Latency in Cascaded Wired/Wireless Communication Networks for Factory Automation

Steven Dietrich[1(✉)], Gunther May[1], Johannes von Hoyningen-Huene[2], Andreas Mueller[2], and Gerhard Fohler[3]

[1] Bosch Rexroth AG, Lohr am Main, Germany
{Steven.Dietrich,Gunther.May}@boschrexroth.de
[2] Robert Bosch GmbH, Renningen, Germany
{Johannes.Hoyningen-Huene,Andreas.Mueller21}@de.bosch.com
[3] Chair of Real-time Systems, Technical University Kaiserslautern, Kaiserslautern, Germany
fohler@eit.uni-kl.de

Abstract. Unlike in the areas of process automation and condition monitoring, current wireless technologies cannot be used for many closed-loop control applications in factory automation. These applications require shorter cycle times, precise synchronicity in the microseconds range and higher reliability with low packet error rates. Furthermore, established Industrial Ethernet communication systems will not be completely replaced in the near future. Therefore, a wireless communication system for factory automation also requires seamless integration into existing networks.

However, a resulting cascaded network will lead to additional latencies, which have a negative effect on the overall real-time performance. In this paper, we analyze the effect of frame structure conversion for different subnetworks with respect to the additional latencies they introduce. Therefore, we introduce an abstracted network model representing various subnetworks. We exemplify different protocol implementations and discuss them in terms of the resulting latencies and optimizations.

Keywords: Industrial Ethernet · Closed-loop control · Wireless communication · Cascaded network · ParSec · Latency

1 Introduction

In industrial communication, flexibility and mobility gains more and more importance, especially in the context of connected industries (in Germany also referred to as Industrie 4.0 [1]). In the recent past, communication networks known from

The work presented in this paper has been partially supported by the German Federal Ministry of Education and Research BMBF (grant agreement no. 16KIS0225 & 16KIS0224).

L.A. Maglaras et al. (Eds.): INISCOM 2016, LNICST 188, pp. 50–61, 2017.
DOI: 10.1007/978-3-319-52569-3_5

the office domain have been established in areas like condition monitoring or process automation. They represent an approach to use components off the shelf (COTS).

Nevertheless, for the area of factory automation wireless communication is hardly considered so far, since there is no wireless communication system that covers all the requirements of industrial closed-loop control applications. Here Industrial Ethernet (IE) systems are most widely used. They provide cycle times below 1 ms, high synchronicity with a timing deviation of less than 1 µs, and high reliability with packet error ratios (PER) of $\leq 10^{-9}$ [2].

Since the benefits of wireless communications are desired in the field of factory automation too [3], the German research project ParSec [4] currently develops a wireless real-time system especially for closed loop control applications.

In the near future, wireless communication will not completely replace the established IE systems [5]. Therefore, hybrid communication networks build on cascaded wired and wireless subnetworks will arise. A seamless integration of the wireless subnetwork into existing wired technologies is mandatory.

Due to the cascading the behavior of the communication network is changed. Especially the timing constraints are affected. Therefore, this paper analyses different approaches of data positioning into a secondary network frame to reduce the resulting latency and jitter. The remainder of this paper is structured as follows: Sect. 2 provides work related to hybrid networks and timing constraints. In Sect. 3 we introduce an abstract network model intended to represent many network protocols used in factory automation. Additionally the wired IE and the wireless ParSec subsystem are shortly introduced. In Sect. 4 we discuss the different approaches of frame conversion from one subnetwork into another one. Finally a summary and a short outlook are given in Sect. 5.

2 Related Work

Cascaded networks for industrial communication were investigated already over the last few years. Ackerberg et al. [6] introduces arising latencies and jitter as mayor research challenges for the use of wireless networks in industry. However, most of the investigated cascaded networks do not face the timing constrains or the combination of several hard real-time protocols. The authors of [7,8] introduce a detailed protocol conversion form Industrial Ethernet to CAN, but without any timing constraints. They use unadapted mechanisms for sending and receiving, which results in high jitter transmission latency. The authors of [9] introduce the hybrid network concept of FlexWare with wired field-level backbone and wireless clusters. Unfortunately, this concept cannot guarantee any hard real-time performance as well.

