This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033159, IEEE Access

Digital Object Identifier

Measures to Improve the Accuracy and Reliability of Clock Synchronization in Time-Sensitive Networking

HAILONG ZHU[1,2], KUN LIU[1], YUANYUAN YAN[2], HUAYU ZHANG[2], (Member, IEEE), and TAO HUANG[1,2]

[1]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunication, Beijing 100876 P.R.C
[2]Purple Mountain Laboratories, Nanjing 211111, P.R.C

Corresponding author: Hailong ZHU (e-mail:zhuhl@bupt.edu.cn).

This work was supported in part by The MIIT of China "SDN-based Industrial Network Interconnection and Collaborative Platform Demonstration Application" and the National Key R&D Program of China 2017YFB0102501.

ABSTRACT

Most of the time-sensitive networking standards are based on a high precise and reliability time reference, which is accomplished by IEEE 802.1 AS. However, the accuracy and reliability of clock synchronization could be affected in practice. To conquer the inaccuracy resulting from the collision between different packets, an algorithm that combines time-slot-based synchronization and priority scheduling is proposed. To improve the reliability, an abnormal recovery strategy of the master clock is proposed. The proposed algorithms are implemented on the OMNet++ simulation platform with an typical in-vehicle network topology. The simulation results show that the clock synchronization accuracy could achieve sub-microsecond level even if there are interference flows, and the abnormal recovery mechanism can maintain the stable state of the clock.

INDEX TERMS Clock Synchronization, Time Slot, Priority, Network Reconstruction, Clock Frequency Synchronization, Time-Sensitive Networking

List of Abbreviations

ADAS	Advanced Driving Assistance System
BMCA	Best Master Clock Algorithm
CAN	Controller Area Network
gPTP	generic Precision Time Protocol
HMI	Human Mechine Interface
IVNs	In-Vehicle Networks
LIN	Local Interconnect Network
TSN	Time-sensitive networking

I. INTRODUCTION

With the development of automated driving, an increasing number of sensors and processors have been equipped on vehicles. Traditionally, vehicle buses, such as CAN, LIN, and FlexRay, are used for IVNs. The growth requirements for bandwidth and reliability have become a major challenge for the current and future IVNs. Ethernet is the most widely used general-purpose communication standard [1]. It has a rich bandwidth, excellent scalability and compatibility. Especially in terms of bandwidth, from 10 Mbps to 100 Gbps, which is an obvious advantage over traditional vehicle buses. Besides, in order to meet the data transmission requirements of ADAS, the network need to have a unified time reference. However, each terminal has its own clock. Due to factors such as the manufacturing process, crystal frequency, and external electromagnetic interference, it is difficult to synchronize the time for all terminals. Without synchronization, the time deviation would become increasingly larger as time passes in ADAS, which causes a large difference in the clock of each terminals. Therefore, the bounder latency and low jitter could not be satisfied.

TSN is a series standards and protocols which are developed by IEEE 802.1 TSN task group. Based on the traditional Ethernet, TSN extend the data-link layer standards with bounded low latency, low jitter and zero congestion. These characters could meet the real-time data transmission requirements for Industrial 4.0 and automated driving. So far, the published TSN standards could be divided into four types, such as time synchronization, latency, reliability, and resource management. Time synchronization is accomplished

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033159, IEEE Access

through the IEEE 802.1 AS standard [2], [3], which uses the gPTP to provide a network wide clock synchronization. The other TSN standards could utilize 802.1 AS to realize the characters of TSN. For example, in IEEE 802.1 Qbv, states of the transmission gate are prone to influence by the accuracy of clock synchronization. If the accuracy is not sufficient, there may be no packets when the transmission gate is opened, or the packet arrives but the transmission gate is closed, which leads to inability of the packet to be opportunely and effectively transmitted. Therefore, the accuracy and reliability of clock synchronization are very important for TSN application.

Figure 1 shows the development process of the clock synchronization technique, and several characteristics are compared, the results are shown in Table 1.

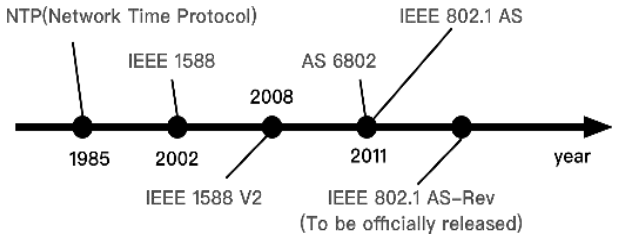

FIGURE 1. The development of clock synchronization technology

The accuracy and reliability of clock synchronization is the basis of corresponding technology of IVN. However, there is little research on the synchronization accuracy when the IEEE802.1AS protocol is applied in practice. The effect of interference flow on the clock synchronization accuracy could be resolved by network planning and traffic scheduling. In addition, if the clock synchronization is unreliable, abnormality of the master clock occurs in the network and some functions will be unavailable.

The main contributions of this paper are listed as follows.

- We construct a TSN clock synchronization function module in the OMNet++ simulation platform based on the INET open source framework. This module realizes basic functions of IEEE802.1 AS, including frequency and non-frequency synchronization for a nonlinear clock. Simulation results prove that frequency synchronization can bring positive results.
- We propose a method of rebuilding the clock synchronization network in the event of an abnormality of the main clock. The method of this study can effectively improve the reliability.
- We analyzed the accuracy of clock synchronization under interference flow. By assigning different time slots for different nodes and setting a higher scheduling priority for the clock synchronization packets, the maximum clock accuracy reaches 860 ns and an average of 270 ns in the presence of interference flow.

The rest of this paper is organized as follows. We introduce the related work in section II. The principle of IEEE802.1AS protocol is illustrated in section III, followed by the module design in section IV. Simulations and a discussion on the proposed model are presented in section V. Finally, conclusions and future works are drawn in section VI.

II. RELATED WORKS

Recently, research on the implementation of highly available clock synchronization has been ongoing. Henning Puttnies *et al* [3] implemented the clock synchronization in the OMNet++ platform and integrated the function into Ethernet network card of INET. The study had following assumptions.

- BMCA algorithm can be replaced with a fixed configuration.
- INET open source framework is included in all the OMNet++ platforms.
- The clock offset is linear.

