Time Synchronization of IEEE P1451.0 and P1451.1.6 Standard-based Sensor Networks

Hiroaki Nishi
Faculty of Science and Technology
Keio University
Yokohama, Japan
west@sd.keio.ac.jp

Eugene Y. Song
Engineering Laboratory
National Institute of Standards and Technology (NIST)
Gaithersburg, MD, USA
eugene.song@nist.gov

Yuichi Nakamura
Global Education Center
Waseda University
Shinjuku, Japan
yuichi@aoni.waseda.jp

Kang B. Lee
IEEE Life Fellow
Gaithersburg, MD, USA
Kang.Lee@ieee.org

Yucheng Liu
Department of Electrical Engineering
City University of Hong Kong
Hong Kong, China
yucliu4-c@my.cityu.edu.hk

Kim Fung Tsang
Department of Electrical Engineering
City University of Hong Kong
Hong Kong, China
ee330015@cityu.edu.hk

IECON 2021 - 47th Annual Conference of the IEEE Industrial Electronics Society | 978-1-6654-3554-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IECON48115.2021.9589904

***Abstract*—This paper introduces the time synchronization approaches to the Institute of Electrical and Electronics Engineers (IEEE) P1451.0 standard-based sensor networks for Internet of Things (IoT) applications. A time synchronization architecture of IEEE P1451.0 standard-based sensor networks is described including two-level time synchronization systems in IEEE P1451.0 and P1451.1.X standards-based wide-area network (WAN) and IEEE P1451.0 and P1451.5.X standards-based local area networks (LANs). However, this paper mainly focuses on the time synchronization approach of IEEE P1451.0 and P1451.1.6 standards-based WANs and provides two implementations of time synchronization of IEEE P1451.0 and P1451.1.6 using wireline and wireless networks with their preliminary results to verify that the time synchronization approach of IEEE P1451.1.6 functions properly. In addition, the time synchronization transducer electronic data sheets (TEDS) of P1451.1.6 is described.**

Keywords—IEEE P1451.0, IEEE P1451.1.6, IoT, LAN, MQTT, sensor network, standard, time synchronization, WAN.

I. INTRODUCTION

Internet of Things (IoT) sensors and actuators play a critical role in providing real-time data for monitoring and control of physical infrastructure systems. These sensors and actuators can communicate with the sensor network gateway via standard protocols such as Bluetooth, ZigBee, 6LowPAN, LoRa, Sigfox, NB-IoT, and IEEE 802.11 protocols. The sensor network gateway (sensor data aggregator) provides some functionality, such as pre-processing or data fusion of the sensor data from different sensors, forwarding sensor data and data fusion results to the cloud services, and sharing with IoT applications. Precision time synchronization to a common time reference (e.g., global position system (GPS) is often required among sensors and actuators, the sensor network gateway, cloud services, and IoT applications. Therefore, time synchronization of sensor networks is critical to IoT applications. However, most IoT devices, such as IoT wireless sensors and actuators using microcontrollers and limited resources, do not have a real-time clock module. These sensor nodes have low-power, miniature footprints, and low-cost embedded microcontrollers. Also, time accuracy and time synchronization requirements are based on different IoT applications and time synchronization methods and protocols.

The Institute of Electrical and Electronics Engineers (IEEE) 1451 family of standards for smart transducer interfaces for sensors and actuators define specifications for device-level interfaces ranging from mixed-mode transducers interface (P1451.4), serial wireline (P1451.2) and wireless interfaces (P1451.5), and RFID-to-sensor interface (P1451.7) as shown in Fig. 1. At the network level, the family consists of standards for P1451.1.4 (P21451-1-4, extensible messaging and presence protocol (XMPP)), P1451.1.5 (P21451-1-5, simple network management protocol (SNMP)), and P1451.1.6 (P21451-1-6, message queue telemetry transport (MQTT)) network interfaces, as well as a network interface for the harmonization with other IoT verticals as defined in P1451.99. All of these interfaces are based on the core family of standards, IEEE P1451.0 that is a revision of the IEEE 1451.0-2007 standard based on IoT requirements, such as global identity, security, and time synchronization for IEEE P1451.0-based sensor networks. IEEE P1451.0 define a common function and messaging protocols, common data formats, and metadata such as the transducer electronic data sheets (TEDS) that enable the access of sensors and actuators data and information and pass them to IoT, Industrial Internet of Things (IIoT), and cyber-physical systems (CPS) applications via various network interfaces, through both IEEE 1451 and non-IEEE 1451 networks and systems. The latest revised IEEE P1451.0 standard specifies common functions, network services, transducer services, and TEDS formats for members of the IEEE 1451 family of standards to follow to achieve interoperable with each other in both network interfaces and transducer interface [1]. It defines the common functions and characteristics that are to be performed by a network-capable application processor (NCAP) working as a 1451 server or gateway of IEEE P1451.0 standard-