Investigations of timing and synchronization in cascaded hard real-time networks are mostly based on distributed clocks. For example Ferrari et al. [10] show a method of synchronized communication over an intermediate network. However, the mechanism of distributed clocks might not always be available. While reaching the required synchronization for factory automation, the throughput

can not be guaranteed since the PROFINET IP protocol in not designed for these requirement. The authors of [3,11] introduce a low jitter point to point wireless communication. Since both investigations are based on basic WLAN, the scaling to multi point communication would result in increased latencies and jitter.

Addad et al. [12] introduce a delay evaluation of networked control systems which includes also the field devices due delay. Since it is based on Switched Ethernet, it does not consider the additional delay arising when using heterogeneous cascaded networks. Accordingly, a gap could still be found in the investigation of timing constraints when combining several hard real-time protocols. Especially the protocol conversion and data placement will have a big influence on the latency and jitter, that was hardly investigated in detail so far.

3 Cascaded Communication System

By combining different communication networks with each other, their individual performances will influence the overall system. Not only every subsystem has to be optimized but also cross attributes needs to be adapted to still meet the requirements of closed-loop control applications. In this section we introduce a model for a subnetwork that enables the analyses detached from any concrete network protocol and shortly present exemplary subsystems.

3.1 Abstract Cascaded Network

A communication network for closed loop control application is often built on master-slave relation. One control unit, the master, operates one or multiple sensor or actuator devices, the slaves. The communication takes place cyclically, such that real-time (RT) performance can be guaranteed. Mostly also non-real-time (NRT) communication can take place whenever there is no RT communication. In this paper, we focus on RT data, since they require higher performance and NRT data can be scheduled with higher latencies and jitter.

We assume full duplex networks, since high throughput and low cycle times (t_{cycle}) are required. Therefore, we can analyze downlink (DL) from master to slave and uplink (UL) from slave to master separately. Here we will focus on the analysis of the DL, since the knowledge about the latencies there are important to set a global network synchronization time. The resulting latencies for UL can be investigated similarly. We do not consider transmission errors, since industrial communication networks often provide advanced mechanisms to minimize the error rate. Remaining errors can then be compensated by the application.

All networks have a limited resource space, in which the DL data are encapsulated (Fig. 1). This can be subdivided into several resource elements (RE) by multiple channel access mechanism and further coding. Thus, one RE consist of n_{BpRE} bits.

We assume that a preamble or header with a size of S_{Pre} RE needs to be transmitted periodically for example for synchronization and signalization.

A frame of S_{Frame} RE consists of S_{Pre} RE for the preamble and S_{Data} RE for the data and is transmitted with a RE rate R_{RE}. A possible idle duration of t_{Idle} can separate two successive frames. For simplification, this could also be seen as a number of RE $S_{\text{Idle}} = t_{\text{Idle}} \cdot R_{\text{RE}}$

A cascaded communication network can now be analyzed with two or more of these models with different parametrization. In Fig. 2 we illustrate the network structure of an exemplary cascaded network built on three subnetworks. The primary subnetwork consists of the primary master (PN-M), several primary slaves (PN-S) and the master of the secondary subnetwork (SN-M). A slave of the secondary subnetwork (SN-S) is then the basis of a tertiary subnetwork with several tertiary slaves (TN-S).

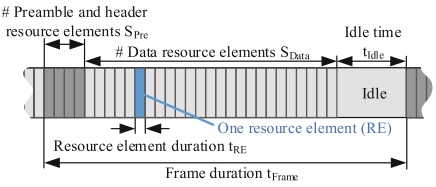

Fig. 1. Frame resource space.