The research performed a simulation verification based on the designed topology. Results showed that the new NIC module could work normally, and the error ratio of the delay measurement value was less than 1.8%. They contributed their code to INET open source community. However, the constructed network card with the clock synchronization function modified the INET code, which had an impact on INET.

Hyung-taek Lim *et al* [4] built a simulation framework for the IEEE802.1AS protocol based on OMNet++. To simplify simulation, the research made following assumptions.

- The equipment in IVN is statically connected, so master and slave clock are fixed without BMCA process.
- No external clock source is used in simulation.
- Discontinuity of the phase and frequency is not considered.

The clock synchronization accuracy under the background stream was considered, and a simulation scenario of daisy-chain topology was designed. Performance of the algorithm was measured by the transmission delay and master-slave clock deviation. The simulation results showed that the different filter algorithms designed in this study could stabilize the transmission delay at approximately 40 ns, with a positive and negative deviation of 10 ns. In the case of a large background flow and clock synchronization cycle of 125 ms and 62.5 ms, the accuracy could reach approximately 400 ns and 200 ns, respectively. The method provided by the research could achieve sub-microsecond synchronization in a daisy-linked vehicular network. The higher of master-slave clock synchronization frequency was, the higher accuracy of clock synchronization. However, the framework built by the institute was not flexible enough to be modular, and the framework also modified the source code of INET, which had an impact on INET. Although background traffic was considered in this study, the port did not participate in clock synchronization. In essence, the best-effort traffic and clock synchronization packet exhibited separate transmission, so the background traffic had little influence on the clock synchronization packet.

Maryam Pahlevan *et al* [5] built a simulation framework based on the OPNET platform, which supported both

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033159, IEEE Access

TABLE 1. Comparison of several clock synchronization methods

Clock synchronization methods	Application scenario	Synchronization accuracy
IEEE 1588	Industry	Microsecond
IEEE 1588 v2	Industry	Submicrosecond
AS6802	Aerospace	Submicrosecond
IEEE 802.1 AS	Industrial, automotive, and multimedia applications	Submicrosecond

IEEE802.1Qbv and IEEE802.1Qci. The study realized a clock synchronization function, including the BMCA. A train communication network topology was designed based on practical applications, which verified the correctness of the simulation framework. In addition to realizing the clock synchronization function, this paper also studied the abnormal recovery of the master clock. A third-party plug-in was used to inject exceptions into the master clock at a specific time, which resulted in the failure of clock synchronization in the network. After a period of time, a recovery signal was injected into the master clock. The network resumed clock synchronization, and all functions were normal after recovery. IEEE802.1Qbv, IEEE802.1Qci and IEEE802.1AS were combined in this study to verify the clock synchronization function, and it was found that all the protocols could work normally. However, the clock synchronization accuracy of this study was low, the average master-slave clock deviation was approximately 300 μs, and the maximum value was 500 μs, which did not reach the sub-microsecond clock synchronization accuracy.

Unlike the studies discussed above, this paper first realizes the high precision clock synchronization function in OMNet++ and then modularizes it. The clock synchronization module can be flexibly combined with other modules to build nodes with different functions. In addition, this paper proposes an algorithm of time slot adjustment plus priority scheduling, which improves the clock synchronization accuracy in the case of background traffic. This paper also studies the beneficial effect of clock frequency synchronization and some countermeasures against abnormal master clocks.

III. ESTABLISHMENT AND TROUBLESHOOTING OF THE CLOCK SYNCHRONIZATION NETWORK

The establishment process of the clock synchronization network was first introduced, and then the reconstruction process was designed.

A. MASTER CLOCK SELECTION AND PORT ROLE DETERMINATION

The first step is to select the master clock. Due to IEEE 802.1 AS protocol, the network uses BMCA to select the master clock, where all the nodes in network participate. The main process of BMCA algorithm is that each node exchanges the parameters with adjacent nodes until the best performance node is selected and the spanning tree is built. The master clock node is the root, and it mainly assigns synchronization information to other nodes. After that, all the ports of network will be assigned to the following four states.

- Master port: Ports for the master clock or forwarding nodes to distribute clock synchronization frames.
- Slave port: Ports for receiving the clock synchronization frame and performing local clock synchronization.
- Disabled port: Ports that do not support the protocol.
- Passive port: Ports that do not meet the above conditions.

The Disabled Ports are ports in TSN domain, but do not support the IEEE 802.1 AS protocol. The Passive Ports are ports do not belong to TSN domain.

B. TROUBLESHOOTING OF THE MASTER CLOCK

During normal operation, the master clock node will periodically send ANNOUNCE messages. When the clock count does not change, the messages are no longer sent. At the same time, if the slave clock node does not receive this message within two clock synchronization cycles, the network will be deemed to have failed, and the clock synchronization network needs to be rebuilt, re-selecting the master clock. However, IEEE802.1 AS does not clearly stipulate the network reconstruction process. We design the detection of master clock failure, the selection of a new master clock and the process of regenerating the clock synchronization tree. The basic process is shown in Figure 2.

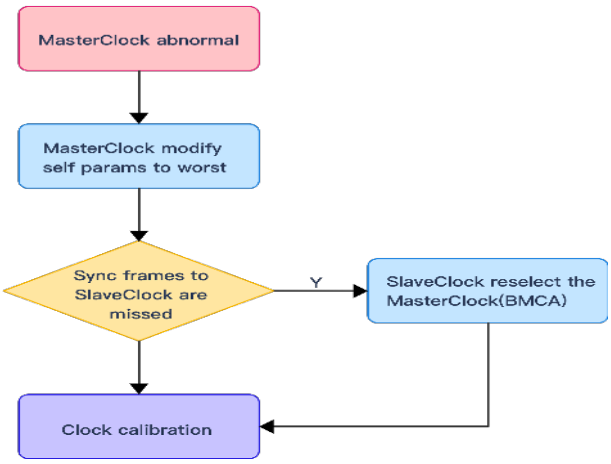

FIGURE 2. Process of network fault handling

When the slave node cannot receive the synchronization packet within a certain period of time, it will be determined that whether the master clock or the path is faulty. At this time, the slave clock initiates BMCA and sends its own

parameters to neighboring nodes. In addition, the master clock node sets its own clock parameter to the worst case and notifies each slave clock of the worst parameter to prevent it from being selected as the master clock again. The master clock re-selection process is the same as the initial selection process.

