (b) throughput

Fig. 12. PRR and throughput for different solutions.

Network Capacity

In this section, we examine the capacity of BEANet and compare it with other protocols.

We test the PRR and throughput of each network under various settings of peripheral amount. We initially assess the transmission reliability of a collision-free peripheral in our experimental environment. This involves configuring a single peripheral to periodically upload data packets to the central device for a duration exceeding 12 h. The obtained result demonstrates a 96.3% success rate in packet reception, which we establish as the baseline for PRR. For simplicity, we unify the interval unit to besecond, as one G_{1s} is almost equivalent to 1 second.

Figure 12 shows the PRR and throughput of the network when using the three protocols. For the self-initiated method, the PRR decreases as the number of peripherals increases. When 150 peripherals transmit data with an interval of 2 s, the PRR drops below 15% and the throughput is even lower than that with 100 peripherals. The random delay has a limited effect when many peripherals transmit data with a short interval, as packets are more likely to collide. In contrast, the other two protocols are not affected by the number of peripherals, as precise synchronization enables each peripheral to transmit in separate slots with little conflict.

The PRR of BEANet is almost the same as the baseline, and both PRR and throughput are slightly better than those of the naive-sync mechanism with a 1 s synchronization interval. To ensure synchronization precision, beacons are transmitted only on channel 37, but the peripherals must scan for them in a round-robin fashion, as described in Section 4.2. In the naive-sync solution, it takes peripherals an average of 3 s to receive a beacon, which may cause some peripherals with large clock skew to become out of sync and transmit data packets at the undesired DATA slot. However, the clock skew is eliminated in BEANet, which leads to better performance than naive-sync.

Latency and Collection Duration

Since peripherals broadcast data packets periodically, successful packet reception can be identified as retransmission of previously lost packets. Based on this, we measured the average

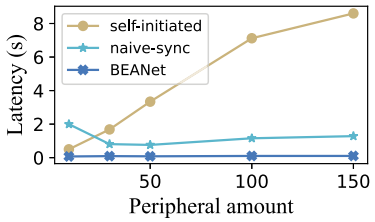
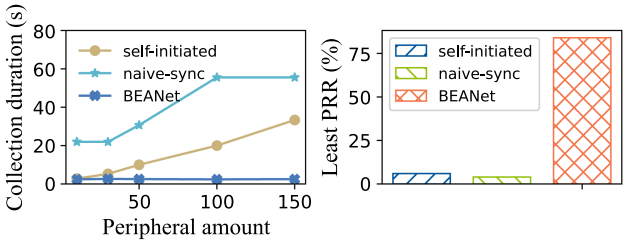


Fig. 13. Average latency.



(a) Collection duration (b) Least PRR of a peripheral

Fig. 14. Collection duration for different solutions.

transmission latency shown in Figure 13. As the number of peripherals increases in the self-initiated mechanism, the PRR decreases, leading to an increase in latency that reaches over 8 s on average when 150 peripherals transmit data. In contrast, there is little difference in latency between the other two mechanisms, even when peripherals pile up, due to their high PRR, as analyzed in the previous section. With a slightly worse PRR, the transmission latency with the naive-sync mechanism is larger than that with BEANet, which provides a mean latency of 0.1 s with up to 150 peripherals.

Another significant indicator of an EAN is the duration for collecting data from all peripherals for one round. Figure 14(a) shows the collection duration for the three mechanisms. BEANet exhibits the shortest duration of around 2.5 s regardless of the number of peripherals. The self-initiated broadcast mechanism shows similar results when the number of peripherals is small, but the duration increases to around 33.3 s, 13 times that of our mechanism, when 150 peripherals communicate with the central device. The performance of the naive-sync is the worst, with about 55 s collection duration.

The collection duration depends on the peripheral that takes the longest time to get one of its data packets received by the central device [22]. Therefore, we analyzed the least PRR among the 150 peripherals when they are set to broadcast at an interval of 2 s and depict the results in Figure 14(b). The peripherals implementing BEANet exhibit a minimum PRR of around 84%, whereas the least PRR of the peripherals that employ a self-initiated broadcast mechanism is around 6% due to intensive collision. The least PRR of the naive-sync peripheral node is even worse, around 4%, because the clock skew between a specific peripheral and central device is so great that the data packets sent by the peripheral fall into the undesired DATA slot and clash with other packets, even though it has just synchronized with the central clock within 3 s. The experiment also demonstrates that BEANet guarantees a collection duration of no more than 2.5 s when there are no more than 150 peripherals with our proposed configurations, which fits the requirement of the factory testing fixture scenario.

