# Fieldbus based isochronous automation application

Max Felser
Bern University of Applied Sciences
Engineering and Information Technology
Jlcoweg 1, 3400 Burgdorf, Switzerland
max.felser@bfh.ch

## Abstract

*Fieldbus based automation solutions are widely accepted in industrial applications. In the last twenty years the number of Profibus nodes is growing every year with more than 5 millions nodes.*

*The success of this technology is also based on a continuous development and adoption of new mechanism to increase the functionally of the fieldbus technology. One of the adopted features is the possibility to create isochronous automation systems.*

*This paper shows how isochronous automation applications are implemented with Profibus and the performances which can be achieved. An outlook is given to the new Ethernet based fieldbus Profinet and the possibilities to reach the same performances as with the fieldbus.*

## 1. Introduction

Over several decades the fieldbus adapted to the needs of automation environment: Easy manipulation by installation personal, robust in dirty environment, cheap and simple cabling. The star-based traditional cabling was replaced by the more efficient bus- or tree-topology. One of the key features still is the strict limitation of the transmitted data amount to the optimal minimum and the maximization of the real-time performance with a minimal investment in resources.

One of the most successful fieldbus today is Profibus according to IEC 61158 [1] and IEC 61784-1 [2] defined with the compliance family CPF3/1. Until today about 30 millions of nodes are installed with a growing rate of more than 5 Million nodes every year. One of the reasons of this success is the technical improvements defined during the last 20 years for the communication models and technology of the communication protocols.

On the other hand we see that Ethernet and TCP/IP are widely accepted in the Information Technologies (IT) and allow simple access and integration to the world-wide internet networks and technology. In clean office environment Ethernet and TCP/IP are the standard communication network and therefore very cheap and well known by a lot of IT-Experts. If we want to use this technology also for automation applications, we have to adapt the physical interfaces and installation technology to the "field" environment. The transmission of real-time control Information is demanding also some special requirements. More than a dozen of technical solutions are proposed, to make this Ethernet real-time efficient and to allow to compete with the application fields of the traditional fieldbus [3]. One of these solutions is Profinet.

This paper focuses on these real-time requirements and lists first the features and solutions of the classical and proved fieldbus Profibus and its measured performance values. In the last section the mechanisms of Profinet based on IEC 61784-2 [4] compliance family CPF3 are outlined and compared with the results of Profibus.

## 2. Model of an automation system

For this study we use a model of an automation system which is based on the model used in IEC 61784-3-3 [5] to describe the system requirements. This model is very similar to the model defined by [6] for the evaluation of performances of Profinet systems class 1 and also suited to describe an automation system based on Profibus.

We consider a small but typical automation application with several sensors connected to field devices, a logic program within a controller and one or several actuators such as motors or valves connected to field devices as outlined in figure 1. The different field devices are connected to the controller over one common Fieldbus.

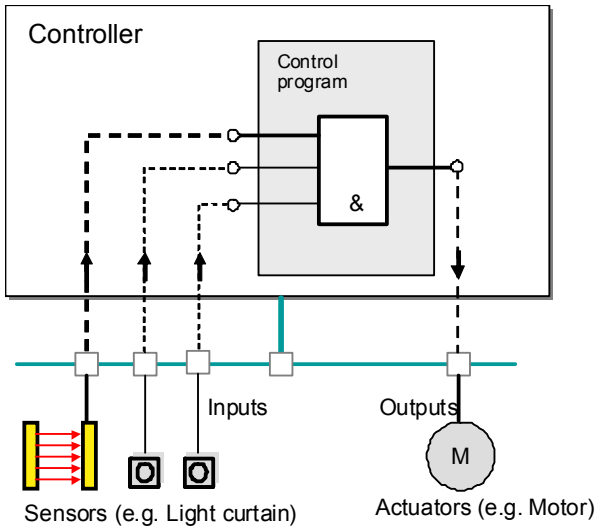



**Figure 1. Typical automation application**

The response time is an important parameter for such an automation system. The response time is defined as the time from the moment a sensor detects an event until we have a reaction at the actuator. This response time is composed of different delays on the signal path. In the used example the signal path is consisting of a sensor device, the bus transfer to the controller, the computing in the controller by the control program, another transfer to the output device, and the output device as the final element. This is outlined in figure 2.