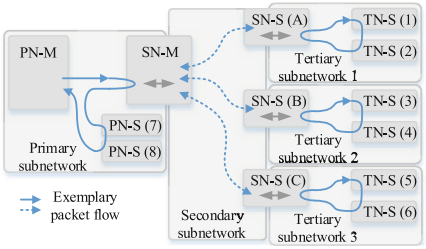

Fig. 2. Schematic network concept with data flow.

3.2 Wired Industrial Ethernet Communication

There are multiple wired IE networks well established in factory automation. The most common IE protocols for closed loop control applications are Sercos III, EtherCAT, and PROFINET IRT [5,13–15]. All of them are based on cyclic master-slave communication.

Sercos III and EtherCAT are mostly organized in a ring or line topology. D_{Down} and D_{Up} are transmitted by summation telegrams carrying data of several slaves. These telegrams are building the DL and the UL. They provide minimal latencies and the required low cycle times within the IE network.

PROFINET on the other hand is not based on summation telegrams. By synchronized clocks, the slave devices are able to generate UL packets on their own. Through special Ethernet switches also star topologies can be build up.

Referring to the abstract model, the frame structure of the introduced IE networks is built on Fast Ethernet [16]. S_{Pre} corresponds to the necessary number of bytes for the Ethernet header and S_{Data} to the payload (max. 1500 RE per frame). The net data rate at the link layer is 100 Mbit/s. A fixed number of one or more frames of individual length each is transmitted per communication cycle. The idle time between successive frames can vary. Once set, these variable parameter are constant in each cycle.

3.3 Wireless ParSec Communication

One exemplary wireless network for industrial closed loop applications we will focus on in this paper will be developed within the ParSec Project [4]. Within this project we focus on the design of a highly deterministic wireless communication network. This includes especially minimum latency and high reliability. This section presents the current working hypothesis regarding integration. The communication is also based on master-slave relations.

Referring to the abstract model, the communication is based on the Parallel Sequence Spread Spectrum (PSSS) technique [17,18]. 255 orthogonal codes can be used in parallel to build up a RE. However, the number of bits per code can be varied to increase the reliability of the communication or the data rate. Either code division duplexing (CDD) or frequency division duplexing (FDD) is used for full duplex transmission of DL and UL. There is no idle time between successive frames.

Here we assume one bit per code and therefore $n_{\mathrm{BpRE}} = 255$. With a RE duration t_{RE} of 14.25 µs, R_{RE} is about 70.175 kHz and, thus, the data rate is about 17.9 Mbit/s. This data rate is further reduced by the necessary error correction mechanism and therefore significantly smaller than 100 Mbit/s used in IE technologies. Here we have to find special approaches for rate conversion, data selection and placements of the IE data into the wireless frame.

Further information about the project will be published soon and can be found on the project homepage [4].

4 Data Positioning into a Secondary Network Frame

According to the cascaded network structure, additional latencies arise due to the placement of the data from the primary subnetwork (PN) into the secondary subnetwork (SN). Latency in this context means the time between the full reception of the PN data and the complete placement into the SN frame. Constant latencies could be partially compensated by the application by adapting the control algorithm. Therefore, they must be completely known and kept as low as possible. In addition, the jitter must be minimal to enable adequate control. Here, jitter means the fluctuation between the best and the worst case latency.

Each PN cycle, only a subset of the PN DL date $D_{\mathrm{Down(PN)}}$ must be transmitted into the SN, since some slaves might be located already in the PN. The resulting amount of SN DL data $D_{\mathrm{Down(SN)}}$ results as $D_{\mathrm{Down(SN)}} \leq D_{\mathrm{Down(PN)}}$. The number of required RE per PN cycle is equal to

$$n_{\mathrm{RE(SN)}} = \left\lceil \frac{D_{\mathrm{Down(SN)}} + M_{\mathrm{(SN)}} \cdot B_{\mathrm{OH(SN)}}}{n_{\mathrm{BpRE(SN)}}} \right\rceil . \tag{1}$$