based sensor networks. It also defines a set of network services application programming interfaces (API) comprised of requests and responses to access sensor data and TEDS data from the NCAP (1451 server) for IoT applications (1451 clients). It also defines common functions and characteristics that are to be performed by a transducer interface module (TIM). In addition, it defines a set of transducer services that include a set of commands and responses to facilitate the setup and control of TIMs as well as reading and writing the data used by the sensor systems. APIs are defined to facilitate communications between the NCAPs and TIMs, and between applications and NCAPs. The IEEE P1451.0 also includes specifications for global identity, security, time synchronization based on IoT requirements, and security and time synchronization TEDS for IEEE 1451.0-based sensor networks. The objective of IEEE P1451.0 is to achieve sensor data interoperability in the network interface and transducer interface. IEEE P1451.1.6 (or P21451-1-6) standard defines a method for transporting IEEE P1451.0 network service messages over a network using MQTT to establish a lightweight, simplified protocol structure to handle IEEE 1451 communications [2].

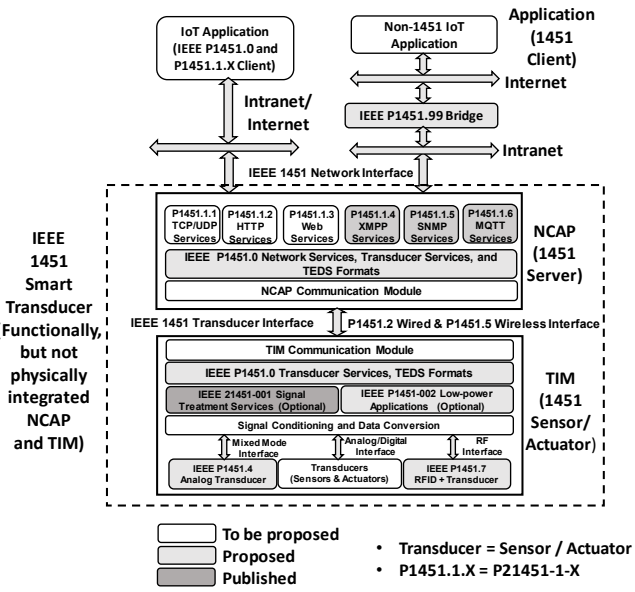


Fig. 1. Architecture of IEEE 1451 family of standards.

This paper mainly focuses on the time synchronization of IEEE P1451.0 and P1451.1.6 standard-based sensor networks for IoT applications. This paper is organized as follows. Related works are described in Section II. The time synchronization of IEEE P1451.0 and P1451.1.6 standard-based sensor networks is described in Section III. An implementation of time synchronization of IEEE P1451.0 and P1451.1.6 WANs with preliminary results to verify functionality is described in Section IV. Summary and conclusion are provided in Section V.

II. Related Works

Ramos et al. implemented the NCAP in a personal computer and using the universal serial bus (USB) to communicate with the TIM. The IEEE 1588 precision time protocol (PTP) protocol is used to synchronize the real-time clock of two TIMs connected to the USB hub. An application to determine the temporal precision of the two modules obtained with the protocol is also described in detail [3]. Wobschall and Ma introduced a method of precise time synchronization of IEEE 1451.0-2007 and 1451.5-6LowPAN standard-based wireless sensor networks using modified IEEE 1588 protocol. A wireless TIM (WTIM) was designed and fabricated using the IEEE 802.15.4 transceiver model TI CC2430 which allows access to a hardware sync signal. The results show that the synchronization precision is better than 10 µs for short synchronization intervals but increases to about 100 µs for longer synchronization intervals [4]. Dueck et al. studied Ethernet-based time synchronization for Raspberry Pi** network based on IEEE 1451-based sensor networks using the IEEE 1588 standard synchronization protocol [5]. As a supplementary to these existing works, the time synchronization of IEEE P1451.0 and P1451.1.6 standards-based WANs is discussed in this paper.