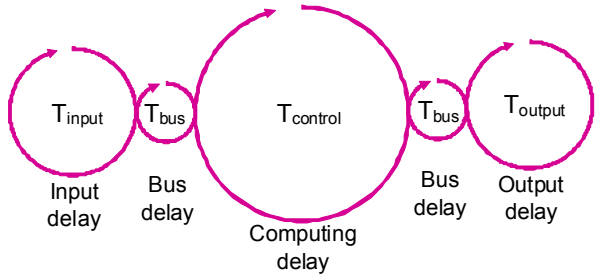



**Figure 2. Delays composing the response time**

Each sensor has its own signal path and thus a particular typical response time. This typical response time consists of several individual time values including the bus delays as shown in the simplified typical response time model of figure 2.

$$T_{response} = T_{input} + T_{control} + T_{output} + 2\,T_{bus} \quad (1)$$

Every of these time elements $T_x$ is composed of a processing time $T_{p,x}$ and a waiting time $T_{w,x}$ to get processed.

$$T_x = T_{w,x} + T_{p,x} \quad (2)$$

This results in having a minimum response time $minT_{response}$ corresponding to the sum of the processing times and a maximum delay time $maxT_{response}$ corresponding to the sum of the processing plus the sum of the waiting times. The actual delay may be any time between these values.

$$minT_{response} = \Sigma_{All}\, T_{p,x} \quad (3)$$

$$maxT_{response} = \Sigma_{All}\, T_{p,x} + \Sigma_{All}\, T_{w,x} \quad (4)$$

The processing time of the bus delay $T_{p,bus}$ is based on the signal propagation delay and the transmission time for a data frame. The waiting time for the bus $T_{w,bus}$ is in the case of a cyclic scheduling based on the cycle time of the bus.

The processing time of the control delay $T_{p,control}$ takes typically less time than the waiting time $T_{w.control}$ introduced with the cyclic execution of the program in the controller. The field devices for the inputs and outputs are supposed to have similar structures with cyclic control programs.

The difference between the $maxT_{response}$ and $minT_{response}$ is called the jitter. So the jitter is the sum of the different waiting times. This jitter is important for the usage of an automation system for closed loop control. It is possible to compensate a defined and fixed response time, but it is almost impossible to scope with a jitter which is larger than the response time.

Reducing the jitter also implies a reduction of the max response time.

So we are looking for a method to keep the response time as stable as possible by reducing the jitter as much as possible. This can be achieved, by reducing the waiting times on every different path. These waiting times are based on the cyclic computing on the different levels.

$$\text{Free running: } T_{jitter} = \Sigma\, T_{w,x} \quad (5)$$

To synchronize all cycles will reduce the jitter to the waiting time of the sensor or to one cycle of the synchronous automation system.

$$\text{Isochronous: } T_{jitter} = T_{cycle} = T_{w,input} \quad (6)$$

## 3. Synchronization with Profibus

There exist different types of stations in a Profibus network: The controller, the engineering station and the field devices. The controller is named the master of class 1 (MC1) and takes the control over an application. It is typically implemented as a Programmable Logic Controller (PLC) or a Process Control System (PCS). In one network there is typically just one such MC1 master or the other way around: for every MC1 master one Profibus network is planned. The second type of stations

in a Profibus network are the field devices called the slaves. They are the valves, remote I/O, transmitters, drives or other devices connected directly to the field. The third types of stations are engineering stations called masters of class 2 (MC2). These engineering stations are used to access the parameters inside the different field devices.

The masters are active and the slaves are passive stations. The passive stations wait to get a service request by the master and take no action by it selves. The MC1 master sends cyclic the output data to the assigned slaves and receives the input data from these slaves as an immediate answer.

First we show how long a bus cycle with Profibus is. Then we make it with a constant length and afterwards we use this stable bus cycle for the synchronization of the application cycles in the field devices and the controller.