During network reconstruction, if no measures are taken, the clock skew will continue to increase, and it will exceed the normal range. Therefore, it is necessary to be able to synchronize the clock even if the master clock is lost in the network. At this time, the period of clock synchronization is unchanged, and the calculation method is:

$$t = t_0 + T_{offs} \tag{1}$$

T_{offs} is the difference between the last synchronization time and the last local time before synchronization.

IV. CLOCK SYNCHRONIZATION WITH HIGH PRECISION

On the basis that the network is successfully established and capable of troubleshooting, we focus on improving the accuracy of clock synchronization.

A. DELAY MEASUREMENT

The basic process of the delay measurement between master clock node and slave clock nodes is as follows: master clock sends a synchronization packet and records the transmission time t_{1s} of the synchronization packet. Slave clock records the time t_{2s} when the synchronization packet is received. Then, the delay measurement request packet is sent to the master clock, and the time t_{3s} is recorded when the delay measurement request packet is sent. The master clock records the time t_{4s} when the delay measurement request packet is received. The process of delay measurement is shown in Figure 3. The sending time delay T_s is expressed as,

$$T_s = \frac{[(t_{2s} - t_{1s}) + (t_{4s} - t_{3s})]}{2} \tag{2}$$

Without loss of generality, the receiving time delay T_r is expressed as,

$$T_r = \frac{[(t_{2r} - t_{1r}) + (t_{4r} - t_{3r})]}{3} \tag{3}$$

B. SLAVE CLOCK LOCAL TIME CALIBRATION

Synchronization time t for sending the master clock to the slave clock:

$$t = t_0 + T \tag{4}$$

t_0 is the local clock of the master clock. Then the slave clock receives this synchronization time and modifies its local time.

C. CLOCK FREQUENCY SYNCHRONIZATION

The inconsistency of frequencies between the clocks is a main reason for the continuous drift of the clocks. Especially when frequency is not stable, a deviation between the clocks may increase, resulting in an increment of the clock offset. Therefore, clock frequency synchronization can effectively reduce the clock offset per unit time and respond to the impact of clock frequency changes.

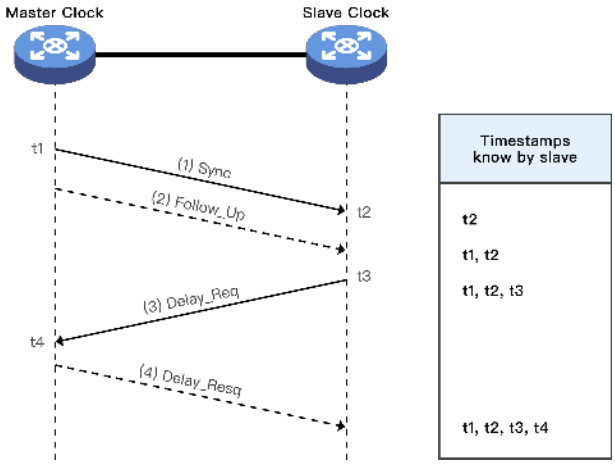

FIGURE 3. Process of delay measurement

After receiving the clock synchronization packet from the master clock, behavior of the slave clock node is shown in Figure 4.

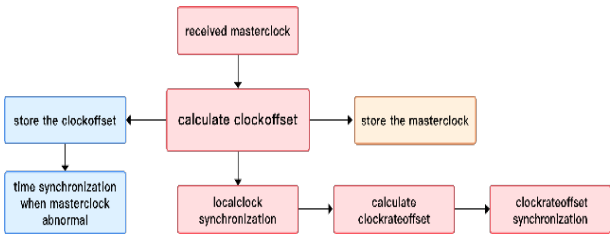

FIGURE 4. Behavior of the slave clock node

Frequency of the clock is determined by the oscillation period of the crystal oscillator and the frequency spreader. Frequency synchronization here is designed to align the change in clock count for each hop of the clock with time interval between two adjacent hops. What changes during synchronization is the change in clock count, which is calculated as,

The clock counting interval is set to T_0, the local time of each count changes to J_0, and the difference $T_{offset1}$ between the synchronization time and the local time before synchronization is first calculated

$$T_{offset1} = t_{master} - t_{local} \tag{5}$$

t_{master} is the time of the received master clock, and t_{local} is the local time of the slave clock. At the same time, the slave clock changes the local time to t_{master} and records it as t_{sync}. At this time, the time between two adjacent synchronization times $T_{interval1}$ can be calculated.

$$T_{interval1} = t_{sync} - t_{sync0} \tag{6}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033159, IEEE Access

where t_{sync0} is the time when the clock was last synchronized, from which the frequency difference T_{offset} of the slave clock and the master clock can be calculated.

$$T_{offset} = T_0 \times \frac{T_{offset1}}{T_{interval1}} \tag{7}$$

The value of the clock counting interval J is:

$$J = J_0 + T_{offset} \tag{8}$$

The clock frequency is synchronized.

D. TIME SLOTTING AND PRIORITY SCHEDULING

When the synchronization packets of the master clock and the delay measurement are sent to multiple slave clocks through the switch node, or when there are multiple slave clocks sending the delay measurement request packets to the master clock at the same time through the switch node, the clock synchronization packets appear in the queue of switch node and they collide with each other.In addition, the best-effort traffic in the network could collide the clock synchronization packets in the queue of switch node. After the collision occurs, it will cause clock synchronization packets to be queued for transmission. The timestamp of the clock synchronization packet arriving at the opposite end contains the queuing waiting time. In TSN network, when the master-slave clock time offset calculation is performed, the queuing waiting time is uncertain, which can result in a large error in the calculation results. In the case of a large number of nodes or large background traffic, there is a complex collision, a long queue waiting time, and a large error in the calculation result of the master-slave clock time offset, resulting in a low clock synchronization accuracy.