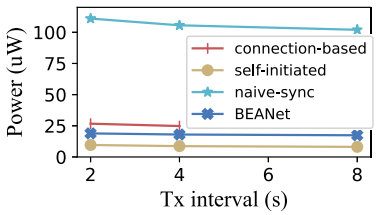


Fig. 15. Average power.

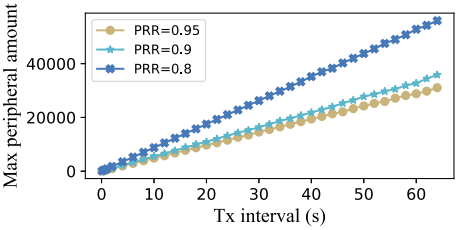


Fig. 16. The maximum number of peripheral devices that BEANet can support under different PRR settings.

Table 3. Parameters for Simulation

Packet Length	Advertisement Interval	Transmission Duration	Bits per Packet
192 µs	220 µs	1.016 µs	72

Power and Cost

By measurement, Stage I in the BEANet synchronization process consumes 32.4 mW in around 39 s, while the power consumption for Stage II along with communication in the OTAA or data collection phase is around 18.9 µW afterward when the broadcast interval is set to 2 s. Considering a lifetime of 3 months for peripherals, which is possible for a common CR2032 battery, the average power is 19.0 µW, and the impact of relatively high power consumption in Stage I is mitigated.

We present the power consumption of our solution against the others in Figure 15. Compared with self-initiated broadcast, BEANet brings a little synchronization overhead of 9.0 µW, while the power consumption for naive-sync grows to around 100.0 µW, 6 times as much as BEANet. In addition, BEANet consumes 7.8 µW less than connection-based BLE communication, which allows at most a 4 s communication interval.

The SoC we use is EFR32BG22, which costs about $1. The expense of building up an EAN with our solution is low.

5.2 Simulation

To explore the performance boundaries of BEANet, we conduct a series of simulations using Python. We emulate a large number of peripheral devices that implement self-initiated broadcast, naive-sync, or BEANet to communicate with the central device. In each round, every peripheral device uploaded hundreds of data packets with a certain transmission interval, and the entire simulation consists of tens of rounds. The simulation employs parameters as detailed in Table 3. Continuous transmission of data packets across the three channels is essential during each advertisement event since one of the channels can be identical to the specific scanning channel of a receiver. The advertisement interval specified in the table represents the channel-switching time within the event. Packet length and advertisement interval measurements are conducted using an oscilloscope. The transmission duration consists of three packet lengths and two advertisement intervals.

For the **self-initiated** mechanism, the central device cyclically scans all three advertising channels with a 20 ms interval. Peripherals introduce a random delay within 10 ms to each transmission interval. In the case of the **naive-sync** and **BEANet** protocols, the central device broadcasts a beacon on a single channel every second and scans the three channels in the rest of time, maintaining a clock frequency of 32,768 Hz. Peripherals implementing **naive-sync** synchronize with the central device every second, uploading data in their respective slots. The clock frequency of each

peripheral is randomly chosen from a normal distribution with $\mu = 32{,}768$ and $\sigma = 107.57$, derived from measurements and fitting with 50 EFR32 devices. The exact time point for each data transmission is then calculated based on both the real clock frequency and the assumed 32,768 Hz clock frequency. In the case of BEANet simulation, with a specific DATA slot number configuration, the required Err_{limit} is calculated using the slot length and the transmission duration. Subsequently, the synchronization interval is calculated with Equation (5) by setting the interval in Stage II equal to the synchronization Stage I interval T to ensure uniform synchronization intervals. Peripherals then transmit data on their designated slots. The clock frequency of each peripheral is set to 32,768 Hz $\times(1 + e_{\Delta})$, where e_{Δ} is randomly selected after each synchronization from the jitter distribution described in Section 3.4. The exact time point for each data transmission is calculated based on both the real clock frequency and the assumed 32,768 Hz clock frequency.