M denotes the number of slaves to be addressed and B_{OH} the signaling overhead (OH) to address them. With the number of RE for data per frame (S_{Data}), the number of required SN frames per PN cycle is

$$n_{\mathrm{Frames(SN)}} = \left\lceil \frac{D_{\mathrm{Down(SN)}} + M_{\mathrm{(SN)}} \cdot B_{\mathrm{OH(SN)}}}{n_{\mathrm{BpRE(SN)}} \cdot S_{\mathrm{Data(SN)}}} \right\rceil . \tag{2}$$

The required number of RE for data and OH together with the overall number of preamble and idle RE must be smaller or equal to the possible number of RE during one PN cycle, as shown in Eq. (3). Otherwise there will be no balanced relation between data sending and receiving.

$$t_{\mathrm{Cycle(PN)}} \geq \left\lceil \frac{n_{\mathrm{RE(SN)}} + n_{\mathrm{Frames(SN)}}(S_{\mathrm{Pre(SN)}} + S_{\mathrm{Idle(SN)}}) - S_{\mathrm{Idle(SN)}}}{R_{\mathrm{RE(SN)}}} \right\rceil \tag{3}$$

For the following analyses, we assume that this condition is always met.

We propose two different approaches with several variants each to place the received data $D_{\mathrm{Down(SN)}}$ into the SN frame: by static allocation or by a streaming approach. In the following subsections we will analyze them to enable the best choice regarding lowest latency and jitter according to the target application and cascaded network. Figure 3 illustrates the approaches with their different subclasses in their worst case scenario according to the network structure shown in Fig. 2. Here, the PN addresses six slaves (1–6) wireless with one DL and UL frame every cycle. The SN consists of three slaves (A–C). Each of these SN slaves has to receive the data for two of the slaves addressed in the PN (A: 1&2, B: 3&4, C: 5&6). The transmission duration of the packets in the SN is pictured to be longer than in the PN. The number of required frames to transmit the PN data depends on the implementation approach. The data reception time could be seen as constant offset. Therefore, the approaches are independent of the PN structure and the SN indexing could be neglected.

4.1 Static Allocation

Static allocation (Fig. 3.1–4) in this context means, that each SN data packet has a reserved slot within the SN frame and can only be placed there. If this slot is missed, for example when the PN data arrive too late, the next slot has to be awaited. This might be one or multiple frames later. In the meantime, no other RT data can be transmitted, since their order must be satisfied. All SN data placements can be predefined and signaled during startup. The SN-S only need to observe always the same position within the SN frame. Therefore, we

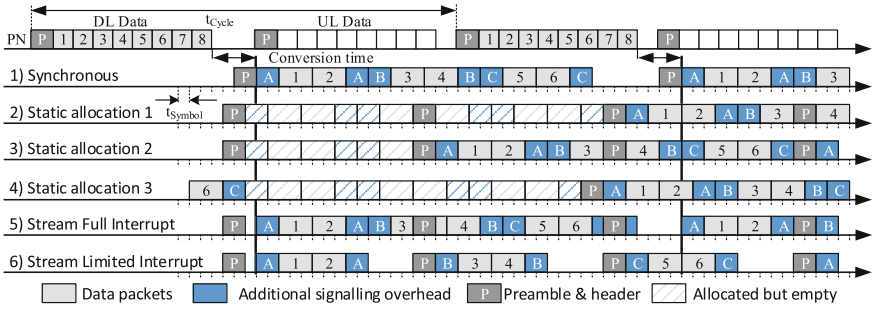

Fig. 3. Exemplary data positioning approaches (worst case).

assume that the signaling overhead for this approach will be lower compared to the other cases.

A further differentiation within the static allocation is possible by the comparison of the SN frame duration with respect to the PN cycle time. If the SN frame duration is smaller or equal to the PN cycle time, one or more SN frames will fit into the time between two successive PN DL frames.