To avoid collisions between clock synchronization packets, an algorithm is designed in which different nodes perform time adjustments according to time slots. To avoid collisions between time synchronization and best-effort traffic packets, based on the time slotting algorithm, it is necessary to set the priority of the clock synchronization packets to be higher than the priority of the best-effort traffic packets. By using these methods, the interference of queuing waiting time to the calculation of the time offset of the master and slave clocks can be reduced, and the accuracy of clock synchronization can be improved.

Adopting the time slotting algorithm, when the clock synchronization cycle starts, the master clock sends a synchronization packet. When different slave clock nodes receive the synchronization packet of the master clock, they record relevant information, but instead of sending the delay measurement request packet immediately, they wait before sending the delay measurement request packet. The slave clock node needs to correct the sending time of the delay measurement request packet by the waiting slot time, and the value T_w is

$$T_w = N \times L \tag{9}$$

That is, the number of time slots (N) times the length of each time slot (L), where these two values are configured in the configuration file. The length (L) of each time slot is

$$L = \sum_{i=1}^{n} txtime_i \tag{10}$$

where n is the maximum number of links in the topology, and $txtime$ is the packet sending time of the longest link, where its value is

$$txtime = \frac{max(len)}{B_w} \tag{11}$$

len is the length of the packet transmitted on the link, and B_w is the transmission bandwidth of the link.

The process of time adjustment by the time slot is shown in Figure 5.

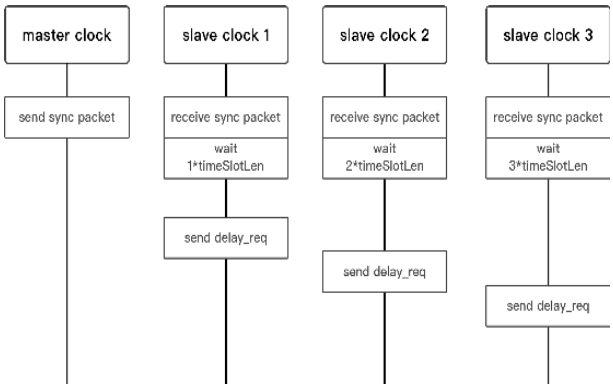

FIGURE 5. Timing chart of the time adjustment by the time slot

Time Slotting And Priority Scheduling algorithm is described as below. In the algorithm description code, L is the

Algorithm 1 Time Slotting And Priority Scheduling

Require: L, N

$P \leftarrow P_H$
$T_w \leftarrow L*N$
$T_s \leftarrow T_R + T_w$
sendPacket at T_s
if collision happened in sending queue **then**
  send P_H first
end if

calculated slotting length, N is the serial number of device, P is the priority for clock synchronization packets and measure delay packets, P_H represents the highest priority, T_s is the sending time of a packet, and T_R is the time when a device received a clock synchronization packet or measure delay packet.

The strategy of using priority scheduling is designed.When there are both clock synchronization and best-effort flow packets, and they are transmitted through the same network card, configuring the clock synchronization packet as the highest priority. When scheduling, the scheduler will first

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033159, IEEE Access

schedule the clock synchronization packet for transmission to reduce the queue waiting time of the clock synchronization packet in the sending queue and improve the clock synchronization accuracy. Meanwhile, if increasing the synchronization frequency, the synchronization accuracy could be more higher with the proposed algorithm. Meanwhile, if increasing the synchronization frequency, the synchronization accuracy could be more higher with the proposed algorithm.

V. SIMULATION DESIGN AND RESULT ANALYSIS

This section designs an in-car vehicular Ethernet topology, introduces the various simulation modules implemented in OMNet++, simulates the clock synchronization function and accuracy in several different scenarios and finally analyzes the simulation results.

A. SIMULATION DESIGN

1) Simulation topology

A car vehicular network scenario is considered, the topology of which contains both star and ring structures, in which the ADAS structure is simulated. In addition, to show the influence of hops, the topology should include clock nodes with different hops to the master clock node. The designed simulation verification topology is shown in Figure 6.

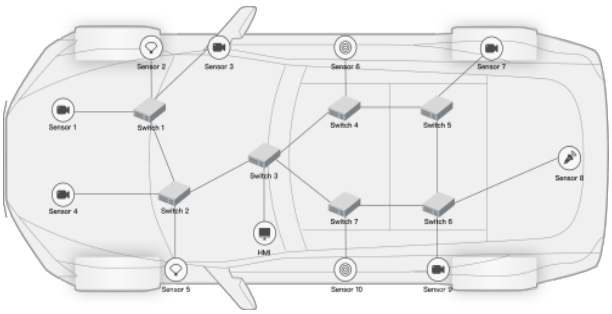

FIGURE 6. Simulation topology of the IVN network

As shown, the topology contains a total of 7 switches and 11 terminal devices, which include 10 sensors and a HMI. The link between switches in the network uses Ethernet with a bandwidth of 10 Gbps. The link between switch, sensor 9 and sensor 10 is 100 Mbps Ethernet, and the other connection is 1 Gbps. The maximum number of hops between different nodes is 6.

2) Node module design

Based on the architecture of Ethernet terminal and switch node in INET open source framework, the clock module AsClock and the clock synchronization module AsElem are added, and the terminal and switch node that support the Ethernet service flow and clock synchronization function are constructed as shown in Figure 7 and Figure 8. The eth, trafGenApp, relayunit, filteringDatabase and other modules in the terminal node and the switch node come from the open source framework INET, and we have use and modify

