Time Synchronization

Time (or clock) synchronization is a large and vital field of research, as synchronization is a precondition for many applications. A few example applications are distributed data acquisition, distributed databases, and real-time communication.

1. Introduction

There are many applications of time synchronization in the IoT (Internet of Things) and IIoT (Industrial Internet of Things). In a smart factory, the production robots which work together on a production line must be synchronized precisely. Exact time synchronization is also important when multiple motors move one mechanical load [1][2] or, in general, when drives work together [3]. In a smart power plant or in the area of energy supply, time synchronization is of decisive importance both for the analysis of blackouts and for controlling the stability of the power grid [4].

In addition, time synchronization is essential for communication based on TDMA (Time Division Multiple Access). TDMA is used, e.g., in real-time networks. Each device is given an exclusive transmission time slot. Compliance with these time slots prevents collisions. The better the devices are synchronized, the more precisely the time slots can be adhered. Consequently, the available bandwidth can be utilized better [5]. However, if the devices are inaccurately synchronized, slack times must be inserted to guarantee that only one device is transmitting at a specific point in time. This leads to unused bandwidth. TDMA methods are used in practice in many real-time Ethernet protocols such as Ethernet Powerlink or SERCOS III. Furthermore, wireless real-time communication is typically based on TDMA [6]. In addition to real-time networks, TMDA is also used in WSNs (Wireless Sensor Networks). Here, the sensor nodes communicate, e.g., only for a short time period and otherwise remain in a sleep mode to save energy [7][8].

Another application of time synchronization is distributed data acquisition. An example is the nuclear fusion experiment W7-X (Wendelstein 7-X). The W7-X experiment has a relatively large spatial expansion as the diameter of the vacuum vessel is approximately 16 meters. Above all, it is a highly dynamic system (magnetically enclosed plasma). Both the control of the system and the scientific evaluation of the experiments require measurement timestamps with accuracies in the range of nanoseconds [9]. Measured parameters are, e.g., temperature, electron density, and plasma properties [10]. These requirements also apply to similar large-scale experiments such as the particle accelerators at CERN (European Organization for Nuclear Research) [11]. Another example of distributed data acquisition as an application of time synchronization is the interconnection of several radio telescopes in order to achieve a significantly higher resolution. If several such telescopes are distributed over different continents, the data needs to be timestamped precisely in order to be fused in postprocessing. With this method, the Event Horizon Telescope could for the first time take a picture of the direct vicinity of a black hole [12]. Another example is the observation of the stability of bridges, stadiums, and high-rise buildings using wireless sensors. Here, a synchronization accuracy of at least 120 μs is required to enable detecting vibrations [13].

In general, time synchronization enables ensuring the sequence of events [5]. Therefore, it is also essential for the financial world and online trading as the sequence of transactions is particularly important here [5]. Time synchronization is also important for distributed databases to ensure consistency as well as to optimize throughput and latency [5].

2. Time Synchronization Protocols

This section deals with existing protocols for time synchronization. These primarily relate to wired networks. Table 1 shows a tabular overview of this.

Table 1. Overview of protocols for time synchronization. It shows their standard compatibility (SC), the estimation method used, the accuracy, as well as conditions and restrictions (MV = mean value, RTT = round-trip time, KF = Kalman filter, and PLL = phase-locked loop).

Approach	SC	Estimator	Accuracy	Cond./restr.
NTP [14]	Yes	RTT	1ms	None
PTP [15]	Yes	RTT	<1µs	Custom HW
gPTP [16]	Yes	RTT	<1µs	Custom HW
PTCP [17][18]	Yes	RTT	<1µs	Custom HW
PTCP-KF [19]	Yes	KF	20-200ns	Eval. with Matlab
SyncE [20]	Yes	Offset: MV Freq.: PLL	Freq.: 4 ppm	Custom HW
c SyncE + Offset [21]	No	Offset: MV Freq.: PLL	NI Freq.: 4 ppm	Custom HW

2.1. NTP

The Network Time Protocol (NTP) [14] is standard-compliant, uses the RTT (round-trip time) as estimator, and has an accuracy of approximately 1 ms. It is also the most commonly used synchronization protocol on the Internet. NTP is a pure SW protocol, so that no changes are required at the lower network layers. NTP clients send packets to the NTP server at certain times and wait for the response. The packet's RTT can be calculated from the time the response was received. From this, in turn, the latency can be estimated assuming symmetrical packet delays. An accuracy of approximately 1 ms can be achieved on the Internet with this protocol. On the network level, there is a hierarchical structure of participating servers. The advantage is that large networks can be synchronized more efficiently with several servers and the load on the server is reduced. However, the accuracy decreases in this case due to the propagation of errors along the hierarchy. NTP is not suitable for real-time IIoT applications due to its limited accuracy, which decreases even further when delay variations occur [22].