In the simplest case, the duration of one (or several) SN frames is equal to the PN cycle time and we have a *synchronous* cyclic relation (Fig. 3.1). Here, the signaling overhead is minimal since the whole SN frame structure can be predefined and would not change during runtime. If the start of the SN frame is adapted according to the arrival of the DL data, the latency can be minimized. The worst case latency will result as in Eq. (4) and there will be no jitter.

$$t_{\text{Place}}[\text{Synchronous}] = ((n_{\text{Frames}} - 1)(S_{\text{Pre}} + S_{\text{Idle}}) + n_{\text{RE}})R_{\text{RE}}^{-1} \tag{4}$$

However, either the PN cycle time has to be set according to the SN frame duration or the other way around. Also, despite unsynchronized clocks, there must be no phase drift between the clocks of each subnetwork.

If there is an asynchronous relation between the PN cycle time and the SN frame duration, the reception of the PN data will vary within the SN frame. A possible phase drift provides another additional shift within the SN frame. For the static allocation this means that we cannot use every SN frame for RT data. If the SN frame duration is smaller than the PN cycle time and multiple SN frames are required, we can further distinguish between two different approaches. In the first one, later called *static allocation 1*, all SN frames are statically allocated and have a fixed successive order (Fig. 3.2). If the SN data are only available after the start of the according SN frame is already over, all n_{Frames} frames remain empty for RT transmission. The worst case latency results as in Eq. (5) whereas the jitter is calculated in Eq. (6).

$$t_{\text{Place}}[\text{Static 1}] = (n_{\text{Frames}}(S_{\text{Data}} + 2S_{\text{Pre}} + 2S_{\text{Idle}}) - S_{\text{Pre}} - S_{\text{Idle}} + n_{\text{RE}})R_{\text{RE}}^{-1} \tag{5}$$

$$t_{\text{J}}[\text{Static 1}] = (n_{\text{Frames}}(S_{\text{Data}} + S_{\text{Pre}} + S_{\text{Idle}}) - S_{\text{Idle}})R_{\text{RE}}^{-1} \tag{6}$$

In the second approach, later called *static allocation 2*, all SN frames still are statically allocated. However, we can mark one frame as empty and repeat its transmission, if the PN data are received too late (Fig. 3.3). Therefore, only one SN frame remains empty for RT transmission, but we need special signalization. This results in an increased OH. The worst case latency results as in Eq. (7) whereas the jitter is calculated in Eq. (8).

$$t_{\text{Place}}[\text{Static 2}] = (S_{\text{Data}} + n_{\text{RE}} + n_{\text{Frames}}(S_{\text{Pre}} + S_{\text{Idle}}) - S_{\text{Idle}})R_{\text{RE}}^{-1} \tag{7}$$

$$t_{\text{J}}[\text{Static 2}] = (S_{\text{Data}} + S_{\text{Pre}} + S_{\text{Idle}})R_{\text{RE}}^{-1} \tag{8}$$

If the SN frame duration is larger than the PN cycle time, later called *static allocation 3*, the SN data must fit multiple times completely into the SN frame to avoid congestions (Fig. 3.4). As advantage of this approach, we need to transmit

less SN preambles. However, the system must stay highly synchronous even for long intervals without new preambles. We assume that this approach requires a higher signaling overhead than the previous ones. The worst case latency results as in Eq. (9) whereas the jitter is calculated in Eq. (10).

$$t_{\text{Place}}[\text{Static 3}] = (2n_{\text{RE}} + S_{\text{Pre}} + S_{\text{Idle}})R_{\text{RE}}^{-1} \tag{9}$$

$$t_{\text{J}}[\text{Static 3}] = (n_{\text{RE}} + S_{\text{Pre}} + S_{\text{Idle}})R_{\text{RE}}^{-1} \tag{10}$$

4.2 Streaming Approach

In this approach, we consider the SN frames not statically allocated but as continuous stream. Here, we can place the SN data together with according signaling as they arrive. This transmission can be interrupted by the SN preamble. We constitute two different cases how this can be implemented:

- By full interrupt, where the SN preamble can interrupt data arbitrarily, or
- by limited interrupt, where only complete data packets for one PN-S or SN-S can be interrupted.