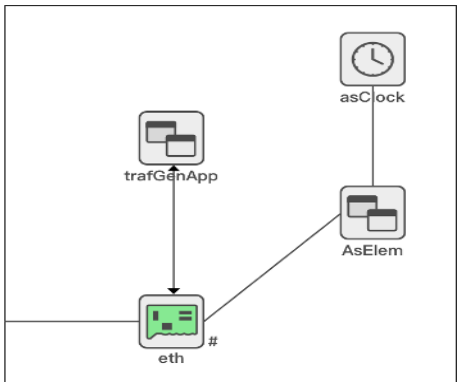

FIGURE 7. Structure of the terminal

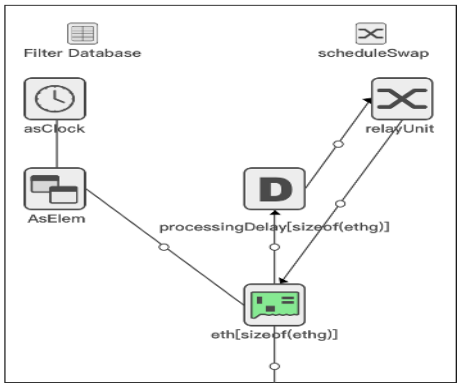

FIGURE 8. Structure of the switch

some of its functions. The eth module is a network card function module that implements the MAC and physical layer functions of the link layer and some LLC functions. The trafGenApp module is a flow generation module that can generate Ethernet packets. The relay unit module is a forwarding unit that forwards the packet according to the routing table generated by the filtering Database. The filtering Database module is a routing table generator that generates routing tables for switches based on the content configured in xml. When the network card module eth receives the packet, it first determines whether it is a clock-synchronized packet. If so, the packet is transferred to the AsElem module for processing; otherwise, it is transferred to other traffic processing modules of the INET for processing.

3) Clock module

In the experiment, each node has a local clock. The clocks of some nodes will be accurately timed at a counting interval of 10 ns, that is, the clock is synchronized to the OMNET++ system simulation time every 10 ns. Other clock frequencies have different degrees of drift and can be set to fixed or time-varying values (only linear changes in frequency are considered in the simulation), and the local time is modeled as:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033159, IEEE Access

$$T_L = \frac{J}{F} \times T_s \tag{12}$$

T_L is the local clock, T_s is the system simulation time, F is the interval of each count of the local clock, the value of F is set at the time of simulation initialization, and J is the time added by the clock after one count. When F is equal to J, the local clock value is the same as the system simulation time, and the initial values of F and J are both set to 10 ns. As the network runs, due to the difference in the clock frequency, the local time of each node of the system will continue to be offset. Therefore, it is necessary to perform continuously synchronization at certain intervals to ensure that the clock deviation of each node of the system is within a certain range.

4) Clock synchronization module

According to the main content of the IEEE802.1AS, the clock synchronization module AsElem is divided into Node, Sync, and Encapsulation submodules. The functions of each submodule are developed and implemented in C++ language. The internal structure of the clock synchronization module is shown in Figure 9. The node module records the clock parameters of each clock and is responsible for the main clock election function. The sync module is responsible for the periodic time correction function, which is mainly used for delay measurements and time corrections.

The encapsulation module is used for addressing the network topology and sending and receiving packets between other nodes.

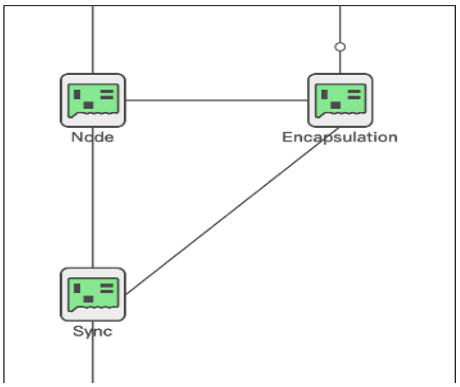

FIGURE 9. The structure of AsElem

5) Simulation parameter settings

Each clock will participate in the main clock election. The clock parameters of sensor 1 are optimal, and the clock parameters of sensor 4 are suboptimal. The clocks except sensor 1 and sensor 4 all have clock drift. There are two types of clock drift: some clocks have positive drift, and some clocks have negative drift. When setting the frequency offset, the frequency is 1 femtosecond drift every 1 millisecond. In the simulation test, the first clock synchronization starts when the simulation reaches 2 milliseconds, and the clock synchronization is periodically performed every 1 millisecond.

6) Simulation Scenarios

For network with complex topology, various complicated scenarios are generated during operation. In the simulation, following scenarios are designed to test the clock synchronization effect. Due to the importance of high precision and high reliability of the clock synchronization in TSN, some of the designed scenarios are used to verify the improvement of clock synchronization accuracy brought by the algorithm proposed in this paper, and the improvement of the clock synchronization reliability brought by the algorithm proposed in this paper.

Scenario one

There is no other traffic in the network, only the generation and transmission of clock synchronization packets. The clock synchronization packets are designed with no actual size. This scenario verifies the basic functions of clock synchronization and checks the correctness of the simulation program. In this case, there is no packet collision in the network.

Scenario two

On the basis of scenario one, some of the clock is set to a nonlinear clock with clock drift. At this time, the initial frequency of the clock is the standard value. The clock synchronization results are tested with and without clock frequency synchronization to verify the gain brought by clock frequency synchronization.

Scenario three

On the basis of scenario one, the master clock is set to fail. When the simulation goes to 7.5 ms, the master clock fails. At this time, network reconstruction is required, that is, the re-election of the master clock and the regeneration of the clock synchronization tree. This scene is set to test whether the measures of the network reliability are effective, verify whether the master clock can be reselected and continue clock synchronization.

Scenario four

The other settings are the same as those in scenario one, but the clock synchronization packet uses the Ethernet packet. This scenario simulates the simplest situation of the actual network. The clock synchronization packets in the network will be transmitted according to the standard Ethernet packet transmission method. When there are many nodes, collisions will occur between the synchronization packets, and the clock synchronization accuracy will decrease. This scenario verifies the impact of collisions on the accuracy of clock synchronization and whether an algorithm based on a time slot can improve the accuracy of clock synchronization.

Scenario five

On the basis of scenario four, interference flow is added and the clock synchronization is tested. In this case, in addition to the collision between the clock synchronization packets, a complex collision with the interference flow will also occur. Whether clock synchronization can be effectively performed in this case is a standard to verify whether the clock synchronization scheme is effective.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033159, IEEE Access

B. SIMULATION RESULTS AND ANALYSIS

Simulations are conducted for the five scenarios in the section above. The simulation results use the time offset value between the master clock and the slave clock as the measurement metric, which is defined as:

$$T_0 = t_L - t_M \tag{13}$$

t_L is the local time of the slave clock, and t_M is the standard time of the master clock.

1) Clock synchronization basic function test

In this part, the clock changes of the slave clock nodes in the network are basically the same. The T_0 of sensor 6 was selected to record the changes in the slave clock. Simulation results for scenario one are shown in Figure 10.

