http://dx.doi.org/10.25673/39563

Timing Influencing Aspects of Industrial Applications

Steven Dietrich, Ludwig Leurs, Maximilian Schüngel
Bosch Rexroth AG
{steven.dietrich, ludwig.leurs, maximilian.schuengel}@boschrexroth.de

Abstract: Industry 4.0 offers not only new opportunities but also additional challenges for industrial applications and their communication systems. This requires a detailed knowledge of corresponding timing requirements to optimally use new technologies such as Time-Sensitive Networking (TSN) and the fifth generation cellular network technology (5G). Therefore, we provide an overview on application related timing aspects and implementation related constraints. Finally, we match application requirements towards network properties, in order to give a more detailed understanding on various timing aspects in industrial communication.

1 Introduction

Industrial applications require a communication network that fully covers their corresponding needs, such as determinism, short transmission latencies and reliability [D+17]. Increasing demands regarding flexibility and mobility require for new technologies, such as Time-Sensitive Networking (TSN) or the fifth generation cellular network technology (5G). These offer novel possibilities of data exchange but may need additional optimization and adaptation concerning the corresponding applications.
However, the data transmission models of today's typical Industrial Ethernet (IE) or fieldbus systems are only defined to the transmission mechanism, cf. [PRO15, ser13, ODV17]. They are very protocol specific and do not fully relate the corresponding manufacturing requirements. This makes it difficult to map the new communication technologies to this transmission models and to benefit from the novel possibilities.
Therefore, the goal of this work is to further analyze and classify timing constraints in industrial networks allowing a more detailed comparison towards the corresponding application requirements. The remainder of this work is structured as follows:
In Sec. 2, we present an overview of related work, followed by an overview of timing constraints we derive from the industrial applications point of view in Sec. 3. In Sec. 4, we analyze current protocol related timing aspects. In Sec. 5, we compare the corresponding timing constraints and relate them towards the data transmission models. Additionally, we derive possibilities for an application specific optimization, which lead to new degrees of freedom for the data transmission that could be used by new communication technologies. Finally, in Sec. 6, we draw the conclusion.

2 Related Work

Timing constraints of industrial communication are related on application based requirements or on transmission based conditions. The latter are derived from the evolutions of industrial communication systems, starting from field bus systems to IE based networks [Sau10]. The use of IE allows a better information exchange across all levels of the automation pyramid. However, the lower layer implementations of the typically used protocols differ in parts very strongly. This results in very different mechanisms for synchronization and the derivation of the corresponding temporal behavior. A comparison and classification of typically used IE protocols is for example presented in [J+07, Dec09].

In order to derive application based timing constraints, applications are typically first divided into different classes. One of the most common classification is the division into the application classes condition monitoring, process automation and factory automation [D+17, GH13, VDI09, ZVE09], which also classifies to different timing aspects. The corresponding data are typically transmitted as cyclic real-time data, acyclic real-time data, and non real-time data [PN09]. This already allows deriving certain requirements towards the real-time capability of the data exchange. Here, a distinction is typically made between non real-time, soft real-time, hard real-time and isochronous real-time capable data transmission [PN09]. More specific transmission requirements are very application related. Some specific application requirements are described for example in [D+17, F+14, VDI09, 5G 20].

Nevertheless, the actual requirements of the application and the implementation-related requirements are still frequently mixed here. Therefore, we will look at these explicitly in individual cases in the following.

3 Application Related Timing Aspects

Industrial applications usually require the exchange of data between a controller, the master, and its sensors and actuator, the slaves. This data exchange is typically referred to a downlink and uplink communication. A communication cycle therefore contains at least the transmission of control data from the master in downlink direction and actual data from the slaves in uplink direction. For synchronized applications, a so-called global sampling point (GSP) typically serves as temporal reference. It usually occurs once per communication cycle with a static relation to the beginning of the communication cycle. It defines when the slaves have to activate their command data and capture their actual data.

The structure of the actual communication cycle and control-loop depends on the application. In [Die21] this is discussed from the controller's point of view. From this perspective, there are four different timing methods possible:

- command data optimized timing,
- actual data optimized timing,
- closed-loop optimized timing, or
- cycle time optimized timing.