III. Time Synchronization of IEEE P1451.0 and P1451.1.6 Standard-based Sensor Networks

A. Architecture of Time Synchronization of IEEE P1451.0 Standard-based Sensor Networks for IoT Applications

Fig. 2 shows an architecture of time synchronization of IEEE P1451.0 standard-based sensor networks for IoT applications. As shown in Fig. 2, an IEEE P1451.0 and P1451.1.X standards-based WAN consists of several IoT applications and NCAPs via Internet/Intranet (Ethernet or cellular communications). The communications between IoT applications and NCAPs are based on IEEE P1451.0 network services and IEEE P1451.1.X interfaces. Also, an IEEE P1451.0 and P1451.5.X standards-based wireless local area network (WLAN) consists of a number of NCAPs and their respective WTIM via wireless mediums (e.g., 802.11, Bluetooth, ZigBee, 6LowPAN, NB-IoT, SigFox, and LoRa). The communications between the NCAP and WTIMs are based on IEEE P1451.0 transducer services and IEEE P1451.5.X interfaces.

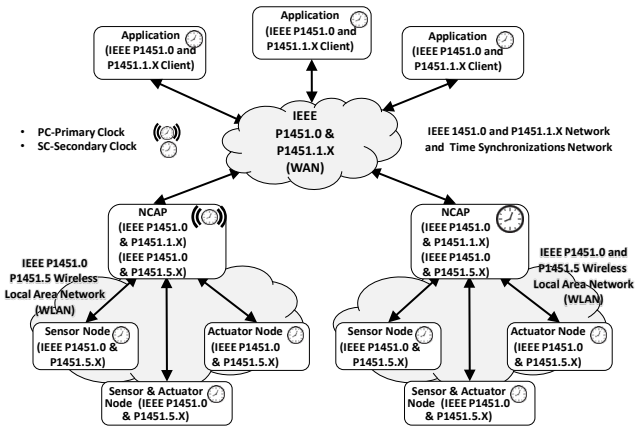


Fig. 2. Architecture of time synchronization of IEEE P1451.0 standard-based sensor networks.

As shown in Fig. 2, the time synchronization protocols for IEEE P1451.0 standard-based WANs and WLANs could be the same or different, depending on application requirements and protocol implementations. For example, applications may require time synchronization to a universal coordinated time

(UTC)-traceable source or a local source. The IEEE P1451.0 standard-based sensor networks should be able to adopt any existing time synchronization standards or protocols, and the adopted time synchronization standard information should be defined in the TimeSync TEDS to achieve sensor data interoperability among these sensor networks. In the IEEE P1451.0 standard-based WANs, all of the secondary clocks, including those in the applications and NCAPs are synchronized to the primary clock (e.g., in one of NCAPs) via one of the following time signals (e.g., GPS, Inter-Range Instrumentation Group - Time Code Format B (IRIG-B), pulse-per-second (PPS), or others). In the IEEE P1451.0 standard-based WLANs shown in Fig. 2, all of WTIMs (wireless sensor nodes or actuator nodes) that are secondary clocks should be synchronized with the NCAP (primary clock), which means that time synchronization messages should be communicated or exchanged via the same IEEE P1451.0-based WAN and IEEE P1451.5-based WLANs.

TEDS are electronic datasheets containing metadata in a standardized data format that describe the characteristics of sensors and actuators associated with the TIMs and NCAPs in the form of transducer channels. The structures of different types of TEDS are described in the P1451.0 standard [1]. They are intended to be stored in nonvolatile memory within a TIM or NCAP. However, there are applications in which this is not practical, so the standard allows them to be stored in other places in the user's system. When stored in some location other than the TIM, they are referred to as "virtual" TEDS. The TimeSync TEDS is optional. The function of the TimeSync TEDS shall be to make available at the interface all of the information needed to gain access to any NCAPs and TIMs/WTIMs. TimeSync TEDS, defined as a text-based TEDS, describes time synchronization protocol names and versions in the standard [1]. Table I shows the details.