2.2. PTP

The Precision Time Protocol (PTP) [15][23][24] specified in IEEE 1588 is standard-compliant. It uses the RTT or a simple sum as an estimator and achieves an accuracy below 1 µs. However, PTP requires specialized HW to achieve this accuracy (at least on the end nodes). Since there are three versions of PTP, these should be briefly classified here. The first version was PTPv1 1588-2002 [24]. The next version, PTPv2 or 1588-2008 [15], has no compatibility with PTPv1. The latest version, 1588-2019 or PTPv2.1, however, is fully downward compatible with PTPv2. PTP is designed to allow precise synchronization between devices. It is available as both a HW and a SW version. PTP synchronization begins with determining the device which has the most stable and accurate time. The node with the reference time (the so-called grandmaster clock) is selected using the Best Master Clock Algorithm (BMCA). Reference times are then sent to the slaves, which compare them with their own time. Slaves respond to the master with their own times at regular intervals. The delay from the master to the slave and the delay from the slave to the master can be determined from the timestamps in the messages in order to be able to synchronize the slaves. In the best case, the HW variant achieves an accuracy in the nanosecond range. The existing SW version uses timestamps from the application layer and, with a HW-supported reference time, achieves an accuracy of the order of microseconds. However, PTP assumes that the network delay is constant and symmetrical. Therefore, delay variations reduce the accuracy of PTP, as measured for example in [25][26] and simulated by the authors of this work in [27]. Such variations can occur when switches do not support PTP at the Media Access Control (MAC) layer. Good descriptions of PTP and an accuracy analysis can be found in [1][26][28]. Of all the

synchronization protocols that use HW timestamps, PTP is the most widely used protocol. Consequently, the IEEE-TSN committee used PTP to derive gPTP (Generalized Precision Time Protocol) as a TSN substandard. The left side of Figure 1 shows an example of a PTP network, and Figure 2 shows PTP's message exchange and delay estimation using RTT.

Figure 1. Example of a Precision Time Protocol (PTP) network and a Generalized Precision Time Protocol (gPTP) network: M marks that the corresponding Ethernet port is in the master role, and S marks the slave role. (We created the diagrams on our own based on [15][16]).

Figure 2. PTP's message exchange and delay estimation using RTT. The master starts by sending a Sync message and timestamps the sending moment as $t1$. The slave timestamps the receiving moment as $t2$. The master sends a Follow_Up message, if it cannot embed $t1$ directly into the Sync message. The slave sends a Delay_Req message and timestamps the sending moment as $t3$. The master timestamps the receiving moment as $t4$. Afterwards, the master sends a Delay_Resp message, which comprises $t4$. The slave estimates the delay as $[(t2-t1)+(t4-t3)]/2)$ assuming a symmetric delay ($tms=tsm$). (We created the diagrams on our own based on [15]).

2.3. gPTP

gPTP or IEEE 802.1AS [16] is standard-compliant. It uses the RTT or a simple sum as an estimator and achieves an accuracy below 1μs, but it is limited to 7 hops [16]. However, gPTP requires a specialized HW. In contrast to PTP, it requires this not only on the end nodes but also on the switches (Although gPTP itself is a standard, it massively improves the complexity of Ethernet's MAC layer. Therefore, we still refer to it as specialized HW). The authors of give a good overview of the gPTP standard, which is part of a series of standards originally developed by the Audio/Video Bridging (AVB) task group. In 2012, this group was renamed TSN. The aim of the TSN group is to develop an open standard for Ethernet-based real-time communication. In particular, they focus on the timing and synchronization, the reservation of resources and queues, as well as data forwarding. Basically, the synchronization of gPTP is similar to PTP. A modified version of PTP's BMCA is used. Each port of a so-called time-sensitive system measures the delay to its neighbors in a way that is similar to PTP. A bridge or an end station that meets the requirements of IEEE 802.1AS is referred to as a time-sensitive system. All nodes in the network (bridges and end stations) must be such time-sensitive systems. gPTP is a so-called PTP profile. As a result, a PTP implementation can be compliant with gPTP if it supports the gPTP profile. However, this is not required because a PTP implementation is considered to be standard-compliant if it supports the default PTP profile. The support of further PTP profiles (e.g., gPTP) is optional. In contrast to PTP, gPTP is also specified for wireless networks [16]. The right side of Figure 1 shows an example of a gPTP network.