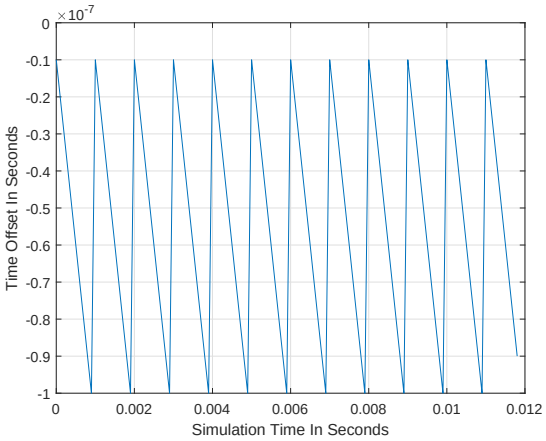

FIGURE 10. The simulation results for scenario one.

At this time, the clock can be accurately synchronized, and the simulation results are consistent with the ideal situation. The maximum time offset between the master and slave clocks is less than 200 ns. After the network is stable, the deviation is within 100 ns.

Figure 11 shows the simulation results of scenario two. For this scenario, we conducted simulation tests on whether to add clock frequency synchronization. It can be seen from the simulation results that when the clock frequency synchronization is not performed, due to the clock frequency difference increasing between the master clock and slave clock, the drift amount of the clock per synchronization period will continue to increase. It reaches 1 ms at approximately 9 ms. Although our experiment uses a higher frequency offset for the clear simulation results, it is also shown from the experimental results that the deviation of each clock continues to substantially increase so that it cannot meet the accuracy requirements of clock synchronization after a certain time.

In the simulation with clock frequency synchronization, when the simulation time reaches 9 ms, the maximum drift of the clock is 100 ns, which is a great improvement compared to the case where clock frequency synchronization is not performed. As the time changes, the deviation does not continue to increase, so in this case, the simulation can achieve sub-microsecond level clock synchronization accuracy.

2) Synchronous network reconstruction test

The node clock may fail in the network. For the IEEE802.1AS protocol, the most serious impact is the failure of the master clock, and the network will lose a unified time reference. For the simulation of this scenario, recovery time $\mathrm{T_r}$ is established as a measure of the network performance.

$$T_r = t_n - t_m \tag{14}$$

t_n is the first time the slave clock adjusts its local clock after the master clock breaks down, and t_m is the last time a clock synchronization message is received from the master clock before it breaks down.

We tested three situations without taking measures, only letting the network re-select the master clock and the clock synchronize itself during the fault. Simulation results are shown in Figure 12.

When no measures are taken, the network loses the same time reference due to the failure of sensor 1, and the individual clocks continue to drift due to frequency differences.

In the re-election test of the master clock, after the network is re-elected, sensor 4 is used as the new master clock node. From the simulation results, it can be seen that during the network failure, the maximum deviation of the timing of each clock node is 400 ns, the average deviation is 250 ns and the recovery time is approximately 4 ms. Although it also achieves sub-microsecond synchronization accuracy, the deviation is larger than the simulation results of other scenarios. Although the network has been repaired, the time deviation during the failure will affect the normal operation of the network. Therefore, the synchronization between the network clock nodes during this period is also important.

After adding the autonomous correction mechanism in the event of a master clock failure in each clock, it can be seen from the experimental results that the clock deviation during network reconstruction is within 200 ns and the recovery time is approximately 2 ms.

3) Tests using Ethernet packets

This part uses the fully encapsulated Ethernet packet in combination with INET framework, that is, the clock synchronization packet is a fully encapsulated Ethernet packet.

First, we test the clock synchronization effect at this time. After using Ethernet packets, collisions between synchronization packets will occur. This collision will cause synchronization packets to be queued in the sending queue, and the results of clock synchronization are no longer accurate. We use different nodes to adjust the time according to different time slots to test the clock synchronization results. The time slot for each node are divided, the time slot of sensor 1 is 0, and then each node adds 1 in turn. According to the calculation method of this study, the time slot length is

