Article

An Enhanced Method for Nanosecond Time Synchronization in IEEE 1588 Precision Time Protocol

Fei Li [1], Wenyi Liu [1,2,*], Yueyan Qi [2], Qiang Li [1] and Gaigai Liu [1]

[1] Key Laboratory of Instrumentation Science & Dynamic Measurement, Ministry of Education, North University of China, Taiyuan 030051, China; b1906056@st.nuc.edu.cn (F.L.); b20210620@st.nuc.edu.cn (Q.L.); b200610@st.nuc.edu.cn (G.L.)
[2] Shanxi Engineering Research Center for New Industrial Bus, Taiyuan 030051, China; 18234139186@163.com
* Correspondence: liuwenyi@nuc.edu.cn; Tel.: +86-139-3460-7107

Abstract: The performance of time-critical systems depends heavily on time synchronization accuracy. Therefore, it is crucial to have a synchronization method that can achieve high time synchronization accuracy. In this paper, we propose a new underlying transmission architecture and new synchronization messages. On the basis of these, aiming at the time error problem of the slave clock, we propose an enhanced time synchronization method based on new synchronization messages. Furthermore, we evaluate the performance of the enhanced time synchronization method on the OMNeT++ simulator. In addition, we compare the impact of different crystal oscillator accuracies and different crystal oscillator frequencies on time synchronization accuracy, respectively. Simulation results show that the time offset is at most ± 1 clock period using the enhanced time synchronization method. We realize the purpose of timing the master clock and the slave clock by counting the period of the clock signal. Therefore, we needed to round down the time count to an integer. This is the reason why -1 and 1 appear at the same time. When the crystal oscillator frequency used is 80 MHz, the system can achieve a time synchronization accuracy of ± 12.5 ns; that is, a nanosecond-level time synchronization accuracy can be achieved. With the reduction of the crystal oscillator accuracy of the slave clock, the synchronization accuracy of ± 1 clock period can still be achieved. With the increase in the crystal oscillator frequency, the time synchronization accuracy that can be achieved also improves. The method proposed in this paper provides a new way of thinking and has certain guiding significance for improving the time synchronization accuracy of time-critical systems.

Keywords: clock synchronization; IEEE 1588 Precision Time Protocol; OMNeT++; time synchronization accuracy

check for updates

Citation: Li, F.; Liu, W.; Qi, Y.; Li, Q.; Liu, G. An Enhanced Method for Nanosecond Time Synchronization in IEEE 1588 Precision Time Protocol. *Processes* **2023**, *11*, 1328. https://doi.org/10.3390/pr11051328

Academic Editors: Silvia Carpitella, Manuel Herrera, Bruno Melo Brentan and Joaquín Izquierdo

Received: 19 March 2023
Revised: 18 April 2023
Accepted: 21 April 2023
Published: 25 April 2023

1. Introduction

In time-critical systems, tasks and messages meet their deadlines by applying Time Division Multiple Access (TDMA) or time-triggered scheduling. Each node is given an exclusive transmission time slot. It sends data within the allocated time slot, which can prevent collisions [1]. The nodes in the network use their local clocks to determine the position of the time slot. Even if these local clocks use crystal oscillators with the same frequency, the time error of the crystal oscillator will occur due to the instability of their own operating state and the influence of external environmental factors, which will also cause a time offset between their local clocks [2]. Therefore, a clock synchronization mechanism is necessary to make the time of each node the same. Time synchronization is important for time-critical systems because the performance of the system depends heavily on the time synchronization accuracy. Furthermore, if the synchronization accuracy between nodes becomes higher, the time slots are more precise. In this way, the utilization rate of the bandwidth is higher [3]. What time synchronization needs to do is to make the system have a reference clock and synchronize each node with this reference clock.

Currently, time synchronization technologies for wired networks mainly include Network Time Protocol (NTP) and IEEE 1588 Precision Time Protocol (PTP) [4]. NTP [5] is a communication protocol for time synchronization between the server and the client. The path delay and the time offset between the server and the client are calculated through the handshake mechanism. The client adjusts the local clock according to these two parameters to achieve time synchronization with the server. NTP is simple and easy to implement. It does not require additional hardware investment and can achieve time synchronization only by running NTP on the computer. NTP provides millisecond time synchronization accuracy. Therefore, it can only meet the needs of fields with low time accuracy requirements. IEEE 1588 PTP enables accurate and precise synchronization of the local clocks of nodes in networked distributed systems [6]. It provides sub-microsecond time synchronization accuracy [6,7]. Compared with NTP, IEEE 1588 PTP can achieve higher time synchronization accuracy through hardware-assisted timestamping and is suitable for time-critical systems that require high time accuracy. In addition, it has low overhead and can be applied in large distributed networks while consuming few resources. Therefore, this paper proposes an enhanced time synchronization method based on IEEE 1588 PTP to provide nanosecond time synchronization accuracy.

IEEE 1588 PTP is a packet-based two-way message exchange protocol that can achieve high-precision time synchronization. However, some factors such as the accuracy of timestamp acquisition, link asymmetry, and time error of the crystal oscillator affect the time synchronization accuracy [8–10]. Therefore, researchers have made some enhancements to IEEE 1588 PTP to solve the erroneous results produced by these factors. To the authors' knowledge, currently, no researchers have made enhancements to the synchronization messages themself. Therefore, this paper proposes a new underlying transmission architecture and new synchronization messages. On the basis of these, aiming at the time error problem of the slave clock, an enhanced time synchronization method based on new synchronization messages is proposed. Furthermore, we build a simulation model on OMNeT++ simulator and evaluate the performance of the proposed method. In addition, we compare the impact of different crystal oscillator accuracies and different crystal oscillator frequencies on time synchronization accuracy, respectively. The enhanced time synchronization method is based on IEEE 1588 PTP, which provides a new way of thinking and has certain guiding significance for improving time synchronization accuracy.

Therefore, we summarize our main contributions, which are as follows:

- New underlying transmission architecture and new synchronization messages are proposed, which not only reduce the transmission time of synchronization messages, but also improve bandwidth utilization.
- Aiming at the time error problem of the slave clock, we propose an enhanced time synchronization method based on new synchronization messages.
- The enhanced time synchronization method proposed in this paper is based on IEEE 1588 PTP, which improves the time synchronization accuracy. It provides a new way of thinking and has certain guiding significance for improving the time synchronization accuracy of time-critical systems.

The research content of this paper is arranged as follows. We present the background and related work in Section 2. We elaborate on the enhancements we have made based on IEEE 1588 PTP in Section 3. Simulation-based evaluation and analysis of the results for the enhanced time synchronization method are presented in Section 4. Finally, Section 5 concludes the paper.

2. Background and Related Work

2.1. IEEE 1588 PTP Clock Synchronization Mechanism

The IEEE 1588 PTP clock synchronization mechanism has two types: delay request-response mechanism and peer delay mechanism. Since this paper is an enhancement on the delay request-response mechanism, we describe this mechanism in detail below. The delay request-response mechanism, also known as the End-to-End (E2E) delay measurement mechanism, is implemented through the exchange of *Sync* message, *Follow_Up* message, *Delay_Req* message, and *Delay_Resp* message. IEEE 1588 PTP has two clock modes for a one-step method and a two-step method. The two-step clock mode requires a *Follow_Up* message to send the egress time of the *Sync* message to the slave clock, while the one-step clock mode does not require a *Follow_Up* message. Through the exchange of the above four synchronization messages, the slave clock knows four timestamps, and then calculates the path delay and the time offset between the master clock and the slave clock. According to the obtained time offset, the slave clock synchronizes its local clock to the master clock [11]. The specific synchronization process is demonstrated in Figure 1, which includes the following eight steps:

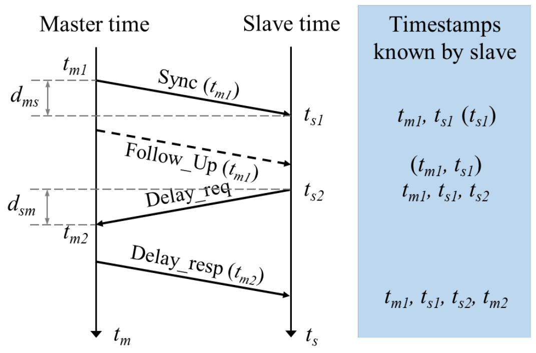

Figure 1. Timing diagram of IEEE 1588 PTP synchronization messages, with t_m being the time of the master clock and t_s the time of the slave clock.