In both cases, the SN frame duration matters only in the number of SN preambles we have to transmit.

In case of the *full interrupt* (Fig. 3.5), the capacity of the SN frame will be used fully. Since the data from the PN arrive in periodical cycles, we can place the SN data in regular RE distances into the SN frame. This requires that the SN preamble can be identified and subtracted from the number of received RE. However, phase drift might occur in this approach as well. This must be compensated by new resource allocation during runtime and can result in a complete change of the resource plan. Therefore, we assume the signaling OH for a streaming approach higher than with a static allocation. The worst case latency results as in Eq. (11) whereas the jitter is one preamble duration, as shown in Eq. (12).

$$t_{\text{Place}}[\text{Stream}] = (n_{\text{RE}} + n_{\text{Frames}}(S_{\text{Pre}} + S_{\text{Idle}}) - S_{\text{Idle}})R_{\text{RE}}^{-1} \tag{11}$$

$$t_{J}[\text{Stream}] = (S_{\text{Pre}} + S_{\text{Idle}})R_{\text{RE}}^{-1} \tag{12}$$

In the case of streaming with *limited interruptibility* (Fig. 3.6), it is possible that we have to stall the RT data transmission, since a SN preamble might be privileged. Therefore, we have to delay the transmission of a SN data packet until the SN preamble is sent, if this packet does not fit completely in front of it. Depending on the SN frame duration, this limits the maximum size of SN data packets we are able to transmit. Further, the shift of one SN packet into the next frame will influence subsequent packets, which then might also be shifted. This can lead to an instable communication relation between receiving and sending. Latency calculations cannot be performed in a general manner, since the whole SN DL needs to be analyzed packet by packet at every possible position within the SN frame. Therefore, also the signaling overhead and the additional required computation performance increases. Due to these substantial disadvantages, we will not consider this approach further.

Table 1. Assumed parameters for exemplary application.

Parameter	Value
Number of slaves (M)	20
RE duration (t_{RE})	14.25 µs
Bit per RE (n_{BpRE})	255
Frame size (S_{Frame})	50 RE
Preamble size (S_{Pre})	6 RE
Idle duration (t_{Idle})	0 s

Table 2. Estimated signaling overhead.

Signaling overhead	Bytes per slave
Synchronous	1
Static allocation 1	2
Static allocation 2	2
Static allocation 3	4
Stream 1	10
Stream 2	30

4.3 Comparison of the Approaches

In this section we compare the different approaches with each other with respect to their latency. We assume an exemplary application based on the networks presented in Sect. 3. The parameters used are listed in Table 1 and derived from the current working hypothesis of the ParSec project. Figure 4 illustrates the calculated latency for different amounts of payload per slave. Here, for simplification, we assume that each SN-S has to receive only one payload packet. Thus, for each SN-S the payload and the signaling overhead is transmitted only once. However, the actual size of the SN signaling overhead for the different cases is not yet fully analyzed. Here we assume the conservatively estimated values listed in Table 2.

In Fig. 4 we show that the *synchronous approach* results in the lowest latency. Here we do not have to consider any additional worst case RE for the data placement. Even with higher OH the latency of synchronous data positioning would remain below that of streaming with the same or higher amount of OH. The steps in the calculated curves occur, whenever the amount of SN RE for the according SN data and OH increases that much, that we need one more SN frame. However, as mentioned earlier, this approach will only be hardly implementable.