### 3.1. Best effort cycle

The Profibus MC1 master by default tries to keep the cyclic polling of his assigned slaves as short as possible. The bus cycle time $T_{cycle,bus}$ is composed of the time needed by the master for the organization of the token passing and the time needed to poll all slaves. The organization overhead for the MC1 $T_{C1}$ is composed by the time needed to pass the token to the next master device, the time needed to update the gap, the time to send a Global Control (GC) for all stations and the time to send an acyclic message to one of the other stations. All these actions except the passing of the token are optional and will be executed not on every cycle.

$$T_{Token} \leq T_{C1} \leq T_{Token} + T_{GAP} + T_{GC} + T_{ac} \quad (7)$$

Very similar is the overhead used by the Profibus MC2 $T_{C2}$ as outlined in formula (8). The only difference is the missing Global Control message.

$$T_{Token} \leq T_{C2} \leq T_{Token} + T_{GAP} + T_{ac} \quad (8)$$

The second component of the cycle time is the time needed to poll the slaves. The polling time for one slave $T_{slave}$ is given by the maximum value of the possible data exchange, the waiting time if a station does not reply and the length of a diagnostic, parameter or configuration message sequence as outlined in formula (9).

$$T_{slave}=\max(T_{DataExchange},T_{NoResponse},T_{Diag},T_{SetPrm},T_{ChkCfg}) \quad (9)$$

If we assume, that there are no communication error once a communication relation is established, $T_{slave}$ is equal to the $T_{p,bus}$ for this slave. This assumption is reasonable according to the study [7].

So the time for a complete data exchange $T_{DX}$ is the sum of all data exchanges with all slaves according to (10).

$$T_{DX} = \Sigma\, T_{slave} \quad (10)$$

The total Bus cycle time $T_{cycle,bus}$ is now the sum of the token master overhead of all masters plus the time for the data exchange.

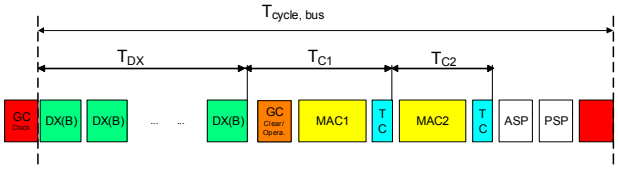
$$T_{cycle,bus} = T_{C1} + T_{DX} + T_{C2} \quad (11)$$

This cycle time gives also a value for the maximal bus delay. The variation and jitter of this $T_{cycle,bus}$ is based on almost all parts: the masters may execute optional services or one or the other slave may be missed or just ask for a diagnostic message. This will result in jitter in $T_{cycle,bus}$ which is out of control of the application.

Most research publications about Profibus show results about the optimization of the token passing [8], [9], the estimation of the optimal token cycle for short bus cycle [10], [11] or the possibility to synchronize the outputs in a distributed system with the help of the SYNC Global Control (GC) command [12]. All these propositions do not reduce the jitter of the bus cycle.

### 3.2. Isochronous cycle

At first the bus cycles have to have the same length in time. In [1] part 5-3 this is called isochronous. For this the cycle is planned according the layout of figure 3.




GC = Clobal Control (Clock; Clear / Operate = MC1 state)
DX(B) = Data Exchange (Broadcast) or Config, PRM, Diag.
MAC1 = One acyclic service of MC1
MAC2 = One acyclic service of MC2
TC = Token
ASP = Active Send Pause
PSP = Passive Send Pause

**Figure 3. Isochronous cycle**

At the beginning of the cycle a first global control (GC) telegram is send to all stations. This telegram contains the indication of group 8, which is reserved for this purpose to signal the start of the cycle. In the next phase the MC1 master exchanges all its cyclic data with its assigned slaves as already calculated in formula (10). An optional additional Global Control may be send by the MC1 to signal its processing state to all related slaves. Afterwards one and only one acyclic service may be performed by the MC1 and the token is passed to another master available on the same bus. This MC2 master is allowed to send now at most one acyclic service on its own. The token is send back to the MC1 master. The MC1 master gets the token back after different delays, depending the optional acyclic services,

repeated messages or other delays. To fill the time he sends now Status Request telegrams to its own address. This signals to the bus that it is still working and busy (ASP). As soon as the resting delay is shorter as one $T_{slot}$ a passive waiting period is added (PSP). The MC1 master is now able to send the next GC telegram at the specified time delay with a jitter of less than 1 µs.

### 3.3. Isochronous applications

With the isochronous cycle the MC1 master sends a GC frame coming always at the same, planned time interval, independent of the load of the bus.