In order to generalize the naming, we change the definition here to the network view which interchanges command and atual data view compared to [Die21].

In the command data optimized timing, the communication cycle is designed such

that the device can get the latest command data with minimal latency. Additionally the devices are synchronized to the contoller cycle with an offset to the cycle start, so that all devices can use their command values at the same point in time (GSP). This is useful in applications where the actuators need to follow the command values very fast.
The opposite case is the actual data optimized timing. Here the controller needs the latest actual data from the slaves. Additionally to the communication time the sample points (GSP) offset can be optimized to minimize the overall time between sampling in the devices and usage in the controller. This is especially important for fast processes attached locally at the controller.
In the closed-loop optimized timing both, command and actual data, must always be based on the latest actual data. Therefore, sufficient time for processing must be scheduled for both master and slaves. As a result, the cycle time is longer compared to the other timing variants considering the same setup. Using shorter communication cycles for closed loop will be considered in Sec. 4.
The shortest network cycle time could be achieved with the cycle time optimized timing. Here, neither the command data nor the actual data have to be based on latest generated data. Therefore, it is only possible to react to changed values with at least one cycle offset. Due to the correspondingly shorter cycle time, appropriate control can still be achieved.
For this consideration, it is assumed that all data are equivalent in terms of timing requirements. This is a typical assumption of today's industrial application, which saves overhead for additional signaling channels and guarantees a fast transmission using the existing IE based networks. However, from an application point of view, a further differentiation might be beneficial, since not all data actually have the same timing requirements.
Form the applications point of view, we can distinguish between:

- open-loop control,
- local-loop control, and
- closed-loop control.

In open-loop control, the control action is decoupled from the process output. It is based on the calculation of command data while taking the actual data without high real-time requirements for system state calculation only. Therefore, either the communication cycle needs to be faster than the response time of the process to the control data or the system state is stable in time and delayed transmission does not lead to different behavior. Typical examples are on/off switching of motors, heaters or lights over a certain time, where a sufficient control result could be assumed for the process without a need for feedback. The advantage of using open-loop control is the reduction of control complexity and required components for the control loop. It allows individualized communication cycles that correspond to the calculation of new command data, the transmission and activation of these data at the slaves. Here the command data do not have to be based on latest actual data, which allows a further reduction of the network cycle time. Typical programming might be a sequential flow chart (SFC) using states and transitions and enabling easy diagnostics through missing signals necessary for the next step.
Local loop control is used to decouple fast sub-processes from the application controller. It allows to eliminate communication time counting as dead time in the closed

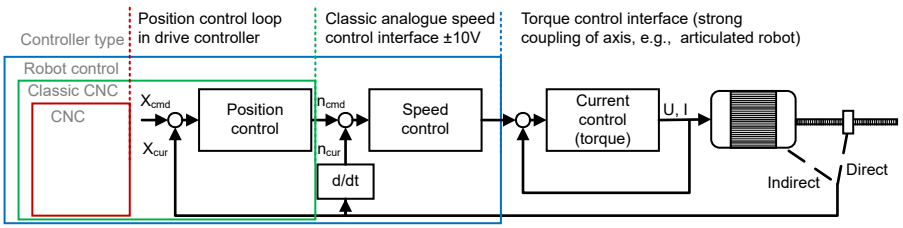

Figure 1: Cascaded control structure.

loop control. Typical examples are drive controller in position set point mode. The command values are target position and start signal plus optional maximum of speed, acceleration and jerk. Motion planner and closed loop control functions are included. In closed loop control the cycle time must be fast enough to capture the upper frequency of the process by fulfilling the Shannon/Nyquist theorem. As we see later in Sec. 4, there are two strategies: one that optimizes the overall loop cycle time and the other that minimizes the network cycle time.

Closed loop control needs synchronization of devices in the system and also of the data transmission. As described before, most applications implement a so called global sampling point (GSP) as temporal reference to activate the command data and capture the actual data. Alignment of the local clocks is either done by additional messages with corresponding synchronization information (e.g. IEEE 1588) or by the encapsulated of these information in the regular cyclic real-time data transmission (e.g. Sercos III). Both might have individual advantages for the timing behavior of the data transmission (cf. Sec. 4).