TABLE I. TIME-SYNC TEDS

Field type	Field name	Description	Data type	# octet	M/O
—		Length	UInt32	4	M/O
0-2	—	Reserved	—	—	
3	TEDSID	TEDS Identification Header	UInt32	4	M
4-9	—	Reserved	—	—	
Time synchronization related information					
10	Name	Time Synchronization Protocol Name	UInt8	1	M
11	Version	version	UInt8	1	M
21-127	—	Reserved	—	—	
128-255	—	Open to manufacturers	—	—	
—		Checksum	UInt16	2	M

B. Time Synchronization of IEEE P1451.0 and P1451.1.6 standards-based Network

Fig. 3 shows the time synchronization of IEEE P1451.0 and P1451.1.6 standards-based network that consists of many IoT applications, an IEEE P1451.1.6 MQTT broker, and many NCAPs via Internet/Intranet (Ethernet or cellular communication). The communications between IoT applications and NCAPs are via the MQTT broker based on IEEE P1451.0 network services and IEEE P1451.1.6 MQTT interfaces. In general, any node in the network can be a primary clock. In Fig. 3, one NCAP is a primary clock, while the other NCAPs and applications are secondary clocks. All secondary clocks should be synchronized with the primary clock based on the time synchronization protocol defined in Clause 6 of IEEE P1451.1.6 [2].

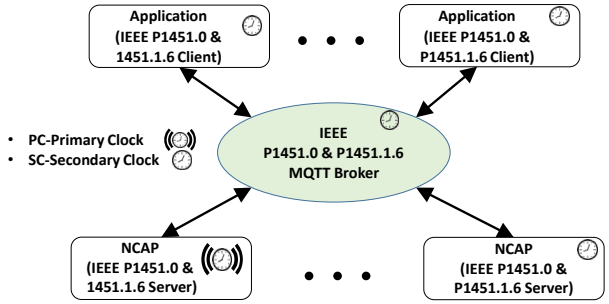


Fig. 3. Time synchronization of IEEE P1451.0 and P1451.1.6 standards-based WAN.

MQTT systems are based on publisher and subscriber structure, with data exchanging via the MQTT broker. MQTT v5 supports a property that enables extensive data inclusion into a message independent of the original message body. It provides additional and separated messages filed as a property of the message. This property can be used as a subchannel communication line independent from the main message, and it can be used as a low-cost additional message channel. The IEEE P1451.1.6 provides a new timestamp property of MQTT v5 that enables cost-effective time-synchronization and maintains the timestamps in the IoT devices (e.g., NCAPs or applications in this case) [2]. By using this method, the IoT devices can synchronize the local clock with an MQTT broker or other time-synchronized MQTT clients (NCAPs or applications in this case). However, microcontrollers may not have a battery-backup real-time clock (RTC) module. For these IoT devices, local timestamping is still a big topic to be resolved**.** These IoT devices often go to sleep mode to extend battery life, and time counting for the sleep duration can be done without requiring an RTC module. For IoT devices used in plant management, factory automation, and abnormal detection applications, timestamping on the order of milliseconds might be needed.

Although MQTT is widely used for these resource-limited microcontrollers, it does not provide time synchronization functions. However, when it is required, MQTT clients and MQTT broker (server) may use external time synchronization protocols, such as network time protocol (NTP) [6], PTP [7], GPS [8], and radio clock such as the National Institute of Standards and Technology (NIST)'s WWV [9]. Time synchronization between the MQTT client and server can be achieved by exchanging MQTT messages via a special synchronization topic. There is an existing study that describes time synchronization using MQTT [10].

IEEE P1451.1.6 provides two types of network communication methods for time synchronization: request-response synchronization (RR-Sync) and broadcasting synchronization (BR-Sync) [2]. This paper mainly focuses on the request-response time synchronization method based on the

client and server operation mode of NTP of RFC 958, RFC 5905, and RFC 7822. It does not consider the stratum of NTP; however, it can additionally compensate for the delay jitter in an MQTT broker [2].