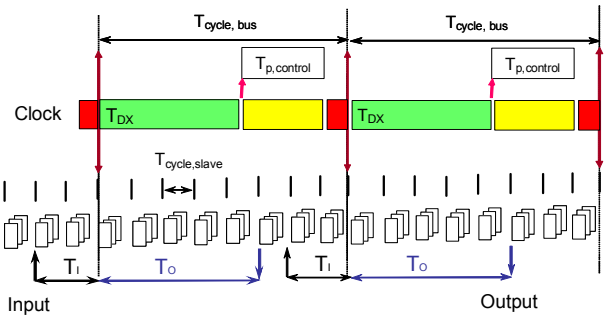



**Figure 4. Isochronous application**

This frame is now used to synchronize the applications in the field devices to this bus-cycle.

On the reception of the GC frame with the Group 8 indication set, the slave triggers his local clock. This signal is passed to a local Phase-Lock-Loop (PLL), which is synchronized with a multiple of the bus cycle.

With additional parameters is fixed to which local time in advance of the global clock the inputs have to be fetched $T_I$ and at which time the outputs have to be set $T_O$ (see figure 4). So all devices in one network are able to fetch all input signals at the same microsecond. The conditions to get a working system are summarized in formulas (12).

$$T_I > T_{p,input}$$

$$T_O > T_{p,output} \qquad (12)$$

$$T_{cycle,bus} > T_{DX} + T_{p,control}$$

The controller application cycle is synchronized at the end of the cyclic data exchange $T_{DX}$ as shown in figure 4. This is the moment, where all cyclic input data is available and the application controller may start its calculations. Instead of a time triggered application we need an event triggered application, which is triggered by this end of the cyclic communication. This results in a stable and defined system reaction time as shown in (13).

$$T_{reaction} = T_I + T_{cycle,bus} + T_O + T_{jitter} \qquad (13)$$

$$0 < T_{jitter} < T_{cycle,bus}$$

The only jitter is given due to the first caption of the event, the rest of the system is synchronous. In fact we replaced the asynchronous automation system by a synchronous control system. Based on the isochronous bus-cycle the complete automation system is now also isochronous.

## 4. Reference benchmark

We need a simple configuration to measure the possible quality of synchronization reachable with such an installation. For this we define a benchmark test with only the output part of the response time. The controller toggles the value of one digital output every 3 ms and we measure the signal at the remote output transmitted over a Profibus-DP network running with 12 MBit/s.

### 4.1. Test bed

For the hardware a simple controller CPU 315-2 PN/DP from Siemens and a remote IO system ET200S with the IM151-3 High Feature from the same manufacturer was selected.

The output signal is measured with a standard scope with storage and the timing of the Profibus is monitored with a commercial monitor allowing statistical measurements not only of the frames but also of the content of the data frames [13]. This permits to measure the timing of the output signal also on the Profibus network with a time resolution of less than one microsecond.

### 4.2. Test configurations

The signal is generated with a simple output toggling application which is located in different organization blocks (OB) of the controller. OB35 can be configured to be executed at a time interval of 3ms. OB61 can be executed synchronous with the Profibus network. Profibus itself can be configured to be free running or to run with isochronous cycles of a specified length and the remote I/O can be free running or synchronized on the bus cycle.

With these configurations the different configurations as listed in table 1 are possible.

**Table 1: Possible configurations**

| Config | Controller | Bus | Remote I/O |
|---|---|---|---|
| 1 | OB35 (3ms) | Free | Free |
| 2 | OB35 (3ms) | Cyclic (1ms) | Free |
| 3 | OB35 (3ms) | Cyclic (1ms) | Synchronic |
| 4 | OB61 | Cyclic (3ms) | Free |
| 5 | OB61 | Cyclic (3ms) | Synchronic |

In the case where there is no synchronization of the controller application with the bus, the cycle of the bus is

selected to be three times smaller than the cycle of the application.

### 4.3. Test results

Some of the typical results are shown in figure 5. As expected with configuration 2 we have a non negligible jitter of the output signal. You can see, that the signal reaches never the expected duration of 3ms and the jitter is in the range of 2,4 ms.

In configuration 3 the jitter based on the field device is eliminated and we see that the jitter stays for 1ms. This is exactly the cycle time of the Profibus network. We verified this cycle also with a monitor on the bus and it is stable with 1 ms and a jitter of 1µs. The user signal has the jitter of the output signal already on the bus. The monitor shows cycles of 2 – 4 ms for the user data. This result is not surprising: the OB35 cycle is not synchronized with the bus cycle and one cycle may drift against the other. This results in the measured signal.