The approaches for *static allocation 1* and *2* have the same trends as long as we only need one SN frame for the transmission of all PN data. For more SN frames, the latency of the *static allocation 1* takes bigger steps, because we have to transmit in worst case all required SN frames twice. For *static allocation 2* only one frame needs to be retransmitted. As mentioned earlier, the *static allocation 3* requires that the PN data with the SN OH fit at least twice into the SN frame. Therefore, we can only use this approach as long as Eq. (13) holds.

$$\left\lceil \frac{D_{\mathrm{Down}} + M \cdot B_{\mathrm{OH}}}{n_{\mathrm{BpRE}}} \right\rceil \leq \frac{S_{\mathrm{Data}}}{2} \tag{13}$$

Up to this point, the latency is lower than for *static allocation 1* and *2* and the *streaming approach* with high OH. Here we have to consider less SN preambles and the SN frame utilization is higher. However, the latency increases depending

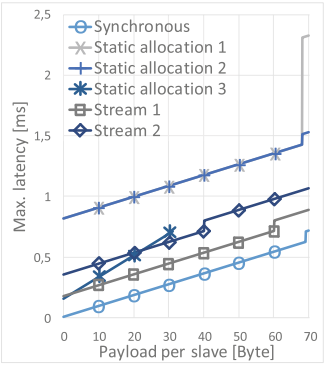

Fig. 4. Calculated worst case latency for 20 slaves and varying payload per slave.

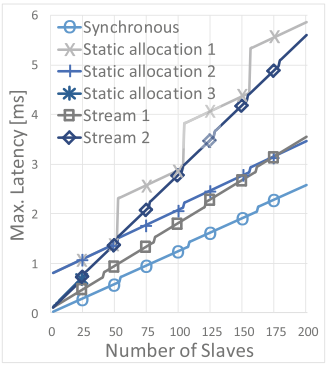

Fig. 5. Calculated worst case latency for a varying number of slaves and 25 byte payload per slave.

on the payload twice as fast than with the other approaches. For implementation reasons the *static allocation* generally has the advantage that the resource allocation for the SN data remains the same even with a phase drift between PN and SN. We only need to consider more frames for worst case estimations.

The *streaming approach with full interrupt* results in the best latency after the *synchronous approach*. Even with the estimated high signaling overhead of 30 bytes per slave and 20 SN-S this approach results in shorter latency times as with the *static allocation*. In comparison to completely empty SN frames, the influence of the OH RE remains insignificant. However, the latency of data positioning with the *streaming approach* increases faster with increasing number of SN-S than with the *static allocation* approach, since we estimate the signalization OH to be higher. This is exemplified in Fig. 5 for a varying number of slaves with 25 byte data per slave (other parameters being equal as before).

As mentioned above, the streaming approach requires the capability of new resource allocation and therefore the implementation will be more challenging than the static allocation.

With help of this comparison, we will now be able to choose the optimal data positioning according to latency and jitter for the investigated application and cascaded network. Considering also the latencies of each subnetwork, we will now be able to calculate the overall latency and the maximum jitter. As a result, we can adapt the control algorithm for the according network structure.

5 Conclusion

In this paper, we focused on the analysis of cascaded real-time communication networks for industrial closed-loop control application with respect to their latencies. We have introduced an abstract model for the subnetworks. This could be used for a wired IE protocols as well as for the currently developed wireless ParSec protocol and simplifies the latency analyses.

Using two combined subnetwork models, we analyzed different approaches to place the data of the primary network into the secondary one with respect to their latencies. We showed that only two approaches are of interest, since a synchronous approach is hard to achieve: Static allocation with only one empty frame, and streaming with full interruptible data.

Within a theoretical evaluation with estimated signaling overhead, we showed that the streaming approach results in shorter latencies, even for conservative assumptions and high numbers of slaves. We further indicated that the streaming approach depends on the ability of the system to reallocate resources during runtime. Therefore, if the latency is less important, we exposed the static allocation approach to be easier to implement. Here, even with phase drift, the resource allocation does not need to be changed during runtime.