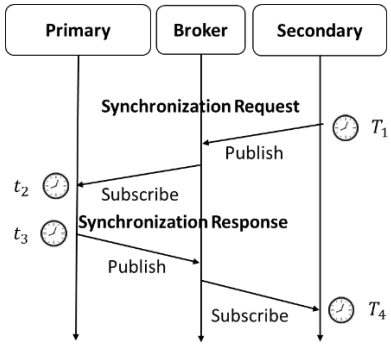


Fig. 4. Time synchronization protocol of IEEE P1451.1.6.

A request-response method-based time synchronization protocol of IEEE P1451.1.6 shown in Fig. 4 is based on the client and server operation mode of NTP. The protocol operates as follows:

a) Synchronization request (SRQ) message: SRQ message is published from the secondary clock client. SRQ message has the timestamp of the secondary clock (T_1). The topic name has to end with TIME/SRQ. The primary clock client subscribes to the topic in advance and memories both the timestamp of the secondary clock (T_1) in SRQ message and the timestamp of its primary clock (t_2). As an option of a transparent clock function, the MQTT broker can add the actual processing time of the SRQ message in the "Correction" property as a millisecond unit for MQTT v3.

b) Synchronization response (SRS) message: The primary clock client sends an SRS message with the timestamp of the primary clock (t3). The parameters in the SRS message are T_1, t_2, and t_3. The topic name of this message has to end with TIME/SRS. As an option of a transparent clock function, the MQTT broker can add the actual processing time of the SRS message in the "Correction" property as a millisecond unit for MQTT v3. The correction of SRQ and SRS will have a similar value. More precise time synchronization can be achieved by compensating the $t_2 - T_1$ value by SRQ correction and $T_4 - t_3$ value by SRS correction because the message processing delay variability of the broker server can be compensated.

c) Estimation: The secondary clock client receives the SRS message by subscribing to the topic for SRS and memories of the timestamp of its secondary clock (T_4). Then, the round trip time (RTT), delay time, and offset can be calculated using the following equations:

$$RTT = (t_2 - T_1) + (T_4 - t_3)$$

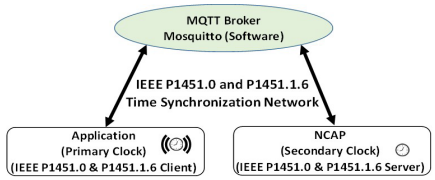
$$One - Way\ Delay = \frac{(t_2 - T_1) + (T_4 - t_3)}{2}$$

$$Offset = \frac{(t_2 - T_1) - (T_4 - t_3)}{2}$$

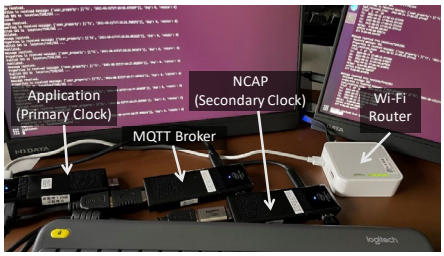
IV. Implementation of Time Synchronization of IEEE P1451.1.6

A. Implementation of Time Synchronization of IEEE P1451.1.6

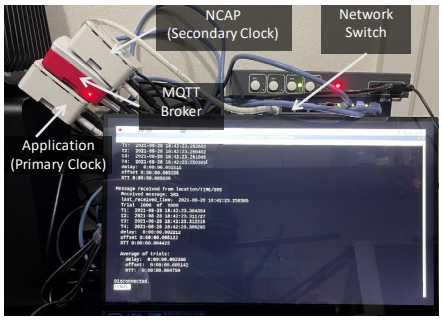
A time synchronization protocol of IEEE P1451.1.6 was implemented using MQTT networks and Python 3.8.5. The architecture of the implemented networks for testing is shown in Fig. 5(a), which consists of an MQTT broker and two MQTT clients – an application and an NCAP. The application had a primary clock, and the NCAP had a secondary clock. The NCAP secondary clock was synchronized with the application's primary clock. Fig. 5(b) shows a time synchronization implementation of IEEE P1451.1.6 using a wireless network (Wi-Fi). The Intel Compute Stick m3 with Core m3-6Y30 (without Ethernet Ports), 4 GB RAM, 64 G eMMC, and Ubuntu 20.04 were the microcontrollers used for the MQTT network. Application, NCAP, and MQTT broker were separately implemented into three different microcontrollers. All of the microcontrollers were connected via a Wi-Fi router.