In configuration 4 the application in the controller is synchronized with the bus but not the application in the remote I/O. We see now the output cycle of the field device producing a jitter. In fact the output is never at the expected 3 ms but only at the values of 2,8 and 3,2 ms with a difference of 0,4 ms. The verification with the monitor on the bus shows, that the data frames are on time and the user data on the bus is toggling as expected and without any exceptions. So the jitter is introduced by the remote I/O device. We can interpret that the output in this remote I/O device is cyclic, with a fixed cycle of 0,4 ms but by default synchronised to the Profibus network.

With configuration 5 we get the expected perfect output signal.

### 4.4. Conclusions

As a first conclusion we can see, that we have only full control over the timing and jitter free output signals, if the application in the controller and the application in the remote I/O are synchronised on the Profibus network.

## 5. Profinet based automation

The structure of an automation system based on Profinet is very similar to a Profibus system. This is done by purpose. Table 2 gives an overview of the correspondence of the different types of stations in an automation network.

**Table 2: Different stations in Profibus and Profinet**

| Profibus-DP | Profinet-IO |
|---|---|
| DP-System | IO-System |
| DP-Master Class 1 | IO-Controller |
| DP-Master Class 2 | IO-Supervisor |
| DP-Slave | IO-Device |

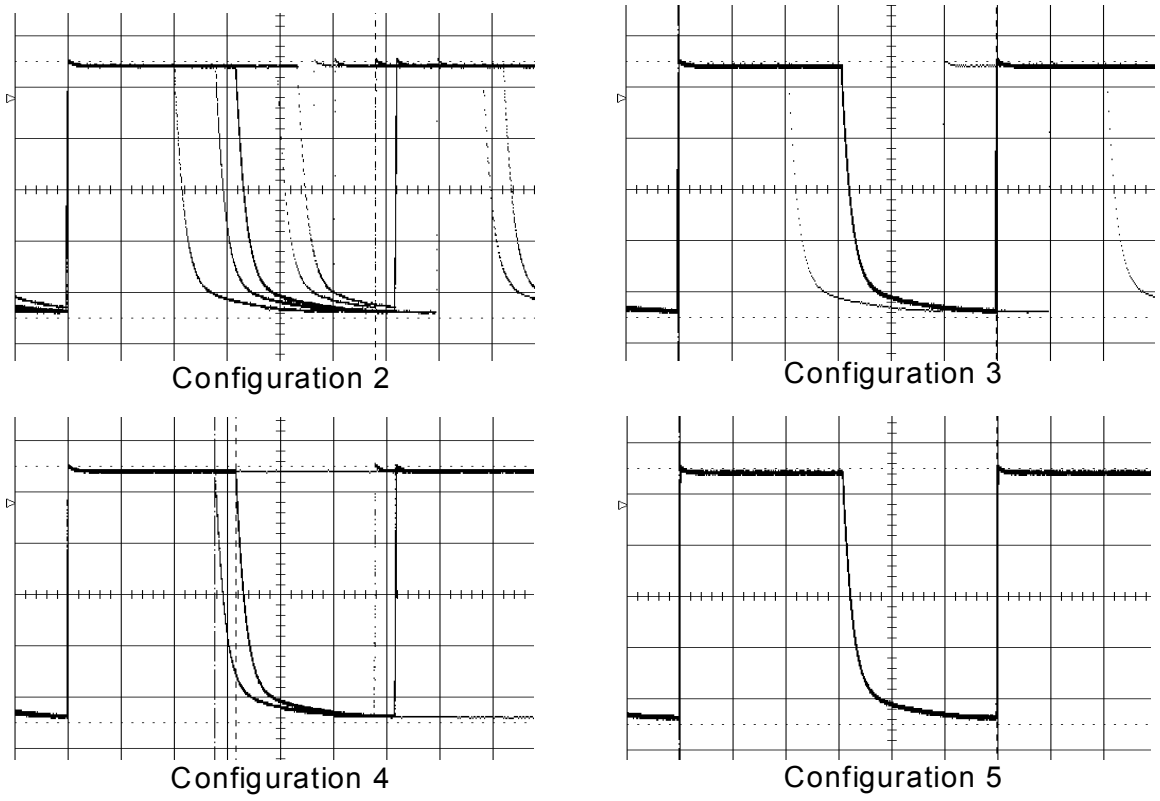



**Figure 5. Output signals of the different configurations (scale = 1ms)**

The Profinet IO-System is well introduced and explained in [14].

