






First Analysis of Time Synchronization for TSN Networks with Hot Standby GM


Mahin Ahmed 
Silicon Austria Labs GmbH
Linz, Austria

Lucas Haug 
University of Stuttgart
Stuttgart, Germany

Raheeb Muzaffar 
Silicon Austria Labs GmbH
Linz, Austria

Damir Hamidovic 
Silicon Austria Labs GmbH
Linz, Austria

Armin Hadziaganovic 
Silicon Austria Labs GmbH
Linz, Austria

Hans-Peter Bernhard 
Silicon Austria Labs GmbH
and Johannes Kepler University
Linz, Austria

Abstract—The rise of industrial applications such as autonomous systems, industrial automation, and precision manufacturing has increased the need for resilient and continuous time synchronization. Time-sensitive networking (TSN) addresses these requirements using the IEEE 802.1AS standard, which implements the generalized precision time protocol (gPTP) for sub-microsecond accuracy. However, best timeTransmitter clock algorithm (BTCA) used for grandmaster (GM) selection is slow to recover from failures and cannot detect transient faults, resulting in unstable synchronization. The IEEE 802.1ASdm standard addresses these limitations by disabling BTCA and introducing a static configuration with a hot standby GM. This paper presents a first analysis of time synchronization in TSN networks with hot standby. Using OMNeT++, we compare BTCA and hot standby performance in terms of out-of-sync time and clock offset during failures. The results demonstrate a close to zero out-of-sync time with hot standby as compared to BTCA.

Index Terms—Time synchronization, TSN, resilient time synchronization, fault tolerant, hot standby

I. INTRODUCTION

The emergence of new industrial use cases, such as autonomous systems, industrial automation, and precision manufacturing, has significantly increased the demand for high-performance communication networks that can ensure ultra-reliable and low-latency data exchange [1]. In these environments, continuous and resilient time synchronization is essential to guarantee coordinated operation among distributed devices, maintain system integrity, and enhance overall efficiency. Time-sensitive networking (TSN) has emerged as a critical technology to address these requirements by providing deterministic networking capabilities over standard Ethernet infrastructure.

Time synchronization is one of the important building blocks for TSN, where precise timing coordination ensures reliable data transmission with minimal latency and jitter. The IEEE 802.1AS standard, building upon the IEEE 1588v2 precision time protocol (PTP), serves as the cornerstone for synchronization within TSN networks [2]. It specifies the generic precision time protocol (gPTP) as its PTP profile, offering robust mechanisms to achieve sub-microsecond synchronization accuracy across industrial communication systems. In the upcoming sections, we follow the inclusive terminology as specified in [3] to describe the time synchronization. Whereby the following should be noted: master is replaced

with timeTransmitter (tT), slave is replaced with timeReceiver (tR), grandmaster (GM) remains grandmaster, and best master clock algorithm (BMCA) is replaced with best timetransmitter clock algorithm (BTCA). The IEEE 802.1AS uses BTCA for the selection of a GM within a domain. BTCA runs independently on each device and compares the clock attributes in the *Announce* messages to select a timing source within a domain. In case of a link or device failure, BTCA is triggered again to choose a new GM. However, BTCA's response to failures is considered slow. The selection of a new GM can take several seconds, primarily due to BTCA and the recovery time of clock's servo mechanisms [4]. Furthermore, BTCA is incapable of detecting transient faults in clocks, which can lead to unstable time synchronization [5]. The recent IEEE 802.1ASdm standard [6] addresses these limitations by disabling BTCA and implementing a static configuration with a hot standby GM for TSN networks.

Time synchronization performance for TSN networks has been studied before. Authors in [7] present the IEEE 802.1AS standard. Time synchronization performance with IEEE 802.1AS is presented in [8]–[13]. The selection of a GM in IEEE 802.1AS has been discussed in [14]. An initial discussion on extending hot standby for 6G-TSN networks is presented in [15]. To the best of author's knowledge, there is no prior work analyzing the time synchronization in TSN networks with a hot standby. In this paper we present the first analysis of time synchronization in TSN networks with a hot standby. We evaluate the time synchronization performance in OMNeT++ using the INET simulation framework for TSN features with extensions to include BTCA and hot standby. We compare its performance with BTCA in terms of out-of-sync time and clock offset in the presence of failures.