Step 1: The master clock sends a *Sync* message containing egress time t_{m1} of the *Sync* message to the slave clock periodically if the system uses the one-step clock mode. Otherwise, the master clock sends a *Sync* message to the slave clock periodically and records egress time t_{m1} of the *Sync* message.

Step 2: The slave clock receives the *Sync* message and records ingress time t_{s1} of the *Sync* message. At this time, the slave clock knows timestamps t_{m1} and t_{s1} if the system uses the one-step clock mode. Otherwise, the slave clock only knows timestamp t_{s1}.

Step 3: Then, the master clock sends a *Follow_Up* message containing timestamp t_{m1} to the slave clock if the system uses the two-step clock mode. Otherwise, this step does not need to be performed.

Step 4: The slave clock receives the *Follow_Up* message if the system uses the two-step clock mode. At this time, the slave clock knows timestamps t_{m1} and t_{s1}. Otherwise, this step does not need to be performed.

Step 5: Afterward, the slave clock sends a *Delay_req* message to the master clock and records egress time t_{s2} of the *Delay_req* message. At this time, the slave clock knows timestamps t_{m1}, t_{s1}, and t_{s2}.

Step 6: The master clock receives the *Delay_req* message and records ingress time t_{m2} of the *Delay_req* message.

Step 7: Then, the master clock sends a *Delay_resp* message containing timestamp t_{m2} to the slave clock.

Step 8: The slave clock receives the *Delay_resp* message. At this time, the slave clock knows timestamps t_{m1}, t_{s1}, t_{s2}, and t_{m2}.

We assume that the path delay is symmetric, i.e., the path delay d_{ms} from the master clock to the slave clock is equal to path delay d_{sm} from the slave clock to the master clock. Hence, according to the time offset calculated from Equations (1) and (2), the slave clock can synchronize its local clock to the master clock.

$$d_{ms} = d_{sm} = \frac{t_{s1} - t_{m1} + t_{m2} - t_{s2}}{2} \tag{1}$$

where d_{ms} represents the path delay between the master clock and the slave clock and d_{sm} represents the path delay between the slave clock and the master clock.

$$d_{offset} = t_{s1} - t_{m1} - d_{ms} \tag{2}$$

where d_{offset} is the time offset between the master clock and the slave clock and d_{ms} represents the path delay between the master clock and the slave clock.

2.2. Related Work

It is necessary to have a high time synchronization accuracy for time-critical systems. Therefore, improving time synchronization accuracy is a major concern for researchers. Since the asymmetry of paths would produce erroneous results for the calculation of the time offset, an improved method was proposed by Shuai Lv et al. [12]. They evaluated the performance of the proposed method through simulation and performed a comparative analysis with traditional methods. These results suggest that this method improves the accuracy of time synchronization. In the process of measuring peer delay and residence time, different rates of the master clock and the transparent clock may cause erroneous results. To solve the problem, Zongpeng Du et al. [13] improved the traditional mechanism by using a fixed delay ratio method. Giada Giorgi et al. [8] analyzed the impact of the clock instability, the synchronization message exchange rate, and the timestamping accuracy on the time synchronization accuracy and the relationships between these factors. At the same time, they proposed a Kalman-filter-based clock servo and analyzed its impact on the performance of time synchronization through simulation. After that, they proposed a composite algorithm consisting of two Kalman filter clock algorithms. These two algorithms run in parallel. This composite algorithm increases the robustness of the system and achieves better time synchronization accuracy [14]. Ryuichiro Maegawa et al. [15] built a discrete model of the system using the Proportional Integral (PI) clock servo in PTPd, a software-only implementation of IEEE 1588 PTP. In addition, they analyzed the stability of the entire system and investigated the impact of the parameters of the clock servo on the performance of the system. To estimate and correct the random delay's bias, Mohammad Javad Hajikhani et al. [16] proposed a recursive method. They compared the proposed method with the boot-strap method. The results show that this method improves the time synchronization accuracy of IEEE 1588 PTP. Aiming at the problem of the path delay error of synchronization messages, Wei Yajun et al. [17] proposed a method based on the Diffserv-based Packets Scheduling Model (DPSM). In addition, they built the hardware platform. This method achieves a time synchronization accuracy of 500 ns under a crystal oscillator frequency of 100 MHz. To solve the problems of frequency drift of the crystal oscillator and quantization errors of the timestamps, Zhang Junhao et al. [10] designed the clock servo based on Linear Extended State Observer (LESO) in IEEE 1588 PTP networks. In addition, they compared the performance of the LESO-based clock servo with the Kalman-PI clock servo through numerical simulations. The LESO-based method achieves a time synchronization accuracy of ±50 ns under a crystal oscillator frequency of 50 MHz. Yin Hongtao et al. [18] presented a Field Programmable Gate Array (FPGA) implementation of IEEE 1588 PTP. It achieves 97.76% of the time offset within the range of ±40 ns. D. Pedretti et al. [19] described an FPGA implementation of IEEE 1588 PTP. This hardware implementation achieves a time synchronization accuracy of ±16 ns under a crystal oscillator frequency of 62.5 MHz. Aamir Sohail Nagra et al. [20] described an FPGA implementation of IEEE 1588 PTP. It achieves a time synchronization accuracy of 18 ns under a crystal oscillator frequency of 100 MHz. Since path asymmetry affects the time synchronization accuracy, Qi Wu et al. [21] proposed a method to solve this problem. This method achieves a time offset of less than 20 ns under a crystal oscillator frequency of 125 MHz. Aiming at the problem that a possible path asymmetry makes it difficult to estimate the clock skew and offset, Anantha K. Karthik et al. [9] presented a robust iterative clock skew and offset estimation scheme and verified it by numerical simulation. Xiaojiang Liu et al. [22] proposed an embedded clock skew estimation scheme in industrial networks.

This scheme improves time synchronization accuracy without increasing bandwidth overhead. Yue Zuo et al. [23] modeled the clock deviation and the frequency deviation. In addition, they corrected the clock deviation. Finally, they verified the effectiveness of the method. To improve the time synchronization accuracy, Vinh Quang Nguyen et al. [24] proposed an adaptive Fuzzy-based Proportional–Integral (Fuzzy-PI) clock servo. In addition, a test bench was built to demonstrate the proposed method. Sascha Einspieler et al. [25] investigated the performance of software-based and hardware-supported time synchronization method over the Controller Area Network (CAN). It achieves sub-microsecond time synchronization accuracy, which outperforms traditional methods. In addition to focusing on time synchronization accuracy, energy consumption and computational complexity are also extremely important in the research of Wireless Sensor Network (WSN) time synchronization. Xintao Huan et al. [26] proposed a beaconless energy-efficient time synchronization scheme. This scheme can conserve up to 95% of energy consumption on the basis of achieving microsecond-level time synchronization accuracy. In WSNs, based on the Extended Kalman Filter (EKF), Heng Wang et al. [27] proposed a timestamp-free clock skew and time offset joint tracking algorithm. To improve the energy efficiency of the time synchronization, an EKF clock skew tracking algorithm was proposed. They verified the effectiveness of the proposed algorithms through simulations. To estimate the fluctuated delay, Zihan Fang et al. [28] proposed a Delay-Compensated Consensus Kalman-based Time Synchronization (DCCKTS) algorithm. This algorithm is superior to conventional algorithms. Researchers have made various enhancements targeting different influencing factors to improve the time synchronization accuracy. However, they are all based on IEEE 1588 PTP synchronization messages.

For the most demanding applications, the description of a new high-accuracy time synchronization profile for IEEE 1588-2019 [29], Ref. [30] has been recently approved in 2019, which is based on the White Rabbit (WR) technology. However, cost of FPGA implementation of WR technology is relatively high.