Based on this classification, it is possible to optimize data transmission for a specific application. One method to represent industrial communication for such an optimization is automata theory. Here, the manufacturing processes or machines are represented as finite automata with a series of states and transitions. A mapping of real-time systems is shown for example in [Pet99].

A main field for synchronized devices is motion control applications. For explanation we consider the control structure of a drive together with its motion controller, computerized numerical control (CNC) or robot control (RC), examplified in Fig. 1.

For Cartesian systems like most machine tools, the axes are independent from each other and can be controlled locally in the drive controller. The CNC interpolates the motion path and sends each position command value to the respective axis. The current position value is only needed for monitoring purpose in the CNC and is not very critical in timely transmission. This constellation can be regarded as open loop control for the CNC and local loop control for the drive controller. Classical CNC systems integrate the position control loop, transfer the speed command value to the speed control loop and need the position feedback value for closed loop operation. As the inner control loops need to be faster, this results to a time critical communication.

For non-cartesian systems like articulated robots, a simple position control loop in the drive does not work because of disturbing momentum of other axes. As the RC

knows the robots kinematic, it is common to use the torque control interface. This leads to even shorter cycle times (typically 125 µs), simplifies the drive controller, but needs a powerful RC. For interoperability of motion controller and drive controller, the interface needs to be standardized. This was done by the fieldbus organizations in regard of the communication aspects by defining appropriate device profiles.
These need to consider timing from the controller application, the network transport and device control loops.

4 Implementation Related Timing Aspects

When looking at different devices profiles for synchronized motion we discover that there are 1- cycle, 2-cycle and even multi-cycles communication models. In this chapter, we describe the background of these models and assign corresponding applications.
Closed loop control always targets to minimize latency inside the loop. In old analogue implementations direct wiring for velocity and feedback position was used. For cost reason serial communication came into industrial controls in the beginning of the 1990^{th}, which helped also to get rid of drift and noise problems but introduced additional latency. The first standard introduced to make the interface between CNC and drives digital was Sercos Interface. A drive controller can be regarded as three cascaded control loops for position, velocity, and torque (cf. Fig. 1). The computer-based NC contained path generation, interpolation, and position control loop. The analogue based drive controller contained velocity and torque (current) control loops. Sercos was designed to interface to any of those control loops in regard of different application requirements. Looking at the communication requirements the outer loops are slower but need higher resolution of the communicated signals. High dynamic machines require fast control loops, so the best would be to have the three loops for position, velocity, and torque running locally in the drive controller without any latency caused by communication. This works perfectly for Cartesian systems like machine tools where there is no cross-dependence between the axis. High performance could be reached by coarse interpolation in the CNC and additional fine interpolation in the drive controller. Supposing that the drive controller can guarantee the required acceleration, the feedback position to the CNC is only needed for monitoring of errors. So, the position interface is just an open loop interface for command position. (Reference to output stream optimization). The cycle time in this case is determined by the minimum cycle time of the controller and the next multiple of control loop cycle time of the drives to keep controller and drives in synchronization, cf. Fig. 2.
For closed loop control it is usual advantageous to choose a 1-cycle model. That would calculate the cycle time by adding the maximum controller calculation time, maximum transport times for sending outputs and inputs including preparing in controller and device. This example is shown in CIP[1] Motion specification, cf. Fig. 2.
The controlling task in the controller would receive the current feedback value, calculate the new command value and send this to the device. The cycle time needs to be integer multiples of the drive control loop time. A common suggestion would be

[1] Common Industrial Protocol

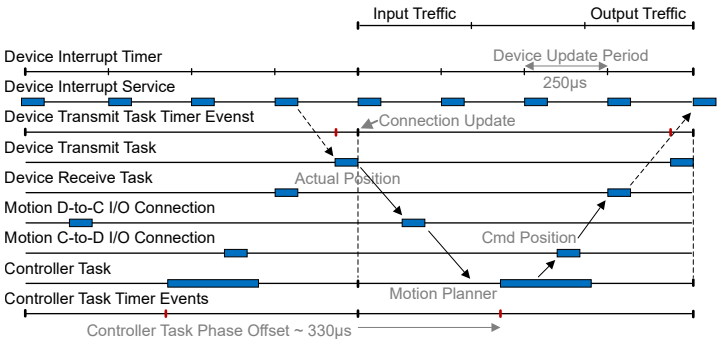

Figure 2: CIP Motion 1-Cycle Timing Model [ODV17].