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033159, IEEE Access

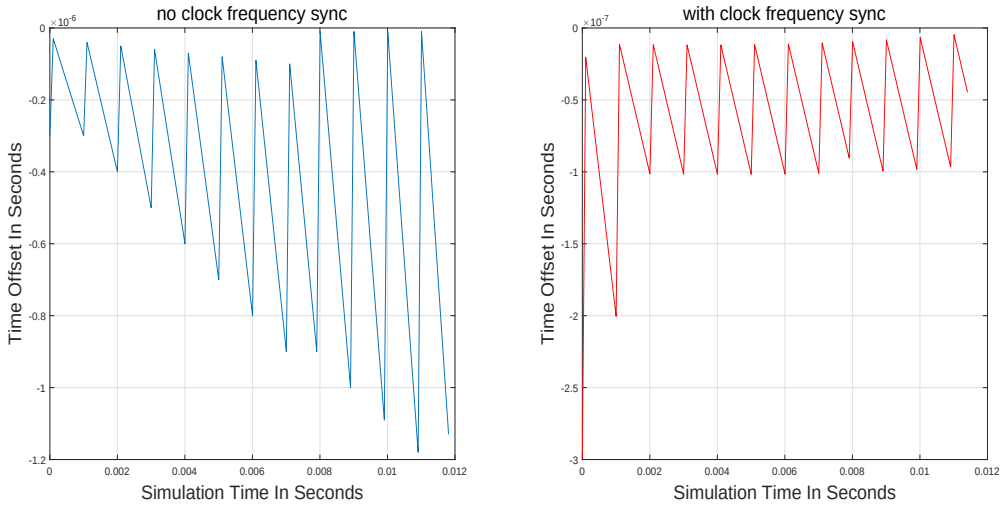

FIGURE 11. The simulation results for scenario two

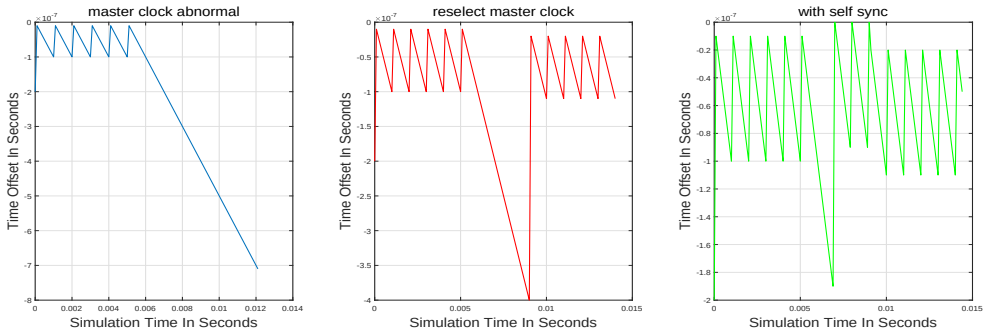

FIGURE 12. The simulation results for scenario three

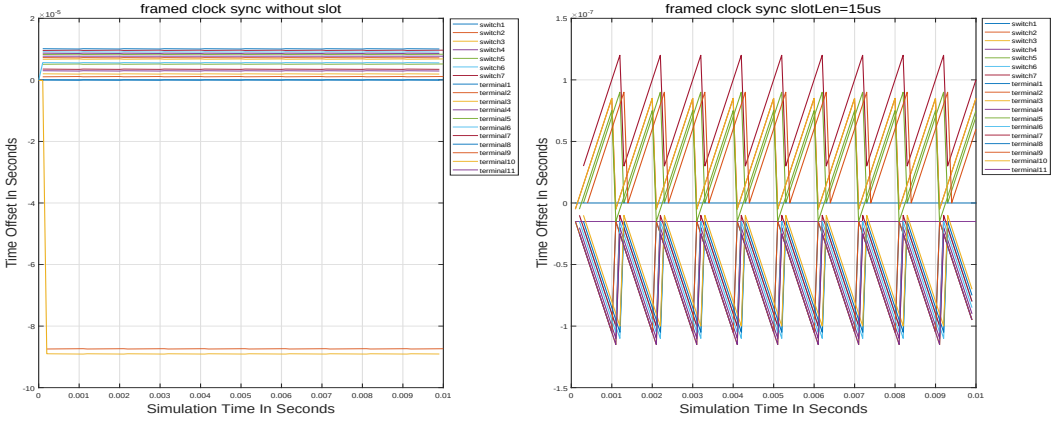

FIGURE 13. The simulation results for scenario four

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033159, IEEE Access

calculated to be 15 μs. The simulation results are shown in Figure 13.

We fit the time offset values of different nodes together, and each color in Figure 13 represents a node. Figure 13 shows that without dividing the time slot, after the first clock synchronization between the master and slave clocks, the maximum time offset between the master and slave clocks is 89.085 μs. The average time offset of most of the slave clocks is 14.658 μs, and the clock synchronization accuracy is low, which does not meet the requirements of clock synchronization accuracy.

After dividing the time slot, before clock synchronization, the maximum time offset between all the slave clocks and the master clock is 120 ns. After the first clock synchronization between the master clock and slave clocks, the maximum time offset between all the slave clocks and the master clock is 120 ns, and the average value is 51 ns. The precision is high, reaching the clock synchronization precision of the sub-microsecond level.

4) Adding clock synchronization test with interference flow

To verify scenario five in the simulation scenario design, we add interference flow to the scene to analyze the influence of interference flow on the clock synchronization accuracy.Sensor 1 to sensor 10 simulate a talker, which generates interference flow. The HMI simulates the listener and receives the interference flow. The parameters of the interference flow are shown in Table 2. Control or management messages between switches, switches and sensors, and sensors in the vehicular Ethernet are absent, but the priority of these control or management messages is lower than the clock synchronization messages.

TABLE 2. Parameters of interference flow

Length of the packet	Sending cycle	Priority
125 Bytes	300 μs	5

First, the clock synchronization with the interference flow and without using the time slot algorithm are tested. There will be many collisions so that the clock synchronization accuracy is low. Then, we solve the collision problem by dividing the time slot timing and setting priority. Adding the interference flow, the simulation results of the clock synchronization are shown in Figure 14.

We fit the time offset values of different nodes together, and each color in Figure 14 represents a node. When the interference flow is added, the maximum time offset between the master clock and the slave clock is 102.83 μs when clock synchronization is performed without dividing the time slots and setting the priority. After the synchronization of the first clock, the maximum time offset between the master clock and slave clocks is 102.83 μs, and the average time offset for most of the slave clocks is 17.27 μs. Due to the interference of the background flow, the time offset between the master clock and slave clocks increases considerably, the synchronization accuracy substantially deteriorates, and the requirement for high-precision clock synchronization cannot be achieved.The different clock synchronization time slots for each node are divided and the priority of the clock synchronization packet is set to 7. After the first clock synchronization, the maximum time offset between all the slave clocks and the master clock is 860 ns, and the average value of most of the nodes is 270 ns. It can be seen that after using the algorithm of this study, although the impact of interference flow on the clock synchronization accuracy still exists, high-precision clock synchronization can be achieved at the sub-microsecond level.

Considering both scenario four and scenario five, the clock synchronization accuracy in each case is shown in Table 3.

VI. CONCLUSION

TSN can meet the low latency requirements and real-time data transmission requirements of many industrial automation and vehicular applications. The accuracy and reliability of clock synchronization influence the TSN function and network performance. This paper is based on the INET open source framework in the OMNet++ environment to simulate a vehicular network. The simulation results show that the TSN has high synchronization accuracy after adopting the time slot timing and priority scheduling algorithm, and it can reach the sub-microsecond level clock synchronization accuracy in the presence of interference flow. The mechanism of clock frequency synchronization and self-synchronization can allow the master-slave clock to maintain high consistency with the high reliability of clock synchronization. In the future, we will continue to conduct research on the application of TSN protocols to achieve the effective application of TSN in vehicular Ethernet.

REFERENCES

[1] "802.3 Standard for Etherne" [S]. IEEE 2015.

[2] Chenjun Le, Donghai Qiao. "Evaluation of Real-Time Ethernet with Time Synchronization and Time-Aware Shaper Using OMNeT++" [C]. 2019 IEEE 2nd International Conference on Electronics Technology (ICET),May 2019:70-73.