### 5.1. Basic communication

The communication model of Profinet is different to the one in Profibus. There exist an Application Relation (AR) between an IO-Controller and every related IO-Device. Inside these ARs we have at least one Communication Relation (CR) specifying the cyclic data exchange between the IO-Controller and the IO-Device, a module or submodule out of the IO-Device. Every cycle can be different; there is no global cycle for the communication anymore.

Every end-point of a CR is sending on its own clock based cycle. In class A or B applications these cycles are not synchronized and may be dephased and have a slightly different cycle length even if the nominal value is the same. This protocol is implemented in most products today and is known under the naming of Profinet RT for Real-Time. In [6] we find some simulation of such a system and practical measurements are published in [15].

Even if the communication model of Profinet is different from the one of Profibus, the application model is almost the same. So if we consider a similar reference application as shown in figure 1 we have the same response time as outlined in formula (1) with the jitter of formula (5). Increasing the Profibus bit rate from the 12 MBit/s half-duplex to the 100 MBit/s full-duplex of the Profinet system results in the same jitter. The jitter is based on the reachable cycle time of the bus and the bit rate has only small influence.

The goal is therefore very similar as for the Profibus system: first we need an isochronous reference and then we have to synchronize the application to this reference.

### 5.2. Isochronous communication

In Profinet class C applications the time is synchronized based on the protocols defined in IEEE1588 and the extensions of the flying time stamping. These principles are explained and proven by [16]. So for the isochronous communication the scheduling of the different cycles can be synchronized on this isochronous time. These protocols are known and documented in [1] under the name Isochronous Real-Time (IRT).

The IRT protocol does need higher performance in measurement equipment [17] and leads to promising measurement results with prototypes [18].

We have now a system with one cycle of the bus which is the same for all stations in the synchronized domain. This does solve the problem of the network scheduling and transmission delays in the network as explained and outlined in [19]. But this does not reduce the jitter of the response time of the automation application! The jitter of the bus cycle has only a minimal influence to the jitter of the response time in formula (5).

A possibility to synchronize an application to this isochronous cycle is defined [1] part 5-10 clause 8.3.7. With this mechanism it is possible, to realize an isochronous automation application with the same features and parameters as with the Profibus standard.

The open question is now: are these protocols and features already implemented in commercial products, that we can reproduce the simple benchmark also for such a Profinet system?

### 5.3. Verification of the benchmark

The verification was not so easy as expected. It is possible to configure the same controller we used for the Profibus benchmark – the CPU315-2 PN/DP - also for Profinet, but only the RT Version. We specify a CR with a send cycle of 1ms and we toggle the signal as before in OB35 with 3 ms interval. There is no possibility to specify in the remote I/O device IM151-3 a synchronization of the outputs similar to the Profibus version. The output signal is identical with the Profibus configuration 3: It looks as if the application cycle inside the remote I/O device is by default synchronized to the CR cycle. Only the jitter based on the non-synchronization of the control application to the communication is detected.

We did not find any possibility to synchronize the application of the controller to the cycle of the Profinet network.

With this measurement we used only the Real-Time (RT) protocol of Profinet. To use the Isochronous Real-Time (IRT) protocol we have to add an additional communication module CP 343-3 extended to the controller. This module can be configured to run the IRTflex protocol. This leads to synchronized clocks on all devices in the same IO-System on the Profinet network. As we measured with different network monitoring tools [17] the cyclic transmission is isochronous [18].

The Problem with this solution is the missing possibility to synchronize the application on the controller. To move user data from the controller application to the network adapter it takes unexpected long time, which is far longer than 3 ms and includes also unexpected high jitter. The output signal gives impulses in the range of 2 to 7 ms and the system fails completely to reach the required performance.

We had no other, more powerful equipment available in the laboratory to extend the measurements to other configurations.

## 6. Summary and Conclusion

We showed that an automation system based on a Profibus-DP system can be configured in different ways to reach different levels of performance. It is possible to

reach a complete synchronous and isochronous distributed automation system with defined performances in respect to cycle time and jitter. We showed with a simple benchmark test for output signals that these performances can be reached in practical applications with commercial products.

Automation systems based on Profinet-IO networks have from the user point of view by purpose the same structure. But the communication paradigms are different. At the moment it is not possible to reach the same performance and quality for a distributed automation system with Profinet-IO as for a Profibus-DP system with commercial products in the same price class.