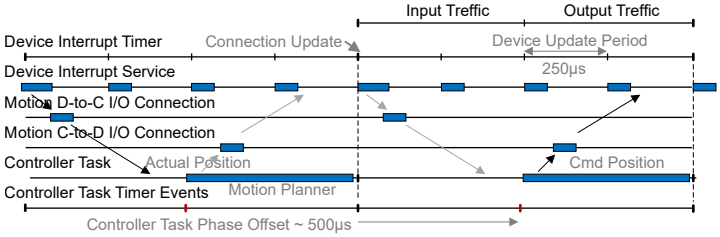

Figure 3: CIP Motion 2-Cycle Timing Model [ODV17].

to divide the cycle time by three for equal parts of calculation time and transport times. This would leave $2/3^{rd}$ of transport and calculation capacity for other purpose supposing full duplex communication.
A 2-cycle timing model would allow a bigger part of the cycle being used for the controller task, cf. Fig. 3.
The drawback is that the command values of the drives are calculated on the feedback values of an earlier cycle. This results in interlaced calculations and requires that the frequency of the signals is below half the frequency of the cycle.
The 3-cycle timing model allows controller calculation and both direction of transport in parallel and spans input, calculation, and output over three cycle. The model is running three interlaced control loops each cycle and therefore could in an extreme case use all resources for computing and transport task, cf. Fig. 4.
The ProfiDrive profile is defined on Profibus/Profinet. As Profibus is a half-duplex system, send and receive directions are not independent and coupled as request and confirmation/response (cf. DX for data exchange in Fig. 5).
A non-optimized multi-cycle timing is shown in Fig. 6.
The inputs can only be used in the next controller cycle and the outputs can only be sent and activated in the following communication cycle. The profile defines three or more communication cycles as the standard for Profidrive. If the Profibus Master is

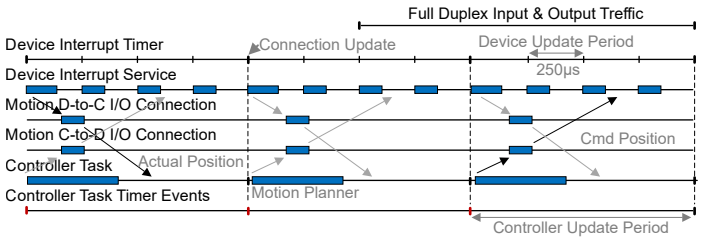

Figure 4: CIP Motion 3-Cycle Timing Model [ODV17].

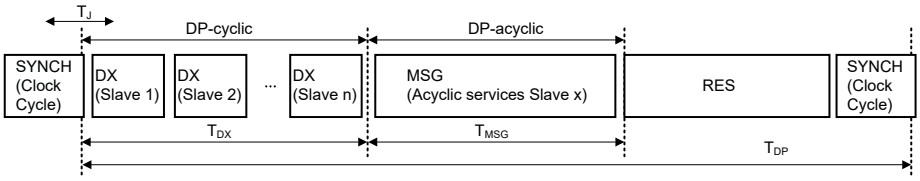

Figure 5: Sequence of an isochronous DP cycle [PRO15].