As far as we know, the existing literature compensates for the factors affecting the time synchronization accuracy without changing the underlying transmission architecture of synchronization messages and synchronization messages themselves. Therefore, we propose a new underlying transmission architecture and new synchronization messages. On the basis of these, aiming at the time error problem of the slave clock, we propose an enhanced time synchronization method based on new synchronization messages. Furthermore, we built a simulation model on the OMNeT++ simulator and performed an analysis for the performance of the proposed method. In addition, we compared the impact of different crystal oscillator accuracies and different crystal oscillator frequencies on time synchronization accuracy, respectively.

3. An Enhanced Time Synchronization Method

Shanxi Engineering Research Center for New Industrial Bus has developed a new generation of high real-time intelligent reconfigurability bus suitable for aerospace, industrial automation, and other fields. The practice shows that the bus technology has better high real-time, high determinism, and high reliability [31].

To improve the time synchronization accuracy, we propose new underlying transmission architecture, new synchronization messages, and—on these bases, to solve the time error problem of the slave clock—an enhanced time synchronization method based on new synchronization messages. Sections 3.1–3.3 describe these three aspects, respectively.

3.1. New Underlying Transmission Architecture

At present, synchronization messages of IEEE 1588 PTP are transmitted based on the Ethernet frame format at the physical layer. The length of an Ethernet frame ranges from 64 to 1518 bytes, of which the data field is at least 46 bytes. If the length of the transmitted data field is less than 46 bytes, the data field is padded. When the transmitted frame is a short frame, using the Ethernet frame format for transmission will cause a waste of the bandwidth and also increase the transmission time of the frame on the link.

A new underlying transmission architecture is proposed for this problem. It does not depend on Ethernet and has a completely new protocol architecture. Its frame structure is shown in Figure 2. The length of the frame is 80 bits, i.e., 10 bytes. The 1-byte MD field is the Mode Domain (MD) of the frame. It defines the type and the function of the frame and indicates the priority of the frame. The 2-byte SA field is the Source Address (SA) of the source terminal device. It is the identification number of the source terminal device. The 2-byte DA field is the Destination Address (DA) of the destination terminal device. It is the identification number of the destination terminal device. The SA field and the DA field are embedded in the frame when the frame is sent, and the frame is forwarded to the correct destination terminal device according to the DA field. All devices live on the network with a unified addressing scheme. Moreover, the identification numbers of all devices must be unique. The 1-byte PD1 field and the 2-byte PD2 field are the Parameter Domain (PD) of the frame. When a frame is transmitted, the attributes of the frame can be set in these two fields. The 1-byte CD field is the Control Domain (CD). The new bus developed by our center is a dynamic intelligent reconfigurability bus, which can select the optimal path for data transmission. This field is used to select the path for data transmission. When a data frame is transmitted, this field is used as the response-or-not field of the data start frame. The 1-byte VD field is the Verification Domain (VD) of the frame. The VD field of the new bus uses the Cyclic Redundancy Check (CRC) method. The length of the frame using the proposed underlying transmission architecture is 10 bytes, which is much smaller than the minimum length of an Ethernet frame, which is 64 bytes. Compared with synchronization messages using the Ethernet frame structure, the transmission time of synchronization messages using the proposed underlying transmission architecture on the link can be greatly reduced, and the utilization rate of the bandwidth can also be increased.

MD	SA	DA	PD1	PD2	CD	VD
79...72	71..56	55..40	39..32	31..16	15..8	7..0
(1 byte)	(2 bytes)	(2 bytes)	(1 byte)	(2 bytes)	(1 byte)	(1 byte)

Figure 2. Frame structure of the new underlying transmission architecture.

3.2. New Synchronization Messages

Based on the new underlying transmission architecture, we propose new synchronization messages for time synchronization. The new synchronization messages include *A1* message, *51* message, and *A2* message. The lengths of these three synchronization messages are all 10 bytes. The type of synchronization message is determined by the value of the 1-byte MD field. That is, the value of the MD field of an *A1* message is 0x*A1*, the value of the MD field of a *51* message is 0x51, and the value of the MD field of an *A2* message is 0x*A2*. The new clock synchronization mechanism is an enhancement based on the IEEE 1588 PTP clock synchronization mechanism. Its specific process is shown in Figure 3. Similar to the IEEE 1588 PTP clock synchronization mechanism, the new clock synchronization mechanism also obtains the time offset through the exchange of timestamps. According to the obtained time offset, the slave clock synchronizes its local clock to the master clock. To eliminate fluctuations caused by data processing and network protocol stacks, timestamps should be as close as possible to the physical layer. In this way, higher timestamp accuracy can be achieved, and the synchronization accuracy that the system can achieve is also higher. Therefore, this clock synchronization mechanism uses a hardware-assisted timestamp. Section 3.3 describes the specific time synchronization process in detail.

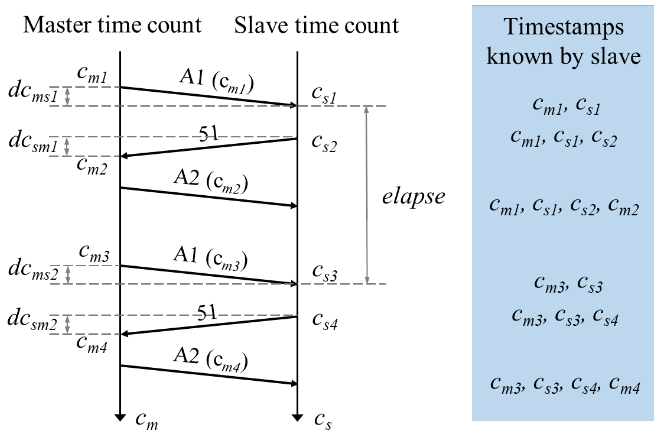

Figure 3. Timing diagram of the new synchronization messages, with c_m being the time count of the master clock and c_s the time count of the slave clock.

3.3. Enhanced Time Synchronization Method

The crystal oscillator frequency of the slave clock is not always the nominal frequency, and the time error will occur due to the instability of its own operating state and the influence of external environmental factors. Nodes can periodically synchronize with each other using IEEE 1588 PTP. However, due to the time error of the slave clock, there is a large gap time between the master clock and the slave clock during the interval between two synchronization processes, which will seriously affect the time synchronization accuracy. Therefore, we propose an enhanced time synchronization method to solve this problem. The specific synchronization process of the proposed new clock synchronization mechanism in Figure 3 includes the following 13 steps:

Step 1: The master clock sends an *A1* message containing egress time count c_{m1} of the *A1* message to the slave clock periodically.

Step 2: The slave clock receives the *A1* message and records ingress time count c_{s1} of the *A1* message. The slave clock knows timestamps c_{m1} and c_{s1} by this time.

Step 3: Afterward, the slave clock sends a *51* message to the master clock and records egress time count c_{s2} of the *51* message. At this time, the slave clock knows timestamps c_{m1}, c_{s1}, and c_{s2}.

Step 4: The master clock receives the *51* message and records ingress time count c_{m2} of the *51* message.

Step 5: Then, the master clock sends an *A2* message containing timestamp c_{m2} to the slave clock.

Step 6: The slave clock receives the *A2* message. At this time, the slave clock knows timestamps c_{m1}, c_{s1}, c_{s2}, and c_{m2}. We assume that the path delay is symmetric; that is, path delay count dc_{ms1} from the master clock to the slave clock is equal to path delay count dc_{sm1} from the slave clock to the master clock. Hence, according to the time offset calculated from Equation (3), the slave clock can synchronize its local clock to the master clock. So far, the first time synchronization round has completed.

$$dc_{offset} = \frac{c_{s1} - c_{m1} + c_{s2} - c_{m2}}{2} \tag{3}$$

where dc_{offset} is the time count offset between the master clock and the slave clock.

Step 7: The master clock sends an *A1* message to the slave clock at a fixed interval. When the interval time arrives, the second time synchronization round starts. Similarly, the master clock sends an *A1* message containing egress time count c_{m3} of the *A1* message to the slave clock periodically.