To reach the same performance figures, the application of the controller has to be synchronized with the synchronized clock of the Profinet Isochronous Real-Time (IRT) system. To get this performance as defined in the specification the products have to be extended with this possibility. As soon as this is the case the test can be repeated with these new products.

## References


[1] IEC 61158: Digital data communications for measurement and control - Fieldbus for use in industrial control systems, available at www.iec.ch

[2] IEC 61784-1: IEC 61784-1: Industrial communication networks - Profiles - Part 1: Fieldbus profiles, available at www.iec.ch

[3] Felser, M.: Real-Time Ethernet - Industry Prospective, Proceedings of the IEEE, vol. no.6, june 2005, pages 1118 ff

[4] IEC 61784-2: “Industrial communication networks – Profiles - Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3”, available at www.iec.ch

[5] IEC 61784-3-3: Industrial communications networks – Profiles – Part 3-3: Functional safety fieldbuses – Additional specifications for CPF 3, available at www.iec.ch

[6] Ferrari, P.; Flammini, A.; Marioli, D.; Taroni, A.; Venturini, F.: New Simulation Models to Evaluate Performance of PROFINET IO Class 1 Systems, 5th IEEE International Conference on Industrial Informatics, 23-27 June 2007, Volume: 1, pages: 237 – 242

[7] Felser, M.: Quality of Profibus Installations, WFCS 2006, 6th IEEE International Workshop on Factory Communication System, June 27-30, 2006, Conference Center Torino Incontra, Torino, Italy

[8] Cavalieri, S.; Monforte, S.; Tovar, E.; Vasques, F.: Evaluating worst case response time in mono and multi-master profibus DP, 4th IEEE International Workshop on Factory Communication Systems, 2002, pages: 233 – 240

[9] Kaghazchi, H.; Hongxin Li; Ulrich, M.: Influence of Token Rotation Time in multi master PROFIBUS networks, IEEE International Workshop on Factory Communication Systems, 2008. WFCS 2008. 21-23 May 2008, pages: 189 – 197

[10] Vitturi, S.: Stochastic model of the Profibus DP cycle time, IEE Proceedings Science, Measurement and Technology, 4 Sept. 2004, Volume: 151 , ISSN: 1350-2344, pages: 335 - 342

[11] Fang He, Weiming Tong: Estimation to the Time Parameters of PROFIBUS Network System, 2nd IEEE Conference on Industrial Electronics and Applications, 2007. ICIEA 2007. 23-25 May 2007, pages: 873 - 876

[12] Fang He, Weiming Tong, Qiang Wang: Synchronization Control Strategy of Multi-motor System Based on Profibus Network, 2007 IEEE International Conference on Automation and Logistics, 18-21 Aug. 2007 in Jinan, pages: 3029 - 3034

[13] http://www.procentec.com/profitrace2

[14] PROFIBUS International: “PROFINET: Technology and Application, System Description”, Document number: 4.132, Issue April 2006, available at http://www.profibus.com

[15] Ferrari, P.; Flammini, A.; Marioli, D.; Taroni, A.; Venturini, F.: Experimental analysis to estimate jitter in PROFINET IO Class 1 networks, IEEE Conference on Emerging Technologies and Factory Automation, 2006. ETFA '06. 20-22 Sept. 2006, pages: 429 - 432

[16] Jasperneite, J. Shehab, K. Weber, K. : Enhancements to the time synchronization standard IEEE-1588 for a system of cascaded bridges, Proceedings. 2004 IEEE International Workshop on Factory Communication Systems, 22-24 Sept. 2004, pages: 239 - 244

[17] Schafer, I.; Felser, M.: Precision of ethernet measurements based on software tools, IEEE Conference on Emerging Technologies & Factory Automation, 2007. ETFA. 25-28 Sept. 2007, pages: 510 - 515

[18] Ferrari, P.; Flammini, A.; Marioli, D.; Taroni, A.; Venturini, F.: Evaluation of timing characteristics of a prototype system based on PROFINET IO RT_Class 3, IEEE Conference on Emerging Technologies & Factory Automation, 2007. ETFA. 25-28 Sept. 2007, page(s): 1254 - 1261

[19] Jasperneite, J. Imtiaz, J. Schumacher, M. Weber, K.: A Proposal for a Generic Real-Time Ethernet System, IEEE Transactions on Industrial Informatics, May 2009, Volume: 5 , Issue: 2, pages: 75 – 85