2.4. PTCP

PTCP (Precision Transparent Clock Protocol) [17][18] is standard-compliant. It uses the RTT or a simple sum as an estimator and achieves an accuracy below 1 μs. However, PTCP requires specialized HW on switches and end nodes. PTCP is used for synchronization in the real-time Ethernet protocol PROFINET and, just like PROFINET, is standardized in IEC 61784-2 [17]. PTCP is also based on PTP. The most important difference is the use of "transparent clocks". In

contrast to PTP, direct neighbors are not synchronized, but only the end nodes with the master are synchronized. This is achieved as the switches compensate the time between receiving the Ethernet frame at the input port and sending the frame at the output port (by adapting the timestamps). These so-called "transparent clocks" have the advantage that the synchronization accuracy remains high even over many hops. In [19], PTPC is extended with a Kalman filter as an estimator. The standard compliance is retained.

2.5. SyncE

Another protocol is Synchronous Ethernet (SyncE) or ITU-T G.8262 [20]. SyncE is actually a real-time Ethernet protocol. Strictly speaking, however, it also offers a synchronization functionality for the frequency or skew. For frequency synchronization, the clock signal is transmitted at the physical (PHY) layer and the devices synchronize using phase-locked loops (PLLs). The achievable frequency accuracy is ±4 ppm. SyncE does not correct the offset. In [21], however, an extension of SyncE to include such an offset correction is suggested. The procedure is then no longer compliant with SyncE, but it was submitted to the ITD-T as an extension proposal.

2.6. Summary of Existing Time Synchronization Protocols

NTP is the most widely used protocol, as is does not need any special HW. PTP achieves a much higher precision than NTP by using a special HW at least on the end nodes. As PTP assumes the network delay to be constant and symmetrical, delay variations can reduce its accuracy. This is, e.g., experimentally shown in [25][26] and simulated by the author of this work in [27]. Such variations might occur, if non-PTP switches are used in the network (note that this fully complies with the PTP specification). The gPTP protocol (part of the TSN standards) demands special HW on all end notes and switches. Therefore, the delay in a gPTP network can be assumed to be constant. One benefit of gPTP and PTP regarding robustness is that both use the BMCA. The BMCA adaptively chooses a new reference clock if the original one fails.

References

1. Scheiterer, R.L.; Na, C.; Obradovic, D.; Steindl, G. Synchronization Performance of the Precision Time Protocol in Industrial Automation Networks. IEEE Trans. Instrum. Meas. 2009, 58, 1849–1857.

2. Chen, B.; Chen, Y.P.; Xie, J.M.; Zhou, Z.D.; Sa, J.M. Control methodologies in networked motion control systems. In Proceedings of the 2005 International Conference on Machine Learning and Cybernetics, Guangzhou, China, 18–21 August 2005; IEEE Operations Center: Piscataway, NJ, USA, 2005; Volume 2, pp. 1088–1093.

3. Felser, M. Real-Time Ethernet—Industry Prospective. Proc. IEEE 2005, 93, 1118–1129.

4. Steinhauser, F.; Riesch, C.; Rudigier, M. IEEE 1588 for time synchronization of devices in the electric power industry. In Proceedings of the International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS), Portsmouth, NH, USA, 27 September–1 October 2010; pp. 1–6.

5. Geng, Y.; Liu, S.; Yin, Z.; Naik, A.; Prabhakar, B.; Rosenblum, M.; Vahdat, A. Exploiting a natural network effect for scalable, fine-grained clock synchronization. In Proceedings of the USENIX Conference on Networked Systems Design and Implementation (NSDI) 2018, Renton, WA, USA, 9–11 April 2018.

6. Mahmood, A.; Exel, R.; Trsek, H.; Sauter, T. Clock synchronization over IEEE 802.11—A survey of methodologies and protocols. IEEE Trans. Ind. Inform. 2017, 13, 907–922.

7. Akhlaq, M.; Sheltami, T.R. RTSP: An Accurate and Energy-Efficient Protocol for Clock Synchronization in WSNs. IEEE Trans. Instrum. Meas. 2013, 62, 578–589.

8. Wu, Y.C.; Chaudhari, Q.; Serpedin, E. Clock Synchronization of Wireless Sensor Networks. IEEE Signal Process. Mag. 2011, 28, 124–138.

9. Schacht, J.; Laqua, H.; Niedermeyer, H. Synchronization of Processes in a Distributed Real Time System Exemplified by the Control System of the Fusion Experiment WENDELSTEIN 7-X. IEEE Trans. Nucl. Sci. 2006, 53, 2187–2194.

10. Wolf, R.C.; Ali, A.; Alonso, A.; Baldzuhn, J.; Beidler, C.; Beurskens, M.; Biedermann, C.; Bosch, H.S.; Bozhenkov, S.; Brakel, R.; et al. Major results from the first plasma campaign of the Wendelstein 7-X stellarator. Nucl. Fusion 2017, 57, 102020–102033.