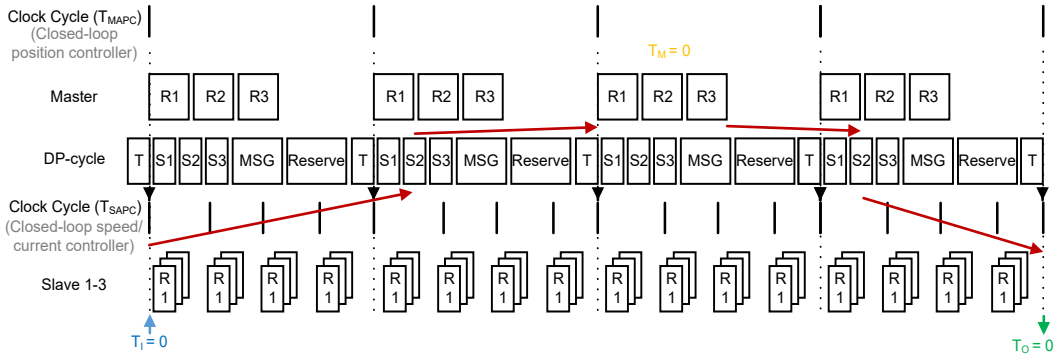

Figure 6: Simple DP cycle [PRO15].

tightly coupled to the control program and the control loops can return their results before the next communication cycle starts, the 2-cycle model is possible, cf. Fig. 7. Profinet is based on Standard Ethernet and uses full-duplex operation, but the 3-cycle model is kept as standard as well. Profinet is working directly on Layer 2 Ethernet frames and also optimizes transport time by dynamic frame packaging (DFP) and cut-through forwarding. The communication cycle time can be lower than in UDP based systems like EtherNet/IP and might compensate for this.
Sercos was the first motion control system and based on TDMA. For synchronization originally a separated telegram was used. With Sercos III this telegram was included

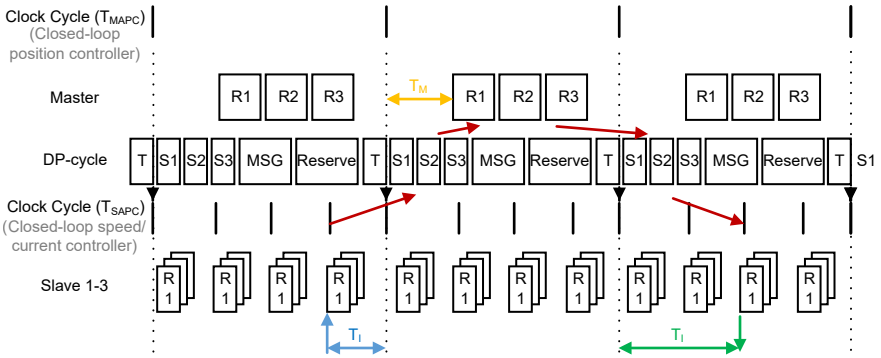

Figure 7: Optimized DP cycle [PRO15].

as a pattern in the first Ethernet frame of the master to device data telegram (MDT0) for data efficiency. The timing defines a real-time channel (RTC) and a channel for universal communication (UCC). Inside the RTC the order of data transport direction can be chosen. This enables optimization of control loops and the latency of command and feedback values. MDT first and AT second enables a 1-cycle timing model. For position control loop locally closed in the drive, the controller internal timing can be aligned to the Sercos timing in both orders of data directions.

5 Matching

After having shown the different applications and their timing requirements together with implementations in the typical fieldbus systems, we want to classify and match the applications to the network properties, as presented in Tab. 1.

Table 1: Matching of application requirements and network properties.

		Applications				
		PLC	Sensor	Drives		CNC/RC
				Position setpoint	Closed-loop control	
Communication	Not synchonized	X[1]	X[1]	X[1]		X[2]
	Synchronized		X[1]		X	X

[1] Optimization possible using asymetric data rate
[2] Only for the integrated PLC