Step 8: The slave clock receives the *A1* message and records ingress time count c_{s3} of the *A1* message. The slave clock knows timestamps c_{m3} and c_{s3} by this time.

Step 9: Afterward, the slave clock sends a *51* message to the master clock and records egress time count c_{s4} of the *51* message. At this time, the slave clock knows timestamps c_{m3}, c_{s3}, and c_{s4}.

Step 10: The master clock receives the *51* message and records ingress time count c_{m4} of the *51* message.

Step 11: Then, the master clock sends an *A2* message containing timestamp c_{m4} to the slave clock.

Step 12: The slave clock receives the *A2* message. At this time, the slave clock knows timestamps c_{m3}, c_{s3}, c_{s4}, and c_{m4}. We still assume that the path delay is symmetric; that is, path delay count dc_{ms2} from the master clock to the slave clock is equal to path delay count dc_{sm2} from the slave clock to the master clock. Hence, according to the time offset calculated from Equation (4), the slave clock can synchronize its local clock to the master clock. At this point, the time of the slave clock is exactly the same as that of the master clock. However, due to the time error of the crystal oscillator of the slave clock, the time offset becomes larger and larger as time goes on. Therefore, before the third time synchronization round, we need to use the enhanced time synchronization method to make the slave clock synchronize its local clock to the master clock. Corresponding to Figure 3, the time count offset calculated by Equation (4) is actually the time count offset generated by the slave clock during *elapse* period when it receives the *A2* message. Therefore, we use Equation (5) to compensate for this time count offset. The slave clock increases or decreases by one clock period every *cnt* clock periods calculated by Equation (5). In this way, during the interval between two synchronization processes, the slave clock can also synchronize its local clock to the master clock, which can compensate the time offset caused by the time error of the crystal oscillator of the slave clock. So far, the second time synchronization round has completed.

$$dc_{offset} = \frac{c_{s3} - c_{m3} + c_{s4} - c_{m4}}{2} \tag{4}$$

where dc_{offset} is the time count offset between the master clock and the slave clock.

$$cnt = \frac{c_{s3} - c_{s1}}{dc_{offset}} = \frac{elapse}{dc_{offset}} \tag{5}$$

where *cnt* is the interval count between two successive compensation algorithm execution for the slave clock, dc_{offset} is the time count offset between the master clock and the slave clock, and *elapse* is the interval count between the ingress time count when the slave clock received the *A1* message and the ingress time count when the slave clock received the last *A1* message.

Step 13: When the sending interval of an *A1* message arrives again, the third time synchronization round starts. Steps 7–12 are repeated. The only difference is that the time of the slave clock is based on the time corrected by using purely IEEE 1588 PTP clock synchronization mechanism when the slave clock received the last *A2* message, instead of the time corrected by using the enhanced time synchronization method.

The master clock sends an *A1* message containing the egress time count of the *A1* message to the slave clock periodically. The master clock can directly send an *A1* frame containing the timestamp to the slave clock. It does not need to additionally send a *Follow_Up* message containing the timestamp information to the slave clock like the two-step clock mode in IEEE 1588 PTP. The enhanced time synchronization method only needs three one-way transmissions of synchronization messages to achieve time synchronization. This can reduce the execution time of the enhanced time synchronization method. On the other hand, the enhanced time synchronization method is carried out on the port of the device. Unlike using the kernel to achieve time synchronization, it does not occupy the data exchange time of the kernel. The port and the kernel are two separate processes that handle tasks independently. Therefore, time synchronization on the port does not affect the data exchange performance of the kernel. For the new bus developed by our center, it has a port detection mechanism. The port detection mechanism periodically detects changes in port connections and dynamically constructs and updates the entire network. The time synchronization message is embedded in the communication packet of the port detection, and no additional communication packet is added. Through the above analysis, the enhanced time synchronization method itself does not additionally occupy the bandwidth of the bus and consume resources. In addition, when the enhanced time synchronization method is implemented in reality, users are allowed to modify the interval for sending an *A1* message through the register according to the specific usage scenario. When the interval for sending an *A1* message is longer, the proportion of time synchronization occupying the bus bandwidth is smaller. Therefore, the bandwidth can be better utilized by modifying the sending interval of an *A1* message. This ensures the performance of the system in reality.

The time synchronization network is vulnerable to cyber-attacks [32–34]. This will reduce the time synchronization accuracy of the network. In more serious cases, it may even cause damage to the entire network [32–34]. The enhanced time synchronization method sends the *A1* message, *51* message, and *A2* message in sequence. If the slave clock receives an *A2* message before receiving an *A1* message, the slave clock does not process the *A2* message and considers it to be an illegal packet. Furthermore, the master clock periodically sends an *A1* message through the sending interval of the *A1* message set by the register. That is, for the master clock, the difference between the time of sending an *A1* message this time and the time of sending an *A1* message the last time is fixed. The sending time of the last *A1* message is known, and if the timestamp carried by a packet minus the sending time of the last *A1* message is not equal to the fixed sending interval of the *A1* message, then the packet is invalid. Sending an *A1* message periodically provides a good barrier to security. In addition, the enhanced time synchronization method sets a threshold for the time count offset. If the time count offset calculated by the slave clock exceeds the threshold, it is considered that the time synchronization has been attacked, and the time synchronization will not be performed this time. When the network using the enhanced time synchronization method is subjected to cyber-attacks, the above mechanism ensures the normal operation of time synchronization.

Through the analysis of the proposed new underlying transmission architecture, new synchronization messages, and the enhanced time synchronization method, We can know that the slave clock can synchronize its local clock to the master clock in real-time. We analyze the performance of the proposed method based on new synchronization messages through concrete simulations in Section 4.

4. Simulation-Based Evaluation and Analysis of the Results

The crystal oscillator runs at the nominal frequency under ideal conditions, and the specific time is obtained by accumulating the number of clock periods running. However, due to the instability of its own operating state and the influence of external environmental factors, the crystal oscillator cannot run at the nominal frequency under normal conditions. The time error of the crystal oscillator mainly includes two parts: systematic error and random error [35,36]. The crystal oscillator may generate a systematic error due to aging, frequency drift, or being affected by different external environmental factors such as temperature and air pressure. It is internal and inherent deviation of the crystal oscillator. The systematic error of the crystal oscillator is generally expressed by its frequency error, including two parts: initial frequency error Δf_0 and frequency drift coefficient D. The unit of the frequency error is Parts Per Million (PPM), which is a dimensionless unit. The total frequency error of the crystal oscillator corresponding to the ideal standard time t is shown in Equation (6).

$$\Delta f(t) = \Delta f_0 + Dt \tag{6}$$

where $\Delta f(t)$ is the total frequency error of the crystal oscillator and Δf_0 is the initial frequency error of the crystal oscillator, which is a fixed value. Different crystal oscillators have different Δf_0 values. D is the frequency drift coefficient of the crystal oscillator. In practice, D usually changes in real-time due to the influence of external environmental factors [37]. The *Dt* item is a one-time item, indicating that it increases or decreases linearly with time.

The crystal oscillator may generate a random error $\varepsilon(t)$ due to the influence of uncertain factors such as noise and jitter during operation. The jitter error is related to the minimum resolution of the crystal oscillator. For example, a crystal oscillator with a frequency of 80 MHz has a jitter error of 12.5 ns. In fact, the random error has a limited impact on the time synchronization accuracy. Time error *e*(*t*) of the crystal oscillator can be calculated by Equation (7) according to the total frequency error of the crystal oscillator calculated by Equation (6).

$$e(t) = e_0 + \int_0^t \Delta f(t)dt + \varepsilon(t) = e_0 + \Delta f_0 t + \frac{1}{2}Dt^2 + \varepsilon(t) \tag{7}$$

where *e*(*t*) is the time error of the crystal oscillator, e_0 is the initial time error corresponding to initial time t = 0, and $\varepsilon(t)$ is the random error generated by the crystal oscillator during operation.