For each data packet transmission, corresponding to a single advertisement event, the order of the three channels is randomly configured. Subsequently, the timing of each packet on each channel is recorded for all peripherals. A packet is identified as a candidate with a reception ratio of 96.3% if it lies on the channel being scanned by the central device and does not overlap in time with other packets on that channel. In the event that none of the three packets during an advertisement event meet the specified requirements, it is categorized as a packet loss event. Based on these criteria, we gathered the PRR of all peripherals and the network throughput for each protocol. Concerning packet transmission, if the aggregated packet reception rate calculated over multiple transmissions surpasses 99.9%, it is considered a successful packet reception. Using this information, we can determine the transmission latency and overall collection duration during the simulation.

Figure 16 shows the maximum number of peripheral devices that BEANet can manage under different PRR settings. The number grows linearly as the transmission interval increases. By lowering the data uploading duty cycle of each device, the unoccupied DATA slot can be allocated to new peripheral devices, thus extending the capacity. Typically, when the transmission interval is set to 2 s, BEANet can accommodate at most around 960, 1,100, and 1,750 peripheral devices, respectively, with a PRR requirement of 95%, 90%, and 80%. However, when the interval decreases to 64 s, the maximum number grows to 31,040, 35,840, and 56,000, respectively.

Next, we compare the performance of the aforementioned three protocols from the aspects of PRR, throughput, latency, and collection duration as shown in Figure 17 to Figure 20. It is worth noting that some points of naive-sync are missing in Figure 19 and Figure 20 because all the packets of a certain peripheral are lost in every single simulation round, making the latency and collection duration incalculable. The self-initiated protocol is even much worse, and we omit its data line in Figure 19 and Figure 20. The simulation result shows little difference from the real implementation. According to the effective data, the self-initiated protocol exhibits the worst performance, with the PRR quickly dropping to nearly zero, resulting in small throughput and large latency. Although naive-sync has similar PRR and throughput as BEANet, BEANet is able to provide much lower and stabler latency and collection duration with an increasing number of peripherals. As discussed in Section 5.1, the peripheral with the worst PRR hampers the network performance when implementing naive-sync. Regarding BEANet itself, there is a tradeoff between network capacity (the maximum number of peripherals that can be held) and network latency (average transmission latency and collection duration). In particular, for scenarios that require nearly real-time data collection (e.g., collection duration < 0.15 s), the transmission interval can be configured to 0.125 s, and the network should accommodate fewer than 70 peripherals. In contrast, for a network with over 30,000 peripherals, it is necessary to lower the transmission interval of each device to be like 64 s, sacrificing the collection duration to around 74 s. According to the simulation, it is possible

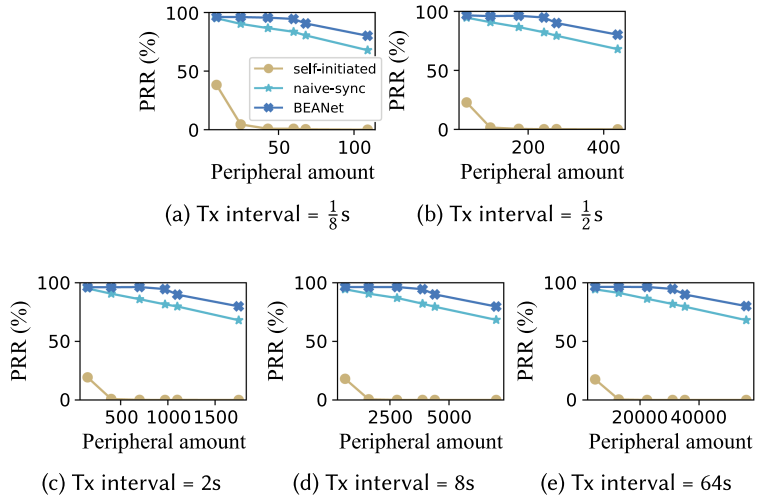


Fig. 17. Simulation of PRR with different transmission intervals for three solutions.