Not synchronized applications like PLC use unsynchronized networks like Profinet RT (Profibus DPV1), Ethernet/IP (DeviceNet). The effect is that at each interface along the data transport an additional buffer is needed and as a worst case a full cycle delay is introduced. The user's approach to reduce the delay and the system reaction time is to set the network cycle time to half of the application cycle time, which generates the requirement for fast network cycle times to the system and device manufacturers. From the device side, most sensors signals are not very time critical and used unsynchronized. Intelligent drives using the setpoint interface include fast closed loop control at local level and can be used with PLC applications for improving precision and simplifying application programming. As devices generate signal of different frequency the data rate could be adjusted by poll/transmit frequency. Especially Sensors are in the first place only transmitting and need an answer only for keep alive of the connection.
Synchronized Applications like CNC/RC need synchronized devices like drives or position feedback sensors in closed loop control mode as well as unsynchronized sensors and actors for their integrated PLC. Modern communication systems based on Ethernet support both synchronized and unsynchronized data transfer in one network. State of the art IE implements this in a more or less proprietary way to each system (Sercos III, EtherCAT, PowerLink, Profinet IRT) or live with higher network cycle time and shift functionality to the device (EtherNet/IP with CIP Sync and CIP Motion). Standardization by IEC/IEEE60802 enable IE to migrate to IEEE802.1 standards like time synchronization (.1AS), quality of service (.1Q) for scheduling (.Qbv), and priorization. Due to synchronization the delay in data transfer can be minimized and buffer resources as well.

6 Summary

In this work, we analyzed and classified timing constraints in industrial networks allowing a more detailed comparison towards the corresponding application requirements. We provided an overview of related work classifying industrial networks and deriving timing requirements. For a specification of application based timing aspects, we compared different timing methods and identified timing constraints of different control structures. We continued with an overview on implementation related timing aspects, investigating typically used Industrial Ethernet (IE) protocols. We compared different cycle models and related them to feasible application control structures. Finally, we concluded our specification with a matching of application requirements towards network properties, allowing a more detailed understanding on various timing constraints in industrial communication.

Bibliography

[5G 20] 5G Alliance for Connected Industries and Automation (5G-ACIA). Key 5G Use Cases and Requirement. Technical report, May 2020.

[D+17] Steven Dietrich et al. Performance indicators and use case analysis for wireless networks in factory automation. In *IEEE International Conference*

on Emerging Technologies And Factory Automation (ETFA'2017). IEEE, September 2017.

[Dec09] Jean-Dominique Decotignie. The many faces of industrial ethernet [past and present]. *IEEE Industrial Electronics Magazine*, 3(1):8–19, 2009.

[Die21] Steven Dietrich. *Methods of Evaluation and Improvement on Cascaded Wired and Wireless Real-Time Communication Networks for Factory Automation*. phdthesis, Brandenburgische Technische Universität Cottbus-Senftenberg, 2021. DOI: https://doi.org/10.26127/BTUOpen-5483.

[F+14] Andreas Frotzscher et al. Requirements and current solutions of wireless communication in industrial automation. In *IEEE International Conference on Communications Workshops (ICC'2014)*. IEEE, 2014.

[GH13] V Çağrı Güngör and Gerhard P Hancke. *Industrial wireless sensor networks: Applications, protocols, and standards*. CRC Press, 2013. ISBN 978-1-1380-7620-4.

[J+07] Juergen Jasperneite et al. Limits of increasing the performance of industrial ethernet protocols. In *IEEE International Conference on Emerging Technologies And Factory Automation (EFTA'2007)*. IEEE, 2007.

[ODV17] ODVA Inc. The CIP Networks Library Volume 9: CIP Motion. Edition 1.1, April 2017.

[Pet99] Paul Pettersson. *Modelling and Verification of Real-Time Systems Using Timed Automata: Theory and Practice*. phdthesis, Uppsala University, February 1999.

[PN09] Carlos E. Pereira and Peter Neumann. *Handbook of Automation*, chapter Industrial Communication Protocols, pages 981–999. Springer Berlin Heidelberg, 2009. ISBN 978-3-540-78831-7.

[PRO15] PROFIBUS Nutzerorganisation e.V. Technical Specification for PROFIBUS and PROFINET. Version 4.2, October 2015. Order No.: 3.172.

[Sau10] Thilo Sauter. The three generations of field-level networks—evolution and compatibility issues. *IEEE Transactions on Industrial Electronics*, 57(11):3585–3595, 2010.

[ser13] sercos international e.V. sercos the automation bus - Communication Specification. Version 1.3.1-1.12, December 2013.

[VDI09] VDI/VDE Society for Measurement and Automatic control (GMA). VDI/VDE 2185 Part 1: Radio based communication in industrial automation. Technical report, 2009.

[ZVE09] ZVEI Automation Division. Coexistence of wireless systems in automation technology. White Paper, 2009.