[3] Junhui Jiang, Yuting Li, Seung Ho Hong, et al. "A Simulation Model for Time-sensitive Networking(TSN) with Experimental Validation" [C]. IEEE Communications Letters,2019,18(2):153-160.

[4] Henning Puttnies, Peter Danielis, Enkhtuvshin Janchivnyambuu, et al. "A Simulation Model of IEEE 802.1AS gPTP for Clock Synchronization in OMNeT++" [J]. EPiC Series in Computing,2018,56(1):63-72.

[5] H.-T. Lim, D. Herrscher, L. Völker, and M. J. Waltl, "IEEE 802.1 as time synchronization in a switched ethernet based in-car network" [J]. Vehicular Networking Conference (VNC),2011 IEEE,147-154.

[6] Maryam Pahlevan, Balakrishna Balakrishna, Roman Obermaisser. "Simulation Framework for Clock Synchronization in Time Sensitive Networking" [C]. 2019 IEEE 22nd International Symposium on Real-Time Distributed Computing (ISORC). May. 2019:213-220.

[7] Marina Gutiérrez, Wilfried Steiner et al. "Synchronization Quality of IEEE 802.1AS in Large-Scale Industrial Automation Networks" [C].2017 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS),April.2017:273-282.

[8] Martin Wollschlaeger, Thilo Sauter, and Juergen Jasperneite. "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0" [J]. IEEE Industrial Electronics Magazine,11(1),March.2017:17-27.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033159, IEEE Access

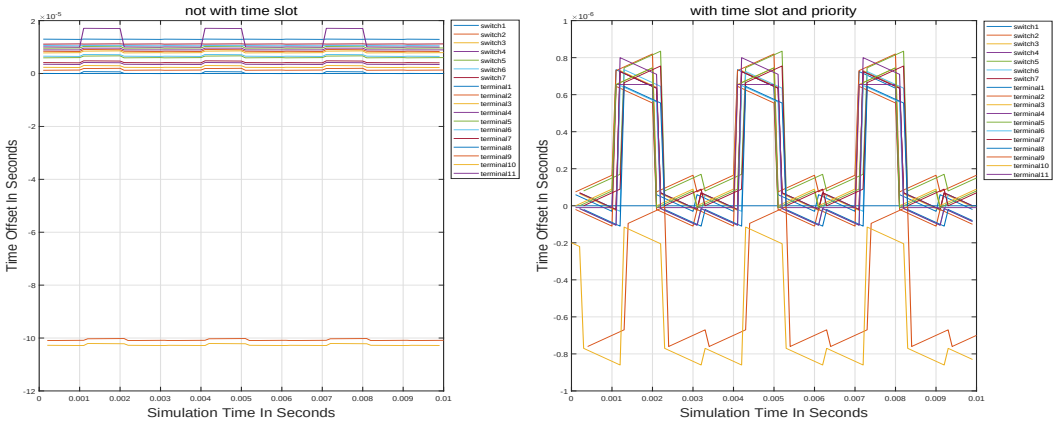

FIGURE 14. The simulation results for scenario five

TABLE 3. Clock synchronization accuracy

Situations	Average Time Offset(ns)	Max Time Offset(ns)
Ethernet packet with no time slot	14658	89085
Ethernet packet with a time slot	51	120
Interference flow with no time slot	17270	102830
Interference flow with a time slot and priority	270	860

[9] Jonathan Falk,David Hellmanns,Ben Carabelli,et al. “NeSTiNg: Simulating IEEE Time-sensitive Networking (TSN) in OMNeT++” [C]. 2019 International Conference on Networked Systems (NetSys),March.2019:1-8.

[10] Ahmed Nasrallah, Akhilesh S. Thyagaturu, Ziyad Alharbi, et al. “ltra-Low Latency (ULL) Networks: The IEEE TSN and IETF DetNet Standards and Related 5G ULL Research” [C].IEEE Communications Surveys & Tutorials,Sept.2018:88-145.

[11] Yong Ju Kim, Jin Ho Kim, Bo Mu Cheon, Young Seo Lee, et al. “Performance of IEEE 802.1AS for Automotive System using Hardware Timestamp” [C]. The 18th IEEE International Symposium on Consumer Electronics (ISCE 2014),June.2014:1-3.

[12] Irfan Allahi, Rabbia Idrees et al. “Performance evaluation of IEEE 1588 protocol with modified LibPTP in OMNet++” [C]. 2018 Annual IEEE International Systems Conference (SysCon), April 2018:1-5.

[13] Martin Levesque and David Tipper. “Improving the PTP Synchronization Accuracy Under Asymmetric Delay Conditions” [C]. 2015 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS), Oct.2015:88-93.

[14] Brunner S, Roder J, Kucera M, et al. “Automotive E/E-architecture enhancements by usage of ethernet TSN” [C].Intelligent Solutions in Embedded Systems (WISES),2017 13th Workshop on. IEEE,2017:9-13.

[15] G. M. Garner and H. Ryu, “Synchronization of Audio/Video Bridging Networks Using IEEE 802.1AS” [J]. IEEE Communications Magazine, Vol.49, No.2, 2011: 140-147.

[16] Feilong Shan, Jianguo Yu, Kaile Li, Zhifang Wang, Yongtao Huang. “Research on High Precision Master-slave Clock Synchronization Based on 1588 Protocol” [C].2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC),July. 2019.

[17] Shiying He, Liansheng Huang, Jun Shen, et al. “Time Synchronization Network for EAST Poloidal Field Power Supply Control System Based on IEEE 1588” [J]. IEEE Transactions on Plasma Science,July.2018:2680-2684.

[18] Feilong Shan, Jianguo Yu et al. “research on High-accuracy Clock Synchronization Based on IEEE 1588 Protocol” [C]. 2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT),Oct. 2019:343-347.

[19] Ryuichiro Maegawa, Daiki Matsui et al. “A Discrete Model of IEEE 1588-2008 Precision Time Protocol with Clock Servo using PI Controller” [C]. 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC),July 2019:531-536.

[20] Ting Yang, Yuqing Niu, Jiexiao Yu. “Clock Synchronization in Wireless Sensor Networks Based on Bayesian Estimation” [J]. IEEE Access(8),April 2020:69683-69694.