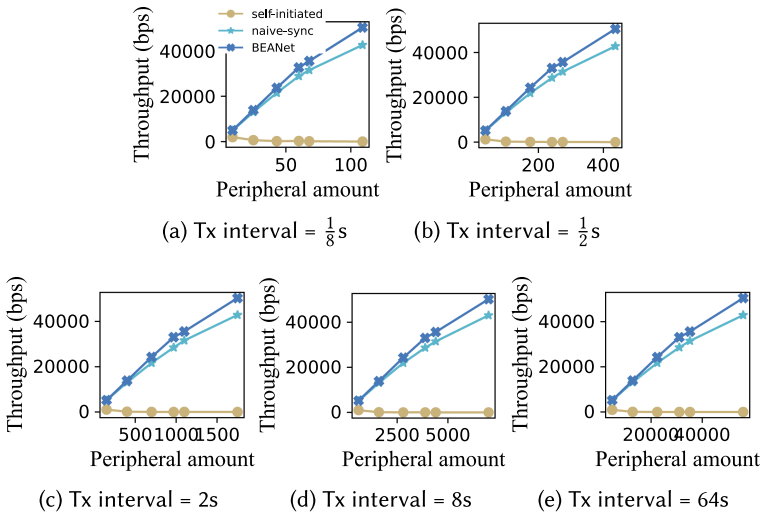


Fig. 18. Simulation of throughput with different transmission intervals for three solutions.

that at most around 1,750 dial indicators with BLE SoCs on a testing fixture implementing BEANet (Tx interval = 2 s) can satisfy the 3-second collection duration requirement of the factory.

5.3 Comparison

At last, we present the comparison between BEANet and other efficient transmission protocols in Table 4.

As for power consumption comparison, in order to eliminate the impact brought by different hardware, we evaluate the duty cycle of peripherals implementing different protocols with an identical 2 s broadcast interval. The peripherals in BEANet synchronize to the central device every 39 G_{1s}. The result exhibits that the duty cycle of the peripherals in BEANet is only half that of the

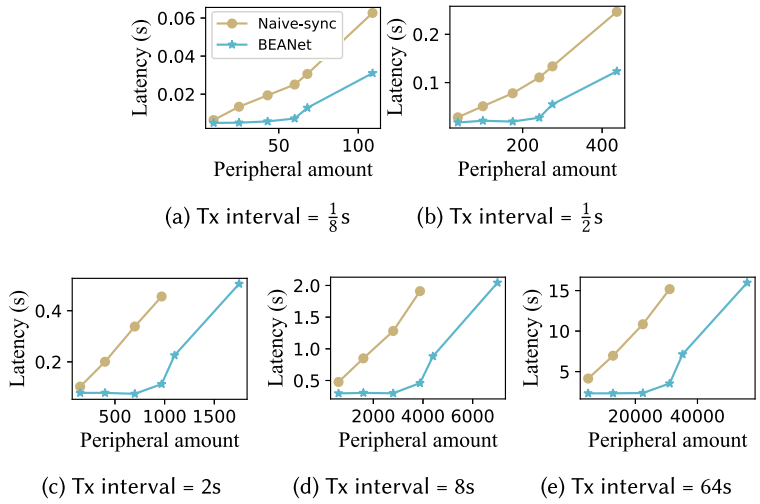


Fig. 19. Simulation of transmission latency with different transmission intervals for three solutions.

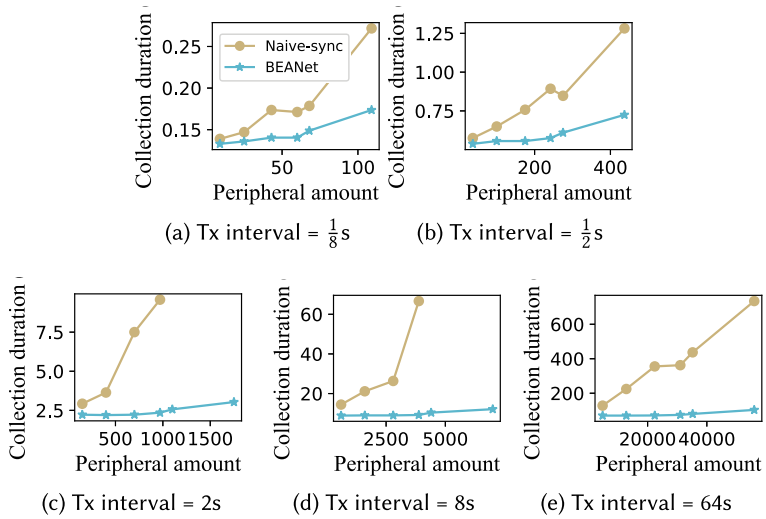


Fig. 20. Simulation of collection duration with different transmission intervals for three solutions.