(a) Architecture of time synchronization implementation



(b) Time synchronization implementation using wireless network



(c) Time synchronization implementation using wired network

Fig. 5. Time synchronization implementation of IEEE P1451.0 and P1451.1.6 standards-based network.

Fig. 5(c) shows a time synchronization implementation using a wireline network. Raspberry Pi 3+ and Raspberry Pi operating system (OS) were used for the application and NCAP implementations, respectively. Raspberry Pi 4 and Ubuntu 20.04 were used for the implementation of the MQTT broker. In both wireline and wireless network implementations, the Mosquitto MQTT broker server v1.6.12, which supports both MQTT v5.0

and v3.11, was deployed. These configurations are suitable for typical sensor networks. In these two types of implementations, it is possible to adjust the NCAP secondary clock to synchronize it with the primary clock; however, in our implementations, we only showed the time delays and offsets, but we did not adjust the NCAP secondary clock to synchronize it with the application's primary clock. The implementation of the P1451.1.6 MQTT time synchronization protocol based on NTP was tested and evaluated by running the installed system described in Fig. 4. All measured parameters such as T_1, t_2, t_3, T_4, one-way delays, offsets, and RTTs between the primary and secondary clock time are captured in a computer screenshot and shown in Fig. 6.

```
Message received from location/TIME/SRS
  Received message: SRS
  last_received_time:  2021-08-28 18:42:23.205441
  Trial  999  of  1000
  T1:  2021-08-28 18:42:23.252692
  t2:  2021-08-28 18:42:23.260462
  t3:  2021-08-28 18:42:23.261045
  T4:  2021-08-28 18:42:23.258305
  delay:  0:00:00.002515
  offset 0:00:00.005255
  RTT 0:00:00.005030

Message received from location/TIME/SRS
  Received message: SRS
  last_received_time:  2021-08-28 18:42:23.258305
  Trial  1000  of  1000
  T1:  2021-08-28 18:42:23.304384
  t2:  2021-08-28 18:42:23.311727
  t3:  2021-08-28 18:42:23.312215
  T4:  2021-08-28 18:42:23.309295
  delay:  0:00:00.002212
  offset 0:00:00.005132
  RTT 0:00:00.004423

  Average of trials:
    delay:  0:00:00.002380
    offset:  0:00:00.005142
    RTT:  0:00:00.004759

Disconnected.
```

Fig. 6. Screenshot of time synchronization implementation of IEEE P1451.1.6.

We have also evaluated the average of 100, 200, 500, 1000, 2000, and 5000 measurements of time synchronization using the setup of Fig. 5. The results are shown in Table II. The implementation of the IEEE P1451.1.6 time synchronization protocol has achieved microseconds (μs)-level of uncertainty according to the precision of "gettimeofday" system calls of Linux, and the returned values of the function calls were used for the measurements of delays, offsets, and RTTs. The uncertainty of one-way delays, offsets, and RTTs includes the cumulative results of the uncertainty of primary and secondary clocks, MQTT broker delay, and wireless and wireline network delays. As shown in Table II, the magnitudes of one-way delays, offsets, and RTTs of the wireline network are less than that of the wireless network, so that the time synchronization uncertainty of the wireline network is considered to be better than that of the wireless network.

Fig. 7 shows the histogram of the offset values of 5000 measurement data points. In the wireless network implementation, 95% of the measured offsets (2-sigma standard deviation) are within the range from 230,160 to 239,169 μs. In the wireline network implementation, that is within the range from 3,582 to 5,425 μs. The wireless network case shows a very large offset mean value of 234,327 μs is primarily due to the time offset between the primary and secondary clocks.