The rest of the paper is organized as follows: Section II explains the time synchronization in TSN networks with a focus on BTCA and hot standby. Section III presents the analysis of time synchronization for TSN with hot standby. Finally, section IV concludes the paper with some discussions.

II. TIME SYNCHRONIZATION IN TSN NETWORKS

Time synchronization within TSN is specified by the IEEE 802.1AS standard [16], which builds upon the IEEE 1588v2

PTP while introducing functionalities tailored to meet the requirements of cyber-physical systems. IEEE 802.1AS specifies the gPTP as a PTP profile. gPTP is a packet-based protocol designed for disseminating timing information and selecting a GM within local area networks (LANs). It enables network-wide synchronization with sub-microsecond precision while demanding minimal computational resources [7]. Networks employing gPTP are known as time-aware networks and comprise multiple gPTP instances, collectively referred to as time-aware systems. A gPTP instance is an instance of the IEEE 802.1AS protocol and operating within a single time aware system within exactly one domain. The IEEE 802.1AS standard supports redundancy in multiple GMs and domains to enhance fault tolerance and meet the time-critical application requirements for continuous time synchronization. Each domain is identified by a unique domain number.

The gPTP protocol ensures simultaneous synchronization across time and frequency domains. Frequency synchronization involves adjusting a clock's frequency offset to match the GM clock. Conversely, time synchronization entails fine-tuning the clock's absolute time to align as closely as possible with the GM's reported time. Propagation delay mechanism involves the exchange of *Pdelay* messages between adjacent clocks for the calculation of neighbor rate ratio and propagation delay. Neighbor rate ratio is the ratio between the clock frequencies of two adjacent clocks. Propagation delay accounts for the delay experienced by a message traversing the network path between two clocks. The transport of timing information in a time-aware network for a two-step process is carried out by the *Sync* and *Follow_Up* messages. The correction field is calculated and used to compensate for delays introduced by timing message distribution within the network. It is continuously updated at each clock that propagates timing information from the GM. The GM is selected either using BTCA or external port configuration.

A. Best timeTransmitter Clock Algorithm (BTCA)

The BTCA dynamically selects the GM within each gPTP domain in a time-aware network, as outlined in [17] and [16]. The synchronization hierarchy is determined by independent port decisions made at each PTP instance, meaning PTP instances do not negotiate which one should act as the GM. The synchronization hierarchy is established as a spanning tree to prevent synchronization loops. BTCA comprises two main components: a dataset comparison algorithm and a port decision algorithm. The dataset comparison algorithm evaluates the information within *Announce* messages, which are sent by each time-aware system to others within the gPTP domain. These messages contain clock and topology attributes—such as *priority1*, *clockClass*, *clockAccuracy*, *offsetScaledLogVariance*, *priority2*, and *clockIdentity*—arranged in descending order of significance [17]. By comparing the received *Announce* messages at each port with the local clock, the algorithm determines the best clock for each port. It then applies the same comparison process to identify the optimal clock for the entire PTP instance. The port decision algorithm,

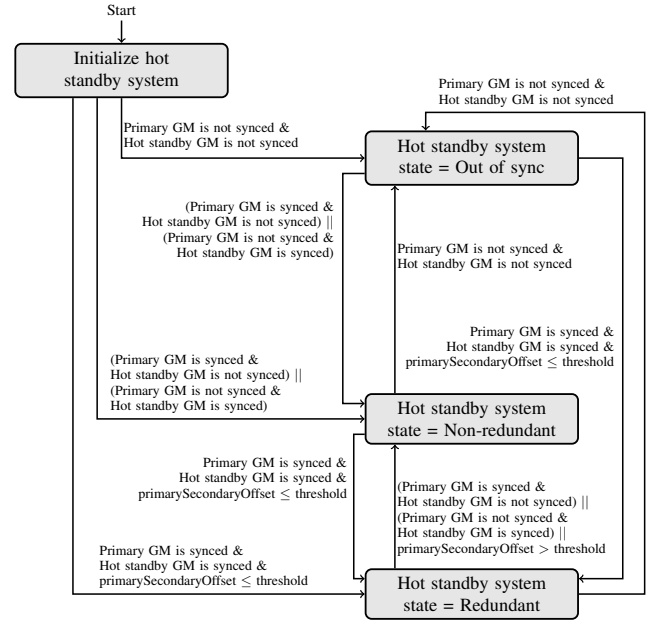


Fig. 1. Hot standby system state machine, abstracted from [6].

executed on each PTP instance, assigns port states based on the best dataset per instance, the best dataset per port, and the local dataset. Ports can function as either a tT, tR, or passive port. The tT ports send synchronization messages, while tR ports receive them. Passive ports, on the other hand, help prevent synchronization loops.