Table 4. Comparison among Different Protocols with 2 s Broadcast Interval

Protocol	LLDN	TSCH	Glossy	BlueFlood	BEANet
Power consumption (duty cycle)	0.15%	0.18%	0.16%	0.41%	**0.077%**
Oscillator type	crystal	crystal	crystal	crystal	**RC**
Clock accuracy tolerance (ppm)	20	20	20	40	**10,000**
PHY technology	802.15.4	802.15.4	802.15.4	BLE	BLE

other classical mechanisms. As we discuss in Section 2, the synchronization overhead in LLDN is much greater than BEANet, and peripherals implementing TSCH should waste a mass of extra energy waiting for an ACK after each data transmission, while devices with Glossy and BlueFlood have to spend extra energy on relaying packets.

The other protocols rely on a much more precise external crystal oscillator as the clock source, with a clock skew of below 20 ppm or 40 ppm. In contrast, BEANet is able to survive with the less accurate but on-chip RC oscillator, whose clock skew can reach as high as 10,000 ppm in our measurements. If the listed TDMA mechanisms, i.e., TSCH and LLDN, were to use the same RC oscillator as the clock source, their synchronization precision would likely be greatly affected. TSCH would likely have to shorten its synchronization interval, resulting in higher power consumption [30]. LLDN would probably need to add extra guard time to each slot to ensure collision-free transmission, which would decrease capacity and increase power consumption [29]. Using the RC oscillator, with its large clock skew, as the clock source in Glossy or BlueFlood could lead to significant variations in the transmission time that accumulate after several hops at the end of two independent paths, causing destructive interference at the receiver nodes and resulting in reception failure [13].

Compared with them, we design a more efficient synchronization mechanism and a more simplified communication protocol in BEANet, so that in the premise that the requirement of EAN is fulfilled, more energy gets saved.

6 DISCUSSION

Enlarge Network Capacity. A practical approach for factories is to manually number the peripherals from 1 to m. After that, the OTAA phase can be substituted with another data collection phase and a batch of peripherals with a diverse number between 1 and m can directly advertise data packets in corresponding slots after synchronization. With the adjustment, twofold throughput as well as around half collection duration can be attained.

Enhance Transmission Robustness. Considering the overall PRR of over 95% in BEANet, it is adequate to reserve less than 10 slots out of the 150 DATA slots for **group ACK (GACK)** and data retransmission just like LLDN [29] to enhance transmission robustness while sacrificing the capacity and power consumption for a little bit.

Bidirectional Communication. The most traffic in BEANet comes from the unidirectional data collection phase. As for the applications that require bidirectional communication, such as industrial control, periodic global management instructions can be placed on beacon packets. Besides, the peripheral-specific control commands can be transmitted in the corresponding slots as a reply to the upload packets, as the OTAA operation does, or the peripheral may be configured to be in scan mode in a certain slot, waiting for commands.

Security. Currently, BEANet transmits data in plaintext. Since BEANet adopts a non-connection design and the communication is one way for now, key exchange algorithms cannot be applied. In such a private network, pre-shared keys with regularly updated nonces embedded in the beacons may be better. As for authentication, lightweight signature algorithms like ECDSA or EdDSA are possible choices.

7 CONCLUSION

In this work, we first uncover the industrial need for a new type of network named Equipment Area Network whose identity is cable-free, large capacity, low latency, and low power. Then we present BEANet and set up a network with COTS BLE devices to fulfill the requirements of EAN. A key innovation is to improve the synchronization mechanism and greatly reduce the synchronization overhead by around 40 times compared with the naive mechanism. Combined with the TDMA protocol, the network can accommodate over 150 peripherals within 1 G_{1s} and achieve a high PRR of 95.4% on average. When the broadcast cycle is set to 2 G_{1s}, a short collection duration of at most 2.5 G_{1s} is ensured and each peripheral consumes 18.9 μW of power, and the duty cycle of the peripherals is only half of those adopting LLDN or TSCH. The simulation reveals that by

lowering the transmission duty cycle of the peripheral devices to one packet per 64 s, BEANet has the potential to accommodate over 30,000 peripherals.