TABLE II. RESULT OF EVALUATIONS

a) Wireless network

No. of Samples		Delay(μs)	Offset(μs)	RTT(μs)
100	Ave.	9,459	233,374	18,919
	Std. Dev.	4,869	3,038	9,737
200	Ave.	8,006	233,037	16,012
	Std. Dev.	3,769	2,408	7,538
500	Ave.	8,553	232,220	17,105
	Std. Dev.	4,765	3,646	9,530
1000	Ave.	8,154	232,706	16,307
	Std. Dev.	5,366	3,132	10,731
2000	Ave.	7,560	233,215	15,120
	Std. Dev.	4,104	2,693	8,208
5000	Ave.	7,294	234,327	14,588
	Std. Dev.	3,067	2,478	6,133

b) Wireline network

No. of Samples		Delay(μs)	Offset(μs)	RTT(μs)
100	Ave.	5,475	4,314	10,950
	Std. Dev.	142	413	285
200	Ave.	5,484	4,315	10,969
	Std. Dev.	146	402	291
500	Ave.	5,523	4,339	11,047
	Std. Dev.	195	457	391
1000	Ave.	5,525	4,314	11,050
	Std. Dev.	194	463	388
2000	Ave.	5,500	4,354	11,000
	Std. Dev.	195	464	389
5000	Ave.	5,521	4,394	11,041
	Std. Dev.	214	469	428

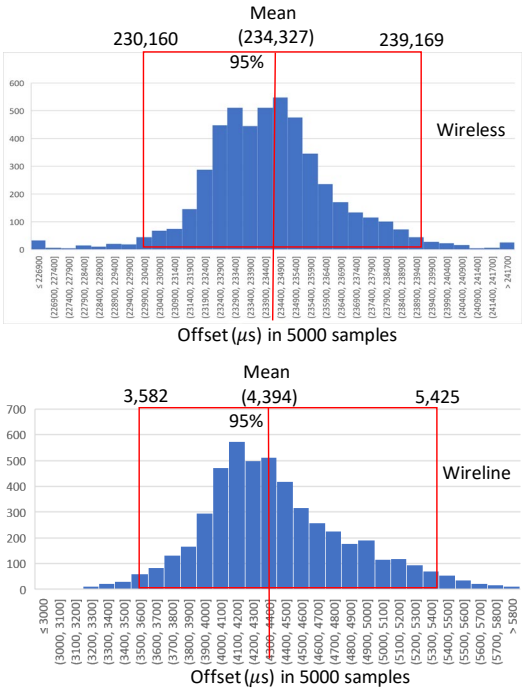


Fig. 7. Offset histogram of 5000 samples.

B. *Time Synchronization TEDS of IEEE P1451.1.6*

The time synchronization TEDS or TimeSync TEDS in the NCAPs was used to describe time synchronization-related information between NCAP and applications. The TimeSync TEDS parameters of IEEE P1451.1.6 are shown in Table III.

TimeSync TEDS consists of a number of fields, such as TEDS identifier (TEDSID), time synchronization protocol name, and version. Table IV shows a list of time synchronization protocols that can be used with IEEE P1451.1.6, such as NTP, PTP, and GPS. Table V shows the version of the IEEE P1451.1.6 time protocol.

Table III shows an example of TimeSync TEDS that includes the TEDSID, protocol name, and version of current time synchronization implementation in this paper. The TEDSID field value is 16, 17, 1, and 1. The value of 16 means IEEE P1451.1.6 in Table IV. The value of 17 means the TimeSync TEDS access code in P1451.0. The value of 1 means that the version number of IEEE P1451.1.6 is 1.0. The value of 1 means that the tuple length is 1. The field value of the time synchronization protocol name in Table III corresponds to the enumerated value of 1 in Table 4, which means IEEE P1451.1.6. The version value 1 in Table III means version 1 of IEEE P1451.1.6, as shown in Table V.

TABLE III. AN EXAMPLE OF AN IEEE P1451.6 TIMESYNC TEDS

Field type	**Field name**	**Description**	**Data type**	**# octet**	**M/O**	**Field Value**	**Field Description**
—		Length	UInt32	4			
0-2	—	Reserved	—	—			
3	TEDSID	TEDS Identification Header	UInt32	4		16 17 1 1	16:P1451.1.6 17: access code 1:Version 1: Tuple length
4-9	—	Reserved	—	—			
Time synchronization related information							
10	Name	Time Synchronization Protocol Name	UInt8	1	M	1	IEEE P1451.1.6 Standard
11	Version	version	UInt8	1	M	1	V1
21-127	—	Reserved	—	—			
128-255	—	Open to manufacturers	—	—			

TABLE IV. TIME SYNCHRONIZATION PROTOCOLS FOR IEEE P1451.1.6.