If the current GM fails or stops transmitting timing information, BTCA is triggered after an *announceReceiptTimeout* to select a new GM. However, BTCA's response time in failure scenarios is relatively slow, causing a downtime before a new GM is elected [4], which is undesirable for time-critical applications. Additionally, BTCA lacks a mechanism to detect transient faults in a GM clock. If such faults go undetected, they may result in a phenomenon known as the "ping-pong effect," where BTCA repeatedly switches between multiple candidate clocks, preventing stable synchronization [5].

B. IEEE 802.1ASdm — Hot standby amendment

The recent IEEE 802.1ASdm standard [6], also known as the hot standby amendment, addresses the limitations of the BTCA by enabling redundancy in GM within a time-aware network. It introduces a mechanism for selecting and configuring multiple GMs, allowing for a hot standby system that ensures continuous synchronization even in case of device or link failure. Instead of relying on BTCA for GM selection within a domain, the hot standby amendment utilizes the external port configuration option to establish a static configuration. The two GMs are selected by an external management entity (e.g., the TSN network controller).

With the hot standby system activated, each time-aware system operates two PTP instances simultaneously, each belonging to a distinct domain. These domains are designated as the primary domain and the hot standby domain, with

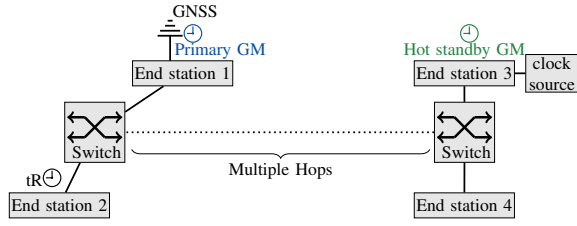


Fig. 2. Simulated time synchronization network with multiple hops.

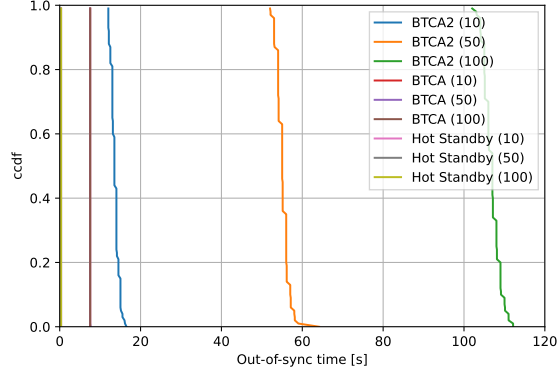


Fig. 3. Complementary cumulative distribution of the out-of-sync time for end station 2. The number in parentheses indicates the number of hops between the end station and the GM. The plots for the different number of hops for hot standby and BTCA overlap each other.

their respective GMs referred to as the primary GM and the hot standby GM. As long as both domains are available and the system is in a redundant state, the time-aware system primarily uses the primary domain for its applications. In the event that the primary domain becomes unavailable, the system seamlessly switches to the hot standby domain to maintain synchronization. To ensure timing consistency between the two domains, the hot standby GM synchronizes itself with the primary GM before transmitting timing messages within its domain. This synchronization is maintained within a specified tolerance range, guaranteeing minimal deviation between the two GMs. The hot standby system continuously monitors the difference between the primary and hot standby GM times through the *primarySecondaryOffset* variable. If this offset exceeds a predefined threshold, the system transitions from a redundant state (with both domains available) to a non-redundant state (with only one domain active). The state machine governing the hot standby process is depicted in Fig. 1, where the term "is synced" indicates that the synchronization requirements of the PTP instance have been met.

The hot standby amendment also introduces an optional split functionality that acts as an interworking feature, transferring synchronized time from a synchronized PTP instance to an out-of-sync PTP instance. This functionality helps reduce the *primarySecondaryOffset* to within the acceptable threshold, allowing the hot standby system to recover from a non-redundant state back to a redundant state, thereby restoring the availability of both domains.