According to Equation (7), we can obtain the recursive form of the time error of the crystal oscillator shown in Equation (8).

$$e(t) = e(t') + \Delta f_0(t - t') + \frac{1}{2}D(t - t')^2 + \varepsilon(t) - \varepsilon(t') \tag{8}$$

where t is the current monitoring time and t' is the previous monitoring time.

Therefore, corresponding to the ideal standard time t, the local time *LocalTime*(*t*) of the crystal oscillator is shown in Equation (9).

$$LocalTime(t) = t + e(t) = t + e(t') + \Delta f_0(t - t') + \frac{1}{2}D(t - t')^2 + \varepsilon(t) - \varepsilon(t') \tag{9}$$

where *LocalTime*(*t*) is the local time of the crystal oscillator corresponding to the ideal standard time t and *e*(*t*) is the time error of the crystal oscillator.

In practice, it is difficult to obtain an extremely accurate clock model because the model of the crystal oscillator is very complex [36,38]. Therefore, to simplify the clock model of the crystal oscillator, we assume that frequency drift coefficient D of the crystal oscillator is a fixed value [39].

We use the OMNeT++ simulator [40] to build the simulation model shown in Figure 4. The master clock is perfectly ticking at 80 MHz with no time error. The slave clock has a crystal oscillator with initial frequency error Δf_0 of 80 PPM, frequency drift coefficient D of 10^{-10} PPM/s, jitter of 12.5 ns, and frequency of 80 MHz. TimeOffsetObserver is a module that monitors the time count offset at regular intervals. We set its monitoring interval to 0.15 ms in this paper. The master clock and the slave clock exchange messages through a 250-Mbps Low-Voltage Differential Signaling (LVDS) twisted pair cable.

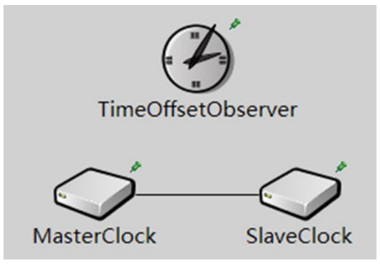

Figure 4. Simulation model built using the OMNeT++ simulator.

The delay experienced by a message sent from the sending node (the master clock or the slave clock) to the receiving node (the slave clock or the master clock) is composed of three parts, namely, sending delay, propagation delay, and receiving delay, as shown in

Figure 5. The sending delay is composed of processing delay and transmission delay. The specific meaning of each part of the delay is described in detail below.

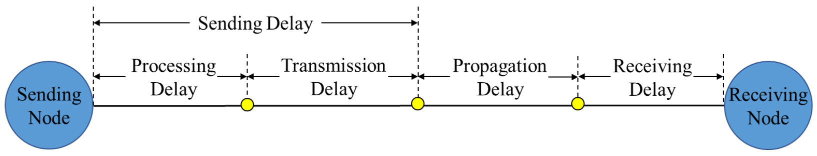

Figure 5. Delay experienced by a message sent from the sending node to the receiving node.

Sending delay: This is defined as the duration between when the last bit of the data of an FPGA or a Micro Control Unit (MCU) is available to the communication services of the sending node and when the last bit of the corresponding frame is transmitted on the transmission medium. As can be seen from its definition, it is composed of two parts: processing delay and transmission delay. The processing delay is the time for the sending node to process the sent message, and it is generally a fixed value. For the new bus developed by our center, the processing delay of the sending node is 80 ns. The transmission delay is the time it takes for a frame to transmit on the transmission medium, which is equal to the length of the frame divided by the bandwidth of the transmission link. The frame length of the new synchronization message proposed in this paper is 10 bytes.

Propagation delay: This is defined as the time it takes for an electromagnetic signal or optical signal to travel a certain distance in a transmission medium. Specifically, it is equal to the length of the transmission channel divided by the propagation rate of the signal in the transmission medium. The propagation delay of the twisted pair cable is typically 5 ns per 1 m cable [41]. The transmission medium of Figure 4 uses a 2 m long twisted pair cable. Thus, the network has a propagation delay of 10 ns.

Receiving delay: This is defined as the duration between when the last bit of a frame is received on the transmission medium and when the last bit of the corresponding data is available to the FPGA or the MCU of the receiving node. It is also generally a fixed value. For the new bus developed by our center, the receiving delay of the receiving node is 80 ns.

Simulation environment configurations for Figure 4 are shown in Table 1 according to the specific description of the simulation model above.

Table 1. Simulation environment configurations.

Parameters	**Configuration**
master clock frequency	80 MHz
slave clock frequency	80 MHz
slave clock initial frequency error Δf_0	80 PPM
slave clock frequency drift coefficient D	10^{-10} PPM/s
slave clock jitter	12.5 ns
monitoring interval	0.15 ms
A1 message sending interval	15.625 ms
transmission link bandwidth	250 Mbps
frame length	10 bytes
propagation delay	10 ns
processing delay of the sending node	80 ns
receiving delay of the receiving node	80 ns

First, we analyze the performance of purely IEEE 1588 PTP. The slave clock is set to synchronize with the master clock every 15.625 ms. We run 1000 time synchronization rounds, which is a total of 15,625 ms. Since the data points of the time count offset curve monitored by the TimeOffsetObserver module from 0 ms to 15,625 ms are too dense to see the trend of the curve clearly, we only show the time count offset curve from 0 ms to 2062.5 ms as shown in Figure 6. The time count offset curve after 2062.5 ms is exactly the same as the curve in Figure 6. Figure 7 shows a distribution histogram of the time count offset.

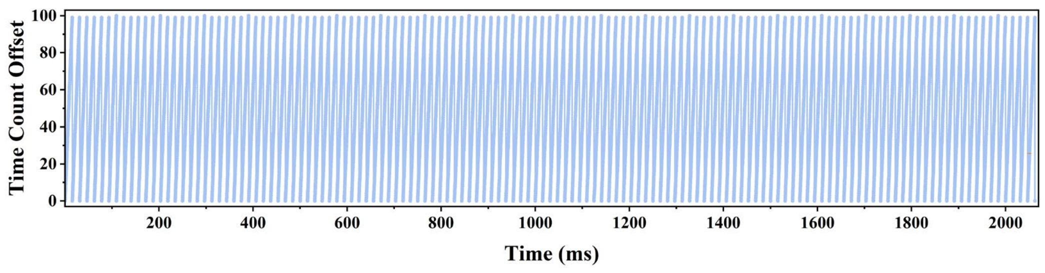

Figure 6. Time count offset curve using purely IEEE 1588 PTP.

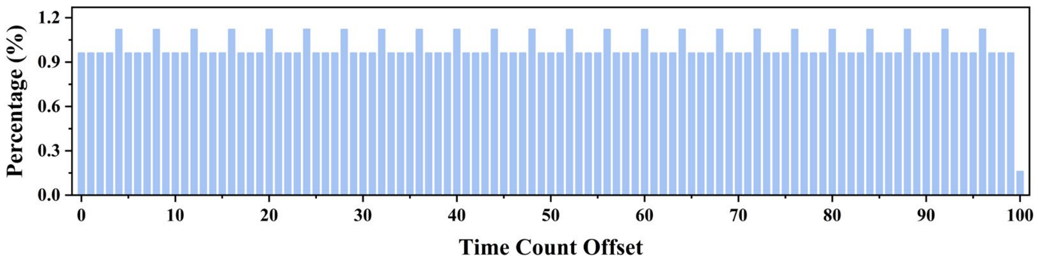

Figure 7. Distribution histogram of the time count offset using purely IEEE 1588 PTP.