REFERENCES

[1] [n. d.]. EFR32BG22 Wireless Gecko SoC Family Data Sheet. Retrieved June 27, 2022 from https://www.silabs.com/documents/public/data-sheets/efr32bg22-datasheet.pdf

[2] Hiroki Asano, Tetsuya Hirose, Keishi Tsubaki, Taro Miyoshi, Toshihiro Ozaki, Nobutaka Kuroki, and Masahiro Numa. 2016. A 1.66-nW/kHz, 32.7-kHz, 99.5 ppm/°C fully integrated current-mode RC oscillator for real-time clock applications with PVT stability. In *Proceedings of 42nd European Solid-state Circuits Conference*. IEEE, 149–152.

[3] Aloÿs Augustin, Jiazi Yi, Thomas Clausen, and William Mark Townsley. 2016. A study of LoRa: Long range & low power networks for the Internet of Things. *Sensors* 16, 9 (2016), 1466.

[4] Bluetooth. [n. d.]. Bluetooth® Core Specification. Retrieved June 27, 2022 from https://www.bluetooth.com/specifications/specs/core-specification-5-3/

[5] Martin Bor, John Vidler, and Utz Roedig. 2016. LoRa for the Internet of Things. In *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*. 361–366.

[6] Ryan Brummet, Md Kowsar Hossain, Octav Chipara, Ted Herman, and David E. Stewart. 2021. Recorp: Receiver-oriented policies for industrial wireless networks. *ACM Transactions on Sensor Networks* 17, 4 (2021), 1–32.

[7] Xinlei Chen, Aveek Purohit, Carlos Ruiz Dominguez, Stefano Carpin, and Pei Zhang. 2015. Drunkwalk: Collaborative and adaptive planning for navigation of micro-aerial sensor swarms. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. 295–308.

[8] Xinlei Chen, Aveek Purohit, Shijia Pan, Carlos Ruiz, Jun Han, Zheng Sun, Frank Mokaya, Patric Tague, and Pei Zhang. 2017. Design experiences in minimalistic flying sensor node platform through Sensorfly. *ACM Transactions on Sensor Networks (TOSN)* 13, 4 (2017), 1–37.

[9] Xinlei Chen, Carlos Ruiz, Sihan Zeng, Liyao Gao, Aveek Purohit, Stefano Carpin, and Pei Zhang. 2020. H-DrunkWalk: Collaborative and adaptive navigation for heterogeneous MAV swarm. *ACM Transactions on Sensor Networks (TOSN)* 16, 2 (2020), 1–27.

[10] Fan Dang, Xi-Kai Sun, Ke-Bin Liu, Yi-Fan Xu, and Yun-Hao Liu. 2023. A survey on clock synchronization in the industrial internet. *Journal of Computer Science and Technology* 38, 1 (2023), 146–165.

[11] Christof Ebert and Carlos Henrique C. Duarte. 2018. Digital transformation. *IEEE Software* 35, 4 (2018), 16–21.

[12] Jeremy Elson, Lewis Girod, and Deborah Estrin. 2002. Fine-grained network time synchronization using reference broadcasts. *ACM SIGOPS Operating Systems Review* 36, SI (2002), 147–163.

[13] Federico Ferrari, Marco Zimmerling, Lothar Thiele, and Olga Saukh. 2011. Efficient network flooding and time synchronization with glossy. In *Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks*. IEEE, 73–84.

[14] Alessandra Flammini, Daniele Marioli, Emiliano Sisinni, and Andrea Taroni. 2009. Design and implementation of a wireless fieldbus for plastic machineries. *IEEE Transactions on Industrial Electronics* 56, 3 (2009), 747–755.

[15] Saurabh Ganeriwal, Ram Kumar, and Mani B. Srivastava. 2003. Timing-sync protocol for sensor networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*. 138–149.

[16] Danielle Griffith, Per Torstein Røine, James Murdock, and Ryan Smith. 2014. 17.8 A 190nW 33kHz RC oscillator with ±0.21% temperature stability and 4ppm long-term stability. In *Proceedings of 2014 IEEE International Solid-state Circuits Conference Digest of Technical Papers*. IEEE, 300–301.