Nevertheless, further analyses and real-life implementations have to show how large the signaling overhead for the different data positioning approaches really is, considering also transmission errors. In addition, the ability of resource reallocation needs to be analyzed. Implementations of adjustments in the global synchronization time must prove whether all requirements of the closed-loop control applications still can be met.

References

1. Jazdi, N.: Cyber physical systems in the context of industry 4.0. In: 2014 IEEE International Conference on Automation, Quality and Testing, Robotics, pp. 1–4. IEEE (2014)
2. Frotzscher, A., Wetzker, U., Bauer, M., Rentschler, M., Beyer, M., Elspass, S., Klessig, H.: Requirements and current solutions of wireless communication in industrial automation. In: 2014 IEEE International Conference on Communications Workshops (ICC), pp. 67–72. IEEE (2014)
3. Bauer, M., May, G., Jain, V.: A wireless gateway approach enabling industrial real-time communication on the field level of factory automation. In: Emerging Technology and Factory Automation (ETFA), 2014 IEEE, pp. 1–8. IEEE (2014)
4. BMBF-Project. ParSec. www.parsec-projekt.de
5. Sauter, T.: The three generations of field-level networks - evolution and compatibility issues. IEEE Trans. Ind. Electron. **57**(11), 3585–3595 (2010)
6. Akerberg, J., Gidlund, M., Bjorkman, M.: Future research challenges in wireless sensor and actuator networks targeting industrial automation. In: 2011 9th IEEE International Conference on Industrial Informatics (INDIN), pp. 410–415. IEEE (2011)
7. Zhang, Y., Feng, X., Guo, Y.: Design of ethernet-can protocol conversion module based on STM32. Int. J. Future Gener. Commun. Netw. **7**(1), 89–96 (2014)
8. Eum, S.: Implementation of protocol conversion control board for industrial communication. Int. J. Control Autom. **9**, 201–208 (2016)
9. Sauter, T., Jasperneite, J., Bello, L.L.: Towards new hybrid networks for industrial automation. ETFA **9**, 1141–1148 (2009)
10. Ferrari, P., Flammini, A., Rinaldi, S., Sisinni, E.: On the seamless interconnection of IEEE1588-based devices using a PROFINET IO infrastructure. IEEE Trans. Ind. Inf. **6**(3), 381–392 (2010)

11. Mahmood, A., Ring, F.: Clock synchronization for IEEE 802.11 based wired-wireless hybrid networks using PTP. In: IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication Proceedings, pp. 1–6. IEEE (2012)
12. Addad, B., Amari, S., Lesage, J.-J.: A virtual-queuing-based algorithm for delay evaluation in networked control systems. IEEE Trans. Ind. Electron. **58**(9), 4471–4479 (2011)
13. International Electrotechnical Commission (IEC). IEC 61784–2: 2014 industrial communication networks - profiles - part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3-CP 16/3 (Sercos III), 07 2014
14. International Electrotechnical Commission (IEC). IEC 61784–2: 2014 industrial communication networks - profiles - part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3-CP 12/1 (EtherCAT Simple IO) and CP 12/2 (EtherCAT Mailbox and Timing Synchronization), 07 2014
15. International Electrotechnical Commission (IEC). IEC 61784–2: 2014 industrial communication networks - profiles - part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3-CP 3/4 (PROFINET Conformance Class A), CP 3/5 (PROFINET Conformance Class B) and CP 3/6 (PROFINET Conformance Class C), 07 2014
16. IEEE 802.3 standard for ethernet (2016)
17. Schwetlick, H., Wolf, A.: PSSS-parallel sequence spread spectrum a physical layer for RF communication. In: 2004 IEEE International Symposium on Consumer Electronics, pp. 262–265 (2004)
18. Gowda, K.K., Messinger, T., Wolf, A.C., Kraemer, R., Kallfass, I., Scheytt, J.C.: Towards 100 Gbps wireless communication in THz band with PSSS modulation: a promising hardware in the loop experiment. In: 2015 IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB), pp. 1–5. IEEE (2015)