From Figure 6, we can see that it is a sawtooth-shaped curve. At initial moment t = 0, the time count offset is 0. This means that, at the initial moment, the time of the master clock is exactly the same as that of the slave clock. Afterward, with the progression of time, the time offset becomes larger and larger due to the time error of the slave clock, up to a maximum of 100 clock periods. This process corresponds to the first oblique line in Figure 6. The clock period is 12.5 ns under a crystal oscillator frequency of 80 MHz. Therefore, 100 clock periods correspond to 1.25 us. When the slave clock receives an *A2* message in the first time synchronization round, the slave clock synchronizes with the master clock. At this point, the time of the slave clock is exactly the same as that of the master clock. Corresponding to Figure 6, the curve suddenly drops from the first highest point to the lowest point. This is a result of time synchronization in action. However, after that, with the progression of time, the time offset becomes larger and larger again due to the time error of the slave clock. This process corresponds to the second oblique line in Figure 6. When the slave clock receives an *A2* message in the second time synchronization round, the slave clock synchronizes with the master clock. At this point, the time of the slave clock is again exactly the same as that of the master clock. Corresponding to Figure 6, the curve suddenly drops from the second highest point to the lowest point once again. This is still a result of time synchronization in action. The above process is repeated continuously. After 1000 time synchronization rounds are performed, a sawtooth-shaped curve is finally formed. From Figure 7, we can see that the percentage of counts for time count offset with values from 0 to 100 is basically the same during network synchronization. This is because, as we can see from Figure 6, the curve corresponding to each time synchronization round is basically the same. Through the above-mentioned analysis, we conclude that the synchronization accuracy achieved by purely IEEE 1588 PTP is at the microsecond level. By increasing the clock frequency and the synchronization rate, sub-microsecond synchronization accuracy can be achieved. However, the time synchronization accuracy is seriously affected by the time error of the slave clock. Therefore, we must enhance purely IEEE 1588 PTP to solve this problem.

Then, we analyze the performance of the proposed method. The slave clock is set to synchronize with the master clock every 15.625 ms. We run 1000 time synchronization rounds, which is a total of 15,625 ms. The time count offset curve monitored by the TimeOffsetObserver module every 0.15 ms is shown in Figure 8. The figure in the rectangle marked with red-dotted lines in Figure 8 is an enlarged display of the time count offset curve from 0 ms to 31.35 ms. Figure 9 shows a distribution histogram of the time count offset.

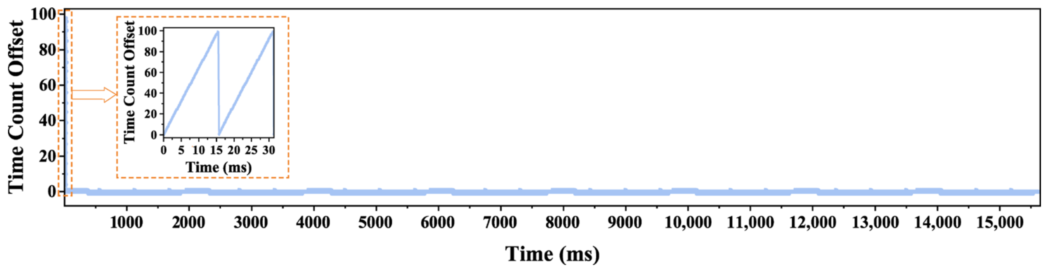

Figure 8. Time count offset curve using the enhanced time synchronization method.

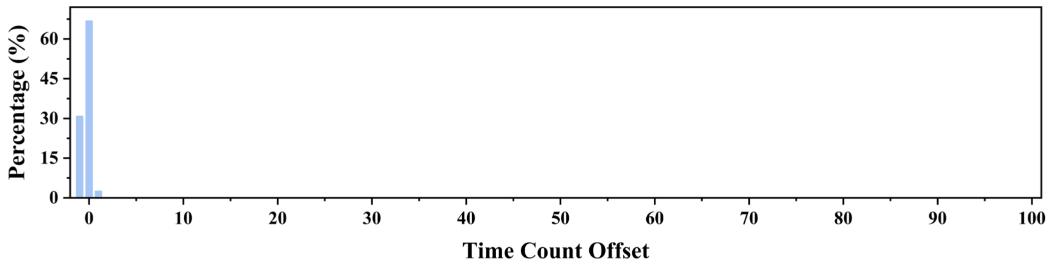

Figure 9. Distribution histogram of the time count offset using the enhanced time synchronization method.

From Figure 8, we can see that at initial moment t = 0, the time count offset is 0. This means that at the initial moment, the time of the master clock is exactly the same as that of the slave clock. Afterward, with the progression of time, the time offset becomes larger and larger due to the time error of the slave clock. This process corresponds to the first oblique line in Figure 8. When the slave clock receives an *A2* message, it synchronizes with the master clock. At this point, the time of the slave clock is exactly the same as that of the master clock. Corresponding to Figure 8, the curve suddenly drops from the first highest point to the lowest point. This is a function of purely IEEE 1588 PTP. However, the time offset becomes larger and larger again over time due to the time error of the slave clock. This process corresponds to the second oblique line in Figure 8. When the slave clock receives an *A2* message again, it synchronizes with the master clock. At this point, the time of the slave clock is again exactly the same as that of the master clock. Corresponding to Figure 8, the curve suddenly drops from the second highest point to the lowest point once again. This is still a function of purely IEEE 1588 PTP. The time synchronization process above is the same as that of purely IEEE 1588 PTP. After that, the time count offset will still be larger and larger due to the time error of the slave clock if purely IEEE 1588 PTP is used for time synchronization. However, the proposed method can compensate for the time error of the slave clock by Equation (5). In this way, the slave clock synchronizes its local clock to the master clock both when it receives an *A2* message and in the interval between two *A2* messages it receives. Therefore, the slave clock is always in a state of being synchronized with the master clock. From Figure 9, we can clearly see that, during network synchronization, the percentage of counts for time count offset with values -1, 0, and 1 is basically close to 100%. This is because, as we can see from Figure 8, the time count offset is either -1, 0, or 1 after the second time synchronization round. We realize the purpose of timing the master clock and the slave clock by counting the period of the clock signal. Therefore, we need to round down the time count to an integer. This is the reason why -1 and 1 appear at the same time. The clock period is 12.5 ns under a crystal oscillator frequency of 80 MHz. From the above analysis, we can see that, after the proposed method works, the time offset is at most ± 1 clock period; that is, ± 12.5 ns under a crystal oscillator frequency of 80 MHz. Therefore, the enhanced time synchronization method can achieve nanosecond-level synchronization accuracy.

Next, we study the impact of the crystal oscillator accuracy of the slave clock on the performance of the proposed method. In addition to the crystal oscillator accuracy, the values of other parameters are shown in Table 1. We still run 1000 time synchronization rounds, that is, a total of 15,625 ms. We choose three different cases where the initial frequency error Δf_0 of the crystal oscillator of the slave clock is 20 PPM, 50 PPM, and 80 PPM, respectively, to analyze their impact on the time synchronization accuracy. After the enhanced time synchronization method runs stably—that is, after the slave clock receives an *A2* message in the second time synchronization round, according to the time count offset monitored every 0.15 ms by the TimeOffsetObserver module starting from 31.35 ms—Table 2 shows the time synchronization accuracy that the slave clocks with different crystal oscillator accuracy can achieve and the mean value and the standard deviation of the time offset.

Table 2. Time synchronization accuracy and time offset's mean value and standard deviation under different crystal oscillator accuracies.

Initial Frequency Error Δf_0 of the Crystal Oscillator (PPM)	Time Synchronization Accuracy (ns)	Mean Value (ns)	Standard Deviation (ns)
20	±12.5	−3.42	6.24
50	±12.5	−3.49	6.25
80	±12.5	−3.56	6.26

From Table 2, we can see that, when the frequency of the master clock and the slave clock are both 80 MHz, although initial frequency error Δf_0 of the slave clock is different, they can all achieve a time synchronization accuracy of ±12.5 ns after the enhanced time synchronization method runs stably. This shows that the enhanced time synchronization method can well-compensate the time error of the slave clock regardless of the crystal oscillator accuracy. As initial frequency error Δf_0 of the slave clock becomes larger and larger, the mean value and the standard deviation of the time offset become larger and larger. This shows that, as the crystal oscillator accuracy of the slave clock becomes lower and lower, the jitter of the achieved time synchronization accuracy becomes larger and larger. This is in line with the actual situation. At the same time, we can also see from Table 2 that the mean value and the standard deviation of the time offset change very little, which can keep the time offset in a stable state.