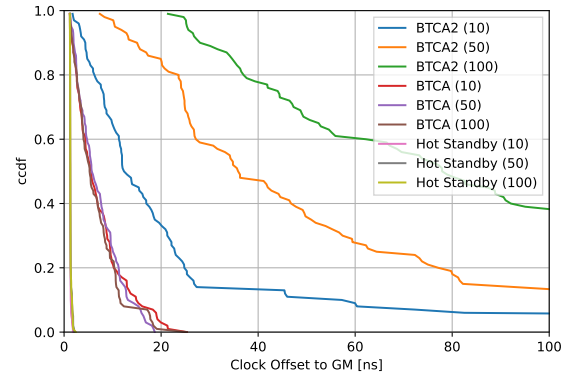


Fig. 4. Complementary cumulative distribution of the clock offset for end station 2. The number in parentheses indicates the number of hops between the end station and the GM. The plots for the different number of hops for hot standby and BTCA overlap each other.

III. SIMULATION ANALYSIS

For the analysis of time synchronization for TSN networks with hot standby, we extend the INET TSN simulator with a BTCA and hot standby implemented in OMNeT++. Our implementation is available open source on GitHub [18]. We simulate a TSN network with multiple hops as shown in Fig. 2. The time synchronization performance is measured at the end station 2, marked with tR in Fig. 2. All gPTP compliant devices are configured according to the parameter settings recommended in the IEEE 802.1AS standard. Specifically, we set the intervals for the *Sync* messages to 125 ms and to 1 s for the *Announce* message, each with a timeout after 3 intervals. In the hot standby scenario, BTCA is disabled and hot standby system is enabled. The end station 1 is chosen as the primary GM, and the end station 3 is chosen as the hot standby GM. Whereas, the BTCA scenario uses BTCA for selecting a new GM and hot standby system is disabled. In the BTCA scenario, end station 1 is selected as the GM.

To analyze the performance, we consider a failure of the primary GM where end station 1 becomes GM incapable. For the hot standby scenario, the hot standby system switches to a non-redundant state after the sync timeout and the end station 2 starts using the hot standby domain immediately. Whereas, for the BTCA scenario, BTCA is triggered to select a new GM. We simulate two BTCA implementations: BTCA, which immediately sends *Announce* messages upon selecting a new GM, and BTCA2, which waits until the current announce interval ends. For a fair comparison, the clock priorities are set in a way that end station 3 is chosen as the new GM by the BTCA. The time synchronization performance is measured at the end station 2 in terms of clock offset to GM and out-of-sync time for both BTCA and hot standby scenario. The out-of-sync time is defined as the time between a *syncReceiptTimeout* and a new *Sync* message. The simulation results are averaged over 100 repetitions.

In the hot standby scenario, out-of-sync time is minimal (as shown in Fig. 3) since devices quickly switch to the hot standby domain after a sync timeout. This leads to a negligible

maximum offset between TSN end station and the GM, as shown in Fig. 4. The number of hops has little effect because the sync timeout occurs independently on each device, with no need to select a new GM. Hence, the lines for different number of hops overlap each other in Fig. 3 and Fig. 4.

In contrast, BTCA implementations respond slower to network failures than the hot standby scenario, resulting in longer out-of-sync times (as shown in Fig. 3) and greater clock offsets (as shown in Fig. 4). This delay occurs for two reasons: (1) The Announce timeout is longer than the Sync timeout, delaying BTCA response to network failures. (2) BTCA requires time to exchange Announce messages and select a new GM, unlike the statically configured hot standby GM. This issue is particularly evident in BTCA2, where the number of intermediate hops significantly increases out-of-sync time and maximum offset as the Announce messages are not sent immediately. The lines for number of hops for BTCA implementation overlap each other in Fig. 3.

Overall, the results confirm that BTCA responds more slowly to failures, making hot standby superior in minimizing out-of-sync time and clock offset.

IV. CONCLUSIONS

This paper presented a preliminary analysis of time synchronization performance in TSN networks comparing the hot standby approach with the BTCA. With simulation analysis in OMNeT++, we demonstrated that hot standby significantly outperforms BTCA in terms of out-of-sync time and clock offset during failures, achieving near-zero out-of-sync time. These findings highlight the limitations of BTCA, particularly its slower response to failures.

However, this analysis is only a first step. Future work should explore a wider range of network scenarios and failure types to comprehensively evaluate the effectiveness of both hot standby and BTCA. Additionally, it remains unclear whether BTCA's dynamic nature may offer advantages over the static hot standby configuration, particularly in networks lacking redundancy mechanisms like redundant paths.