[17] IEEE. [n. d.]. IEEE 1588 IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. Retrieved June 27, 2022 from https://standards.ieee.org/ieee/1588/4355/

[18] IEEE. 2012. IEEE standard for local and metropolitan area networks–part 15.4: Low-rate wireless personal area networks (LR-WPANs) amendment 1: MAC sublayer. *IEEE Std 802.15.4e-2012 (Amendment to IEEE Std 802.15.4-2011)*, 1–225. https://doi.org/10.1109/IEEESTD.2012.6185525

[19] Harrison Kurunathan, Ricardo Severino, Anis Koubaa, and Eduardo Tovar. 2018. IEEE 802.15.4e in a nutshell: Survey and performance evaluation. *IEEE Communications Surveys & Tutorials* 20, 3 (2018), 1989–2010.

[20] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. 2014. Industry 4.0. *Business & Information Systems Engineering* 6, 4 (2014), 239–242.

[21] Guorui Li, Sancheng Peng, Cong Wang, Jianwei Niu, and Ying Yuan. 2019. An energy-efficient data collection scheme using denoising autoencoder in wireless sensor networks. *Tsinghua Science and Technology* 24, 1 (2019), 86–96. https://doi.org/10.26599/TST.2018.9010002

[22] Qiang Ma, Zhichao Cao, Wei Gong, and Xiaolong Zheng. 2021. BOND: Exploring hidden bottleneck nodes in large-scale wireless sensor networks. *ACM Transactions on Sensor Networks* 17, 2 (2021), 1–21.

[23] Nitin Mangalvedhe, Rapeepat Ratasuk, and Amitava Ghosh. 2016. NB-IoT deployment study for low power wide area cellular IoT. In *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications*. 1–6.

[24] Microchip. [n. d.]. Bluetooth® Low Energy (BLE) Physical Layer. Retrieved June 27, 2022 from https://microchipdeveloper.com/wireless:ble-phy-layer

[25] David L. Mills. 1985. *Network Time Protocol (NTP)*. Technical Report.

[26] Beshr Al Nahas, Antonio Escobar-Molero, Jirka Klaue, Simon Duquennoy, and Olaf Landsiedel. 2021. BlueFlood: Concurrent transmissions for multi-hop Bluetooth 5–modeling and evaluation. *ACM Transactions on Internet of Things* 2, 4 (2021), 1–30.

[27] Umber Noreen, Ahcène Bounceur, and Laurent Clavier. 2017. A study of LoRa low power and wide area network technology. In *Proceedings of 2017 International Conference on Advanced Technologies for Signal and Image Processing*. IEEE, 1–6.

[28] J. O'Halloran. 2020. Industrial IoT connections to reach 37 billion by 2025. *ComputerWeekly* 3 (2020).

[29] Celia Ouanteur, Djamil Aïssani, Louiza Bouallouche-Medjkoune, Mohand Yazid, and Hind Castel-Taleb. 2017. Modeling and performance evaluation of the IEEE 802.15.4e LLDN mechanism designed for industrial applications in WSNs. *Wireless Networks* 23, 5 (2017), 1343–1358.

[30] Georgios Z. Papadopoulos, Xenofon Fafoutis, and Pascal Thubert. 2020. Multi-source time synchronization in IEEE std 802.15.4-2015 TSCH networks. *Internet Technology Letters* 3, 2 (2020), e148.

[31] Nuno Pereira, Bjrn Andersson, and Eduardo Tovar. 2007. WiDom: A dominance protocol for wireless medium access. *IEEE Transactions on Industrial Informatics* 3, 2 (2007), 120–130.

[32] Friedrich Pukelsheim. 1994. The three sigma rule. *American Statistician* 48, 2 (1994), 88–91.

[33] Jiyuan Ren, Yanggang Xu, Zuxin Li, Chaopeng Hong, Xiao-Ping Zhang, and Xinlei Chen. 2023. Scheduling UAV swarm with attention-based graph reinforcement learning for ground-to-air heterogeneous data communication. In *Adjunct Proceedings of the 2023 ACM International Joint Conference on Pervasive and Ubiquitous Computing & the 2023 ACM International Symposium on Wearable Computing*. 670–675.

[34] Anthony Rowe, Rahul Mangharam, and Raj Rajkumar. 2006. RT-Link: A time-synchronized link protocol for energy constrained multi-hop wireless network. In *Proceedings of the 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, Vol. 2. IEEE, 402–411.