Finally, we study the impact of the crystal oscillator frequency on the performance of the proposed method. In addition to the crystal oscillator frequency, the values of other parameters are shown in Table 1. We still run 1000 time synchronization rounds, that is, a total of 15,625 ms. We choose three different cases where the frequency of the master clock and the slave clock are both 50 MHz, 80 MHz, and 125 MHz, respectively, to analyze their impact on the time synchronization accuracy. After the enhanced time synchronization method runs stably—that is, after the slave clock receives an *A2* message in the second time synchronization round, according to the time count offset monitored every 0.15 ms by the TimeOffsetObserver module starting from 31.35 ms—the time synchronization accuracy under different crystal oscillator frequencies is shown in Table 3.

Table 3. Time synchronization accuracy under different crystal oscillator frequencies.

Crystal Oscillator Frequency (MHz)	Time Synchronization Accuracy (ns)
50	±20
80	±12.5
125	±8

From Table 3, we can see that when initial frequency error Δf_0 of the crystal oscillator of the slave clock is 80 PPM, the time synchronization accuracy that can be achieved by the system with the crystal oscillator frequency of 50 MHz, 80 MHz, and 125 MHz is ±20 ns, ±12.5 ns, and ±8 ns, respectively, after the enhanced time synchronization method runs stably. We can see from the results in Table 3 that, as the frequency of the crystal oscillator increases, the time synchronization accuracy achieved by the system also increases. Therefore, we can improve the time synchronization accuracy by increasing the frequency of the crystal oscillator.

The time synchronization accuracy achieved by the method proposed in this paper and the methods proposed in the existing literature are shown in Table 4. Ref. [10] achieves a time synchronization accuracy of ±50 ns under a crystal oscillator frequency of 50 MHz by compensating the frequency drift of the crystal oscillator and the quantization errors of the timestamps. Ref. [17] achieves a time synchronization accuracy of 500 ns under a crystal oscillator frequency of 100 MHz by compensating the path delay error of synchronization messages. Both Ref. [18] and Ref. [19] present FPGA implementations of IEEE 1588 PTP. Ref. [18] achieves 97.76% of the time offset within the range of ±40 ns. Ref. [19] achieves a time synchronization accuracy of ±16 ns under a crystal oscillator frequency of 62.5 MHz through a dedicated high-cost hardware circuit for specific applications. Ref. [21] achieves a time offset of less than 20 ns under a crystal oscillator frequency of 125 MHz by compensating the path asymmetry. The proposed method in this paper compensates the time error of the slave clock based on the proposed new underlying transmission architecture and new synchronization messages. It achieves a time synchronization accuracy of ±12.5 ns under a crystal oscillator frequency of 80 MHz by using simple protocol and low-cost hardware such as conventional circuits and cables. It realizes double unification of high-precision time synchronization and economy.

Table 4. Comparison of time synchronization accuracy achieved by different methods.

Different Methods	Crystal Oscillator Frequency (MHz)	Time Synchronization Accuracy (ns)
[10]	50	±50
[17]	100	500
[18]	Not stated	±40
[19]	62.5	±16
[21]	125	20
Proposed method in this paper	80	±12.5

5. Conclusions

Based on IEEE 1588 PTP, we proposed a new underlying transmission architecture, new synchronization messages, and an enhanced time synchronization method on these bases. First, we proposed a new underlying transmission architecture with a frame length of 10 bytes and new synchronization messages. Then, to solve the time error problem of the crystal oscillator of the slave clock, we proposed an enhanced time synchronization method using new synchronization messages. Finally, we built a simulation model on the OMNeT++ simulator to evaluate the performance of the proposed method. Furthermore, we compared the impact of different crystal oscillator accuracies and different crystal oscillator frequencies on time synchronization accuracy respectively. Simulation results show that the time offset is at most ±1 clock period. That is, a time synchronization accuracy of ±12.5 ns can be achieved under a crystal oscillator frequency of 80 MHz. With a reduction in the crystal oscillator accuracy of the slave clock, synchronization accuracy of ±1 clock period can still be achieved. With an increase in the crystal oscillator frequency, time synchronization accuracy that can be achieved also improves. The evaluations show that a nanosecond-level time synchronization accuracy can be achieved by the method proposed in this paper. It provides a new way of thinking and has certain guiding significance for improving the time synchronization accuracy of time-critical systems.

Author Contributions: Conceptualization, F.L. and W.L.; methodology, F.L.; software, F.L.; validation, F.L. and Y.Q.; formal analysis, F.L. and Y.Q.; investigation, F.L., Y.Q. and Q.L.; resources, W.L.; data curation, F.L., Q.L. and G.L.; writing—original draft preparation, F.L.; writing—review and editing, F.L. and W.L.; visualization, F.L. and G.L.; supervision, W.L., Q.L. and G.L.; project administration, F.L. and W.L.; funding acquisition, W.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Innovative Research Group Project of the National Science Foundation of China, China (51821003) and the National Science Foundation of Shanxi Province, China (201701D121065).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alvarez, I.; Ballesteros, A.; Barranco, M.; Gessner, D.; Djerasevic, S.; Proenza, J. Fault Tolerance in Highly Reliable Ethernet-Based Industrial Systems. *Proc. IEEE* **2019**, *107*, 977–1010. [CrossRef]
2. Idrees, Z.; Granados, J.; Sun, Y.; Latif, S.; Gong, L.; Zou, Z.; Zheng, L.R. IEEE 1588 for Clock Synchronization in Industrial IoT and Related Applications: A Review on Contributing Technologies, Protocols and Enhancement Methodologies. *IEEE Access* **2020**, *8*, 155660–155678. [CrossRef]
3. Puttnies, H.; Danielis, P.; Sharif, A.R.; Timmermann, D. Estimators for Time Synchronization—Survey, Analysis, and Outlook. *IoT* **2020**, *1*, 398–435. [CrossRef]
4. Kero, N.; Puhm, A.; Kernen, T.; Mroczkowski, A. Performance and Reliability Aspects of Clock Synchronization Techniques for Industrial Automation. *Proc. IEEE* **2019**, *107*, 1011–1026. [CrossRef]
5. Mills, D.; Martin, J.; Burbank, J.; Kasch, W. Network time protocol version 4: Protocol and algorithms specification. Technical Report. 2010. Available online: https://tools.ietf.org/html/rfc5905/ (accessed on 18 March 2023).
6. *IEEE Std 1588-2008*; IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems; Revision of IEEE Std 1588-2002. IEEE: New York, NY, USA, 2008; pp. 1–300.
7. Liu, H.; Liu, J.; Bi, T.; Li, J.; Yang, W.; Zhang, D. Performance analysis of time synchronization precision of PTP in smart substations. In Proceedings of the 2015 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS), Beijing, China, 11–16 October 2015; pp. 37–42. [CrossRef]
8. Giorgi, G.; Narduzzi, C. Performance Analysis of Kalman-Filter-Based Clock Synchronization in IEEE 1588 Networks. *IEEE Trans. Instrum. Meas.* **2011**, *60*, 2902–2909. [CrossRef]
9. Karthik, A.K.; Blum, R.S. Robust Clock Skew and Offset Estimation for IEEE 1588 in the Presence of Unexpected Deterministic Path Delay Asymmetries. *IEEE Trans. Commun.* **2020**, *68*, 5102–5119. [CrossRef]
10. Zhang, J.; Zhang, W. A Disturbance Rejection Control Approach for Clock Synchronization in IEEE 1588 Networks. *J. Syst. Sci. Complex.* **2018**, *31*, 1437–1448. [CrossRef]
11. Seijo, O.; Lopez-Fernandez, J.A.; Bernhard, H.P.; Val, I. Enhanced Timestamping Method for Subnanosecond Time Synchronization in IEEE 802.11 Over WLAN Standard Conditions. *IEEE Trans. Ind. Inform.* **2020**, *16*, 5792–5805. [CrossRef]
12. Shuai, L.; Lu, Y.; Ji, Y. An Enhanced IEEE 1588 Time Synchronization for Asymmetric Communication Link in Packet Transport Network. *IEEE Commun. Lett.* **2010**, *14*, 764–766. [CrossRef]
13. Du, Z.P.; Lu, Y.M.; Ji, Y.F. An Enhanced End-to-End Transparent Clock Mechanism with a Fixed Delay Ratio. *IEEE Commun. Lett.* **2011**, *15*, 872–874. [CrossRef]
14. Giorgi, G.; Narduzzi, C. A resilient Kalman filter based servo clock. In Proceedings of the 2013 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS) Proceedings, Lemgo, Germany, 22–27 September 2013. [CrossRef]
15. Maegawa, R.; Matsui, D.; Yamasaki, Y.; Ohsaki, H. A Discrete Model of IEEE 1588-2008 Precision Time Protocol with Clock Servo using PI Controller. In Proceedings of the 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), Milwaukee, WI, USA, 15–19 July 2019; pp. 531–536. [CrossRef]
16. Hajikhani, M.J.; Kunz, T.; Schwartz, H. A Recursive Method for Clock Synchronization in Asymmetric Packet-Based Networks. *IEEE ACM Trans. Netw.* **2016**, *24*, 2332–2342. [CrossRef]
17. Wei, Y.; Li, K.; Tian, C.; Zhang, X. Analysis and Correction Methods for Network Time-delay Error of IEEE 1588 Synchronization Clock. In Proceedings of the 2020 IEEE 4th Conference on Energy Internet and Energy System Integration (EI2), Wuhan, China, 30 October–1 November 2020; pp. 4015–4020. [CrossRef]
18. Yin, H.; Fu, P.; Qiao, J.; Li, Y. The implementation of IEEE 1588 clock synchronization protocol based on FPGA. In Proceedings of the 2018 IEEE International Instrumentation and Measurement Technology Conference, Houston, TX, USA, 14–17 May 2018. [CrossRef]
19. Pedretti, D.; Bellato, M.; Isocrate, R.; Bergnoli, A.; Brugnera, R.; Corti, D.; Dal Corso, F.; Galet, G.; Garfagnini, A.; Giaz, A.; et al. Nanoseconds Timing System Based on IEEE 1588 FPGA Implementation. *IEEE Trans. Nucl. Sci.* **2019**, *66*, 1151–1158. [CrossRef]