ACKNOWLEDGEMENTS

This work was supported by the European Union's Horizon Europe project DETERMINISTIC6G under grant agreement No. 101096504 and the COMET-K2 Center of the Linz Center of Mechatronics (LCM) funded by the Austrian federal government and the federal state of Upper Austria. The authors

would like to thank Marilet De Andrade and János Farkas for their support in the completion of this work.

REFERENCES

- [1] G. P. Sharma et al., "Toward deterministic communications in 6G networks: State of the art, open challenges and the way forward," *IEEE Access*, vol. 11, pp. 106 898–106 923, 2023.
- [2] M. K. Atiq, R. Muzaffar, Ó. Seijo, I. Val, and H.-P. Bernhard, "When IEEE 802.11 and 5G meet time-sensitive networking," *IEEE Open J. of the Ind. Electron. Soc.*, vol. 3, pp. 14–36, 2021.
- [3] IEEE, "IEEE Std 1588g-2022: IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Amendment 2: Master-Slave Optional Alternative Terminology," 2023.
- [4] A. Bondavalli, F. Brancati, A. Flammini, and S. Rinaldi, "Master failure detection protocol in internal synchronization environment," *IEEE Trans. on Instrum. and Meas.*, vol. 62, no. 1, pp. 4–12, 2013.
- [5] Y. Kozakai and M. Kanda, "Keeping clock accuracy on a master clock failure in substation network," in *Proc. IEEE Int. Symp. on Prec. Clock Synchronization for Meas., Control and Commun.*, 2010, pp. 25–29.
- [6] IEEE, "IEEE Std 802.1ASdm-2024: IEEE standard for local and metropolitan area networks—timing and synchronization for time-sensitive applications amendment 3: Hot standby and clock drift error reduction," 2024.
- [7] S. Rodrigues and J. Lv, "Synchronization in time-sensitive networking: An introduction to IEEE Std 802.1AS," *IEEE Commun. Standards Mag.*, vol. 6, no. 4, pp. 14–20, 2022.
- [8] M. D. J. Teener and G. M. Garner, "Overview and timing performance of IEEE 802.1 AS," in *Proc. IEEE Int. Symp. on Prec. Clock Synchronization for Meas., Control and Commun.*, 2008, pp. 49–53.
- [9] M. Gutiérrez, W. Steiner, R. Dobrin, and S. Punnekkat, "Synchronization quality of IEEE 802.1AS in large-scale industrial automation networks," in *Proc. IEEE Real-Time and Embedded Technol. and Appl. Symp. (RTAS)*, 2017, pp. 273–282.
- [10] 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," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, 2011, pp. 147–154.
- [11] G. M. Garner, A. Gelter, and M. J. Teener, "New simulation and test results for ieee 802.1 as timing performance," in *Proc. IEEE Int. Symp. on Prec. Clock Synchronization for Meas., Control and Commun.*, 2009, pp. 1–7.
- [12] G. M. Garner and H. Ryu, "Synchronization of audio/video bridging networks using ieee 802.1 as," *IEEE Commun. Mag.*, vol. 49, no. 2, pp. 140–147, 2011.
- [13] S. Tang, X. Hu, and L. Zhao, "Modeling and security analysis of ieee 802.1 as using hierarchical colored petri nets," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2020, pp. 1–6.
- [14] Y. Jeon, J. Lee, and S. Park, "An efficient method of reselecting grand master in IEEE 802.1AS," in *Proc. IEEE Asia-Pacific Conf. on Commun. (APCC2014)*, 2014, pp. 303–308.
- [15] M. Ahmed et al., "Resilient time synchronization in 6G networks: A hot standby solution," *IEEE Commun. Standards Mag.*, vol. 9, no. 1, 2025.
- [16] IEEE, "IEEE Std 802.1AS-2020: IEEE standard for local and metropolitan area networks—timing and synchronization for time-sensitive applications," 2020.
- [17] —, "IEEE Std 1588-2019 (Revision of IEEE Std 1588-2008): IEEE for a precision clock synchronization protocol for networked measurement and control systems," 2019.
- [18] DETERMINISTIC6G, "Time Synchronization Simulator for 6G-TSN networks," https://github.com/DETERMINISTIC6G/6GDetCom_Simulator/pull/2, 2025, Accessed: April 27, 2025.