[35] Phui San Cheong, Johan Bergs, Chris Hawinkel, and Jeroen Famaey. 2017. Comparison of LoRaWAN classes and their power consumption comparison. In *Proceedings of 2017 IEEE Symposium on Communications and Vehicular Technology*. IEEE, 1–6.

[36] Guntram Scheible, Dacfey Dzung, Jan Endresen, and Jan Erik Frey. 2007. Unplugged but connected design and implementation of a truly wireless real-time sensor/actuator interface. *IEEE Industrial Electronics Magazine* 1, 2 (2007), 25–34.

[37] Daniel Sexton. 2007. SP100.11a overview. DOE Award DE-FC36-02GO14001, GE Global Research, Research Triangle Park, NC.

[38] Silabs. [n. d.]. Understanding the Bluetooth Connection Process. Retrieved June 27, 2022 from https://docs.silabs.com/bluetooth/2.13/general/connections/understanding-the-bluetooth-connection-process

[39] Jianping Song, Song Han, Al Mok, Deji Chen, Mike Lucas, Mark Nixon, and Wally Pratt. 2008. WirelessHART: Applying wireless technology in real-time industrial process control. In *Proceedings of 2008 IEEE Real-time and Embedded Technology and Applications Symposium*. IEEE, 377–386.

[40] Yifei Sun, Yuxuan Liu, Ziteng Wang, Xiaolei Qu, Dezhi Zheng, and Xinlei Chen. 2022. C-RIDGE: Indoor CO2 data collection system for large venues based on prior knowledge. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*. 1077–1082.

[41] Benny Vejlgaard, Mads Lauridsen, Huan Nguyen, István Z. Kovács, Preben Mogensen, and Mads Sorensen. 2017. Coverage and capacity analysis of Sigfox, LoRa, Gprs, and NB-Iot. In *Proceedings of 85th Vehicular Technology Conference*. IEEE, 1–5.

[42] Haoyang Wang, Xuecheng Chen, Yuhan Cheng, Chenye Wu, Fan Dang, and Xinlei Chen. 2022. H-SwarmLoc: Efficient scheduling for localization of heterogeneous MAV swarm with deep reinforcement learning. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*. 1148–1154.

[43] Weiping Wang, Zhaorong Wang, Zhanfan Zhou, Haixia Deng, Weiliang Zhao, Chunyang Wang, and Yongzhen Guo. 2021. Anomaly detection of industrial control systems based on transfer learning. *Tsinghua Science and Technology* 26, 6 (2021), 821–832. https://doi.org/10.26599/TST.2020.9010041

[44] Ziteng Wang, Chun Hu, Dezhi Zheng, and Xinlei Chen. 2021. Ultralow-power sensing framework for Internet of Things: A smart gas meter as a case. *IEEE Internet of Things Journal* 9, 10 (2021), 7533–7544.

[45] Yifan Xu, Fan Dang, Rongwu Xu, Xinlei Chen, and Yunhao Liu. 2022. LSync: A universal event-synchronizing solution for live streaming. In *Proceedings of the IEEE Conference on Computer Communications 2022*. 2188–2197. https://doi.org/10.1109/INFOCOM48880.2022.9796933

[46] Chenyu Zhao, Haoyang Wang, Jiaqi Li, Fanhang Man, Shilong Mu, Wenbo Ding, Xiao-Ping Zhang, and Xinlei Chen. 2023. SmoothLander: A quadrotor landing control system with smooth trajectory guarantee based on reinforcement learning. In *Adjunct Proceedings of the 2023 ACM International Joint Conference on Pervasive and Ubiquitous Computing & the 2023 ACM International Symposium on Wearable Computing*. 682–687.

[47] Tang Zhong, Cheng Mengjin, Zeng Peng, and Wang Hong. 2010. Real-time communication in WIA-PA industrial wireless networks. In *Proceedings of the 3rd International Conference on Computer Science and Information Technology*, Vol. 2. IEEE, 600–605.

[48] Eustathia Ziouva and Theodore Antonakopoulos. 2002. CSMA/CA performance under high traffic conditions: Throughput and delay analysis. *Computer Communications* 25, 3 (2002), 313–321.

Received 16 August 2023; revised 20 November 2023; accepted 13 January 2024