20. Nagra, A.S.; Allahi, I.; Pasha, M.A.; Masud, S. Design and FPGA based Implementation of IEEE 1588 Precision Time Protocol for Synchronisation in Distributed IoT Applications. *Aust. J. Electr. Electron. Eng.* **2022**, *19*, 31–39. [CrossRef]
21. Wu, Q.; Yang, L.Z.; Chen, J.H. Enhancement for Real-Time Ethernet Clock Synchronization by Internal Processing Delay Measurement. *IEEE Commun. Lett.* **2019**, *23*, 2063–2067. [CrossRef]
22. Liu, X.J.; Wang, H. Embedded Clock Skew Estimation in Industrial Networks. *IEEE Commun. Lett.* **2022**, *26*, 1873–1877. [CrossRef]
23. Zuo, Y.; Wang, X.; Zhang, B. An optimization method of clock synchronization for large-scale regional power network based on IEEE 1588. In Proceedings of the 2021 International Conference on Power Electronics and Power Transmission (ICPEPT), Hangzhou, China, 24–26 September 2021; pp. 1–5. [CrossRef]
24. Nguyen, V.Q.; Nguyen, T.H.; Jeon, J.W. An Adaptive Fuzzy-PI Clock Servo Based on IEEE 1588 for Improving Time Synchronization Over Ethernet Networks. *IEEE Access* **2020**, *8*, 61370–61383. [CrossRef]
25. Einspieler, S.; Rathakrishnan, N.; Prabhakara, A.; Steinwender, B.; Elmenreich, W. High Accuracy Software-Based Clock Synchronization Over CAN. *IEEE Trans. Syst. Man Cy-S* **2022**, *52*, 4438–4446. [CrossRef]
26. Huan, X.; Kim, K.S.; Lee, S.; Lim, E.G.; Marshall, A. A Beaconless Asymmetric Energy-Efficient Time Synchronization Scheme for Resource-Constrained Multi-Hop Wireless Sensor Networks. *IEEE Trans. Commun.* **2020**, *68*, 1716–1730. [CrossRef]
27. Wang, H.; Lu, R.; Peng, Z.; Li, M. Timestamp-Free Clock Parameters Tracking Using Extended Kalman Filtering in Wireless Sensor Networks. *IEEE Trans. Commun.* **2021**, *69*, 6926–6938. [CrossRef]
28. Fang, Z.; Gao, Y. Delay Compensated One-Way Time Synchronization in Distributed Wireless Sensor Networks. *IEEE Wirel. Commun. Lett.* **2022**, *11*, 2021–2025. [CrossRef]
29. *IEEE Std 1588-2019*; IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control System; Revision of IEEE Std 1588-2008. IEEE: New York, NY, USA, 2020.
30. Girela-López, F.; López-Jiménez, J.; Jiménez-López, M.; Rodríguez, R.; Ros, E.; Díaz, J. IEEE 1588 high accuracy default profile: Applications and challenges. *IEEE Access* **2020**, *8*, 45211–45220. [CrossRef]
31. Li, F.; Liu, W.Y.; Gao, W.J.; Liu, Y.F.; Hu, Y.J. Design and Reliability Analysis of a Novel Redundancy Topology Architecture. *Sensors* **2022**, *22*, 2582. [CrossRef] [PubMed]
32. Valdivia, L.J.; Adin, I.; Añorga, J.; Arrizabalaga, S.; Mendizabal, J. Coexistence of safety and security: Synchronized redundant system with security enhancements. *Qual. Reliab. Eng. Int.* **2019**, *35*, 561–571. [CrossRef]
33. Alghamdi, W.; Schukat, M. Precision time protocol attack strategies and their resistance to existing security extensions. *Cybersecurity* **2021**, *4*, 12. [CrossRef]
34. Alghamdi, W.; Schukat, M. A Security Enhancement of the Precision Time Protocol Using a Trusted Supervisor Node. *Sensors* **2022**, *22*, 3671. [CrossRef]
35. Allan, D.W. Time and Frequency (Time-Domain) Characterization, Estimation, and Prediction of Precision Clocks and Oscillators. *IEEE Trans. Ultrason. Ferroelectr. Freq. Control.* **1987**, *34*, 647–654. [CrossRef]
36. Allan, D.W.; Barnes, J.; Cordara, F.; Garvey, M.; Hanson, W.; Kinsman, R.; Kusters, J.; Smythe, R.; Walls, F.L. Precision Oscillators: Dependence of Frequency on Temperature, Humidity and Pressure. In Proceedings of the 1992 IEEE Frequency Control Symposium, Hershey, PA, USA, 27–29 May 1992; pp. 782–793. [CrossRef]
37. Schriegel, S.; Jasperneite, J. Investigation of Industrial Environmental Influences on Clock Sources and their Effect on the Synchronization Accuracy of IEEE 1588. In Proceedings of the 2007 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, Vienna, Austria, 1–3 October 2007; pp. 50–55. [CrossRef]
38. Gaderer, G.; Nagy, A.; Loschmidt, P.; Kero, N. A Novel, High Resolution Oscillator Model for DES Systems. In Proceedings of the 2008 IEEE International Frequency Control Symposium, Honolulu, HI, USA, 19–21 May 2008; pp. 178–183. [CrossRef]
39. Liu, Y.S.; Yang, C. OMNeT++ Based Modeling and Simulation of the IEEE 1588 PTP Clock. In Proceedings of the 2011 International Conference on Electrical and Control Engineering, Yichang, China, 16–18 September 2011; pp. 4602–4605. [CrossRef]
40. OMNeT++ Discrete Event Simulator. Available online: https://omnetpp.org/ (accessed on 1 January 2023).
41. Prytz, G. A performance analysis of EtherCAT and PROFINET IRT. In Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation, Hamburg, Germany, 15–18 September 2008; pp. 408–415. [CrossRef]