11. Lipinski, M.; Wlostowski, T.; Serrano, J.; Alvarez, P. White rabbit: A PTP application for robust sub-nanosecond synchronization. In Proceedings of the 2011 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, Munich, Germany, 9–12 September 2011; pp. 25–30.

12. Akiyama, K.; Alberdi, A.; Alef, W.; Asada, K.; Azulay, R.; Baczko, A.K.; Ball, D.; Baloković, M.; Barrett, J.; Bintley, D.; et al. First M87 event horizon telescope results. IV. Imaging the central supermassive black hole. Astrophys. J. Lett. 2019, 875, L4.

13. Noel, A.B.; Abdaoui, A.; Elfouly, T.; Ahmed, M.H.; Badawy, A.; Shehata, M.S. Structural Health Monitoring Using Wireless Sensor Networks: A Comprehensive Survey. IEEE Commun. Surv. Tutor. 2017, 19, 1403–1423.

14. Mills, D.; Martin, J.; Burbank, J.; Kasch, W. Network Time Protocol Version 4: Protocol and Algorithms Specification, Internet Engineering Task Force (IETF) Request for Comments (RFC) 5905, June 2020.

15. IEEE 1588-2008 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, 24 July 2008. Available online: https://ieeexplore.ieee.org/document/4579760 (accessed on 10 November 2020).

16. IEEE 802.1AS-2011 Standard for Local and Metropolitan Area Networks—Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks, 30 March 2011. Available online: https://ieeexplore.ieee.org/document/5741898 (accessed on 10 November 2020).

17. IEC 61784-2. Digital Data Communications for Measurement and Control-Part 2: Additional Profiles for ISO/IEC 8802-3 Based Communication Networks in Real-Time Applications; German Institute for Standardisation (Deutsches Institut für Normung): Berlin, Germany, 2005.

18. Pigan, R.; Metter, M. Automating with PROFINET: Industrial Communication Based on Industrial Ethernet; Publicis MCD Werbeagentur GmbH: Somerset, UK, 2015.

19. Fontanelli, D.; Macii, D.; Rinaldi, S.; Ferrari, P.; Flammini, A. Performance analysis of a clock state estimator for PROFINET IO IRT synchronization. In Proceedings of the 2013 IEEE International Instrumentation and Measurement Technology Conference, Minneapolis, MN, USA, 6–9 May 2013; Staff, I., Ed.; pp. 1828–1833.

20. International Telecommunication Union. ITU-I G.8262: Timing Characteristics of a Synchronous Equipment Slave Clock, 29 November 2018. Available online: https://www.itu.int/rec/T-REC-G.8262 (accessed on 10 November 2020).

21. Hann, K.; Jobert, S.; Rodrigues, S. Synchronous ethernet to transport frequency and phase/time. IEEE Commun. Mag. 2012, 50, 152–160.

22. Bletsas, A. Evaluation of Kalman filtering for network time keeping. IEEE Trans. Ultrason. Ferroelectr. Freq. Control 2005, 52.

23. IEEE 1588-2019 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, 16 June 2020. Available online: https://ieeexplore.ieee.org/document/9120376 (accessed on 10 November 2020).

24. IEEE 1588-2002 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, 31 October 2002. Available online: https://ieeexplore.ieee.org/document/1048550 (accessed on 10 November 2020).

25. Lee, K.S.; Wang, H.; Shrivastav, V.; Weatherspoon, H. Globally Synchronized Time via Datacenter Networks. In Proceedings of the 2016 ACM SIGCOMM Conference, Florianopolis, Brazil, 22–26 August 2016; Barcellos, M., Ed.; Association for Computing Machinery: New York, NY, USA, 2016; pp. 454–467.

26. Watt, S.T.; Achanta, S.; Abubakari, H.; Sagen, E.; Korkmaz, Z.; Ahmed, H. Understanding and applying precision time protocol. In Proceedings of the 2015 Saudi Arabia smart grid (SASG), Jeddah, Saudi Arabia, 7–9 December 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 1–7.

27. Puttnies, H.; Danielis, P.; Timmermann, D. PTP-LP: Using Linear Programming to Increase the Delay Robustness of IEEE 1588 PTP. In Proceedings of the IEEE GLOBECOM 2018, Abu Dhabi, UAE, 9–13 December 2018.

28. Correll, K.; Barendt, N.; Branicky, M. Design considerations for software only implementations of the IEEE 1588 precision time protocol. In Proceedings of the Conference on IEEE 1588, Zurich, Switzerland, 10–12 October 2005; pp. 11–15.