No.	Time Synchronization Protocol for IEEE P1451.1.6
0	None
1	IEEE P1451.1.6
2	NTP
3	PTP
4	GPS
5-127	Reserved for manufacturer

TABLE V. VERSION OF IEEE P1451.1.6

Version No.	Year
0	Reserved
1	2021
2-127	Reserved for manufacturer

V. CONCLUSION AND FUTURE WORK

This paper describes the time synchronization approaches of IEEE P1451.0 standard-based sensor networks for IoT applications, which includes two-level time synchronizations in IEEE P1451.0 and P1451.1.X standards-based wide area networks (WANs) and IEEE P1451.0 and P1451.5.X standards-based local area networks (LANs). This paper focuses on the time synchronization of IEEE P1451.0 and P1451.1.6 standards-based WANs. Two implementations of time synchronization of IEEE P1451.0 and P1451.1.6 using wireline and wireless networks and their preliminary results are provided to verify the time synchronization approaches of IEEE P1451.1.6. Also, an example of time synchronization TEDS of P1451.1.6 is described.

Future work will focus on hardware-assisted technologies to improve time synchronization uncertainty.

ACKNOWLEDGMENT

The research work on sensors is supported by MEXT/JSPS KAKENHI Grant (B) Number JP20H02301, and sensor hardware design is supported JST CREST Grant Number JPMJCR19K1. The standardization processes are supported under the commissioned research by the National Institute of Information and Communications Technology (NICT, Grant Number 22004).

** Certain commercial products or company names are identified here to describe our study adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the products or names identified are necessarily the best available for the purpose.

References

[1] IEEE P1451.0 - Standard for a Smart Transducer Interface for Sensors, Actuators, Devices, and Systems - Common Functions, Communication Protocols, and Transducer Electronic Data Sheet (TEDS) Formats, [Online]. Available: https://standards.ieee.org/project/1451_0.html.

[2] IEEE P21451-1-6 - Standard for a Smart Transducer Interface for Sensors, Actuators, and Devices - Message Queue Telemetry Transport (MQTT) for Networked Device Communication, [Online]. Available: https://standards.ieee.org/project/21451-1-6.html.

[3] Helena Maria Ramos, Pedro M. Ramos, Pavel Paces, Development of a IEEE 1451 Standard Compliant Smart Transducer Network with Time Synchronization Protocol, Instrumentation and Measurement, Technology Conference - IMTC 2007, Warsaw, Poland, May 1-3, 2007.

[4] Darold Wobschall and Yuan Ma, Synchronization of Wireless Sensor Networks Using a Modified IEEE 1588 Protocol, 2010 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, Portsmouth, New Hampshire, Sept. 29 - Oct. 1, 2010,

[5] Marcel DUECK, Mario SCHLOESSER, Stefan van WAASEN, Michael SCHIEK, Ethernet based time synchronization for Raspberry Pi network improving network model verification for distributed active turbulent flow control, Control Theory Tech, Vol. 13, No. 2, pp. 204–210, May 2015.

[6] Network time protocol (NTP), [Online]. Available: http://www.ntp.org/

[7] Precision time protocol (PTP), [Online]. Available: https://standards.ieee.org/standard/1588-2019.html.

[8] GPS, [Online]. Available: https://www.gps.gov/.

[9] The National Institute of Standards and Technology (NIST)'s WWV, [Online]. Available: https://www.nist.gov/pml/time-and-frequency-division/radio-stations/wwv/wwv-and-wwvh-digital-time-code-and-broadcast.

[10] Adnan Shaout and Brennan Crispin, Using the MQTT Protocol in Real Time for Synchronizing IoT Device State, The International Arab Journal of Information Technology, Vol. 15, No. 3A, Special Issue 2018, [Online]. Available: https://iajit.org/PDF/Special%20Issue%202018,%20No.%203A/17406.pdf.